

**ANALISIS PERFORMA SISTEM ABSENSI
MENGUNAKAN TEKNIK *LOAD TESTING* DENGAN
JMETER DI SMK TARUNA BAKTI KERTOSONO**

SKRIPSI



Oleh

**Evita Nur Sianturi
E41220819**

**PROGRAM STUDI DI LUAR KAMPUS UTAMA (PSDKU)
TEKNIK INFORMATIKA KAB. NGANJUK
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER
2026**

**ANALISIS PERFORMA SISTEM ABSENSI
MENGUNAKAN TEKNIK *LOAD TESTING* DENGAN
JMETER DI SMK TARUNA BAKTI KERTOSONO**

SKRIPSI



Sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains Terapan
Komputer (S.Tr.Kom) di Program Studi Di Luar Kampus Utama
(PSDKU) Teknik Informatika Kab. Nganjuk
Jurusan Teknologi Informasi

Oleh

**Evita nur Sianturi
E41220819**

**PROGRAM STUDI DI LUAR KAMPUS UTAMA (PSDKU)
TEKNIK INFORMATIKA KAB. NGANJUK
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER
2026**

**KEMENTERIAN PENDIDIKAN TINGGI SAINS, DAN TEKNOLOGI
POLITEKNIK NEGERI JEMBER
JURUSAN TEKNOLOGI INFORMASI**

**ANALISIS PERFORMA SISTEM ABSENSI
MENGUNAKAN TEKNIK LOAD TESTING DENGAN
JMETER DI SMK TARUNA BAKTI KERTOSONO**

Evita Nur Sianturi (E41220819)

Telah diuji pada Tanggal 6 Juli 2026
dan dinyatakan Memenuhi Syarat

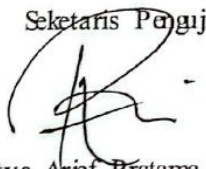
Menyetujui

Ketua Penguji,



Ulfa Emi Rahmawati, SKom., M.Kom.
NIP. 199706282022032018

Sekretaris Penguji,



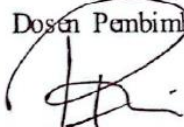
Raditya Arief Pratama SKom., M.Eng
NIP. 199310092024061001

Anggota Penguji,



Puji Hastuti ST., M.Eng
NIP. 199510302024062003

Dosen Pembimbing,



Raditya Arief Pratama SKom., M.Eng
NIP. 199310092024061001

Mengesahkan

Ketua Jurusan Teknologi Informasi



H. Hendra Yuli Riskiawan, SKom, M.Cs
NIP. 19830203 2006041003

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Evita Nur Sianturi

NIM : E41220819

menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam Laporan Akhir/Skripsi/Tesis saya yang berjudul "Analisis Performa Sistem Absensi Menggunakan Teknik *Load testing* Dengan Jmeter Di Smk Taruna Bakti Kertosono" merupakan gagasan dan hasil karya saya sendiri dengan arahan komisi pembimbing dan belum pernah diajukan dalam bentuk apapun pada perguruan tinggi mana pun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan dari penulis lain telah disebutkan dalam teks dan dicantumkan dalam daftar pustaka di bagian akhir Laporan Akhir/Skripsi/Tesis ini.

Nganjuk, 06 Juli 2026



Evita Nur Sianturi

NIM E41220819



**PERNYATAAN
PERSETUJUAN BUBLIKASI
KARYA ILMIAH UNTUK KEPENTINGAN
AKADEMIS**

Yang bertandatangan di bawahini, saya:

Nama : Evita Nur Sianturi
Nim : E41220819
Program Studi : Teknik Informatika
Jurusan : Teknologi Informasi

Demi pengembangan Ilmu Pengetahuan, saya menyetujui untuk memberikan kepada UPT.Perpustakaan Politeknik Negeri Jember, Hak Bebas Royalti Non-Eksklusif (Non-Exclusive Royalty Free Right) atas Karya Ilmiah berupa Laporan Skripsi saya yang berjudul :

**ANALISIS PERFORMA SISTEM ABSENSI MENGGUNAKAN TEKNIK
LOAD TESTING DENGAN JMETER DI SMK TARUNA BAKTI
KERTOSONO**

Dengan Hak Bebas Royalti Non-Eksklusif ini UPT, Perpustakaan Politeknik Negeri Jember berhak menyimpan, mengalih media atau format, mengelola dalam bentuk Pangkalan Data (Database), mendistribusikan karya dan menampilkan atau mempublikasikannya di Internet atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis atau pencipta.

Saya bersedia untuk menanggung secara pribadi tanpa melibatkan pihak Politeknik Negeri Jember, Segala bentuk tuntutan hukum yang timbul atas Pelanggaran Hak Cipta dalam Karya ilmiah ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Nganjuk
Pada Tanggal : 06 Juli 2026
Yang menyatakan,

Nama : Evita Nur Sianturi
Nim : E41220819

MOTTO

"Setiap revisi adalah langkah menuju versi terbaik dari diri sendiri."

(Penulis)

PERSEMBAHAN

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa atas segala kasih, berkat, dan penyertaan-Nya sehingga saya dapat menyelesaikan skripsi ini dengan baik. Proses penyusunan skripsi ini memberikan banyak pelajaran, pengalaman, perjuangan, serta tantangan yang mampu membentuk saya menjadi pribadi yang lebih kuat dan lebih baik, khususnya kepada :

1. Kedua orang tua tercinta, yang selalu memberikan doa, dukungan, kasih sayang, pengorbanan, dan semangat tanpa henti selama saya menempuh pendidikan hingga menyelesaikan skripsi ini. Terima kasih atas segala perjuangan dan kepercayaan yang selalu diberikan kepada saya.
2. Almarhum kakek dan nenek tercinta yang semasa hidupnya selalu memberikan kasih sayang, doa, perhatian, serta semangat kepada saya untuk terus melanjutkan pendidikan hingga ke bangku perkuliahan. Terima kasih atas semua kenangan, nasihat, dukungan, dan cinta yang telah diberikan. Semoga beliau bangga melihat saya mampu sampai di tahap ini dan menyelesaikan pendidikan dengan baik
3. Dosen pembimbing yang telah memberikan arahan, ilmu, masukan, serta bimbingan dengan penuh kesabaran sehingga skripsi ini dapat terselesaikan dengan baik.
4. Untuk pemilik NIM E41221880 yang selalu menemani, mendengarkan keluh kesah, memberikan semangat, dukungan, perhatian, serta motivasi selama proses penyusunan skripsi ini hingga selesai.
5. Diri saya sendiri yang telah berjuang, bertahan, bangkit, dan tidak menyerah dalam menyelesaikan seluruh proses hingga berada di titik ini. Terima kasih karena sudah mampu melewati semua tantangan dengan baik.

**ANALISIS PERFORMA SISTEM ABSENSI MENGGUNAKAN TEKNIK
LOAD TESTING DENGAN JMeter DI SMK TARUNA BAKTI
KERTOSONO**

Dibimbing oleh Raditya Arief Pratama, S.Kom., M.Eng.

Evita Nur Sianturi

Program Studi D-IV Teknik Informatika
Jurusan Teknologi Informasi

ABSTRAK

Penelitian ini bertujuan untuk menganalisis performa sistem absensi berbasis *website* dan *Mobile* di SMK Taruna Bakti Kertosono menggunakan teknik *load testing* dengan Apache JMeter. Latar belakang penelitian ini adalah meningkatnya penggunaan sistem absensi digital berbasis *face recognition* dan *QR Code* yang digunakan oleh siswa, guru, dan admin secara bersamaan, sehingga berpotensi memengaruhi stabilitas dan kecepatan *respons* sistem, terutama pada jam sibuk. Kondisi tersebut menimbulkan kekhawatiran terkait kemampuan sistem dalam menangani banyak pengguna secara bersamaan dan pentingnya pengujian performa sebelum sistem digunakan secara optimal. Metode penelitian yang digunakan adalah *load testing* dengan bantuan Apache JMeter sebagai *tools* pengujian performa sistem. Pengujian dilakukan secara bertahap dengan peningkatan jumlah pengguna dan dilakukan berulang kali untuk mengetahui konsistensi performa sistem. Selain itu, penelitian ini juga menggunakan pendekatan *gorilla testing* untuk menguji ketahanan sistem terhadap penggunaan secara intensif dan berulang. Pengujian difokuskan pada fitur *scan QR Code* dan kelola perizinan pada aplikasi *mobile*, serta fitur rekap absensi pada *website*. Hasil penelitian menunjukkan bahwa aplikasi *mobile* siswa memiliki performa yang baik berdasarkan parameter Load Time, Latency, Throughput, dan Error Rate. Nilai Response Time pada sebagian besar skenario masih berada di atas waktu *respons* optimal sebesar 0,1 detik. Pengujian pada *website* dan aplikasi *mobile* guru menunjukkan beberapa fitur mengalami Error Rate di atas 5% akibat keterbatasan lingkungan pengujian, sedangkan *gorilla testing* menunjukkan bahwa sebagian besar fitur tetap stabil selama eksekusi berulang. Hasil penelitian ini menjadi dasar evaluasi untuk pengembangan sistem absensi.

Kata Kunci: Sistem Absensi, *Load testing*, Apache JMeter, *Gorilla testing*, Performa Sistem

**ANALISIS PERFORMA SISTEM ABSENSI MENGGUNAKAN
TEKNIK *LOAD TESTING* DENGAN JMETER DI SMK TARUNA
BAKTI KERTOSONO**

*(PERFORMANCE ANALYSIS OF AN ATTENDANCE SYSTEM USING LOAD
TESTING TECHNIQUES WITH JMETER AT SMK TARUNA BAKTI
KERTOSONO)*

Supervised by Raditya Arief Pratama, S.Kom., M.Eng.

Evita Nur Sianturi

Study Program of Applied Bachelor in Informatics Engineering

Department of Information Technology

ABSTRACT

This study aims to analyze the performance of the web-based and mobile attendance system at SMK Taruna Bakti Kertosono using the load testing technique with Apache JMeter. The background of this study is the increasing use of a digital attendance system based on face recognition and QR Code, which is accessed simultaneously by students, teachers, and administrators. This condition has the potential to affect the system's stability and response time, particularly during peak hours. Therefore, performance testing is required to ensure that the system can handle multiple concurrent users before being fully implemented. The research method employed load testing using Apache JMeter as the performance testing tool. Load testing was conducted by gradually increasing the number of virtual users, while gorilla testing was performed using repeated execution (looping) on the same test scenarios to evaluate the system's robustness under continuous operations. The testing focused on the QR Code scanning and leave request management features in the mobile application, as well as the attendance report feature on the website. The results indicate that the student mobile application demonstrated good performance in terms of Load Time, Latency, Throughput, and Error Rate. However, the Response Time in most test scenarios remained above the optimal response time of 0.1 seconds. The website and teacher mobile application showed several features with an Error Rate exceeding 5% due to testing environment limitations. Meanwhile, the gorilla testing results showed that most features maintained stable performance during repeated execution. These findings provide an evaluation that can be used as a basis for further improvement and development of the attendance system.

Keywords: *Attendance System, Load testing, Apache JMeter, Gorilla testing, System Performance.*

RINGKASAN

Analisis Performa Sistem Absensi Menggunakan Teknik *Load testing* Dengan Jmeter Di Smk Taruna Bakti Kertosono, Evita Nur Sianturi, NIM E41220819, Tahun 2026, 139 hlm., Teknik Informatika, Politeknik Negeri Jember, Raditya Arief Pratama, S.Kom., M.Eng (Pembimbing).

Penelitian ini dilakukan untuk menganalisis performa sistem absensi berbasis website dan mobile di SMK Taruna Bakti Kertosono yang menerapkan teknologi face recognition dan QR Code. Meningkatnya jumlah pengguna yang mengakses sistem secara bersamaan berpotensi memengaruhi stabilitas dan kecepatan respons sistem. Selain itu, penelitian terdahulu masih memiliki keterbatasan, seperti skenario pengujian yang sederhana, jumlah pengguna yang terbatas, serta belum mengombinasikan load testing dan gorilla testing. Penelitian ini bertujuan menganalisis kemampuan sistem dalam menangani beban pengguna, mengevaluasi performa berdasarkan parameter Load Time, Response Time, Throughput, Latency, dan Error Rate, serta menguji ketahanan sistem menggunakan gorilla testing.

Pengujian dilakukan menggunakan metode load testing dengan bantuan Apache JMeter, sedangkan gorilla testing dilakukan melalui perulangan (looping) pada skenario yang sama untuk menguji ketahanan sistem terhadap eksekusi fitur secara terus-menerus. Hasil penelitian menunjukkan bahwa aplikasi mobile siswa memiliki performa yang baik dengan Load Time di bawah 3 detik, Latency di bawah 1 detik, Throughput yang stabil, dan Error Rate di bawah 5%. Namun, Response Time pada sebagian besar skenario masih berada di atas waktu respons optimal sebesar 0,1 detik. Pengujian pada website dan mobile guru menunjukkan beberapa fitur memiliki Error Rate di atas 5% akibat keterbatasan lingkungan pengujian, sedangkan gorilla testing menunjukkan bahwa sebagian besar fitur tetap stabil selama eksekusi berulang.

Hasil penelitian ini memberikan gambaran mengenai kemampuan sistem dalam menangani beban pengguna serta menjadi bahan evaluasi bagi pengembang untuk meningkatkan performa dan keandalan sistem absensi digital di SMK Taruna Bakti Kertosono.

PRAKATA

Puji syukur penulis panjatkan ke hadirat Allah SWT atas berkat rahmat dan karunia-Nya sehingga penulisan skripsi yang berjudul " Analisis Performa Sistem Absensi Menggunakan Teknik *Load testing* Dengan Jmeter Di Smk Turuna Bakti Kertosono" dapat diselesaikan dengan baik dan lancar. Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Terapan Komputer (STrKom) pada Program Studi Teknik Informatika, Jurusan Teknologi Informasi Sebagai bentuk apresiasi penulis menyampaikan penghargaan dan ucapan terima kasih yang sebesar-besarnya kepada pihak-pihak berikut:

1. Bapak Suiful Anwar, S.TP, MP. selaku Direktur Politeknik Negeri Jember.
2. Bapak Ir.Hendra Yufit Riskiawan, SKom, M.Cs. selaku Ketua Jurusan Teknologi Informasi.
3. Ibu Ulfa Eri Rahmawati, SKom, M.Kom. selaku Koordinator Program Studi Teknik Informatika PSDKU Nganjuk.
4. Bapak Raditya Arief Pratama, SKom, M.Eng selaku dosen pembimbing yang telah mduangkan waktu, memberikan arahan, masukan, serta ilmu yang sangat bermanfaat selama proses penyusunan skripsi ini.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun demi perbaikan di masa mendatang. Semoga penelitian ini dapat memberikan manfaat

Jember, 06 Juli 2026



Evita Nur Sianturi
NIM E41220819

DAFTAR ISI

	Halaman
HALAMAN SAMPUL	i
HALAMAN JUDUL	ii
HALAMAN PENGESAHAN PROPOSAL	iii
KEMENTERIAN PENDIDIKAN TINGGI SAINS, DAN TEKNOLOGI	iii
SURAT PERNYATAAN	iv
MOTTO.....	vi
PERSEMBAHAN.....	vii
ABSTRAK	viii
<i>ABSTRACT</i>	ix
RINGKASAN.....	x
PRAKATA	xi
DAFTAR ISI	xii
DAFTAR TABEL	xiv
DAFTAR GAMBAR	xv
DAFTAR LAMPIRAN	xviii
BAB 1 PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah.....	5
1.3. Tujuan	5
1.4. Manfaat	5
1.5. Batasan Masalah	6
BAB 2 TINJAUAN PUSTAKA	7
2.1. <i>Software testing</i>	7
2.2. <i>Performance testing</i>	7
2.3. <i>Load testing</i>	8
2.4. Apache Jmeter	11
2.5. <i>Website</i>	11
2.6. <i>Aplikasi Mobile</i>	12
2.7. <i>Application Programming Interface (API)</i>	13
2.8. <i>Populasi dan Sampel</i>	14

2.9	<i>Software Testing Life Cycle (STLC)</i>	15
2.10	<i>Test Plan</i>	20
2.11	<i>Test case</i>	20
2.12	<i>Gorilla testing</i>	21
2.13	<i>Automation testing</i>	21
2.14	<i>State of the Art</i>	22
BAB 3 METODOLOGI PENELITIAN.....		29
3.1	Waktu dan Tempat Pelaksanaan	29
3.2	Alat dan Bahan.....	29
3.3	Tahapan Penelitian	30
3.4	Jadwal Pelaksanaan Penelitian.....	33
BAB 4 HASIL DAN PEMBAHASAN.....		35
4.1.	Identifikasi Masalah.....	35
4.2.	Studi Literatur.....	36
4.3.	<i>Requirement Analysis</i>	37
4.4.	<i>Test Planning</i>	39
4.5.	<i>Test case Development</i>	40
4.6.	<i>Test Envionment setup</i>	47
4.7.	<i>Test Execution</i>	68
4.8.	<i>Test Closure</i>	95
4.9.	Hasil Analisis.....	98
BAB 5 KESIMPULAN DAN SARAN		106
5.1.	Kesimpulan.....	106
5.2	Saran.....	107
DAFTAR PUSTAKA.....		108
LAMPIRAN		113

DAFTAR TABEL

	Halaman
Tabel 2.1 Contoh <i>Test Plan</i>	17
Table 2.2 Contoh <i>Test Case</i>	18
Tabel 3. 1 Jadwal Penelitian	33
Tabel 4. 1 Rincian Skenario Pengujian	41
Tabel 4. 2 Skenario <i>Load testing</i> 2 <i>Mobile</i> Siswa	41
Tabel 4. 3 Skenario <i>Load testing</i> 3 <i>Mobile</i> Guru.....	41
Tabel 4. 4 Skenario <i>Gorilla testing</i> 3	42
Tabel 4. 5 <i>test case load testing website</i>	43
Tabel 4. 6 <i>test case load testing Mobile</i> siswa.....	44
Tabel 4. 7 <i>test case load testing Mobile</i> guru	44
Tabel 4. 8 <i>test case Gorilla testing</i>	46
Tabel 4. 9 Spesifikasi Lingkungan Pengujian	47

DAFTAR GAMBAR

	Halaman
Gambar 2.1 STLC (Ruliansyah dkk., 2023).....	16
Gambar 2.2 Contoh Pengumpulan Data (Siti hannaniyah Sucipto, 2025).....	18
Gambar 3. 1 Tahapan penelitian	31
Gambar 4. 1 <i>Thread Group</i> Pada pengujian <i>Website</i>	49
Gambar 4. 2 <i>HTTP Request</i> mengirim fungsi <i>login</i>	50
Gambar 4. 3 <i>HTTP Request</i> mengakses halaman <i>QR Code</i>	50
Gambar 4. 4 <i>HTTP Request</i> mengakses halaman absensi harian	51
Gambar 4. 5 <i>HTTP Request</i> mengakses halaman absensi mapel	52
Gambar 4. 6 <i>HTTP Request</i> mengirim data foto	52
Gambar 4. 7 <i>Thread Group</i> pada pengujian <i>Mobile</i> siswa.....	53
Gambar 4. 8 <i>HTTP Request</i> mengirim fungsi <i>login</i>	54
Gambar 4. 9 <i>HTTP Request</i> mengirim <i>scan QR Code</i>	55
Gambar 4. 10 <i>HTTP Request</i> Menampilkan data <i>perizinan</i>	56
Gambar 4. 11 <i>HTTP Request</i> mengirim perizinan	57
Gambar 4. 12 <i>HTTP Request</i> Mengakses data jadwal	57
Gambar 4. 13 <i>HTTP Request</i> Mengakses data pengumuman.....	58
Gambar 4. 14 <i>Thread Group</i> pada pengujian <i>Mobile</i> guru	59
Gambar 4. 15 <i>HTTP Request</i> halaman Absensi Kehadiran	60
Gambar 4. 16 <i>HTTP Request</i> halaman perizinan.....	61
Gambar 4. 17 <i>HTTP Request</i> Mengirim persetujuan perizinan.....	62
Gambar 4. 18 <i>HTTP Request</i> Mengakses halaman jadwal personal	62
Gambar 4. 19 <i>HTTP Request</i> halaman jadwal kelas	63
Gambar 4. 20 <i>Thread Group</i> pada pengujian <i>Gorilla testing</i>	64
Gambar 4. 21 <i>HTTP Request</i> Mengirim <i>Scan QR</i> (Mobile Siswa)	65
Gambar 4. 22 <i>HTTP Request</i> Mengirim <i>Update</i> Data Perizinan (<i>Mobile Guru</i>) .	66
Gambar 4. 23 <i>HTTP Request</i> Mengakses Halaman Absensi Kehadiran (<i>Website</i>)	67
Gambar 4. 24 <i>HTTP Request</i> Mengakses Halaman Absensi Mapel (<i>Website</i>).....	67

Gambar 4. 25	<i>Summary Report Load testing pada website</i>	69
Gambar 4. 26	<i>View Results in Table Load testing pada Website</i>	69
Gambar 4. 27	<i>Summary Reports Load testing pada mobile siswa 92 sampel</i>	71
Gambar 4. 28	<i>View Results in table Load testing pada mobile siswa 92 sampel</i> ..	72
Gambar 4. 29	<i>Summary Reports Load testing pada mobile siswa 297 sampel</i>	74
Gambar 4. 30	<i>View Results in table Load testing pada mobile siswa 297 sampel</i>	74
Gambar 4. 31	<i>Summary Reports Load testing pada mobile siswa 1034 sampel</i> ..	76
Gambar 4. 32	<i>View Results in table Load testing pada mobile siswa 1034 sampel</i>	77
Gambar 4. 33	<i>Summary Reports Load testing pada mobile guru 38 sampel</i>	80
Gambar 4. 34	<i>View Results in table Load testing pada mobile guru siswa 38 sampel</i>	80
Gambar 4. 35	<i>View Results Tree Load testing pada mobile guru 38 sampel</i>	80
Gambar 4. 36	<i>Summary Reports Load testing pada mobile guru 52 sampel</i>	83
Gambar 4. 37	<i>View Results in table Load testing pada mobile guru siswa 52 sampel</i>	84
Gambar 4. 38	<i>View Results Tree Load testing pada mobile guru 52 sampel</i>	84
Gambar 4. 39	<i>Summary Reports Load testing pada mobile guru 59 sampel</i>	87
Gambar 4. 40	<i>View Results in table Load testing pada guru siswa 59 sampel</i>	88
Gambar 4. 41	<i>Summary Reports gorilla testing pada scan qr Code absensi mapel (mobile siswa)</i>	91
Gambar 4. 42	<i>View Results in table gorilla testing pada scan qr Code absensi mapel (mobile siswa)</i>	92
Gambar 4. 43	<i>Summary Reports gorilla testing pada post perizinan (mobile siswa)</i>	92
Gambar 4. 44	<i>View Results in table gorilla testing pada post perizinan (mobile guru)</i>	93
Gambar 4. 45	<i>View Results Tree gorilla testing pada post perizinan (mobile guru)</i>	93
Gambar 4. 46	<i>Summary Reports gorilla testing pada Website</i>	94

Gambar 4. 47 <i>View Results in table gorilla testing</i> pada <i>website</i>	94
---	----

DAFTAR LAMPIRAN

	Halaman
Lampiran 1 Kegiatan Wawancara dengan admin	113
Lampiran 2 Kegiatan Wawancara dengan Siswa.....	114
Lampiran 3 Kegiatan Wawancara dengan Guru	115
Lampiran 4 Hasil Wawancara dengan Admin	116
Lampiran 5 Hasil Wawancara dengan Siswa	118
Lampiran 6 Hasil Wawancara dengan Guru.....	120

BAB 1 PENDAHULUAN

1.1. Latar Belakang

Absensi merupakan salah satu aspek penting dalam dunia pendidikan, terutama dalam memantau kehadiran siswa secara sistematis (Nababan dkk., 2022). Pencatatan kehadiran yang baik dapat membantu sekolah dalam menilai kedisiplinan siswa, mengevaluasi proses pembelajaran, serta menyusun laporan yang akurat. Namun, metode absensi tradisional, seperti pencatatan secara manual atau penggunaan daftar kehadiran, sering kali menghadapi masalah, seperti ketidakakuratan data, risiko kehilangan catatan, serta proses rekapitulasi yang memakan waktu lama (Azizah dkk., 2024). Oleh karena itu, berbagai penelitian terus dilakukan untuk meningkatkan efisiensi sistem pencatatan kehadiran, termasuk melalui pendekatan berbasis teknologi.

Salah satu faktor penting dalam pengembangan sistem yang berbasis teknologi adalah memastikan kinerja yang maksimal, terutama terkait dengan kecepatan dan kestabilan sistem ketika digunakan oleh banyak pengguna. Penelitian sebelumnya tentang pengujian beban telah dilakukan oleh (Tejaya dkk., 2023) dalam penelitian mereka yang berjudul "Pengujian situs web Invitees dengan metode pengujian beban menggunakan Apache JMeter". Hasil dari nilai *Error Rate* 0% tidak mencerminkan kestabilan sistem, karena performa menurun drastis saat jumlah pengguna bertambah, menunjukkan sistem belum siap menangani akses secara serentak. Penelitian lain dari (Raweyai & Widiyari, 2024) menemukan *website* menunjukkan performa baik secara keseluruhan, terdapat indikasi potensi penurunan performa jika trafik melebihi batas pengujian saat ini, serta kurangnya data teknis tambahan seperti *Throughput*, beban maksimum, dan pemakaian sumber daya, yang penting untuk analisis kapasitas dan perencanaan skalabilitas jangka panjang. Pada studi oleh (Abda'u dkk., 2024) terdapat skenario pengujian yang sederhana, skala beban yang terbatas, serta *Error Rate* yang tinggi tanpa analisis penyebab. Selain itu, tidak ada monitoring sumber daya sistem dan uji tidak dilakukan

berulang. Metrik yang digunakan juga terbatas, sehingga hasil pengujian kurang komprehensif dan belum mencerminkan kondisi nyata secara menyeluruh. Pada Penelitian (Ginasari dkk., 2021) kelemahan dari *stress testing* dalam studi ini terletak pada batas maksimum yang terlalu ringan di 75 sampel, skenario pengujian yang tidak cukup rumit, serta kurangnya analisis mengenai reaksi dan faktor penyebab kegagalan sistem ketika menghadapi tekanan yang tinggi. Pengujian yang dilakukan pada penelitian ini juga diperkuat dengan beberapa penelitian terdahulu yang menggunakan metode *load testing* untuk mengukur kemampuan sistem dalam menangani sejumlah pengguna secara bersamaan, serta metode *gorilla testing* yang difokuskan pada pengujian satu modul tertentu secara berulang untuk mengetahui tingkat kestabilan sistem pada kondisi beban tinggi (Desy Intan Permatasari, 2020).

Terutama di SMK Taruna Bakti Kertosono, sistem kehadiran telah diperbaharui menggunakan pendekatan AI dengan menggunakan pendekatan *face recognition* untuk mencatat kehadiran secara otomatis, siswa juga melakukan absensi digital pada aplikasi *mobile*. Admin bertanggung jawab untuk mengelola dan memverifikasi data absensi melalui situs *website*, sedangkan siswa dan guru akan dapat mengakses informasi kehadiran melalui aplikasi *Mobile* yang sedang dirancang. Akan tetapi, kinerja sistem menjadi elemen yang penting harus diteliti lebih mendalam karena dengan bertambahnya jumlah pengguna akan muncul kemungkinan masalah terkait stabilitas dan kecepatan sistem. Berdasarkan hasil wawancara menunjukkan bahwa pihak admin mengungkapkan kekhawatiran bahwa performa sistem dapat turun saat diakses secara bersamaan, terutama pada jam sibuk. Guru dan siswa menyampaikan harapan agar sistem ini dapat stabil ketika digunakan secara bersama sama, karena jumlah pengguna mencapai di atas 1000 pengguna. Masalah tersebut menunjukkan betapa pentingnya melakukan pengujian kinerja sistem sebelum diterapkan. Apabila tidak diambil tindakan, risiko seperti keterlambatan akses, serta gangguan saat banyak pengguna mengakses bersamaan dapat terjadi. Oleh sebab itu, pengujian kinerja sistem diperlukan untuk memastikan bahwa sistem absensi dapat beroperasi secara

maksimal dalam situasi penggunaan yang tinggi. Salah satu cara yang sering dipakai dalam uji performa adalah *load testing*, yaitu pendekatan yang digunakan untuk menilai kemampuan sistem dalam menangani banyak pengguna secara bersamaan (Dwinur Andrianto dan Fatrianto Suyatno, 2024). Teknik ini dianggap paling sesuai karna mampu menilai kinerja aplikasi saat menghadapi peningkatan beban akses drari sejumlah pengguna secara bertahap (Alga Saputra, 2023). Apache JMeter digunakan sebagai alat uji, Aplikasi ini dipilih karena merupakan salah satu *tools open-source* yang banyak digunakan dalam pengujian performa sistem, Apache Jmeter juga memiliki tampilan antarmuka yang ramah bahkan dapat digunakan oleh pengguna yang kurang berpengalaman (Ushakova dkk., 2022). sistem dapat direplikasi dalam kondisi tekan tinggi sehingga kemungkinan celah dapat ditemukan lebih awal. Pengujian ini penting untuk memastikan bahwa sistem absensi di SMK Taruna Bakti Kertosono dapat berfungsi dengan stabil, *respons if*, dan dapat diandalkan dalam mendukung proses administrasi kehadiran secara digital tanpa kendala teknis, serta memberikan pengalaman terbaik bagi semua pengguna, baik admin, guru, ataupun siswa.

Penentuan jumlah pengguna pada penelitian ini didasarkan pada hasil perhitungan tingkat kelonggaran *error (margin of error)* sebesar 1%, 5%, dan 10%. Semakin kecil nilai *margin of error* yang digunakan, maka semakin besar jumlah pengguna virtual yang disimulasikan dalam pengujian (Ridwan & Aji, 2021). Nilai *Ramp-up* period pada pengujian ditetapkan sama dengan jumlah *user* pada masing-masing skenario. Pengaturan ini dilakukan agar peningkatan jumlah pengguna virtual terjadi secara bertahap, sehingga *server* tidak menerima lonjakan *request* secara tiba-tiba (Malik Fajar Thaha & Tenriawaru, 2023)

Studi ini akan mengevaluasi kinerja sistem absensi berbasis *webite* dan *Mobile* di SMK Taruna Bakti Kertosono menggunakan metode *load testing* dengan Apache JMeter. Penelitian ini akan mengevaluasi kinerja sistem absensi berbasis website dan mobile di SMK Taruna Bakti Kertosono menggunakan metode load testing dengan Apache JMeter. Pengujian load testing dilakukan

dengan memberikan variasi beban pengguna secara bertahap, mulai dari jumlah pengguna yang rendah hingga tinggi, untuk mengevaluasi kecepatan respons, stabilitas sistem, serta kemampuan sistem dalam menangani peningkatan beban sebelum mengalami penurunan performa.

Gorilla testing dilakukan dengan menerapkan perulangan (looping) pada skenario pengujian yang sama untuk mengidentifikasi kemampuan sistem dalam mempertahankan performanya ketika fitur dijalankan secara berulang (Indrianto, 2023). Penerapan *gorilla testing* pada penelitian ini difokuskan pada fitur utama sistem absensi yang memiliki aktivitas tinggi, yaitu proses pengiriman hasil scan QR Code absensi mapel oleh siswa, pengambilan data perizinan siswa pada *Mobile* guru, serta pengambilan data rekap absensi mapel dan absensi kehadiran pada *website*. Menurut (Indrianto, 2023). Simulasi beban pengguna dapat lebih mendekati kondisi penggunaan sebenarnya di lingkungan sekolah. Nilai *loop count* pada *load testing* ditetapkan sebanyak 1 kali karena pengujian difokuskan pada pengukuran kemampuan sistem dalam menangani akses simultan. Nilai *loop count* pada *gorilla testing* ditetapkan sebanyak 100 kali untuk menguji kestabilan sistem terhadap pengulangan *request* secara intensif.

Parameter pengujian yang digunakan dalam penelitian ini meliputi *Load Time*, *Response Time*, *Latency*, *Throughput*, dan *Error Rate*. Kualitas performa sistem yang baik ditunjukkan oleh waktu muat (*Load Time*), waktu respons (*Response Time*), dan *Latency* yang rendah, *Throughput* yang tinggi, serta *Error Rate* yang rendah. Semakin cepat sistem merespons permintaan pengguna, semakin banyak *request* yang mampu diproses dalam satuan waktu, dan semakin kecil tingkat kegagalan yang terjadi, maka semakin baik kualitas performa sistem dalam melayani banyak pengguna secara bersamaan. Berdasarkan standar yang digunakan dalam penelitian ini, nilai *Load Time* yang baik berada di bawah 3 detik agar halaman *website* maupun aplikasi *mobile* dapat dimuat dengan cepat. Nilai *Response Time* yang optimal berada di sekitar 0,1 detik, sedangkan *Latency* yang baik berada di bawah 1 detik sehingga sistem mampu memberikan *respons awal* dengan cepat kepada pengguna. Nilai *Error Rate* yang kurang dari

5% menunjukkan bahwa sistem memiliki tingkat keandalan yang baik dalam memproses permintaan pengguna. Nilai *Throughput* yang semakin tinggi menunjukkan bahwa sistem mampu memproses lebih banyak *request* dalam satuan waktu, sehingga mencerminkan kapasitas pemrosesan sistem yang semakin baik (Harry Setiawan & Depandi Enda, 2025). Pengujian difokuskan pada fitur *login* dan *scan QR Code* pada aplikasi mobile serta fitur rekap absensi pada *website*. Berdasarkan hasil analisis tersebut, penelitian menghasilkan rekomendasi perbaikan yang dapat digunakan sebagai acuan dalam pengembangan dan optimalisasi sistem absensi di SMK Taruna Bakti Kertosono.

1.2. Rumusan Masalah

Adapun rumusan masalah yang dibahas pada penelitian ini adalah:

- a. Bagaimana menganalisis performa sistem absensi berbasis *website* dan *Mobile* di SMK Taruna Bakti Kertosono dalam menangani banyaknya pengguna secara bersamaan?
- b. Sejauh mana sistem dapat mempertahankan stabilitas dan kecepatan *respons* saat beban pengguna meningkat?

1.3. Tujuan

Tujuan dari penelitian ini adalah:

- a. Menganalisis performa sistem absensi berbasis *website* dan *Mobile* di SMK Taruna Bakti Kertosono menggunakan teknik *load testing* dengan *JMeter*.
- b. Sistem dapat mempertahankan stabilitas dan kecepatan respon dengan mengidentifikasi batasan maksimum beban yang dapat ditangani.

1.4. Manfaat

Manfaat dari penelitian ini adalah:

- a. Manfaat bagi Siswa
Meningkatkan kenyamanan siswa dalam memantau data kehadiran melalui aplikasi *Mobile* yang cepat dan andal secara langsung.
- b. Manfaat bagi Admin
Meningkatkan kemudahan admin untuk mengatur dan memantau data kehadiran, karena sistem yang sudah melalui pengujian performa dapat

tetap berjalan stabil dan optimal meskipun diakses oleh banyak pengguna secara bersamaan, khususnya pada waktu diakses oleh banyak pengguna.

c. Manfaat bagi Guru

Membantu guru untuk dapat memantau dan mengevaluasi kehadiran dengan lebih praktis dan cepat, karena sistem sudah stabil dan tidak menghambat pekerjaan mereka.

d. Manfaat bagi Peneliti

Menjadi referensi bagi penelitian selanjutnya yang berkaitan dengan evaluasi performa sistem menggunakan teknik *load testing*, khususnya dalam konteks sistem informasi pendidikan berbasis *website* dan *mobile*.

e. Manfaat bagi Sekolah

Menjadi pertimbangan untuk memperbaiki dan mengembangkan sistem IT sekolah agar lebih kuat dan dapat diandalkan, sekaligus untuk mendukung pelaksanaan absensi digital dengan sistem yang stabil.

f. Manfaat bagi Institusi (POLIJE)

Menunjukkan betapa pentingnya melakukan pengujian performa sistem sebelum diterapkan secara menyeluruh, sekaligus menjadi rujukan dalam pengembangan sistem absensi digital yang tangguh dan dapat diterapkan di berbagai institusi pendidikan lainnya.

1.5. Batasan Masalah

Batasan masalah pada penelitian ini, berupa:

- a. Objek untuk melakukan pengujian performa menggunakan *website* dan aplikasi *Mobile* absensi di SMK Taruna Bakti Kertosono di Kabupaten Nganjuk.
- b. Penelitian ini menggunakan pendekatan *load testing*.
- c. Hanya menggunakan *tools* Apache Jmeter

BAB 2 TINJAUAN PUSTAKA

2.1. *Software testing*

Software testing menurut adalah tahap yang dilakukan untuk menilai kinerja dari sebuah aplikasi atau sistem dengan maksud untuk mengidentifikasi kesalahan dan memastikan perangkat lunak berfungsi sesuai dengan kriteria yang telah ditetapkan (Hidayat dkk., 2025). *Software testing* dilakukan untuk menemukan kesalahan, kekurangan, atau kebutuhan yang belum dipenuhi dalam sistem atau perangkat lunak yang sedang dalam proses pengembangan (MARDIATI & SAPUTRA, 2025). Menurut (Zahra dkk., 2025) *Software testing* adalah tahap di mana aplikasi yang telah dibuat oleh tim pengembang dijalankan dan diuji, dengan tujuan menemukan isu atau kesalahan (seperti *bug*) yang mungkin ada. Tujuan utama dari pengujian ini adalah untuk memastikan bahwa aplikasi berfungsi sebagaimana mestinya dan memiliki standar kualitas yang baik. Pengujian sangat penting karena selama proses pembuatan perangkat lunak, seringkali ada elemen yang terlewat atau tidak sempurna, sehingga pengujian menjadi langkah penting untuk mengidentifikasi dan memperbaiki masalah tersebut.

Software testing berperan sebagai langkah akhir yang menentukan apakah sistem siap digunakan, serta menjadi sarana untuk menjamin bahwa aplikasi yang dihasilkan benar-benar sesuai dengan kebutuhan pengguna.

2.2. *Performance testing*

Performance testing menurut Mulana (2022) adalah suatu proses menjalankan aplikasi dengan mensimulasi *virtual user* menggunakan sebuah *tools* seolah olah aplikasi sedang berjalan dan di akses oleh pengguna sebenarnya untuk mengetahui sistem berjalan dengan baik dan memiliki kinerja yang baik. Pada penelitian (Hamidah dkk., 2025) ada beberapa jenis *performance test*, yaitu *stress test*, *load test*, Sebagai berikut:

a. *Stress testing*

Stress testing merupakan bentuk evaluasi lanjutan yang menekan sistem hingga mencapai atau melampaui kapasitas maksimumnya. Dengan memberikan beban yang sangat tinggi, pengujian ini memperlihatkan *respons* sistem saat menghadapi keterbatasan sumber daya, kepadatan koneksi database, atau tekanan berat pada komponen perangkat keras. Metode ini bertujuan untuk mengidentifikasi potensi titik kegagalan dan mengevaluasi kemampuan sistem dalam melakukan pemulihan (Hamidah dkk., 2025).

b. *Load testing*

Load testing adalah pengujian beban dilakukan dengan memberikan beban pengguna dalam berbagai tingkat pada sistem perangkat lunak guna menilai waktu *response*, pemakaian sumber daya, serta performa sistem secara keseluruhan dalam berbagai skenario penggunaan. Metode ini berguna untuk menemukan kendala kinerja, seperti lambatnya waktu *respons* atau kegagalan *server*, yang bisa terjadi ketika jumlah pengguna meningkat (Hamidah dkk., 2025).

Performance testing merupakan proses penting untuk mengevaluasi kinerja sistem dengan cara mensimulasikan kondisi penggunaan nyata. *Load testing* merupakan metode yang paling tepat untuk digunakan dalam penelitian ini. Hal ini karena *load testing* fokus pada pengujian performa sistem dalam kondisi beban pengguna yang bervariasi, mencerminkan situasi nyata saat sistem digunakan oleh banyak pengguna secara bersamaan. Metode ini memungkinkan peneliti untuk mengevaluasi *response* sistem, penggunaan sumber daya, dan mengidentifikasi potensi masalah performa yang relevan dengan kebutuhan pengujian sistem absensi. Sementara itu, *stress testing* lebih cocok digunakan untuk menguji ketahanan sistem dalam kondisi ekstrem yang melebihi kapasitas normal, yang kurang relevan dengan tujuan utama penelitian ini.

2.3. *Load testing*

Load testing menurut (Mutmainnah & Ihsan, 2024) adalah suatu jenis pengujian non-fungsional yang bertujuan untuk memahami bagaimana performa aplikasi ketika dihadapkan pada volume klik pengguna yang besar.

Pengujian ini memungkinkan kita untuk menilai seberapa baik aplikasi dapat menangani lalu lintas yang tinggi. Sedangkan menurut (Setiawan dan Enda, 2025) pengujian beban merupakan pemeriksaan kinerja evaluatif yang menilai *respons* sistem dalam keadaan dan kondisi beban yang beragam. Pelaksanaan pengujian beban dilakukan untuk memastikan kapasitas sistem untuk menangani tantangan spesifik atau beban yang dievaluasi, tergantung pada berbagai parameter situasional. Pengujian berfokus pada lima parameter yakni *Load Time*, *latency*, *Throughput*, *Error Rate* dan *Response Time*. perhitungan dengan rumus sebagai berikut:

a. *Load Time*

Load Time adalah durasi yang diperlukan agar seluruh elemen pada halaman *website* maupun aplikasi *mobile* dapat dimuat dan ditampilkan secara sempurna pada perangkat pengguna. Metrik ini digunakan untuk mengukur kecepatan sistem dalam menyajikan halaman kepada pengguna. Nilai *Load Time* di bawah 3 detik menunjukkan bahwa sistem memiliki waktu muat yang baik sehingga mampu memberikan pengalaman pengguna yang optimal.

b. *Latency*

Latency atau waktu tunda merupakan jeda waktu yang berlangsung antara permintaan dari klien (pengguna) dan tanggapan awal dari *server*. Waktu latensi yang optimal adalah di bawah 1 detik. Angka *Latency* dapat dihitung menggunakan rumus berikut:

$$Latency = \frac{\text{jumlah seluruh nilai latency}}{\text{jumlah sample}} \quad \text{-----} \quad 2.1$$

c. *Throughput*

Throughput mengindikasikan jumlah permintaan (*request*) yang sukses diproses oleh sistem dalam periode waktu tertentu. Indikator ini penting untuk menilai kemampuan pemrosesan sistem. Nilai *Throughput* dapat diperoleh melalui rumus:

$$Throughput = \frac{\text{jumlah request}}{\text{waktu rata - rata request}} \quad \text{-----} \quad 2.2$$

d. *Error Rate*

Error Rate merupakan perbandingan antara banyaknya permintaan (*request*) yang tidak berhasil dengan total jumlah permintaan yang dikirimkan selama proses pengujian. *Metrik* ini digunakan untuk mengukur tingkat kegagalan sistem dalam memproses permintaan. Berdasarkan standar yang digunakan dalam penelitian ini, nilai *Error Rate* kurang dari 5% menunjukkan bahwa sistem memiliki tingkat keandalan yang baik dalam menangani permintaan pengguna. Formula untuk menghitung *Error Rate* adalah:

$$Error Rate = \left(\frac{\text{jumlah sample yang gagal}}{\text{total sample}} \right) \times 100\% \quad \text{-----} 2.3$$

e. *Response Time*

Waktu *respons* mengukur durasi yang diperlukan sistem untuk menjawab permintaan dari pengguna. Waktu *response* yang optimal adalah sekitar 0,1 detik agar pengguna merasakan bahwa sistem memberikan tanggapan secepat mungkin. Perhitungan waktu *response* dapat dilakukan menggunakan rumus:

$$Response Time = \frac{\text{jumlah seluruh nilai sample time}}{\text{jumlah sample}} \quad \text{-----} 2.4$$

Seluruh parameter performa tersebut diuji dalam berbagai kondisi beban (*load condition*) secara bertahap, mulai dari beban ringan, sedang, hingga berat (Barus dkk., 2021). Pengaruh beban terhadap waktu muat halaman (*Load Time*), jeda waktu *response* awal (*latency*), kapasitas pemrosesan (*Throughput*), tingkat kesalahan (*Error Rate*), dan waktu *response* keseluruhan (*Response Time*) dapat dievaluasi secara objektif dan terukur. Penyesuaian kondisi beban menjadikan pengujian lebih nyata terhadap situasi yang mungkin terjadi saat sistem digunakan secara bersamaan oleh banyak pengguna. Selain jumlah pengguna (*virtual user*), hasil *load testing* juga dipengaruhi oleh beberapa faktor lain, seperti kualitas jaringan internet, spesifikasi perangkat keras (*hardware*), kapasitas server, konfigurasi aplikasi, serta kondisi lingkungan pengujian. Faktor-faktor tersebut dapat memengaruhi nilai *Load Time*, *Response Time*, *Latency*, *Throughput*, maupun *Error Rate* yang diperoleh selama proses pengujian. Oleh karena itu, hasil pengujian performa tidak hanya

mencerminkan kemampuan sistem dalam menangani beban pengguna, tetapi juga dipengaruhi oleh kondisi infrastruktur yang digunakan saat pengujian dilakukan.

2.4. Apache Jmeter

Apache JMeter menurut adalah sebuah perangkat lunak yang digunakan untuk melakukan pengujian kinerja atau pengujian beban pada aplikasi *website* dan *mobile*. Memungkinkan pengguna untuk menguji seberapa baik sebuah aplikasi atau situs *website* dapat menangani jumlah pengguna yang besar, seberapa cepat halaman *website* dapat dimuat, dan bagaimana sistem tersebut bertahan terhadap beban yang tinggi. JMeter memungkinkan simulasi berbagai skenario penggunaan untuk mengevaluasi kinerja suatu sistem secara luas (A. M. N. Hidayat dkk., 2024). Apache JMeter merupakan proyek sumber terbuka yang dimanfaatkan dengan memakai bahasa java yang dipakai untuk alat pengujian beban dan kinerja. Apache JMeter akan menciptakan beberapa simulasi pengguna yang akan mengakses *server* dengan banyak pengguna dan berbagai permintaan yang bervariasi bergantung pada pengaturan yang dilakukan di Apache JMeter (Ginasari dkk., 2021). Pada penelitian (Ushakova dkk., 2022) JMeter dapat dengan mudah digunakan untuk pengujian pada berbagai *platform*, seperti objek Java, servlet, server FTP, kueri database, HTTP, SOAP, skrip Perl. JMeter juga dapat dengan mudah digunakan karena antarmuka pengguna yang ramah, bahkan digunakan oleh pengguna yang kurang berpengalaman.

Apache JMeter menjadi salah satu alat yang efektif dan fleksibel dalam melakukan evaluasi performa sistem, khususnya dalam konteks pengujian beban terhadap aplikasi berbasis *webite* dan *mobile*.

2.5 Website

Website adalah sekumpulan halaman yang diakses melalui browser dan internet. *Website* terletak dalam sebuah domain atau subdomain, yang umumnya dikenal dengan sebutan WWW atau *World Wide Web*. *Website* dibuat menggunakan bahasa pemrograman HTML (*Hyper Text Markup Language*) dan dapat diakses melalui protokol di internet (Endra dkk., 2022). Menurut

(Noviana, 2022) *Website* merupakan sekumpulan halaman yang saling terhubung, yang berisi berbagai informasi dalam bentuk teks, gambar, animasi, audio, dan video. Halaman ini dapat diakses melalui koneksi internet dan ditujukan untuk individu, organisasi, serta perusahaan. Terdapat banyak dokumen yang tersimpan di *server* komputer (*web server*), di mana *server-server* ini tersebar di lima benua, termasuk Indonesia, dan saling terhubung melalui jaringan internet. Menurut (Marpaung dkk., 2022) *Website* adalah sebuah *platform* yang terdiri dari berbagai halaman. Masing-masing halaman terbuat dari sejumlah file yang mengandung kode program yang berhubungan satu sama lain, bertujuan untuk menyampaikan informasi dalam bentuk gambar, suara, teks, atau gabungan dari semuanya, baik yang bersifat tetap maupun berubah-ubah.

Website dapat disimpulkan sebagai media digital yang memuat dan menyampaikan informasi melalui jaringan internet, yang terdiri dari halaman-halaman saling terhubung, dan berfungsi sebagai sarana komunikasi, publikasi, serta layanan interaktif bagi berbagai kalangan.

Penggunaan *website* sistem absensi di SMK Taruna Bakti Kertosono yang berperan sebagai media pengelolaan data absensi oleh admin. *Website* ini menyediakan berbagai fitur, seperti *login* admin, pengelolaan data siswa dan guru, pembuatan *QR Code* absensi, pengelolaan kelas dan mata pelajaran, serta rekapitulasi dan pelaporan data kehadiran. *Website* tersebut terintegrasi dengan aplikasi *mobile* siswa dan guru sehingga seluruh data absensi yang diperoleh dari proses *face recognition* dan pemindaian *QR Code* dapat tersimpan, dikelola, dan ditampilkan secara terpusat. *Website* menjadi salah satu objek pengujian performa untuk mengetahui kemampuannya dalam menangani permintaan pengguna serta menjaga stabilitas layanan selama proses pengelolaan data absensi.

2.6 Aplikasi *Mobile*

Aplikasi *Mobile* merupakan perangkat lunak yang dirancang khusus untuk digunakan pada perangkat bergerak, seperti *smartphone* dan tablet. Pengguna dapat mengunduh dan menginstal aplikasi ini melalui toko aplikasi

yang disediakan oleh sistem operasi perangkat mereka, seperti *App Store* untuk iOS dan *Google Play Store* untuk Android (Indriyani dkk., 2025). Pengertian lain aplikasi *Mobile* merupakan perangkat lunak yang beroperasi di perangkat bergerak seperti smartphone atau tablet. *Aplikasi Mobile* juga disebut sebagai aplikasi yang dapat diinstal dan memiliki tujuan tertentu yang meningkatkan kemampuan dari perangkat *Mobile* itu sendiri. Aplikasi ini memiliki berbagai kegunaan, mulai dari meningkatkan produktivitas, menyediakan hiburan, mendukung pembelajaran, hingga memfasilitasi komunikasi. Biasanya, aplikasi *Mobile* diunduh dan dipasang melalui toko aplikasi (contohnya, *App Store* untuk iOS dan *google play store* untuk android) dan bisa diakses dengan mudah oleh pengguna melalui tampilan antarmuka yang ramah pengguna (Hermansyah dkk., 2024).

Aplikasi *Mobile* dapat disimpulkan sebagai solusi perangkat lunak praktis yang mendukung berbagai aktivitas pengguna dalam kehidupan sehari-hari melalui kemudahan akses, fleksibilitas penggunaan, serta fungsionalitas yang beragam sesuai kebutuhan.

Penggunaan aplikasi *mobile* merupakan bagian dari sistem absensi di SMK Taruna Bakti Kertosono yang digunakan oleh siswa dan guru. Aplikasi *mobile* siswa digunakan untuk melakukan *login*, melihat jadwal pelajaran, melakukan absensi melalui pemindaian *QR Code*, mengajukan perizinan, serta melihat informasi pengumuman. Aplikasi *mobile* guru digunakan untuk melakukan *login*, melihat jadwal mengajar, memantau data kehadiran siswa, mengelola perizinan, serta mengakses informasi yang berkaitan dengan proses pembelajaran. Aplikasi *mobile* tersebut terintegrasi dengan *website* sehingga seluruh data absensi dapat tersimpan dan dikelola secara terpusat. Penelitian ini aplikasi *mobile* menjadi objek pengujian performa untuk mengevaluasi kemampuan sistem dalam menangani beban pengguna serta mempertahankan stabilitas layanan saat diakses secara bersamaan.

2.7 Application Programming Interface (API)

API atau *web service* merupakan salah satu faktor yang mempengaruhi performansi suatu sistem. API (Application Programming Interface) merupakan

sekumpulan prosedur yang dapat digunakan oleh aplikasi lain untuk memenuhi kebutuhannya. API sendiri dapat diartikan sebagai antarmuka yang dibangun oleh pengembang sistem agar sebagian atau keseluruhan fungsi sistem dapat diakses secara terprogram. API dibangun dengan tujuan mampu mempersingkat proses pengembangan sebuah perangkat lunak agar fitur-fitur yang sama bisa digunakan oleh perangkat lunak lain dan pengembang tidak perlu membangun fitur yang sama lagi (Barus dkk., 2021). Menurut (Hadinata dan Stianingsih, 2024) *Application Programming Interface* (API) sendiri merupakan sebuah interface yang mampu untuk mengintegrasikan data dan menghubungkan sebuah aplikasi yang berjalan di banyak *platform* sehingga dapat saling terhubung satu sama lain. Selain dapat bertukar data di berbagai *platform* yang berbeda, API juga dapat mempercepat proses development dengan menyediakan function secara terpisah sehingga developer tidak perlu membuat fitur yang serupa.

2.8 Populasi dan *Sampel*

Populasi menurut (Sari dkk., 2025) merupakan totalitas objek yang akan atau dikehendaki untuk diteliti. Populasi ini sering dikenal juga Universal. Anggota populasi yang akan diteliti bisa berupa makhluk hidup maupun benda mati, di mana karakteristik yang dimilikinya dapat diukur atau diamati, sedangkan menurut (Susanto dkk., 2024) populasi dalam penelitian merujuk pada seluruh unit analisis yang memiliki karakteristik serupa atau memiliki hubungan relevan dengan topik yang diteliti. Memahami tingkat dan sifat suatu populasi sangatlah penting untuk memastikan representasi yang akurat dari kelompok tersebut dalam kajian. Populasi penelitian mencakup semua individu, benda, atau kejadian yang menjadi objek studi. Sangat penting untuk memahami dengan baik tentang populasi yang diterjuni. Pengambilan sampel dilakukan dengan menerapkan metode *Stratified Random Sampling*, yang merupakan cara pengambilan sampel secara acak berdasarkan sektor pekerjaan.

Pada penelien (Amin dkk., 2023) berbagai rumus yang umum digunakan untuk menentukan jumlah sampel dalam suatu penelitian, seperti rumus Cochran, Yamane, Krejcie & Morgan, serta Slovin. Pemilihan rumus

disesuaikan dengan karakteristik populasi, ketersediaan data, dan tingkat ketelitian yang dibutuhkan. Peneliti memilih untuk menggunakan rumus Slovin karena jumlah populasi diketahui secara pasti dan tidak tersedia data varians populasi, sehingga rumus ini dianggap paling sesuai dengan kondisi penelitian yang bersifat deskriptif dan menggunakan pendekatan survei.

$$n = \frac{N}{1 + N(a)^2} \quad \text{-----2.5}$$

Keterangan:

n = Ukuran Sampel

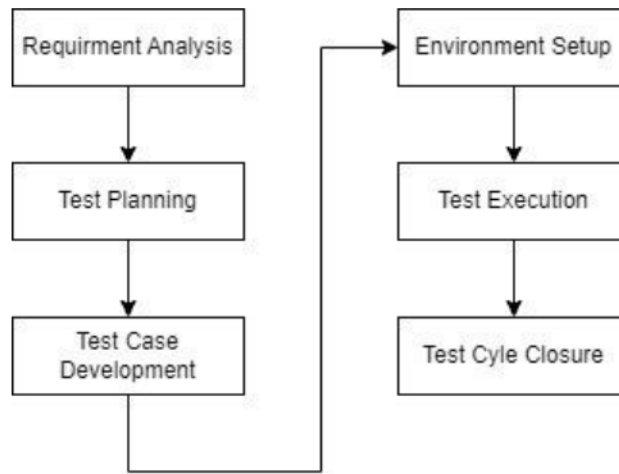
N = Ukuran Pupulasi

(a)² = Persen kelonggaran sampel

Pemilihan metode sampling dan rumus perhitungan jumlah sampel harus disesuaikan dengan karakteristik populasi serta tujuan penelitian, agar hasil yang diperoleh bersifat representatif dan akurat.

2.9 *Software Testing Life Cycle (STLC)*

Software Testing Life Cycle (STLC) adalah tahap-tahap proses pengujian yang dilaksanakan secara sistematis dan terencana. Berbagai kegiatan dilakukan pada proses STLC untuk meningkatkan kualitas produk. STLC mengacu pada tahap-tahap yang spesifik dalam proses pengujian untuk memastikan kualitas produk. Setiap tahap-tahap dari STLC memiliki *Entry Criteria* (ketentuan yang harus dimiliki sebelum pengujian dimulai), *Exit Criteria* (ketentuan yang harus diselesaikan sebelum menyelesaikan pengujian), *Activities dan Deliverable* (hasil yang didapatkan setelah menyelesaikan pengujian) yang terkait (Dhaifullah dkk., 2022). Siklus Hidup Pengujian Perangkat Lunak (*Software Testing Life Cycle/STLC*) adalah serangkaian tahapan sistematis yang dijalankan dalam proses pengujian perangkat lunak (Ruliansyah dkk., 2023). Tahapan-tahapan dalam STLC dapat diuraikan seperti yang terlihat pada gambar 2.1.



Gambar 2.1 STLC (Ruliansyah dkk., 2023)

Proses STLC terdiri dari *Requirement Analysis*, *Test Planning*, *Test case Development*, *Test Environment Setup*, *Test Execution* dan *Test Closure*.

a. *Requirement Analysis*

Meneliti detail perangkat lunak, modul, dan fitur yang akan diuji menurut kriteria yang disediakan oleh pihak yang bersangkutan. Tahap ini melibatkan analisis jumlah pengguna, waktu tanggapan, skenario yang sesuai, dan area pengujian, dilakukan saat *developer* melakukan *Product Backlog* untuk merancang dan mengidentifikasi kebutuhan sistem yang akan diuji (Shalsabilla dkk., 2024).

b. *Test Planning*

Test Planning adalah tahapan yang penting STLC. Tahap ini mencakup penentuan alat uji yang akan digunakan, penetapan tujuan pengujian, penyusunan skenario pengujian, serta identifikasi output yang diharapkan setelah proses pengujian selesai. Selain itu, pada tahap ini juga dilakukan estimasi waktu pelaksanaan, penentuan jumlah pengguna yang akan disimulasikan, dan pengaturan lingkungan pengujian. Seluruh proses ini biasanya dilakukan saat *developer* melakukan *Sprint Planning*, setelah tahap analisis kebutuhan (*requirement analysis*) diselesaikan, guna memastikan bahwa pengujian selaras dengan rencana pengembangan yang akan dilakukan

dalam *sprint* tersebut (Shalsabilla dkk., 2024). Contoh penulisan test plan dapat dilihat pada Tabel 2.1

Tabel 2.1 Contoh *Test Plan*

Modul	Feature
Manajemenen Notifikasi	1. Pengujian fungsionalitas tambah data notifikasi rutin.
	2. Pengujian fungsionalitas tambah data notifikasi khusus.
	3. Pengujian fungsionalitas ubah data notifikasi rutin.
	4. Pengujian fungsionalitas ubah data notifikasi khusus.
	5. Pengujian fungsionalitas hapus data notifikasi rutin.
	6. Pengujian fungsionalitas hapus data notifikasi khusus.

Sumber: (Ruliansyah dkk., 2023).

c. *Test case Development*

Tahap ini melibatkan perancangan skenario pengujian berdasarkan tingkat beban yang bervariasi, mulai dari beban ringan, sedang, hingga berat. Selain itu, dilakukan penyusunan *test case*, pembuatan data uji, serta penentuan target pengujian yang disesuaikan dengan *test case* yang telah dibuat. Pada tahapan ini, kegiatan pengujian dilakukan secara berjalan bersama saat *developer* sedang melakukan *Sprint Execution*, sehingga proses pengujian dapat menyesuaikan dengan fitur atau modul yang sedang dikembangkan dalam *sprint* berjalan (Shalsabilla dkk., 2024). Contoh penulisan test case dapat dilihat pada Tabel 2.2 dan Gambar 2.2.

Table 2.2 Contoh *Test Case*

TC ID	Nama Test Case	Expected Result
TC001	Tambah data notifikasi rutin	Dapat menambahkan data notifikasi rutin
TC002	Tambah data notifikasi khusus	Dapat menambahkan data notifikasi khusus
TC003	Ubah data notifikasi rutin	Dapat mengubah data notifikasi rutin
TC004	Ubah data notifikasi khusus	Dapat mengubah data notifikasi khusus
TC005	Hapus data notifikasi rutin	Dapat menghapus data notifikasi rutin
TC006	Hapus data notifikasi khusus	Dapat menghapus data notifikasi khusus

Sumber: (Ruliansyah dkk., 2023)

Nama	Link
Login	https://silao.situbondokab.go.id/index.php/Login/index
Registrasi	https://silao.situbondokab.go.id/index.php/Login/index2
Data KK	https://silao.situbondokab.go.id/index.php/BerandaController/data_kk
Data KTP	https://silao.situbondokab.go.id/index.php/BerandaController/data_ktp
Data Akta Kelahiran	https://silao.situbondokab.go.id/index.php/BerandaController/data_akta
Data Sinkronisasi	https://silao.situbondokab.go.id/index.php/BerandaController/data_sinkron
Data KIA	https://silao.situbondokab.go.id/index.php/BerandaController/data_kia
Data Akta Kematian	https://silao.situbondokab.go.id/index.php/BerandaController/data_kematian
Data Perpindahan	https://silao.situbondokab.go.id/index.php/BerandaController/data_pindah
Data Pendatang	https://silao.situbondokab.go.id/index.php/BerandaController/data_datang

Gambar 2.2 Contoh Pengumpulan Data (Siti hannaniyah Sucipto, 2025)

d. *Test Environment Setup*

Tahap ini bertujuan untuk mempersiapkan lingkungan pengujian yang sesuai agar proses pengujian perangkat lunak dapat berjalan dengan optimal. Persiapan mencakup penyediaan infrastruktur, perangkat keras (*hardware*),

perangkat lunak (*software*), serta pengaturan konfigurasi yang dibutuhkan untuk mendukung pelaksanaan pengujian. Seluruh aktivitas ini dilakukan saat *developer* sedang melakukan *Sprint Execution*, sehingga pengujian dapat langsung dijalankan ketika fitur atau modul yang dikembangkan sudah siap diuji (Shalsabilla dkk., 2024).

e. *Test Execution*

Pada tahap ini, skenario pengujian dijalankan pada perangkat lunak, dengan pelaksanaan yang mengacu pada skenario yang telah dirancang sebelumnya. Selama proses pengujian berlangsung, data performa sistem didokumentasikan secara menyeluruh, mencakup waktu respon (*Response Time*), kecepatan transfer data (*Throughput*), serta tingkat kesalahan (*Error Rate*). Pengujian ini dilakukan secara bersamaan saat *developer* sedang melakukan *Sprint Execution*, sehingga hasil pengujian dapat langsung memberikan umpan balik terhadap fitur yang tengah dikembangkan (Shalsabilla dkk., 2024).

f. *Test Closure*

Tahap ini berfokus pada penyusunan laporan hasil pengujian yang mencakup data performa sistem, temuan permasalahan, analisis penyebab, serta saran perbaikan bagi tim pengembang. Tujuan utama dari tahap *Test Closure* adalah memastikan bahwa seluruh aktivitas pengujian telah diselesaikan dengan baik dan bahwa perangkat lunak telah siap untuk dipublikasikan atau digunakan secara luas. Proses ini dilakukan saat *developer* melakukan *Sprint Review*, di mana hasil pengujian menjadi bagian dari evaluasi terhadap fitur-fitur yang telah dikembangkan selama *sprint*, serta menjadi bahan masukan untuk perbaikan pada *sprint* berikutnya (Shalsabilla dkk., 2024).

Siklus Hidup Pengujian Perangkat Lunak (*Software Testing Life Cycle*) merupakan proses terstruktur dan sistematis yang terdiri dari beberapa tahapan penting, mulai dari *Requirement Analysis* hingga *Test Closure*. Setiap tahapan memiliki peran krusial dalam menjamin kualitas perangkat lunak sebelum dirilis.

2.10 *Test Plan*

Test Plan adalah dokumen yang berisi definisi tujuan dan sasaran pengujian dalam lingkup iterasi (atau proyek), item-item yang *test plan* adalah dokumen yang berisi definisi tujuan dan sasaran pengujian dalam lingkup iterasi (atau proyek), item-item yang menjadi target pengujian, pendekatan yang akan diambil, sumber daya yang dibutuhkan dan point untuk diproduksi. Dengan kata lain *test plan* dapat disebut sebagai perencanaan atau *scenario* untuk melakukan *testing* yang akan dilakukan baik oleh *expert* atau *user* umum menjadi target pengujian, pendekatan yang akan diambil, sumber daya yang dibutuhkan dan *point* untuk diproduksi. *Test plan* dapat disebut sebagai perencanaan atau *scenario* untuk melakukan *testing* yang akan dilakukan baik oleh *expert* atau *user* umum (Prakasa, 2024). Menurut (Syifa dkk., 2025) *Test plan* merupakan sebuah list atau dokumen yang berisi tujuan serta target pengujian dalam ruang lingkup iterasi, item-item yang menjadi sasaran pengujian, pendekatan yang akan diambil, sumber daya yang dibutuhkan

Test plan merupakan komponen penting dalam proses pengujian perangkat lunak karena berfungsi sebagai panduan menyeluruh yang memastikan pengujian berjalan sistematis, terarah, dan sesuai dengan tujuan yang telah ditentukan. Perencanaan yang matang akan meningkatkan efektivitas serta ketepatan dalam menemukan dan menangani potensi permasalahan dalam sistem.

2.11 *Test case*

Test case menurut (Hasibuan dan Dirgahayu, 2021) adalah pengujian yang dilakukan berdasarkan beberapa masukan seperti kondisi dan hasil yang telah ditentukan sebelumnya. Hasil dari *test case* yang telah dilakukan akan dibandingkan dengan hasil yang telah ditentukan sebelumnya. Jika terdapat perbedaan di antara keduanya maka akan dilakukan perbaikan pada kode program. *Test case* menjadi titik awal untuk melakukan pengujian setelah memasukan nilai-nilai inputan ke sistem. Setelah itu, akan didapat hasil yang definitif dan meninggalkan sistem di beberapa titik akhir atau juga dikenal sebagai *Post condition* eksekusi. Menurut (Dhaifullah dkk., 2022b) *Test case*

adalah sekumpulan skenario yang disusun oleh QA agar sistem yang akan dites dapat memenuhi ketentuan, standar tertentu serta dapat berfungsi dengan baik.

Test case memiliki peran penting dalam proses pengujian karena menjadi acuan utama dalam mengevaluasi apakah sistem telah berjalan sesuai dengan kebutuhan yang ditetapkan. Melalui *test case*, penguji dapat secara sistematis menguji fungsi sistem, mendeteksi kesalahan, dan memastikan bahwa setiap fitur berfungsi sebagaimana mestinya sebelum sistem digunakan secara luas.

2.12 Gorilla testing

Gorilla testing digunakan untuk memastikan bahwa modul berfungsi dengan benar dan tidak terdapat bug. Modul dapat diuji lebih dari seratus kali dengan cara yang sama. *Gorilla testing* sangat berguna untuk menguji ketahanan (*robustness*) suatu aplikasi (Indrianto, 2023). Penggunaan metode *Gorilla testing*, dimana modul program diuji berulang kali bertujuan untuk memastikan ketahanan dan kinerja dari modul tersebut dapat bekerja sesuai dengan fungsinya (Wiyatnanto & Haris, 2021).

Gorilla testing dapat disimpulkan sebagai metode pengujian yang dilakukan secara berulang pada satu modul untuk memastikan bahwa modul tersebut bebas dari *bug*, berfungsi dengan benar, serta memiliki ketahanan (*robustness*) dan kinerja yang stabil sesuai dengan fungsinya.

2.13 Automation testing

Automation testing merupakan proses pengujian perangkat lunak yang dilakukan dengan membuat program atau test script untuk mensimulasikan langkah-langkah pengujian manual menggunakan bahasa pemrograman tertentu serta didukung oleh berbagai tools otomatisasi. Pada pengujian end-to-end, test case yang telah disusun terlebih dahulu dikonversi menjadi test script, kemudian dieksekusi menggunakan tools automation testing untuk memverifikasi fungsi sistem secara otomatis (Dhimas Wicaksono & Rani, 2022).

Automation testing adalah metode pengujian yang memanfaatkan script uji yang telah dibuat oleh penguji untuk dijalankan secara otomatis dalam membandingkan hasil aktual dengan hasil yang diharapkan. Pengujian ini

memiliki beberapa keunggulan, antara lain dapat dilakukan secara berulang, memberikan umpan balik yang lebih cepat, serta mengurangi kebutuhan interaksi manusia selama proses pengujian berlangsung. Dengan demikian, automation testing mampu meningkatkan frekuensi pengujian dan mendukung pelaksanaan regression testing secara lebih efisien (Prasetyo & Silfianti, 2023).

Sebelum *automation testing* banyak diterapkan, proses pengujian umumnya dilakukan secara manual. Namun, metode tersebut memerlukan waktu dan biaya yang lebih besar serta berpotensi menimbulkan kesalahan akibat faktor manusia. Oleh karena itu, automation testing dikembangkan sebagai solusi untuk meningkatkan efisiensi, konsistensi, dan efektivitas dalam proses pengujian perangkat lunak.

2.14 *State of the Art*

Penelitian terkait identifikasi objek telah banyak dilakukan dengan berbagai metode, memberikan referensi berharga bagi penelitian ini. Temuan dari penelitian terdahulu menjadi bagian penting dalam Analisis Performa Sistem Absensi Menggunakan Teknik *Load testing* dengan *JMeter* di SMK Taruna Bakti Kertosono, yang membantu dalam memahami dan mengembangkan metode pengujian performa sistem absensi.

Tabel 2.1 *State Of The Art*

Nama Peneliti & Tahun	Judul Penelitian	Metode	Kekurangan	Pembeda
(Ginasari dkk., 2021)	Pengujian <i>Stress</i> <i>Testing</i> API Sistem Pelayanan dengan	<i>Stress</i> <i>Testing</i>	Kelemahan dari <i>stress testing</i> dalam studi ini terletak pada batas maksimum yang terlalu ringan di 75 sampel, skenario	Menggunakan <i>load testing</i> dan <i>gorilla</i> <i>testing</i> pada sistem absensi berbasis <i>website</i> dan <i>mobile</i> .

Nama	Judul	Metode	Kekurangan	Pembeda
Peneliti &	Penelitian			
Tahun				
	<i>Apache</i>		pengujian yang tidak cukup rumit, serta kurangnya analisis mengenai reaksi dan faktor penyebab kegagalan sistem ketika menghadapi tekanan yang tinggi.	
	<i>JMeter</i>			
(Tejaya dkk., 2023)	Pengujian <i>Load Website Invitees</i> Menggunakan Metode <i>Load testing</i> Dengan Apache Jmeter	<i>Load testing</i>	Nilai <i>Error Rate</i> 0% tidak mencerminkan kestabilan sistem, karena performa menurun drastis saat jumlah pengguna bertambah, menunjukkan sistem belum siap menangani akses secara serentak.	Menambahkan <i>gorilla testing</i> sebagai pengujian lanjutan selain <i>load testing</i> .

Nama Peneliti & Tahun	Judul Penelitian	Metode	Kekurangan	Pembeda
(Raweyai dan Widiasari, 2024)	<i>Performa nce Testing Of Academic Website Using Load testing Method Supported By Apache Jmeter At XYZ University</i>	<i>Load testing</i>	<i>Website menunjukkan performa baik secara keseluruhan, terdapat indikasi potensi penurunan performa jika trafik melebihi batas pengujian saat ini, serta kurangnya data teknis tambahan seperti Throughput, beban maksimum, dan pemakaian sumber daya, yang penting untuk analisis kapasitas dan perencanaan skalabilitas jangka panjang.</i>	Menguji sistem absensi <i>website</i> dan <i>mobile</i> dengan parameter performa yang lebih lengkap, yaitu <i>load time, latency, respons time, throughput, error rate.</i>

Nama Peneliti & Tahun	Judul Penelitian	Metode	Kekurangan	Pembeda
(Abda'udkk., 2024)	Perbandingan Kinerja Antara Gatling dan Apache Jmeter pada Uji Beban RESTful API	<i>Load testing</i>	Skenario pengujian yang sederhana, beban yang terbatas, <i>Error Rate</i> yang tinggi tanpa analisis penyebab. Selain itu, tidak ada monitoring sumber daya sistem dan uji tidak dilakukan berulang. Metrik yang digunakan juga terbatas, sehingga hasil pengujian kurang komprehensif dan belum mencerminkan kondisi riil secara menyeluruh.	Mengombinasikan <i>load testing</i> dan <i>gorilla testing</i> untuk evaluasi performa sistem.
(Arief Anastiar Suharsono, 2024)	Rancang Bangun Analisis Performa Website	<i>Stress Testing</i>	<i>Throughput</i> sistem menurun saat jumlah pengguna meningkat. Ini	Berfokus pada sistem absensi berbasis <i>website</i> dan <i>mobile</i> dengan

Nama Peneliti & Tahun	Judul Penelitian	Metode	Kekurangan	Pembeda
	Dengan Pendekatan n Automation n Stress Testing (Studi Kasus SMA Negeri 1 Jember		terlihat dari penurunan persentase keberhasilan sistem yang drastis, bahkan di bawah 15% ketika ada 1000 pengguna.	kombinasi dua metode pengujian.

Berdasarkan Tabel *State of the Art* 2.1, penelitian yang dilakukan oleh Tejaya dkk. (2023) bertujuan untuk mengevaluasi performa *website Invitees* menggunakan metode *load testing* dengan Apache JMeter. Pengujian dilakukan dengan memberikan beban pengguna secara bertahap untuk mengetahui kemampuan sistem dalam menangani akses secara bersamaan. Hasil penelitian menunjukkan bahwa sistem memperoleh *Error Rate* sebesar 0%. Peningkatan jumlah pengguna menyebabkan performa sistem menurun secara signifikan sehingga menunjukkan bahwa nilai *Error Rate* yang rendah belum tentu mencerminkan kestabilan sistem ketika menerima beban yang lebih tinggi.

Penelitian yang dilakukan oleh Ginasari dkk. (2021) menggunakan metode *stress testing* dengan Apache JMeter untuk menguji performa API sistem pelayanan. Pengujian dilakukan dengan memberikan tekanan kepada sistem hingga batas tertentu untuk mengetahui kemampuan sistem dalam menghadapi peningkatan beban. Hasil penelitian menunjukkan bahwa sistem masih mampu memberikan *respons* pada skenario yang diuji. Keterbatasan penelitian terlihat pada batas maksimum pengujian yang hanya mencapai 75

sampel, skenario pengujian yang belum cukup kompleks, serta belum adanya analisis mengenai penyebab kegagalan sistem ketika menerima tekanan yang lebih tinggi.

Penelitian oleh Abda'u dkk. (2024) membandingkan performa Gatling dan Apache JMeter dalam melakukan *load testing* pada *RESTful API*. Pengujian dilakukan untuk mengetahui efektivitas kedua alat dalam mengevaluasi performa sistem. Hasil penelitian menunjukkan bahwa kedua alat mampu melakukan pengujian beban sesuai skenario yang telah ditentukan. Keterbatasan penelitian meliputi skenario pengujian yang masih sederhana, jumlah beban pengguna yang terbatas, nilai *Error Rate* yang tinggi tanpa analisis penyebab, belum adanya pemantauan penggunaan sumber daya sistem, serta pengujian yang tidak dilakukan secara berulang.

Penelitian yang dilakukan oleh Raweyai dan Widiyari (2024) bertujuan mengevaluasi performa *website* akademik menggunakan metode *load testing* dengan Apache JMeter. Pengujian dilakukan untuk mengetahui kemampuan *website* dalam melayani permintaan pengguna pada kondisi beban tertentu. Hasil penelitian menunjukkan bahwa *website* memiliki performa yang baik selama proses pengujian. Penelitian tersebut masih menunjukkan potensi penurunan performa apabila jumlah *trafik* melebihi batas pengujian serta belum menyajikan informasi mengenai *throughput*, kapasitas beban maksimum, dan penggunaan sumber daya sistem sehingga analisis kapasitas serta skalabilitas sistem belum dilakukan secara komprehensif.

Penelitian yang dilakukan oleh Arief Anastiari Suharsono (2024) bertujuan menganalisis performa *website* menggunakan pendekatan *automation stress testing*. Pengujian dilakukan untuk mengetahui kemampuan sistem dalam menangani peningkatan jumlah pengguna secara bertahap. Hasil penelitian menunjukkan bahwa nilai *throughput* sistem menurun seiring bertambahnya jumlah pengguna. Penurunan performa terlihat dari persentase keberhasilan sistem yang menurun secara signifikan hingga berada di bawah 15% pada beban 1.000 pengguna. Kondisi tersebut menunjukkan bahwa peningkatan beban

pengguna memberikan pengaruh terhadap kemampuan sistem dalam mempertahankan performanya.

Berdasarkan hasil analisis penelitian terdahulu, metode *load testing* menggunakan Apache JMeter banyak digunakan untuk mengevaluasi performa sistem karena mampu mensimulasikan akses dari banyak pengguna secara bersamaan serta menghasilkan parameter pengujian, seperti *response time*, *throughput*, dan *error rate*. Penelitian terdahulu masih memiliki beberapa keterbatasan, antara lain variasi beban pengguna yang terbatas, skenario pengujian yang belum kompleks, kurangnya analisis penyebab kegagalan sistem, serta belum dikombinasikannya metode lain untuk mengevaluasi ketahanan sistem. Penelitian ini menggunakan metode load testing untuk mengevaluasi performa sistem berdasarkan variasi beban pengguna dan gorilla testing untuk mengevaluasi ketahanan sistem melalui perulangan (*looping*) pada skenario pengujian yang sama. Kombinasi kedua metode tersebut menghasilkan evaluasi performa yang lebih komprehensif pada sistem absensi berbasis website dan mobile di SMK Taruna Bakti Kertosono.

BAB 3 METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Pelaksanaan

3.3.1. Tempat Penelitian

Penelitian dengan judul “Analisis Performa Sistem Absensi Menggunakan Teknik *Load testing* dengan Jmeter di SMK Taruna Bakti Kertosono”. Penelitian ini dilakukan di SMK Taruna Bakti Kertosono Ds. Tembarak Kec. Kertosono Kab. Nganjuk, dan dikerjakan di Politeknik Negeri Jember (Polije) PSDKU Kampus 3 Nganjuk.

3.3.2. Waktu Penelitian

Penelitian dengan judul “Analisis Performa Sistem Absensi Menggunakan Teknik *Load testing* dengan Jmeter di SMK Taruna Bakti Kertosono”. dilakukan selama rentan waktu 5 (lima) bulan, dimulai dari bulan Februari 2026 hingga Juli 2026.

3.2 Alat dan Bahan

Alat dan bahan adalah komponen krusial yang perlu diperhatikan dalam melaksanakan penelitian. Berikut adalah rincian mengenai alat dan bahan yang digunakan dalam penelitian berjudul " Analisis Performa Sistem Absensi Menggunakan Teknik *Load testing* dengan Jmeter di SMK Taruna Bakti Kertosono ".

3.2.1. Alat

Alat yang digunakan untuk melakukan kegiatan penelitian ini terdiri dari:

a. *Hardware:*

- 1) *Processor Intel Celeron inside*
- 2) *Memory 3GB*
- 3) *Operating System Windows 8 64-bit*

b. *Software:*

- 1) Apache JMeter Versi 5.4.3

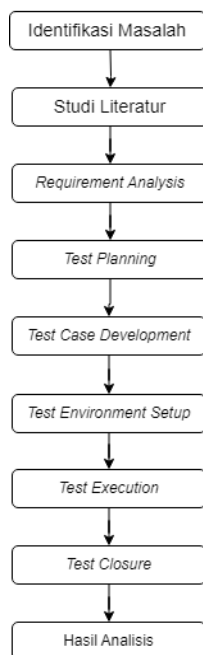
3.2.2. Bahan

Bahan yang digunakan untuk melakukan penelitian ini sebagai berikut :

- a. Hasil wawancara dengan admin, guru, dan siswa SMK Taruna Bakti Kertosono mengenai penggunaan sistem absensi dan kebutuhan pengujian performa.
- b. *Website* absensi SMK Taruna Bakti Kertosono sebagai objek pengujian.
- c. Aplikasi *Mobile* absensi SMK Taruna Bakti Kertosono sebagai objek pengujian.
- d. Dokumen perencanaan pengujian (*Test Plan, Test case, Test Report*).
- e. Dokumen sistem dan dokumen *Software Requirement Specification* (SRS) dan *Software Design Documentation website* dan aplikasi *mobile* sebagai acuan untuk menyusun *test case*.

3.3 Tahapan Penelitian

Tahapan penelitian ini disusun secara sistematis agar prosesnya berjalan terarah dan menghasilkan *output* yang sesuai dengan tujuan. Setiap langkah dilakukan secara runtut dan saling mendukung. Masing-masing tahapan berperan penting dalam membentuk alur berpikir yang logis dan mendukung keakuratan analisis. Gambar 3.1 berikut menyajikan alur tahapan penelitian yang digunakan dalam penyusunan skripsi ini.



Gambar 3. 1 Tahapan penelitian

3.3.1. Identifikasi Masalah

Tahap ini dilakukan untuk mengidentifikasi potensi permasalahan yang dapat muncul pada sistem absensi berbasis *website* dan *Mobile* yang sedang dalam tahap perancangan. Fokus utama adalah pada aspek performa sistem ketika nantinya diakses oleh banyak pengguna secara bersamaan. Pengujian performa dirancang sejak awal untuk memastikan bahwa sistem yang dikembangkan nantinya dapat berjalan dengan stabil, *respons if*, dan mampu menangani beban tinggi saat digunakan secara nyata.

3.3.2. Studi Literatur

Tahapan awal yang dilakukan dengan mengumpulkan berbagai referensi dari jurnal ilmiah, buku, artikel, maupun laporan penelitian terdahulu yang memiliki relevansi dengan topik penelitian. Tujuan dari studi literatur ini adalah untuk memperoleh pemahaman yang komprehensif mengenai konsep dasar sistem absensi, teknologi yang digunakan dalam implementasinya, serta teknik *load testing* sebagai metode evaluasi performa sistem. Studi ini menjadi landasan teoritis yang mendukung perumusan masalah dan penyusunan metodologi penelitian.

3.3.3. *Requirement Analysis*

Pada tahap ini, mengidentifikasi terhadap modul dan fitur yang akan diuji dari sistem absensi berbasis *website* dan *Mobile* di SMK Taruna Bakti Kertosono. Analisis dilakukan untuk mengetahui estimasi jumlah pengguna yang akan disimulasikan, waktu *respons* yang diharapkan, serta skenario pengujian yang relevan. Lingkungan pengujian juga ditentukan sesuai dengan kebutuhan simulasi penggunaan sistem secara nyata.

3.3.4. *Test Planning*

Tahap ini berisi perencanaan pada alat bantu pengujian yang digunakan, yaitu Apache JMeter. Merancang tujuan pengujian, jenis skenario yang akan dijalankan (seperti beban rendah, sedang, tinggi, dan sangat tinggi), jumlah pengguna virtual, waktu pelaksanaan, serta *output* yang diharapkan. Perencanaan ini mengacu pada hasil analisis kebutuhan sebelumnya. Hasil dari tahap ini adalah dokumen *test plan* yang menjadi acuan dalam pelaksanaan pengujian.

3.3.5. *Test case Development*

Pada tahap ini, dibuat skenario beban yang bervariasi, mulai dari kondisi penggunaan ringan hingga beban tinggi. Merancang *test case* dan data uji yang akan digunakan untuk menilai performa sistem pada berbagai variasi beban. Target pengujian ditentukan untuk masing-masing skenario, misalnya respon maksimal, stabilitas sistem, dan tingkat kesalahan. Hasil dari tahap ini adalah *test case* yang akan digunakan dalam proses eksekusi pengujian.

3.3.6. *Test Environment Setup*

Tahap ini penyiapan infrastruktur pengujian, seperti perangkat keras, sistem operasi, server lokal/hosting sistem absensi, dan konfigurasi perangkat lunak pendukung untuk menjalankan Apache JMeter. Lingkungan ini disusun sedemikian rupa agar mampu merepresentasikan kondisi sebenarnya saat sistem diakses oleh banyak pengguna.

3.3.7. Test Execution

Pengujian yang dilakukan untuk menjalankan skenario beban menggunakan Apache JMeter sesuai dengan *test case* yang telah disusun. Pengujian dilakukan secara bertahap dengan peningkatan beban secara bertahap. Selain itu, dilakukan juga pengujian lanjutan menggunakan pendekatan *Gorilla testing*, yaitu pengujian intensif pada modul-modul penting seperti input kehadiran dan rekap data. Selama proses ini, dikumpulkan metrik performa seperti *Response Time*, *Throughput* dan *Error Rate*. Selama proses pengujian, dikumpulkan metrik performa seperti *Response Time*, *Throughput*, dan *Error Rate*. Hasil dari pengujian ini dicatat dan didokumentasikan dalam *test case* sebagai bahan evaluasi performa sistem.

3.3.8. Test Closure

Tahap ini bertujuan untuk menyelesaikan seluruh aktivitas pengujian, mendokumentasikan temuan dan hasil secara lengkap, serta mengevaluasi proses pengujian. Menyusun laporan pengujian yang mencakup rekapitulasi data, kendala yang ditemukan. Hasil dari tahap ini adalah dokumen *test report* yang menjadi acuan untuk perbaikan dan pengambilan keputusan selanjutnya.

3.3.9. Hasil Analisis

Hasil dari semua pengujian selanjutnya dianalisis untuk mengevaluasi performa sistem absensi. Analisis meliputi interpretasi terhadap *Response Time*, kestabilan sistem, dan kapasitas maksimum pengguna yang bisa ditangani. Kesimpulan dapat ditarik berdasarkan hasil pengujian untuk mengidentifikasi kekuatan serta kelemahan sistem.

3.4 Jadwal Pelaksanaan Penelitian

Berikut jadwal penelitian yang dilaksanakan selama 5 bulan pada Tabel

3.1

Tabel 3. 1 Jadwal Penelitian

Tahapan Kegiatan	Bulan ke-				
	1	2	3	4	5
Idenifikasi Masalah					

Studi Literatur

*Requirement
Analysis*

Test Planning

*Test case
Development*

*Test
Environment
Setup*

Test Execution

Test Closure

Hasil Analisis

BAB 4 HASIL DAN PEMBAHASAN

4.1. Identifikasi Masalah

Berdasarkan dokumen *Software Requirement Specification* (SRS), dokumen *Software Desain Documentation* (SDD) dan hasil observasi di SMK Taruna Bakti Kertosono, sistem absensi yang digunakan saat ini telah dikembangkan menjadi sistem digital berbasis *website* dan *Mobile* dengan memanfaatkan teknologi *Face Recognition* dan *QR Code*. Sistem ini dirancang untuk menggantikan metode absensi manual yang sebelumnya menggunakan kertas, sehingga dapat meningkatkan keakuratan dan efisiensi dalam pencatatan kehadiran siswa.

Sistem Presensiku memiliki beberapa karakteristik yang berpotensi menimbulkan permasalahan pada sisi performa. Sistem ini digunakan oleh tiga jenis pengguna, yaitu admin, guru dan siswa, dengan aktivitas yang cukup intens, terutama pada jam masuk sekolah dan saat pergantian mata pelajaran. Pada kondisi tersebut, banyak siswa melakukan proses absensi secara bersamaan melalui aplikasi *mobile*, baik menggunakan *Face Recognition* maupun pemindaian *QR Code*. Di sisi lain, admin juga mengakses *website* untuk memantau dan mengelola data absensi.

Selain itu, sistem ini bergantung pada koneksi internet dan server sebagai pusat pengolahan data. Proses pengenalan wajah, pemindaian *QR Code*, serta penyimpanan data absensi dilakukan secara terpusat pada server. Hal ini menyebabkan sistem berpotensi mengalami penurunan performa ketika menerima banyak permintaan (*request*) dalam waktu yang bersamaan, seperti meningkatnya waktu respon, keterlambatan proses absensi, atau bahkan terjadinya kegagalan pengiriman data.

Berdasarkan ruang lingkup dan batasan sistem yang telah ditetapkan dalam dokumen *Software Requirement Specification* (SRS) yang dapat diakses

melalui <https://bit.ly/DokumenSrs> dan dokumen *Software Design Document* (SDD) yang dapat diakses melalui <https://intip.in/DokumenSDD> sistem Presensiku hanya difokuskan pada proses absensi siswa dan tidak mencakup fitur lain seperti pengelolaan nilai atau keuangan. Fokus utama sistem pada proses absensi menjadikan keandalan dan kecepatan sistem sebagai aspek yang sangat penting, sehingga proses kehadiran siswa dapat berjalan lancar.

Sistem Presensiku belum diketahui secara pasti bagaimana performanya ketika digunakan secara bersamaan oleh banyak pengguna. Oleh karena itu, diperlukan pengujian performa untuk mengetahui kemampuan sistem dalam menangani beban pengguna, khususnya pada kondisi jam sibuk. Pengujian ini dapat memberikan gambaran apakah sistem sudah mampu melayani permintaan pengguna dengan baik atau masih terdapat potensi masalah pada waktu respon, tingkat kesalahan, maupun kestabilan sistem.

Kondisi tersebut menunjukkan bahwa masalah utama yang dapat diidentifikasi adalah belum adanya evaluasi terhadap performa sistem absensi Presensiku ketika diakses oleh banyak pengguna secara bersamaan. Hal ini menjadi dasar perlunya dilakukan analisis performa sistem menggunakan teknik *load testing* dengan bantuan Apache JMeter agar dapat diketahui tingkat keandalan sistem dalam kondisi beban tertentu.

4.2. Studi Literatur

Studi literatur dalam penelitian ini dilakukan untuk memperoleh landasan teori yang mendukung pembahasan mengenai sistem absensi digital dan pengujian performa sistem. Sumber literatur yang digunakan meliputi buku, jurnal ilmiah, artikel penelitian, serta dokumentasi resmi yang berkaitan dengan teknologi absensi, pengujian performa, dan penggunaan Apache JMeter sebagai alat bantu pengujian.

Melalui studi literatur, peneliti mempelajari konsep dasar terkait performa sistem, seperti waktu respon, *latency*, *Throughput*, dan tingkat kesalahan (*Error Rate*). Selain itu, literatur juga digunakan untuk memahami tahapan dalam melakukan *load testing*, mulai dari penyusunan skenario

pengujian, penentuan jumlah pengguna, hingga cara menganalisis hasil pengujian yang dihasilkan oleh JMeter.

Kajian pustaka ini juga berperan dalam membantu peneliti menentukan metode pengujian yang sesuai dengan karakteristik sistem absensi Presensiku yang digunakan di SMK Taruna Bakti Kertosono. Studi literatur memberikan dasar teori yang jelas dalam pelaksanaan pengujian, sehingga pengujian tidak hanya dilakukan berdasarkan perkiraan semata.

Pembahasan secara lebih detail mengenai teori-teori pendukung, konsep *load testing*, parameter pengujian performa, serta penggunaan Apache JMeter dijelaskan secara lengkap pada Bab II (Studi Literatur). Oleh karena itu, pada bagian ini hanya disampaikan gambaran umum sebagai pengantar sebelum masuk ke tahap pengujian dan pembahasan hasil penelitian pada bab berikutnya.

4.3. Requirement Analysis

Tahap analisis kebutuhan pengujian dilakukan berdasarkan hasil observasi dan wawancara yang telah dilakukan di SMK Taruna Bakti Kertosono. Observasi dilakukan untuk mengetahui alur proses absensi yang akan diterapkan menggunakan sistem Presensiku yang saat ini masih dalam tahap pengembangan. Selain itu, wawancara dilakukan dengan pihak admin dan guru untuk menggali kebutuhan serta gambaran penggunaan sistem absensi yang akan diterapkan di lingkungan sekolah.

Berdasarkan hasil observasi dan wawancara, dapat ditentukan kebutuhan pengujian performa sistem. *Load testing* mencakup seluruh modul utama yang digunakan agar dapat mensimulasikan beban pengguna yang mendekati kondisi nyata di sekolah. Cakupan pengujian meliputi pada aplikasi *website*, fungsi *login*, membaca *QR Code*, mengakses halaman Absensi Harian, mengakses halaman Absensi Mata Pelajaran, serta mengirim data foto. *Mobile* siswa, fungsi *login*, mengirim scan *QR Code*, mengakses halaman perizinan, mengirim perizinan, mengakses data jadwal, dan mengakses data pengumuman. *Mobile* guru, fungsi *login*, mengakses halaman Absensi Mata Pelajaran, mengirim pengumuman, mengakses halaman Absensi Kehadiran, mengakses halaman

perizinan, mengirim persetujuan perizinan, mengakses halaman jadwal personal, dan mengakses halaman jadwal kelas. Pengujian ini bertujuan memperoleh gambaran performa sistem saat digunakan secara bersamaan oleh banyak pengguna.

Gorilla testing difokuskan pada empat modul utama yang dianggap krusial untuk kestabilan sistem, yaitu scan QR di aplikasi *mobile*, pengiriman keputusan perizinan siswa di aplikasi *mobile*, melihat riwayat absensi kehadiran di *website*, dan melihat riwayat absensi mata pelajaran di *website*. Pengujian ini dilakukan untuk memastikan sistem tetap stabil saat digunakan secara intensif dalam waktu singkat pada modul-modul kritis tersebut.

Hasil wawancara menunjukkan bahwa pada saat sistem ini nanti digunakan, proses absensi akan dilakukan secara bersamaan, terutama pada jam masuk sekolah dan saat pergantian mata pelajaran. Sistem mampu menangani banyak permintaan dalam waktu yang relatif singkat tanpa mengalami penurunan performa yang signifikan. Pihak sekolah juga menyampaikan agar proses absensi dapat berjalan dengan cepat dan tidak mengganggu kegiatan belajar mengajar.

Berdasarkan hasil observasi dan wawancara tersebut, dapat ditentukan kebutuhan pengujian performa sistem. Pengujian akan difokuskan pada modul-modul utama yang akan digunakan, yaitu modul *login*, modul absensi mata pelajaran menggunakan *QR Code*, serta modul perizinan siswa dan akses data absensi oleh guru. Pengujian juga perlu mensimulasikan jumlah pengguna yang mendekati kondisi nyata di sekolah agar hasil pengujian dapat menggambarkan performa sistem pada saat digunakan secara bersamaan.

Analisis ini digunakan sebagai dasar dalam menentukan skenario pengujian, jumlah pengguna yang akan disimulasikan, serta parameter performa yang akan diukur, seperti *Response Time*, *latency*, *Throughput*, dan *Error Rate*. Lingkungan pengujian disesuaikan dengan spesifikasi sistem yang sedang dikembangkan agar hasil pengujian yang diperoleh dapat mewakili kondisi sistem pada saat diimplementasikan nanti.

Proses pengujian performa yang dilakukan dapat memberikan gambaran awal mengenai kemampuan sistem absensi Presensiku dalam menangani beban pengguna sebelum sistem benar-benar digunakan di lingkungan sekolah.

4.4. *Test Planning*

Tahap *Test Planning* dilakukan setelah kebutuhan pengujian berhasil diidentifikasi. Tahap ini disusun rencana pengujian sebagai acuan pelaksanaan pengujian performa sistem absensi Presensiku. Rencana pengujian mencakup penentuan tujuan pengujian, objek yang akan diuji, metode pengujian, lingkungan pengujian, jumlah virtual user, parameter pengujian, serta kriteria keberhasilan yang digunakan dalam penelitian.

Tujuan pengujian adalah menganalisis kemampuan sistem absensi berbasis *website* dan aplikasi *mobile* dalam menangani beban pengguna secara bersamaan serta mengevaluasi ketahanan sistem ketika fitur dijalankan secara berulang. Objek pengujian terdiri atas *website* admin, aplikasi *mobile* siswa, dan aplikasi *mobile* guru yang merupakan bagian dari sistem Presensiku di SMK Taruna Bakti Kertosono.

Penentuan jumlah virtual *user* pada pengujian *load testing* didasarkan pada jumlah pengguna sistem yang dihitung menggunakan rumus Slovin sebagaimana dijelaskan pada rumus 2.5. Hasil perhitungan tersebut digunakan sebagai dasar dalam penyusunan skenario pengujian sehingga jumlah pengguna yang disimulasikan dapat mewakili kondisi penggunaan sistem di SMK Taruna Bakti Kertosono. Berdasarkan hasil tersebut, pengujian pada aplikasi *mobile* siswa menggunakan 92, 297, dan 1034 virtual *user*, sedangkan aplikasi *mobile* guru menggunakan 38, 52, dan 59 virtual *user*. Pengujian pada *website* dilakukan menggunakan 1 virtual *user* pada setiap fitur dengan tetap mempertimbangkan beban akses yang berasal dari aplikasi *mobile* sehingga kondisi pengujian mendekati penggunaan sistem yang sebenarnya.

Metode pengujian yang digunakan meliputi *load testing* dan *gorilla testing* dengan bantuan Apache JMeter sebagai *tools* pengujian. *Load testing* dilakukan untuk mengetahui performa sistem ketika menerima beban pengguna yang meningkat secara bertahap. *Gorilla testing* dilakukan dengan memberikan

perulangan (*looping*) sebanyak 100 kali pada fitur-fitur yang dianggap kritis untuk menguji ketahanan sistem terhadap eksekusi secara terus-menerus. Pelaksanaan pengujian juga menetapkan *Suspension Criteria* dan *Resumption Criteria* sebagai bagian dari dokumen *Test Plan*. Pengujian akan dihentikan sementara (*suspension*) apabila ditemukan kondisi yang menyebabkan proses pengujian tidak dapat dilaksanakan sesuai skenario, seperti server sistem Presensiku tidak dapat diakses, koneksi internet mengalami gangguan, atau terjadi kesalahan konfigurasi Apache JMeter yang memengaruhi validitas hasil pengujian. Pengujian akan dilanjutkan kembali (*resumption*) setelah penyebab penghentian berhasil diperbaiki, server kembali beroperasi secara normal, koneksi internet stabil, serta konfigurasi Apache JMeter telah sesuai dengan skenario pengujian sehingga seluruh tahapan pengujian dapat dilaksanakan sesuai dengan *Test Plan*.

Parameter performa yang diukur pada penelitian ini meliputi *Load Time*, *Response Time*, *Throughput*, *Latency*, dan *Error Rate*. Kriteria keberhasilan pengujian mengacu pada standar yang digunakan dalam penelitian, yaitu *Load Time* kurang dari 3 detik, *Latency* kurang dari 1 detik, *Error Rate* kurang dari 5%, serta *Response Time* dengan waktu *respons* optimal sebesar 0,1 detik. Nilai *Throughput* digunakan untuk mengevaluasi kemampuan sistem dalam memproses permintaan secara konsisten selama proses pengujian.

Tahap Test Planning menghasilkan dokumen Test Plan yang menjadi acuan dalam penyusunan test case, konfigurasi Apache JMeter, serta pelaksanaan pengujian performa pada tahap berikutnya. Dokumen *test plan* bisa diakses secara lengkap pada link <https://bit.ly/DokumenTestPlan>.

4.5. Test case Development

Tahap pengembangan *test case* dilakukan setelah perencanaan pengujian selesai disusun. Peneliti terlebih dahulu menentukan skenario pengujian yang akan digunakan sebagai dasar dalam pelaksanaan pengujian performa sistem. Pengujian ini terdapat 4 skenario pengujian, yang terdiri dari 3 skenario *load testing* dan 1 skenario *gorilla testing*. Skenario *load testing* digunakan untuk menguji performa sistem berdasarkan jenis pengguna, yaitu pengguna *Mobile*

siswa, *Mobile* guru, dan *website* admin. Sedangkan skenario *gorilla testing* digunakan untuk menguji kestabilan sistem dengan melakukan *request* secara berulang pada *endpoint* API dalam kondisi beban tinggi. Berikut merupakan rincian skenario pengujian yang digunakan:

Tabel 4. 1 Skenario *Load testing* 1 *Website* Admin

No	Jenis Skenario	Skenario
1	SLO1.1	Pengajuan <i>Login</i>
2	SLO1.2	Akses Jadwal
3	SLO1.3	Akses Absensi mapel
4	SLO1.4	Akses Absensi Kehadiran
5	SLO1.5	Kelola Siswa

Tabel 4.1 menunjukkan skenario *load testing* pada sistem berbasis *website* yang digunakan oleh admin. Setiap skenario difokuskan pada pengujian performa fitur utama, seperti *login*, akses jadwal, absensi kehadiran, absensi mata pelajaran, serta pengelolaan data siswa.

Tabel 4. 2 Skenario *Load testing* 2 *Mobile* Siswa

No	Jenis Skenario	Skenario
1	SLO2.1	Pengajuan <i>Login</i>
2	SLO2.2	Absensi mapel
3	SLO2.3	Pengajuan Perizinan
4	SLO2.4	Akses Pengumuman
5	SLO2.5	Akses Jadwal

Tabel 4.2 menunjukkan skenario *load testing* pada aplikasi *Mobile* siswa. Setiap skenario difokuskan pada pengujian performa fitur utama yang sering digunakan oleh siswa, meliputi *login*, absensi mata pelajaran, perizinan, pengumuman, serta jadwal.

Tabel 4. 3 Skenario *Load testing* 3 *Mobile* Guru

No	Jenis Skenario	Skenario
1	SLO3.1	Pengajuan <i>Login</i>
2	SLO3.2	Akses Absensi Mapel
3	SLO3.3	Tambah Pengumuman
4	SLO3.4	Akses Absensi Kehadiran
5	SLO3.5	Akses Perizinan
6	SLO3.6	Kelola Perizinan
7	SLO3.7	Akses Jadwal Personal
8	SLO3.8	Akses Jadwal Kelas
9	SLO3.9	Kelola Laporkan

Tabel 4.3 menunjukkan skenario *load testing* yang dilakukan pada aplikasi *Mobile* guru. Pengujian difokuskan pada berbagai fitur yang digunakan oleh guru dalam aktivitas sehari-hari, seperti *login*, akses absensi mata pelajaran, pembuatan pengumuman, akses absensi siswa, serta pengelolaan perizinan.

Tabel 4. 4 Skenario *Gorilla testing*

No	Jenis Skenario	Skenario
1	SGO1.1	<i>Scan QR Code</i> Absensi Mapel (<i>Mobile</i> Siswa)
2	SGO1.2	Kelola Perizinan (<i>Mobile</i> Guru)
3	SGO1.3	Akses Absensi mapel (<i>Website</i>)
4	SGO1.4	Akses Absensi Kehadiran (<i>Website</i>)

Tabel 4.4 menunjukkan skenario *gorilla testing* yang digunakan dalam pengujian performa sistem. Pengujian ini difokuskan pada beberapa fungsi penting, seperti proses *scan QR Code*, pengelolaan perizinan, absensi mata pelajaran, dan absensi kehadiran.

Test case pada penelitian ini juga disesuaikan dengan modul-modul utama yang terdapat pada sistem absensi, seperti modul *login*, modul absensi kehadiran, modul absensi mata pelajaran, serta modul perizinan dan akses data absensi oleh guru. Secara keseluruhan terdapat 48 *test case* utama, yang terdiri

dari 44 *test case* untuk *load testing* dan 4 *test case* untuk *gorilla testing*. *Test case* tersebut disusun dalam dokumen tersendiri dan dilengkapi dengan *Requirement Traceability Matrix* (RTM) untuk memastikan bahwa setiap kebutuhan pengujian telah terhubung dengan skenario pengujian yang dibuat. Dokumen *test case* dan RTM ini disertakan pada link <https://bit.ly/DokumenRTMPengujian> agar dapat digunakan sebagai pendukung dalam penelitian ini. Tabel *test case* pada tabel 4.5 sampai tabel 4.8 digunakan untuk menjelaskan hubungan antara skenario pengujian dengan *test case* yang mencakup ID skenario, deskripsi pengujian, serta *endpoint* API yang diuji.

Tabel 4. 5 *test case load testing website*

No	ID Test Case	Test case	ID Skenario	API
1	TCW.1	mengirim fungsi <i>login</i>	SLO1.1	<i>/login</i>
2	TCW.2	mengakses <i>QR Code</i>	SLO1.2	<i>/jadwal/qr/1</i>
3	TCW.3	Mengakses halaman Absensi Harian	SLO1.3	<i>/pages/absensi/absensi_harian/1</i>
4	TCW.4	Mengakses halaman Absensi mapel	SLO1.4	<i>/pages/absensi/absensi_mapel/1</i>
5	TCW.5	Mengirim data Foto	SLO1.5	<i>/flaskpresensiku/pipeline/run</i>

Tabel 4.5 menunjukkan daftar *test case* pada sistem berbasis *website* yang digunakan oleh admin. Setiap *test case* disusun berdasarkan skenario pengujian yang telah ditentukan dan direpresentasikan dalam bentuk permintaan ke *endpoint* API yang sesuai. Setiap aktivitas pada *test case*, seperti *login*, mengakses *QR Code*, absensi, dan pengiriman data, berhubungan langsung dengan API yang digunakan oleh sistem. Pengujian dilakukan dengan mengirim *request* ke masing-masing *endpoint* API untuk mengetahui kemampuan sistem dalam merespons permintaan tersebut secara bersamaan.

Tabel 4. 6 *test case load testing Mobile siswa*

No	ID Test Case	Test case	ID Skenario	API
1	TCMS.1	mengirim fungsi <i>login</i>	SLO2.1	/api/login
2	TCMS.2	Mengirim <i>scan QR Code</i>	SLO2.2	/api/qrattendance
3	TCMS.3	Menampilkan data perizinan	SLO2.3	/api/attendance/permission/report
4	TCMS.4	mengirim perizinan	SLO2.3	/api/attendance/permission
5	TCMS.5	Mengakses data jadwal	SLO2.6	/api/classes/1
6	TCMS.6	Mengakses data pengumuman	SLO2.7	/api/student/notifications

Tabel 4.6 menunjukkan *test case* pada aplikasi *Mobile siswa* yang digunakan dalam skenario *load testing*. Setiap *test case* disusun berdasarkan skenario pengujian yang telah ditentukan dan direpresentasikan dalam bentuk permintaan ke *endpoint* API yang sesuai. Setiap aktivitas, seperti *login*, pemindaian *QR Code*, perizinan, serta akses data jadwal dan pengumuman, berhubungan langsung dengan API yang digunakan oleh sistem. Pengujian dilakukan dengan mengirim *request* ke masing-masing *endpoint* API untuk mengetahui kemampuan sistem dalam merespons permintaan pengguna secara bersamaan.

Tabel 4. 7 *test case load testing Mobile guru*

No	ID Test case	Test case	ID Skenario	API
1	TCMG.1	Mengirim fungsi <i>login</i>	SLO3.1	/api/login

No	ID	Test case	Test case	ID Skenario	API
2	TCMG.2	Mengakses halaman Absensi mapel		SLO3.2	/api/teacher/attendance/class/2/2026-05-25
3	TCMG.3	Mengirim pengumuman		SLO3.3	/api/fcm/send-announcement
4	TCMG.4	Mengakses halaman Absensi Kehadiran		SLO3.4	/api/teacher/attendance/daily/class?id_classes=1&date=2026-04-1
5	TCMG.5	Mengakses halaman perizinan		SLO3.5	/api/teacher/attendance/permission/report/1?start_date=2026-05-22&end_date=2026-05-22
6	TCMG.6	Mengirim persetujuan perizinan		SLO3.5	/api/teacher/attendance/permission/accept
7	TCMG.7	Mengakses halaman jadwal personal		SLO3.6	/api/teacher/schedules
8	TCMG.8	Mengakses halaman jadwal kelas		SLO3.7	/api/schedules/2
9	TCMG.9	Mengirim Laporkan siswa		SLO3.8	/api/teacher/attendance/discrepancy/report

Tabel 4.7 menunjukkan *test case* pada aplikasi *Mobile* guru yang digunakan dalam skenario *load testing*. Setiap *test case* disusun berdasarkan skenario pengujian yang telah ditentukan dan direpresentasikan dalam bentuk permintaan ke *endpoint* API yang sesuai. Setiap aktivitas, seperti *login*, pengelolaan absensi mata pelajaran, pengiriman pengumuman, pengelolaan absensi kehadiran, perizinan, serta akses jadwal, berhubungan langsung dengan API yang digunakan oleh sistem. Pengujian dilakukan dengan mengirim *request* ke masing-masing *endpoint* API untuk mengetahui kemampuan sistem dalam merespons permintaan pengguna secara bersamaan.

Tabel 4. 8 *test case Gorilla testing*

No	ID	Test case	ID	API
	case		Skenario	
1	TCGO.1	mengirim <i>Scan QR</i> (<i>Mobile Siswa</i>)	SGO1.1	/api/qrattendance
2	TCGO.2	Mengirim <i>Update</i> data perizinan (<i>Mobile Guru</i>)	SGO1.2	/api/teacher/attendance/ permission/accept
3	TCGO.3	Mengakses halaman Absensi Kehadiran (<i>Website</i>)	SGO1.3	/pages/absensi/ absensi_mapel/1 1
4	TCGO.4	Mengakses halaman Absensi Mapel (<i>Website</i>)	SGO1.4	/pages/kelas/kelas_mapel

Tabel 4.8 menunjukkan *test case* pada skenario *gorilla testing* yang difokuskan pada pengujian fitur-fitur tertentu secara berulang dengan intensitas tinggi. Setiap *test case* merepresentasikan aktivitas yang terhubung langsung dengan *endpoint* API dalam sistem. Pengujian dilakukan dengan mengirim *request* secara berulang pada fungsi tertentu untuk mengetahui tingkat kestabilan sistem dalam menangani beban tinggi dalam jangka waktu tertentu..

4.6. Test Environment setup

Tahap *test environment setup* merupakan proses penyiapan lingkungan pengujian yang digunakan dalam pelaksanaan pengujian performa sistem. Lingkungan pengujian disusun agar mampu merepresentasikan kondisi penggunaan sistem secara nyata, sehingga hasil pengujian yang diperoleh dapat dianalisis secara lebih akurat. Sistem absensi yang diuji dijalankan pada server hosting, sedangkan simulasi pengguna dilakukan menggunakan aplikasi Apache JMeter yang dijalankan pada perangkat komputer penguji. Link <https://bit.ly/DokumenSUD>.

3.9.1. Spesifikasi Lingkungan Pengujian

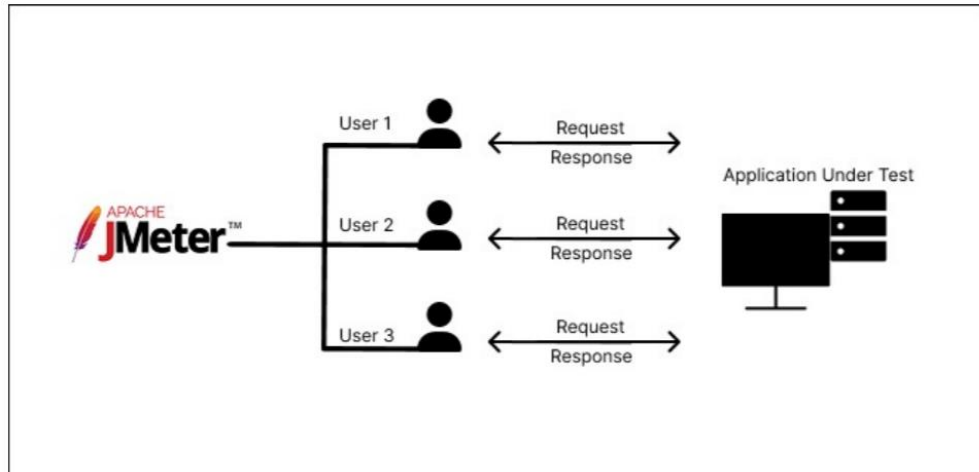
Subbab ini menjelaskan spesifikasi perangkat keras (*hardware*) dan perangkat lunak (*software*) yang digunakan dalam proses pengujian performa sistem absensi Presensiku. Spesifikasi tersebut digunakan sebagai lingkungan pengujian untuk menjalankan simulasi pengguna virtual menggunakan Apache JMeter serta mendukung proses pengujian sistem secara keseluruhan. Adapun spesifikasi perangkat yang digunakan dalam proses pengujian dapat dilihat pada tabel 4.9.

Tabel 4. 9 Spesifikasi Lingkungan Pengujian

No	Komponen	Spesifikasi
1	Perangkat Keras	Laptop / Komputer Pengujian
2	Prosesor	Intel Core i5 / setara
3	RAM	8 GB
4	Sistem Operasi	Windows 10
5	Web Server	XAMPP / Apache
6	Database	MySQL
7	Tools Pengujian	Apache JMeter

Lingkungan pengujian ini digunakan sebagai dasar dalam pelaksanaan proses pengujian performa sistem absensi. Adanya konfigurasi lingkungan pengujian yang jelas, proses eksekusi pengujian dapat dilakukan secara

terstruktur dan hasil yang diperoleh dapat dianalisis secara lebih akurat pada tahap selanjutnya.

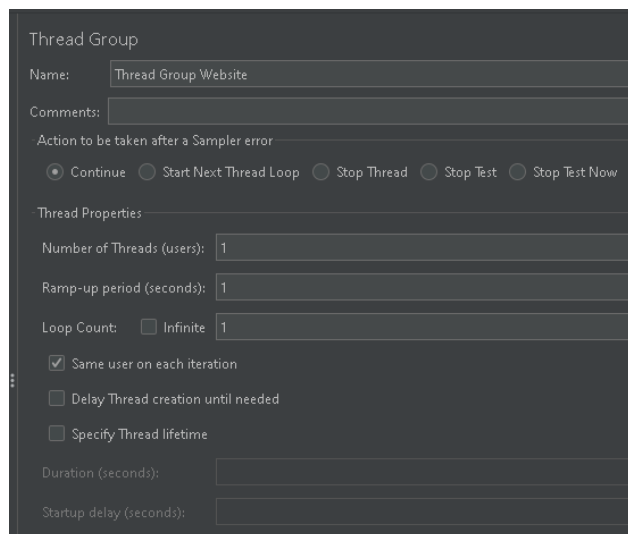


Gambar 4.x Arsitektur Lingkungan Pengujian

Gambar 3.x menunjukkan arsitektur lingkungan pengujian yang digunakan pada penelitian ini. Apache JMeter berperan sebagai alat pengujian yang mensimulasikan sejumlah virtual *user* untuk mengirimkan *request* ke sistem yang diuji (*Application Under Test*). Setiap virtual user menjalankan skenario pengujian sesuai *test case* yang telah dirancang. Sistem kemudian memproses setiap request dan mengembalikan response kepada Apache JMeter. *Response* tersebut digunakan untuk memperoleh nilai parameter performa, meliputi Load Time, Response Time, Throughput, Latency, dan Error Rate.

3.9.2. Konfigurasi Pengujian *Load testing* Pada *Website Admin*

Konfigurasi pengujian *website* pada Apache JMeter diawali dengan pengaturan *Thread Group*. *Thread Group* digunakan untuk mensimulasikan pengguna virtual yang mengakses *website* secara bersamaan. Pada bagian ini, peneliti mengisi parameter utama berupa *Number of Threads* untuk menentukan jumlah pengguna virtual, *Ramp-up Period* untuk mengatur waktu menjalankan pengguna secara bertahap, dan *Loop count* untuk menentukan jumlah pengulangan *request*. Konfigurasi *Thread Group* ini menjadi dasar sebelum penambahan *HTTP Request* pada masing-masing *test case*. Tampilan konfigurasi *Thread Group* ditunjukkan pada Gambar 4.1

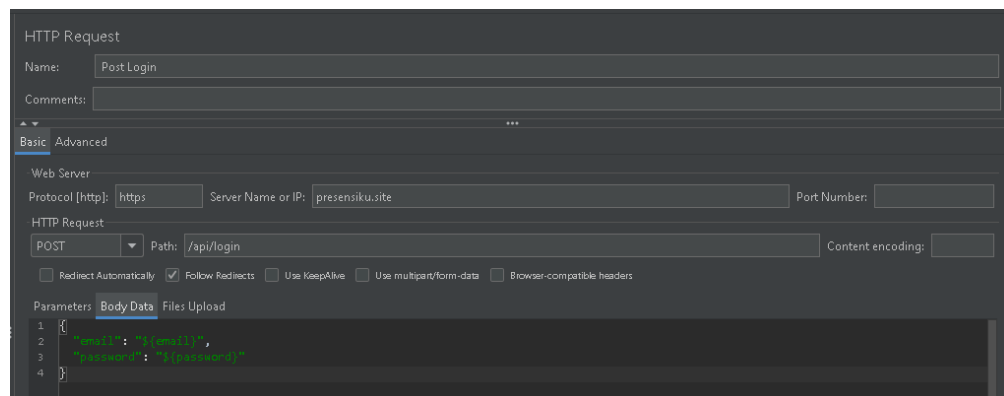


Gambar 4. 1 *Thread Group* Pada pengujian *Website*

Gambar 4.1 menunjukkan Konfigurasi *Thread Group* pada pengujian *website* disesuaikan dengan jumlah pengguna yang tersedia pada sistem. Karena *website* hanya digunakan oleh satu pengguna admin, maka jumlah pengguna virtual pada *Thread Group* diatur sebanyak 1 *user*. Parameter lainnya yang digunakan yaitu *Ramp-up Period* selama 1 detik untuk menjalankan *user* secara bertahap, serta *Loop count* sebanyak 1 kali karena pengujian hanya dilakukan satu kali pada setiap skenario. Konfigurasi ini digunakan untuk mensimulasikan akses admin pada *website* secara sederhana sesuai kondisi penggunaan sebenarnya.

a. Konfigurasi *HTTP Request* mengirim fungsi *login*

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses *login* pengguna pada sistem Presensiku menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengirim permintaan *login* ke server melalui *endpoint* yang telah ditentukan sehingga performa sistem dapat dianalisis saat menerima akses dari pengguna secara bersamaan.

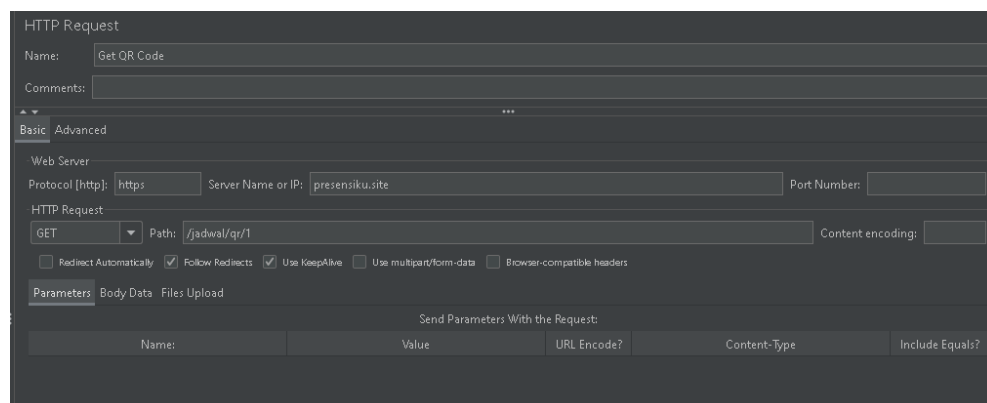


Gambar 4. 2 *HTTP Request mengirim fungsi login*

Berdasarkan konfigurasi pada gambar 4.2, *HTTP Request* menggunakan protokol *https* dengan server *presensiku.site*. Method yang digunakan yaitu *POST* dengan path */login* untuk proses autentikasi pengguna. Parameter yang dikirim meliputi email, password, dan *_token* sebagai token keamanan pada framework Laravel. Konfigurasi ini digunakan untuk mensimulasikan aktivitas *login* pengguna pada sistem selama proses *load testing* berlangsung.

b. Konfigurasi *HTTP Request* mengakses halaman *QR Code*

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman *QR Code* pada sistem Presensiku menggunakan Apache JMeter.



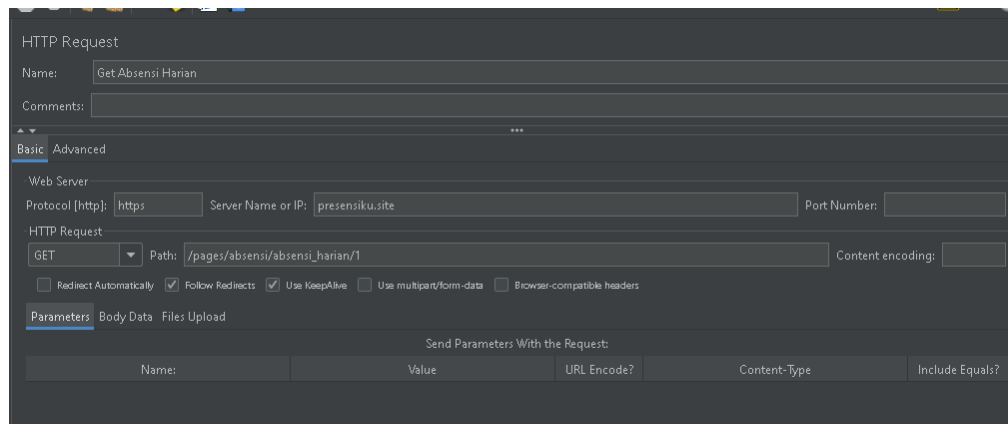
Gambar 4. 3 *HTTP Request mengakses halaman QR Code*

Berdasarkan konfigurasi pada gambar 4.3, *HTTP Request* menggunakan protokol *https* dengan server *presensiku.site*. Method yang digunakan yaitu *GET* dengan *endpoint* */jadwal/qr/1* untuk mengambil data *QR Code*. Angka 1 pada *endpoint* menunjukkan parameter identitas data yang digunakan sistem untuk menampilkan informasi *QR Code* tersebut. Konfigurasi ini digunakan untuk

mensimulasikan aktivitas pengguna saat membuka halaman *QR Code* pada sistem.

c. Konfigurasi *HTTP Request* mengakses halaman absensi harian

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman absensi harian pada sistem Presensiku menggunakan Apache JMeter.

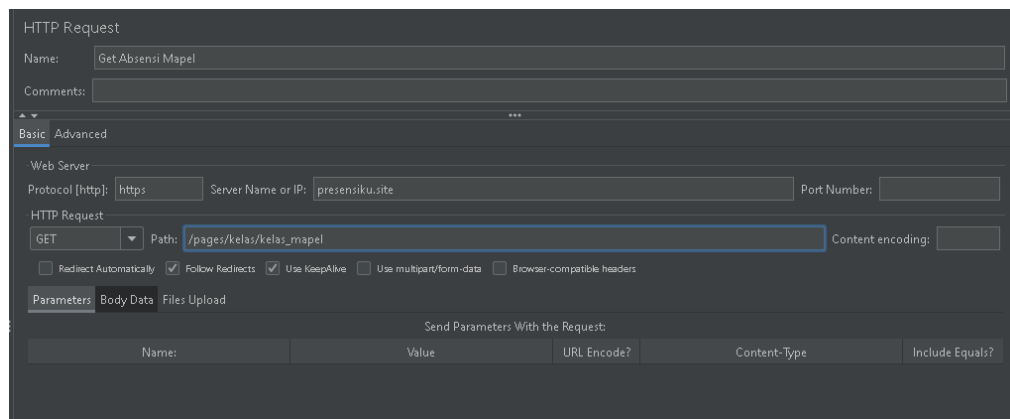


Gambar 4. 4 *HTTP Request* mengakses halaman absensi harian

Berdasarkan konfigurasi pada gambar 4.4, *HTTP Request* menggunakan protokol *https* dengan server *presensiku.site*. *Method* yang digunakan yaitu *GET* dengan *endpoint* */pages/absensi/absensi_harian/1* untuk mengambil data riwayat absensi harian dari server. Angka 1 pada *endpoint* menunjukkan parameter identitas data yang digunakan sistem untuk menampilkan informasi absensi tertentu. Konfigurasi ini digunakan untuk mensimulasikan aktivitas pengguna saat membuka halaman riwayat absensi harian pada sistem.

d. Konfigurasi *HTTP Request* mengakses halaman absensi mapel

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman absensi mata pelajaran pada sistem Presensiku menggunakan Apache JMeter.

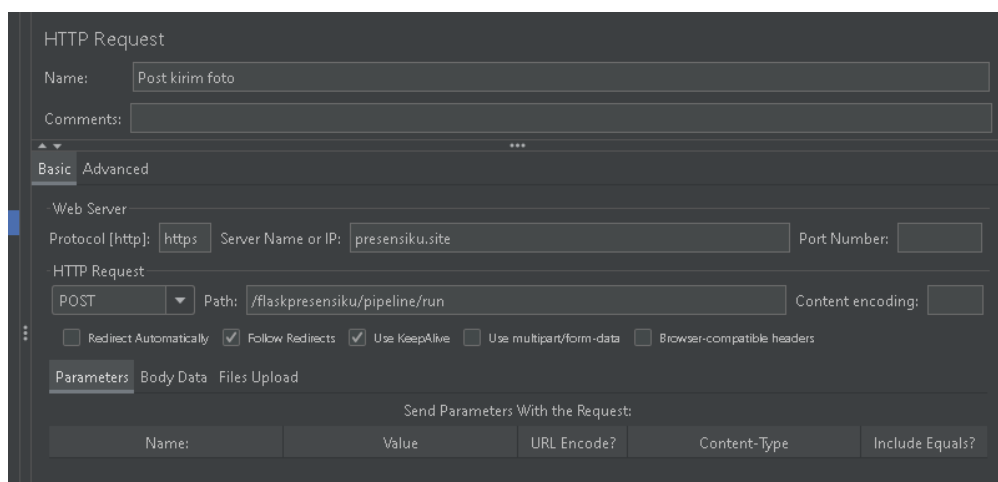


Gambar 4. 5 *HTTP Request* mengakses halaman absensi mapel

Berdasarkan konfigurasi pada gambar 4.5, *HTTP Request* menggunakan protokol *https* dengan server *presensiku.site*. *Method* yang digunakan yaitu *GET* dengan *endpoint* */pages/kelas/kelas_mapel* untuk mengambil data riwayat absensi mata pelajaran dari server. *Endpoint* tersebut digunakan untuk menampilkan informasi absensi berdasarkan mata pelajaran yang tersedia pada sistem. Konfigurasi ini digunakan untuk mensimulasikan aktivitas pengguna saat mengakses halaman riwayat absensi mapel pada sistem.

e. Konfigurasi *HTTP Request* mengirim data foto

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses pengiriman data foto pada sistem Presensiku menggunakan Apache JMeter.



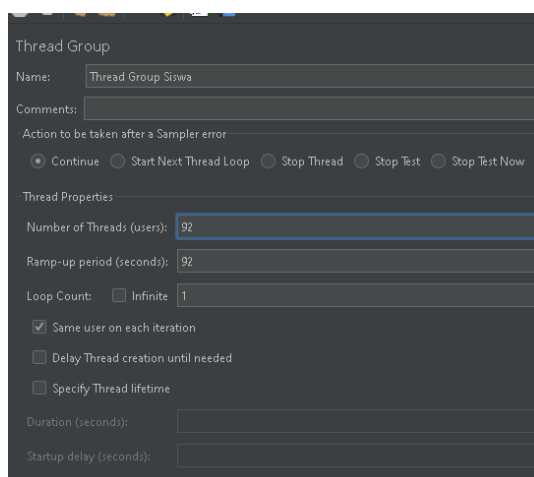
Gambar 4. 6 *HTTP Request* mengirim data foto

Berdasarkan konfigurasi pada Gambar 4.6, *HTTP Request* menggunakan protokol *HTTPS* dengan server *presensiku.site*. *Method* yang digunakan yaitu

POST dengan *endpoint /flaskpresensiku/pipeline/run* untuk mengirim data foto ke server. Endpoint tersebut digunakan untuk memproses foto yang dikirim pengguna sebagai bagian dari proses presensi pada sistem. Konfigurasi ini digunakan untuk mensimulasikan aktivitas pengguna saat melakukan pengiriman foto ke server melalui aplikasi Presensiku. Pengujian dilakukan untuk mengetahui kemampuan sistem dalam menerima dan memproses data foto dari banyak pengguna secara bersamaan selama proses presensi berlangsung.

3.9.3. Konfigurasi Pengujian *Load testing* Pada *Mobile* Siswa

Konfigurasi pengujian *Mobile* siswa pada Apache JMeter diawali dengan pengaturan *Thread Group*. *Thread Group* digunakan untuk mensimulasikan pengguna virtual yang mengakses sistem secara bersamaan melalui aplikasi *mobile*. Peneliti mengisi parameter utama berupa *Number of Threads* untuk menentukan jumlah pengguna virtual, *Ramp-up Period* untuk mengatur waktu menjalankan pengguna secara bertahap, dan *Loop count* untuk menentukan jumlah pengulangan *request*. Konfigurasi *Thread Group* ini menjadi dasar sebelum penambahan *HTTP Request* pada masing-masing *test case*. Tampilan konfigurasi *Thread Group* ditunjukkan pada Gambar 4.7



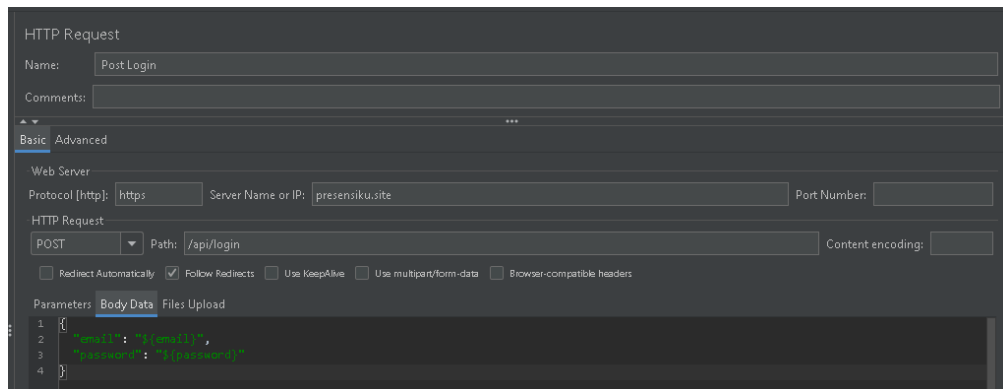
Gambar 4. 7 *Thread Group* pada pengujian *Mobile* siswa

Gambar 4.7 menunjukkan Konfigurasi *Thread Group* pada pengujian *Mobile* siswa disesuaikan dengan jumlah pengguna siswa yang menggunakan aplikasi Presensiku. Jumlah pengguna virtual pada *Thread Group* diatur sesuai

skenario pengujian, yaitu sebanyak 92 *user*, 297 *user*, dan 1034 *user*. Parameter *Ramp-up Period* disamakan dengan jumlah *user* pada setiap skenario sehingga pengguna virtual dijalankan secara bertahap dengan jangka sekitar 1 *user* per detik. Pengaturan ini dilakukan agar peningkatan jumlah *request* berjalan secara bertahap, sehingga *server* tidak langsung menerima beban akses yang besar dalam waktu singkat selama proses pengujian. *Loop count* diatur sebanyak 1 kali karena setiap skenario pengujian dijalankan satu kali sesuai kebutuhan pengambilan data. Konfigurasi tersebut digunakan untuk mensimulasikan akses siswa pada aplikasi *Mobile* secara bersamaan sesuai dengan kondisi penggunaan sistem sebenarnya.

a. Konfigurasi *HTTP Request* mengirim fungsi *login*

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses *login* pengguna pada aplikasi *Mobile Presensiku* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengirim data autentikasi pengguna ke *server* sebagai tahap awal akses aplikasi.

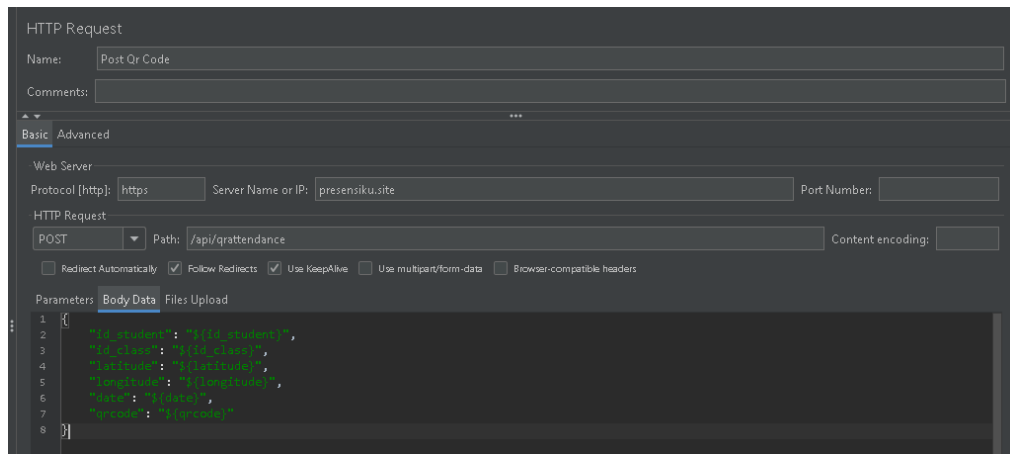


Gambar 4. 8 *HTTP Request* mengirim fungsi *login*

Berdasarkan konfigurasi pada gambar 4.8, *HTTP Request* menggunakan protokol *https* dengan *server* *presensiku.site*. *Method* yang digunakan yaitu *POST* dengan *endpoint* */api/login* untuk mengirim data *login* pengguna ke *server*. Parameter yang digunakan pada *request* terdiri dari *email*, *password*, dan *_token* sebagai data *autentikasi* pengguna saat proses *login* berlangsung. Konfigurasi ini digunakan untuk mensimulasikan aktivitas *login* siswa pada aplikasi *mobile*.

b. Konfigurasi *HTTP Request* mengirim scan QR Code

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses pengiriman data *scan QR Code* pada aplikasi *Mobile Presensiku* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa *server* saat menerima data presensi dari pengguna.

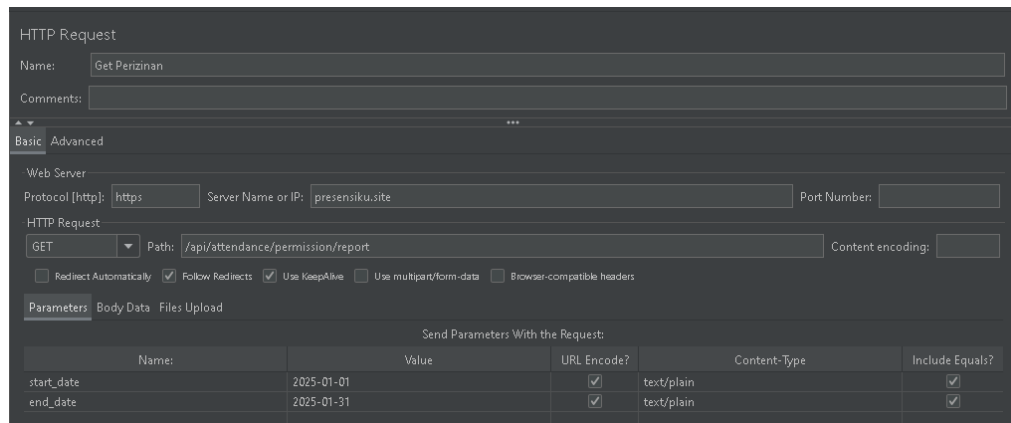


Gambar 4. 9 *HTTP Request* mengirim *scan QR Code*

Gambar 4.9 pada bagian Body Data, *request* mengirimkan data dalam format JSON berupa *id_student*, *id_class*, *latitude*, *longitude*, *date*, dan *qrCode*. Data tersebut diambil dari variabel CSV, seperti *\${id_student}*, *\${id_class}*, *\${latitude}*, *\${longitude}*, *\${date}*, dan *\${qrCode}*. Konfigurasi ini digunakan untuk menguji kemampuan *server* dalam menerima *request* presensi dari banyak pengguna secara bersamaan.

c. Konfigurasi *HTTP Request* Menampilkan data perizinan

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses data perizinan pada aplikasi *Mobile Presensiku* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa *server* saat menampilkan data perizinan pengguna.



HTTP Request

Name: Get Perizinan

Comments:

Basic Advanced

Web Server

Protocol [http]: https Server Name or IP: presensiku.site Port Number:

HTTP Request

GET Path: /api/attendance/permission/report Content encoding:

☐ Redirect Automatically ☒ Follow Redirects ☒ Use KeepAlive ☐ Use multipart/form-data ☐ Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

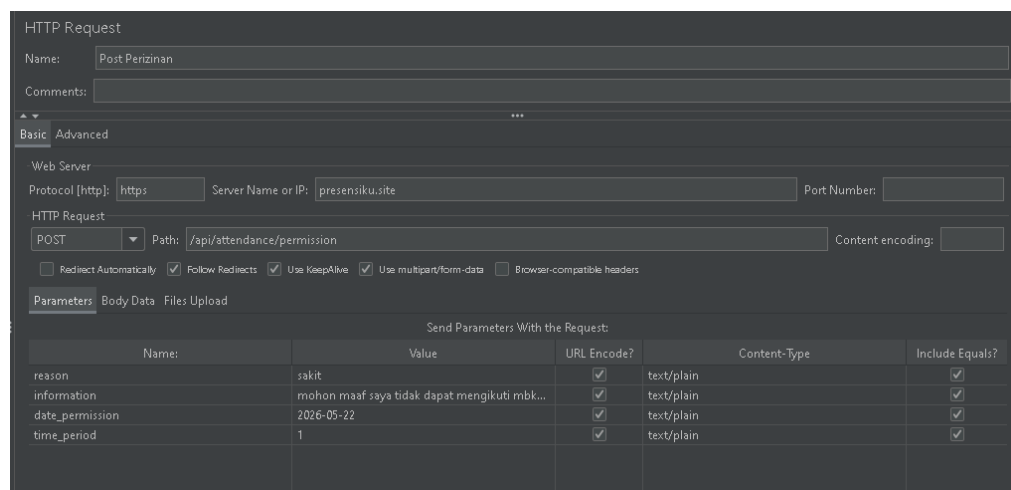
Name:	Value	URL Encode?	Content-Type	Include Equals?
start_date	2025-01-01	☒	text/plain	☒
end_date	2025-01-31	☒	text/plain	☒

Gambar 4. 10 *HTTP Request* Menampilkan data perizinan

Berdasarkan konfigurasi pada gambar 4.10, *HTTP Request* menggunakan protokol *https* dengan server *presensiku.site*. *Method* yang digunakan yaitu *POST* dengan *endpoint* */api/attendance/permission/report* untuk mengambil data perizinan dari *server* berdasarkan rentang tanggal yang ditentukan. Parameter yang digunakan pada *request* terdiri dari *start_date* dan *end_date* sebagai data *filter* untuk menampilkan laporan perizinan pengguna. Konfigurasi ini digunakan untuk mensimulasikan aktivitas pengguna saat membuka data perizinan pada aplikasi *mobile*.

d. Konfigurasi *HTTP Request* mengirim perizinan

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses pengiriman data perizinan pada aplikasi *Mobile Presensiku* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menerima data perizinan dari pengguna.



HTTP Request

Name: Post Perizinan

Comments:

Basic Advanced

Web Server

Protocol [http]: https Server Name or IP: presensiku.site Port Number:

HTTP Request

POST Path: /api/attendance/permission Content encoding:

☐ Redirect Automatically ☒ Follow Redirects ☒ Use KeepAlive ☒ Use multipart/form-data ☐ Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	URL Encode?	Content-Type	Include Equals?
reason	sakit	☒	text/plain	☒
information	mohon maaf saya tidak dapat mengikuti mbk...	☒	text/plain	☒
date_permission	2026-05-22	☒	text/plain	☒
time_period	1	☒	text/plain	☒

Gambar 4. 11 *HTTP Request* mengirim perizinan

Berdasarkan konfigurasi pada gambar 4.11, *HTTP Request* menggunakan protocol https dengan server presensiku.site. Method yang digunakan yaitu *POST* dengan *endpoint* /api/attendance/permission untuk mengirim data perizinan siswa ke server. Parameter yang digunakan pada *request* terdiri dari reason, information, date_permission, dan time_period sebagai data pengajuan perizinan pengguna. Selain itu, *request* juga menggunakan parameter evidence berupa file gambar sebagai bukti pendukung perizinan yang dikirim melalui fitur multipart/form-data. Konfigurasi ini digunakan untuk mensimulasikan aktivitas pengguna saat mengirim pengajuan perizinan pada aplikasi *mobile*.

e. Konfigurasi *HTTP Request* Mengakses data jadwal

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses data jadwal pada aplikasi *Mobile Presensiku* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menampilkan data jadwal kepada pengguna.

HTTP Request

Name: Get Jadwal Siswa

Comments:

Basic Advanced

Web Server

Protocol (http): https Server Name or IP: presensiku.site Port Number:

HTTP Request

GET Path: /api/classes/1 Content encoding:

☐ Redirect Automatically ☒ Follow Redirects ☒ Use KeepAlive ☐ Use multipart/form-data ☐ Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name	Value	URL Encode?	Content-Type	Include Equals?
------	-------	-------------	--------------	-----------------

Gambar 4. 12 *HTTP Request* Mengakses data jadwal

Berdasarkan konfigurasi pada gambar 4.12, *HTTP Request* menggunakan protocol https dengan server presensiku.site. Method yang digunakan yaitu *GET* untuk mengambil data jadwal dari server melalui *endpoint* /api/classes/1. Konfigurasi ini digunakan untuk mensimulasikan aktivitas pengguna saat mengakses data jadwal pada aplikasi *mobile*.

f. Konfigurasi *HTTP Request* Mengakses data pengumuman

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses data jadwal pada aplikasi *Mobile Presensiku* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menampilkan data jadwal kepada pengguna.

Name	Value	URL Encode?	Content-Type	Include Equals?
start_date	2025-01-01	☒	text/plain	☒
end_date	2025-01-31	☒	text/plain	☒

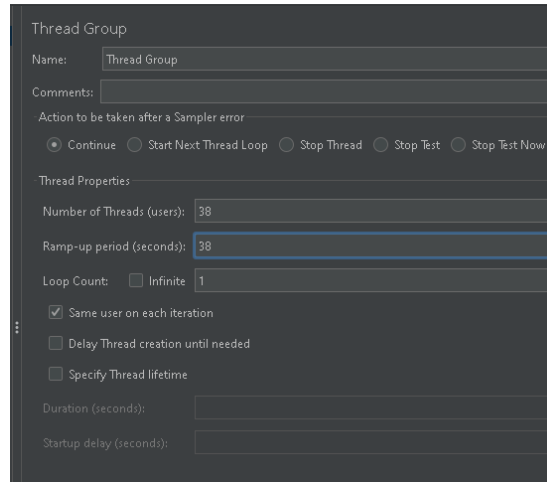
Gambar 4. 13 *HTTP Request* Mengakses data pengumuman

Berdasarkan konfigurasi pada gambar 4.13, *HTTP Request* menggunakan protocol *https* dengan server *presensiku.site*. Method yang digunakan yaitu *POST* untuk mengambil data pengumuman dari server melalui *endpoint* yang telah ditentukan. Parameter yang digunakan pada *request* terdiri dari *start_date* dan *end_date* sebagai data filter untuk menampilkan informasi pengumuman berdasarkan rentang tanggal tertentu. Konfigurasi ini digunakan untuk mensimulasikan aktivitas pengguna saat mengakses data pengumuman pada aplikasi *mobile*.

3.9.4. Konfigurasi Pengujian *Load testing* Pada *Mobile Guru*

Konfigurasi pengujian *Mobile guru* pada Apache JMeter diawali dengan pengaturan Thread Group. Thread Group digunakan untuk mensimulasikan pengguna virtual yang mengakses sistem secara bersamaan melalui aplikasi *Mobile guru*. Pada bagian ini, peneliti mengisi parameter utama berupa Number of Threads untuk menentukan jumlah pengguna virtual, *Ramp-up Period* untuk mengatur waktu menjalankan pengguna secara bertahap, dan *Loop count* untuk menentukan jumlah pengulangan *request*. Konfigurasi Thread Group ini

menjadi dasar sebelum penambahan *HTTP Request* pada masing-masing *test case*. Tampilan konfigurasi Thread Group ditunjukkan pada Gambar 4.14

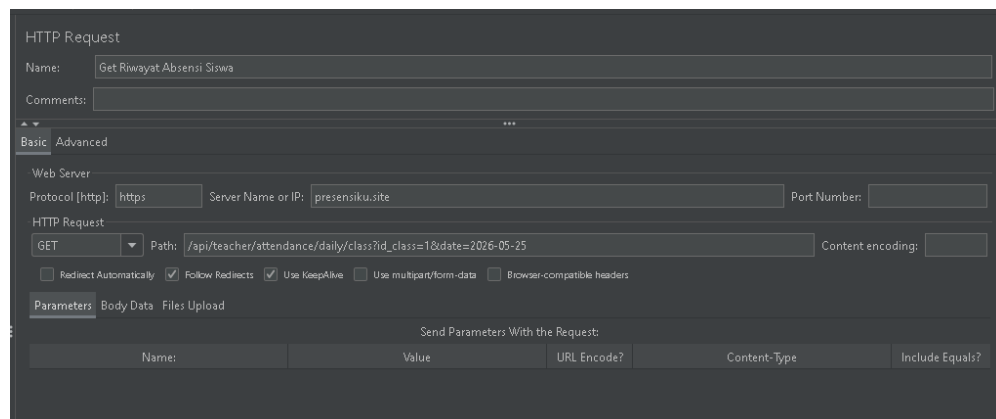


Gambar 4. 14 *Thread Group* pada pengujian *Mobile* guru

Gambar 4.14 Konfigurasi Thread Group pada pengujian *Mobile* guru disesuaikan dengan jumlah pengguna guru yang menggunakan aplikasi Presensiku. Jumlah pengguna virtual pada Thread Group diatur sesuai skenario pengujian yaitu sebanyak 38 *user*, 52 *user*, dan 59 *user*. Parameter *Ramp-up Period* disamakan dengan jumlah *user* pada setiap skenario sehingga pengguna virtual dijalankan secara bertahap dengan interval sekitar 1 *user* per detik. Serta *Loop count* sebanyak 1 kali karena pengujian dilakukan satu kali pada setiap skenario. Konfigurasi ini digunakan untuk mensimulasikan akses guru pada aplikasi *Mobile* secara bersamaan sesuai kondisi penggunaan sistem sebenarnya.

a. Mengakses halaman Absensi Kehadiran

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman absensi kehadiran pada aplikasi *Mobile* guru menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menampilkan data absensi kehadiran kepada pengguna.

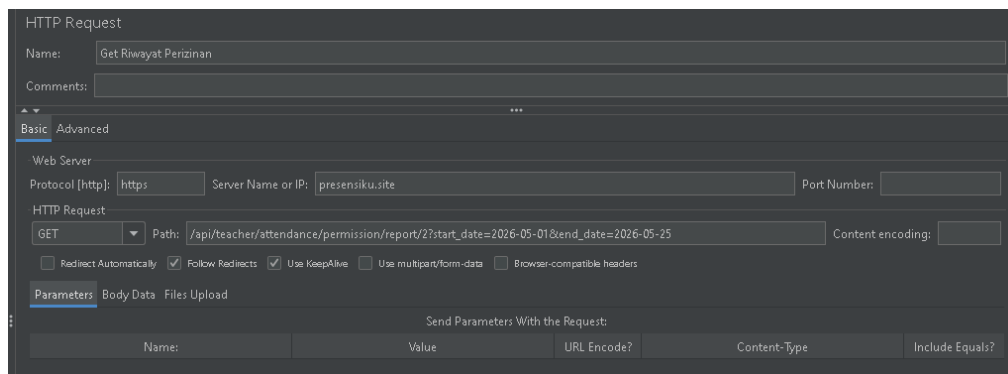


Gambar 4. 15 *HTTP Request* halaman Absensi Kehadiran

Gambar 4.15 menjelaskan konfigurasi *HTTP Request* yang digunakan pada Apache JMeter untuk melakukan pengujian fitur absensi kehadiran guru pada aplikasi *mobile*. *Endpoint* yang digunakan yaitu `/api/teacher/attendance/daily/class?id_class=1&date=2026-05-25`. *Test case* ini menggunakan method *GET* karena sistem hanya melakukan pengambilan data absensi dari server tanpa mengirim atau mengubah data. Protocol *HTTPS* digunakan untuk memastikan proses komunikasi data berlangsung secara aman melalui server `presensiku.site`. Parameter `id_class=1` digunakan untuk menentukan kelas yang diakses, sedangkan parameter `date=2026-05-25` digunakan untuk menampilkan data absensi berdasarkan tanggal tertentu. Konfigurasi ini digunakan untuk mensimulasikan aktivitas guru saat membuka halaman absensi kehadiran pada aplikasi *Mobile* sehingga performa sistem dalam menangani permintaan pengambilan data dapat dianalisis selama proses *load testing* berlangsung.

b. Mengakses halaman perizinan

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman perizinan pada aplikasi *Mobile* guru menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menampilkan data perizinan kepada pengguna.



Gambar 4. 16 *HTTP Request* halaman perizinan

Gambar 4.16 menjelaskan konfigurasi *HTTP Request* yang digunakan pada Apache JMeter untuk melakukan pengujian fitur halaman perizinan pada aplikasi *Mobile* guru. *Endpoint* yang digunakan yaitu `/api/teacher/attendance/permission/report/2?start_date=2026-05-01&end_date=2026-05-25` dengan method *GET* karena sistem hanya mengambil data perizinan dari server tanpa melakukan perubahan data. Protocol *HTTPS* digunakan untuk memastikan proses komunikasi data dengan server `presensiku.site` berjalan dengan aman. Parameter `start_date` dan `end_date` digunakan untuk menentukan rentang tanggal data perizinan yang ditampilkan pada halaman aplikasi. Konfigurasi ini digunakan untuk mensimulasikan aktivitas guru saat mengakses halaman perizinan pada aplikasi *Mobile* sehingga performa sistem dalam menangani permintaan pengambilan data dapat dianalisis selama proses *load testing* berlangsung.

c. Mengirim persetujuan perizinan

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses pengiriman persetujuan perizinan pada aplikasi *Mobile* guru menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menerima data persetujuan perizinan dari pengguna.

The screenshot shows the 'HTTP Request' configuration in Apache JMeter. The 'Name' field is 'Post Perizinan'. The 'Protocol' is 'https' and the 'Server Name or IP' is 'presensiku.site'. The 'Method' is 'POST' and the 'Path' is '/api/attendance/permission'. The 'Content encoding' is empty. The 'Parameters' tab is selected, showing a table of parameters to be sent with the request.

Name	Value	URL Encode?	Content-Type	Include Equals?
reason	sakit	☒	text/plain	☒
information	mohon maaf saya tidak dapat mengikuti mbk...	☒	text/plain	☒
date_permission	2026-05-22	☒	text/plain	☒
time_period	1	☒	text/plain	☒

Gambar 4. 17 HTTP Request Mengirim persetujuan perizinan

Berdasarkan konfigurasi pada gambar 4.17, HTTP Request menggunakan protocol https dengan server presensiku.site. Method yang digunakan yaitu *POST* untuk mengirim data persetujuan perizinan ke server melalui *endpoint* yang telah ditentukan. Konfigurasi ini digunakan untuk mensimulasikan aktivitas guru saat memberikan persetujuan perizinan pada aplikasi *mobile*.

d. Mengakses halaman jadwal personal

Konfigurasi HTTP Request dilakukan untuk mensimulasikan proses akses halaman jadwal personal pada aplikasi *Mobile* guru menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menampilkan data jadwal personal kepada pengguna.

The screenshot shows the 'HTTP Request' configuration in Apache JMeter. The 'Name' field is 'Get Jadwal Personal'. The 'Protocol' is 'https' and the 'Server Name or IP' is 'presensiku.site'. The 'Method' is 'GET' and the 'Path' is '/api/teacher/schedules'. The 'Content encoding' is empty. The 'Parameters' tab is selected, showing a table of parameters to be sent with the request.

Name	Value	URL Encode?	Content-Type	Include Equals?
------	-------	-------------	--------------	-----------------

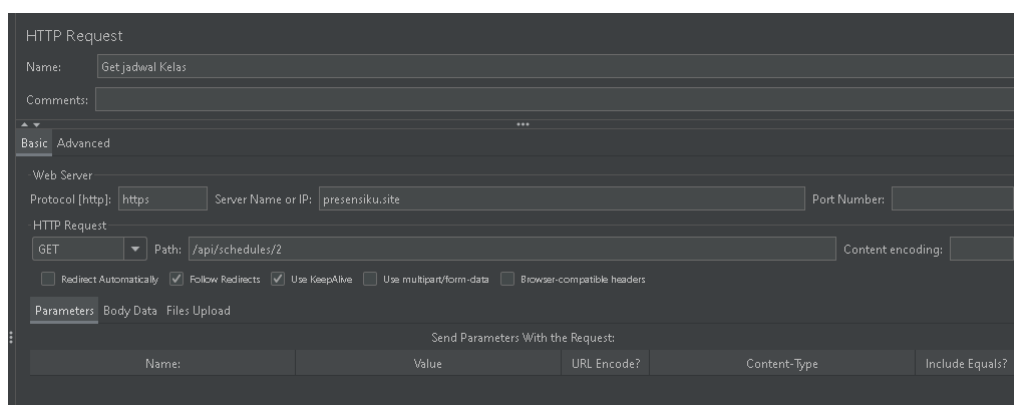
Gambar 4. 18 HTTP Request Mengakses halaman jadwal personal

Berdasarkan konfigurasi pada gambar 4.18, HTTP Request menggunakan protocol HTTPS dengan server presensiku.site. Method yang

digunakan yaitu *GET* untuk mengambil data jadwal personal dari server melalui *endpoint/api/teacher/schedules*. *Endpoint* tersebut digunakan untuk menampilkan informasi jadwal mengajar guru pada aplikasi *Mobile* tanpa melakukan perubahan data pada server. Konfigurasi ini digunakan untuk mensimulasikan aktivitas guru saat mengakses halaman jadwal personal pada aplikasi *Mobile* sehingga performa sistem dalam menangani permintaan pengambilan data jadwal dapat dianalisis selama proses *load testing* berlangsung.

e. Mengakses halaman jadwal kelas

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman jadwal kelas pada aplikasi *Mobile* guru menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menampilkan data jadwal kelas kepada pengguna.



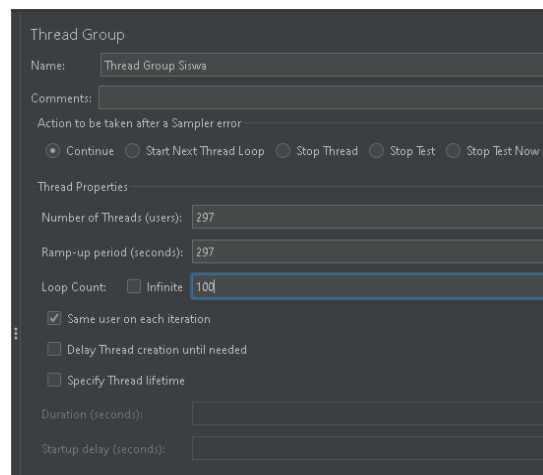
Gambar 4. 19 *HTTP Request* halaman jadwal kelas

Berdasarkan konfigurasi pada gambar 4.19, *HTTP Request* menggunakan protokol HTTPS dengan server presensiku.site. Method yang digunakan yaitu *GET* untuk mengambil data jadwal kelas dari server melalui *endpoint /api/schedules/2*. *Endpoint* tersebut digunakan untuk menampilkan informasi jadwal kelas berdasarkan identitas kelas yang dipilih pada aplikasi *Mobile* tanpa melakukan perubahan data pada server. Konfigurasi ini digunakan untuk mensimulasikan aktivitas guru saat mengakses halaman jadwal kelas pada aplikasi *Mobile* sehingga performa sistem dalam menangani permintaan

pengambilan data jadwal dapat dianalisis selama proses *load testing* berlangsung.

3.9.5. Konfigurasi Pengujian *Gorilla testing*

Konfigurasi pengujian *Gorilla testing* pada Apache JMeter diawali dengan pengaturan Thread Group yang digunakan untuk mensimulasikan aktivitas pengguna pada fitur utama sistem secara berulang. Thread Group berfungsi untuk mengatur jumlah pengguna virtual yang menjalankan *request* ke server selama proses pengujian berlangsung. Pada bagian ini, peneliti mengatur parameter utama berupa Number of Threads untuk menentukan jumlah pengguna virtual, *Ramp-up Period* untuk mengatur waktu menjalankan pengguna secara bertahap, serta *Loop count* untuk menentukan jumlah pengulangan *request* pada setiap skenario pengujian. Tampilan konfigurasi Thread Group *Gorilla testing* ditunjukkan pada Gambar 4.20.



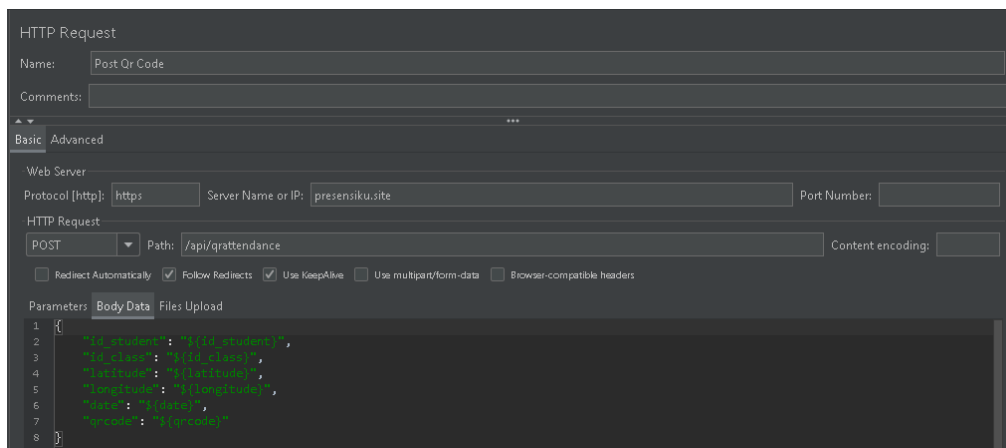
Gambar 4. 20 *Thread Group* pada pengujian *Gorilla testing*

Gambar 4.20 Konfigurasi Thread Group pada pengujian *Gorilla testing* disesuaikan dengan tingkat kelonggaran *error* sebesar 10% dari jumlah pengguna yang sesuai pada sistem. Pengaturan *Loop count* diatur sebanyak 100 kali untuk melakukan pengulangan *request* secara terus-menerus selama proses pengujian berlangsung. Konfigurasi ini digunakan untuk mengetahui kestabilan dan performa sistem ketika fitur utama dijalankan berulang kali dalam waktu tertentu.

Pengujian *Gorilla testing* difokuskan pada fitur utama yang sering digunakan oleh pengguna pada sistem Presensiku. Fitur yang diuji meliputi proses mengirim scan QR pada *Mobile* siswa, mengirim update data perizinan pada *Mobile* guru, mengakses halaman absensi kehadiran pada *website*, serta mengakses halaman absensi mapel pada *website*. Pengujian dilakukan secara berulang menggunakan Apache JMeter untuk mengetahui kemampuan sistem dalam menangani aktivitas pengguna secara terus-menerus pada fitur utama sistem.

a. Konfigurasi HTTP Request Mengirim Scan QR (*Mobile* Siswa)

Konfigurasi HTTP Request dilakukan untuk mensimulasikan proses pengiriman data scan QR Code pada aplikasi *Mobile* siswa dengan beban pengguna sebanyak 297 *user*. Pengaturan ini digunakan untuk mengetahui performa server saat menerima *request* presensi siswa secara berulang selama proses *Gorilla testing* berlangsung.



Gambar 4. 21 HTTP Request Mengirim Scan QR (*Mobile* Siswa)

Berdasarkan konfigurasi pada gambar 4.21, HTTP Request menggunakan protocol https dengan server presensiku.site. Method yang digunakan yaitu *POST* untuk mengirim data hasil scan QR Code ke server melalui *endpoint* /api/qattendance. Konfigurasi ini digunakan untuk mensimulasikan aktivitas presensi siswa melalui scan QR Code secara terus-menerus pada aplikasi *mobile*.

b. Konfigurasi HTTP Request Mengirim Update Data Perizinan (*Mobile* Guru)

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses pengiriman update data perizinan pada aplikasi *Mobile* guru menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menerima perubahan data perizinan secara berulang selama proses *Gorilla testing* berlangsung.

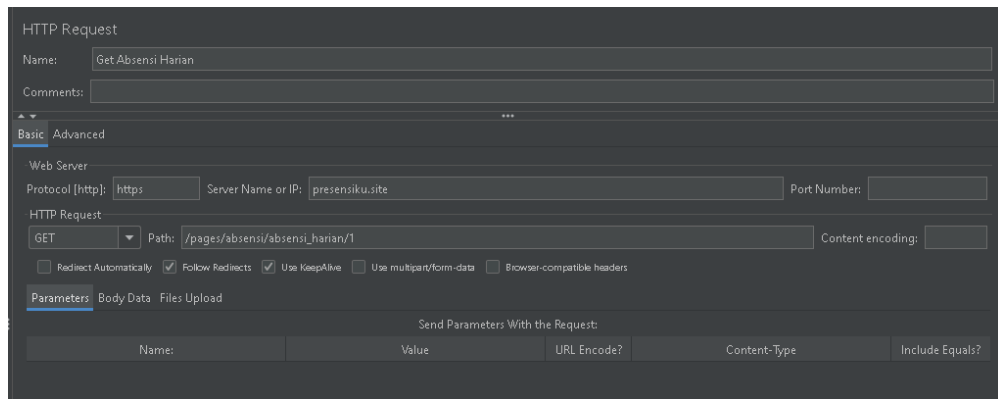
Name	Value	URL Encode?	Content-Type	Include Equals?
reason	sakit	☒	text/plain	☒
information	mohon maaf saya tidak dapat mengikuti mbk...	☒	text/plain	☒
date_permission	2026-05-22	☒	text/plain	☒
time_period	1	☒	text/plain	☒

Gambar 4. 22 HTTP Request Mengirim Update Data Perizinan (Mobile Guru)

Berdasarkan konfigurasi pada gambar 4.22, *HTTP Request* menggunakan protokol *https* dengan server *presensiku.site*. Method yang digunakan yaitu *POST* untuk mengirim data update perizinan ke server melalui *endpoint* yang telah ditentukan. Konfigurasi ini digunakan untuk mensimulasikan aktivitas guru saat melakukan update data perizinan pada aplikasi *Mobile* secara terus-menerus.

c. Konfigurasi *HTTP Request* Mengakses Halaman Absensi Kehadiran (*Website*)

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman absensi kehadiran pada *website* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menerima *request* akses halaman absensi kehadiran secara berulang selama proses *Gorilla testing* berlangsung.

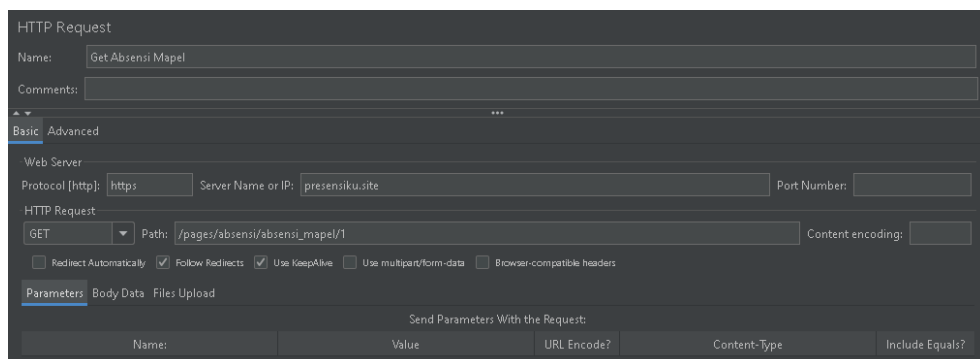


Gambar 4. 23 *HTTP Request* Mengakses Halaman Absensi Kehadiran (*Website*)

Berdasarkan konfigurasi pada gambar di atas, *HTTP Request* menggunakan protocol https dengan server presensiku.site. Method yang digunakan yaitu *GET* untuk mengambil data absensi kehadiran dari server melalui *endpoint* /pages/absensi/absensi_harian/1. Konfigurasi ini digunakan untuk mensimulasikan aktivitas admin saat mengakses halaman absensi kehadiran pada *website* secara terus-menerus.

d. Konfigurasi *HTTP Request* Mengakses Halaman Absensi Mapel (*Website*)

Konfigurasi *HTTP Request* dilakukan untuk mensimulasikan proses akses halaman absensi mapel pada *website* menggunakan Apache JMeter. Pengaturan ini digunakan untuk mengetahui performa server saat menerima *request* akses halaman absensi mapel secara berulang selama proses *Gorilla testing* berlangsung.



Gambar 4. 24 *HTTP Request* Mengakses Halaman Absensi Mapel (*Website*)

Berdasarkan konfigurasi pada gambar 4.24, *HTTP Request* menggunakan protocol https dengan server presensiku.site. Method yang

digunakan yaitu *GET* untuk mengambil data absensi mapel dari server melalui *endpoint* yang telah ditentukan. Konfigurasi ini digunakan untuk mensimulasikan aktivitas admin saat mengakses halaman absensi mapel pada *website* secara terus-menerus.

4.7. Test Execution

4.7.1. Load testing

Pengujian *load testing* dilakukan untuk mengetahui kemampuan sistem dalam menangani permintaan pengguna secara bersamaan. Pengujian ini menggunakan jumlah pengguna yang ditentukan berdasarkan tingkat kelonggaran *error* sebesar 10%, 5%, dan 1% dari total populasi pengguna sistem. Jumlah pengguna pada setiap tingkat kelonggaran *error* dihitung menggunakan Persamaan Rumus 2.5. Hanya lima parameter dianalisis *Load Time* (dari nilai minimum), *Response Time* (rata-rata), *Throughput*, *Error Rate*, dan *Latency* (diukur terpisah). Hasil pengujian diambil dari *Summary Report* dan *View Results in Table* pada JMeter.

Pengujian *load testing* dilakukan pada beberapa *platform* sebagai berikut:

a. Website

Pengujian dilakukan pada sistem berbasis *website* yang digunakan oleh admin untuk mengelola data absensi, jadwal, dan data siswa. Pengujian juga dilakukan dengan skenario di mana 1 *user* admin mengakses halaman melalui *website*, untuk melihat performa sistem saat terjadi akses. Hasil pengujian diperoleh dari tiga tampilan utama pada JMeter, yaitu *Summary Report*, *View Results in Table*, dan *View Results Tree*. *Summary Report* digunakan untuk melihat nilai *Load Time*, *Response Time*, *Throughput*, dan *Error Rate*. *View Results in Table* digunakan sebagai dasar perhitungan nilai *Latency*. Perhitungan lengkap *Latency* dicantumkan pada RTM melalui link <https://bit.ly/DokumenRTMPengujian>. Hasil pengujian *website* pada jmeter ada pada gambar 4.25 dan gambar 4.26.

Summary Report

Name:Summary Report

Comments:

Write results to file / Read from file

Filename

Browse...

Log/Display Only:

☐ Errors

☐ Successes

Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Get Token Logi...	1	4102	4102	4102	0.00	0.00%	14.6/min	1.79	0.03	7502.0
Post Login	1	2342	2342	2342	0.00	0.00%	25.6/min	18.90	0.79	45328.0
Get QR Code	1	207	207	207	0.00	0.00%	4.8/sec	211.82	4.01	44900.0
Get Absensi Ha...	1	4528	4528	4528	0.00	0.00%	13.3/min	585.20	0.19	2713363.0
Get Absensi Ma...	1	166	166	166	0.00	0.00%	6.0/sec	324.01	5.08	55077.0
Post kirim foto	1	37	37	37	0.00	100.00%	27.0/sec	40.28	25.13	1526.0
TOTAL	6	1897	37	4528	1883.89	16.67%	30.4/min	236.27	0.46	477949.3

Gambar 4. 25 Summary Report Load testing pada website

View Results in Table

Name:

View Results in Table

Comments:

Write results to file / Read from file

Filename

Browse...

Log/Display Only

☐ Errors

☐ Successes

Configure

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	21:38:32.826	Thread Group W...	Get Token Login	4102		7502	117	4045	3515
2	21:38:37.341	Thread Group W...	Post Login	2342		45328	1886	565	0
3	21:38:39.714	Thread Group W...	Get QR Code	207		44900	851	165	71
4	21:38:39.921	Thread Group W...	Get Absensi Harian	4528		2713363	870	613	0
5	21:38:44.450	Thread Group W...	Get Absensi Map...	166		55077	863	89	0
6	21:38:44.642	Thread Group W...	Post kirim foto	37		1526	952	37	0

Gambar 4. 26 View Results in Table Load testing pada Website

1) Post Login

Hasil pengujian pada gambar 4.25 dan 4.26 diketahui *Load Time* 2,342 detik, *Response Time* 2,342 detik, *Throughput* 25,6 request/menit, *Error Rate* 0%, dan *Latency* 0,565 detik. Proses *login* berhasil dilakukan tanpa kegagalan *request*. Nilai *Latency* yang lebih rendah dibandingkan *Respons Time* menunjukkan bahwa sebagian waktu digunakan untuk proses validasi data pengguna dan autentikasi pada server.

2) Get QR Code

Hasil pengujian pada gambar 4.25 dan 4.26 diketahui *Load Time* 0,207 detik, *Response Time* 0,207 detik, *Throughput* 4,8 request/menit, *Error Rate* 0%, dan *Latency* 0,165 detik. Pengambilan data QR Code berjalan dengan cepat dan stabil. Nilai *Respons Time* yang rendah menunjukkan bahwa sistem mampu menampilkan data QR Code dengan waktu tunggu yang singkat.

3) Get Absensi Harian

Hasil pengujian pada gambar 4.25 dan 4.26 diketahui *Load Time* 4,528 detik, *Response Time* 4,528 detik, *Throughput* 13,3 request/menit, *Error Rate* 0%, dan *Latency* 0,613 detik. Proses pengambilan data absensi harian berhasil

dilakukan tanpa *error*. Nilai *Respons Time* yang cukup tinggi menunjukkan bahwa sistem perlu memproses dan mengambil data absensi dalam jumlah yang lebih besar dibandingkan *request* lainnya.

4) Get Absensi Mapel

Hasil pengujian pada gambar 4.25 dan 4.26 diketahui *Load Time* 0,166 detik, *Response Time* 0,166 detik, *Throughput* 6 *request*/menit, *Error Rate* 0%, dan *Latency* 0,089 detik. Pengambilan data absensi mata pelajaran berjalan dengan cepat dan efisien. Nilai *Respons Time* dan *Latency* yang rendah menunjukkan bahwa data dapat ditampilkan dengan baik tanpa kendala performa.

5) Post Foto

Hasil pengujian pada gambar 4.25 dan 4.26 diketahui *Load Time* 0,037 detik, *Response Time* 0,037 detik, *Throughput* 27 *request*/menit, *Error Rate* 100%, dan *Latency* 0,037 detik. Proses pengiriman foto memiliki waktu *respons* yang sangat cepat. Namun, seluruh *request* tercatat gagal sehingga menghasilkan *Error Rate* sebesar 100%. Kegagalan tersebut perlu dianalisis berdasarkan hasil *View Results Tree* untuk mengetahui penyebab *error* yang terjadi selama proses pengujian.

Pengujian yang dilakukan pada 5 *test case* yang diuji, sebagian besar *request* berhasil diproses dengan baik tanpa *error*. Nilai *Response Time* tercepat diperoleh pada fitur Post Kirim Foto sebesar 0,037 detik dan nilai *Response Time* tertinggi terdapat pada fitur Get Absensi Harian sebesar 4,528 detik. Hasil pengujian menunjukkan bahwa fitur *login*, QR Code, absensi harian, dan absensi mata pelajaran mampu berjalan dengan stabil. *Error Rate* yang muncul pada pengujian dipengaruhi oleh kegagalan pada fitur Post Kirim Foto. Kegagalan tersebut terjadi karena *server* belum menggunakan VPS yang memadai untuk menangani proses pengiriman dan penyimpanan data foto. Akibatnya, permintaan unggah foto dari pengguna tidak dapat diproses secara optimal sehingga menghasilkan *responsrroor*.

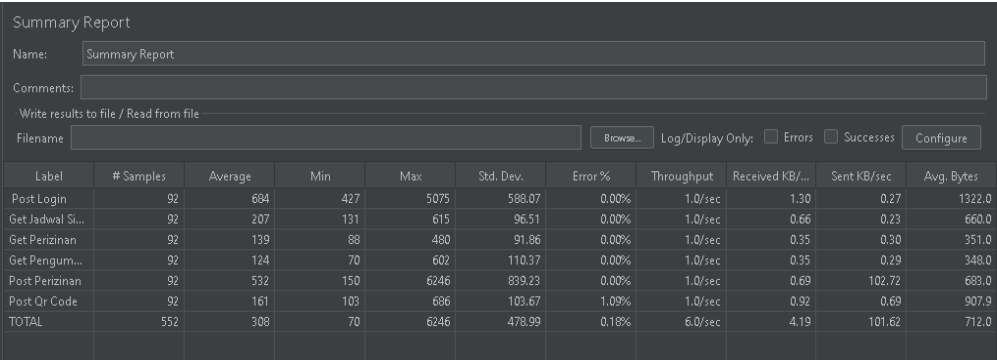
b. Mobile Siswa

Pengujian dilakukan pada aplikasi *Mobile* siswa dengan jumlah pengguna secara bersamaan sebanyak 92, 297, dan 1034 *user*, sesuai dengan tingkat kelonggaran *error* yang telah ditentukan. Hasil pengujian diperoleh dari tiga tampilan utama pada JMeter, yaitu *Summary Report*, *View Results in Table*, dan *View Results Tree*. *Summary Report* digunakan untuk melihat nilai *Load Time*, *Response Time*, *Throughput*, dan *Error Rate*. *View Results in Table* digunakan sebagai perhitungan nilai *Latency*. Perhitungan lengkap *Latency* dicantumkan pada RTM melalui link <https://bit.ly/DokumenRTMPengujian>.

Hasil pengujian pada *test case* ini menunjukkan performa sistem dalam menangani berbagai aktivitas siswa, seperti mengirim fungsi *login*, mengakses halaman absensi mata pelajaran, mengirim *scan QR Code*, mengakses halaman perizinan, mengirim data perizinan, mengakses data jadwal, serta mengakses data pengumuman. Setiap aktivitas diuji dengan mengirim *request* secara bersamaan untuk melihat kemampuan sistem dalam merespons permintaan dari pengguna *Mobile* siswa.

1) Hasil Pengujian *Mobile* Siswa 92 Sampel

Pengujian performa dilakukan pada aplikasi *Mobile* siswa dengan 92 *user*. Hasil pengujian *mobile* siswa dengan 92 sampel ada pada gambar 4.27 dan gambar 4.28.



Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
Post Login	92	684	427	5075	588.07	0.00%	1.0/sec	1.30	0.27	1322.0
Get Jadwal Si...	92	207	131	615	96.51	0.00%	1.0/sec	0.66	0.23	660.0
Get Perizinan	92	139	88	480	91.86	0.00%	1.0/sec	0.35	0.30	351.0
Get Pengum...	92	124	70	602	110.37	0.00%	1.0/sec	0.35	0.29	348.0
Post Perizinan	92	532	150	6246	839.23	0.00%	1.0/sec	0.69	102.72	683.0
Post Qr Code	92	161	103	686	103.67	1.09%	1.0/sec	0.92	0.69	907.9
TOTAL	552	308	70	6246	478.99	0.18%	6.0/sec	4.19	101.62	712.0

Gambar 4. 27 *Summary Reports Load testing* pada *mobile* siswa 92 sampel

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	07:29:53.975	Thread Group ...	Post Login	1283	✓	1322	274	1282	323
2	07:29:53.463	Thread Group ...	Post Login	1765	✓	1322	274	1784	838
3	07:29:54.985	Thread Group ...	Post Login	651	✓	1322	274	650	93
4	07:29:55.372	Thread Group ...	Get Jadwal Sis...	288	✓	660	233	288	46
5	07:29:55.402	Thread Group ...	Get Jadwal Sis...	258	✓	660	233	258	49
6	07:29:55.660	Thread Group ...	Get Perizinan	121	✓	351	294	121	0
7	07:29:55.636	Thread Group ...	Get Jadwal Sis...	227	✓	660	233	227	73
8	07:29:55.781	Thread Group ...	Get Pengumu...	103	✓	348	287	103	0
9	07:29:55.660	Thread Group ...	Get Perizinan	121	✓	351	294	121	0
10	07:29:55.863	Thread Group ...	Get Perizinan	131	✓	351	294	131	0
11	07:29:55.955	Thread Group ...	Get Pengumu...	76	✓	348	287	76	0
12	07:29:55.884	Thread Group ...	Post Perizinan	297	✓	683	101840	297	0
13	07:29:56.181	Thread Group ...	Post Qr Code	197	✓	880	686	197	0
14	07:29:55.994	Thread Group ...	Get Pengumu...	563	✓	348	287	563	0
15	07:29:56.032	Thread Group ...	Post Perizinan	636	✓	683	101847	636	0
16	07:29:56.557	Thread Group ...	Post Perizinan	312	✓	683	101868	312	0
17	07:29:56.668	Thread Group ...	Post Qr Code	205	✓	880	686	205	0
18	07:29:56.869	Thread Group ...	Post Qr Code	161	✓	880	686	161	0
19	07:29:55.973	Thread Group ...	Post Login	1113	✓	1322	274	1112	593
20	07:29:57.086	Thread Group ...	Get Jadwal Sis...	165	✓	660	233	165	72

Gambar 4. 28 *View Results in table Load testing pada mobile siswa 92 sampel*

a) *Post Login*

Hasil pengujian pada gambar 4.27 dan 4.28 diketahui *Load Time* 0,427 detik, *Response Time* 0,684 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,6821 detik. *Login* siswa berjalan lancar tanpa *error*. Nilai *Latency* pada proses *login* relatif lebih tinggi karena sistem perlu melakukan validasi data pengguna dan *autentikasi* sebelum pengguna dapat masuk ke aplikasi

b) *Get Jadwal Siswa*

Hasil pengujian pada gambar 4.27 dan 4.28 diketahui *Load Time* 0,131 detik, *Response Time* 0,207 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,2078 detik. Pengambilan data jadwal siswa berjalan cepat dan stabil. Hal ini menunjukkan bahwa sistem mampu menampilkan data jadwal dengan waktu tunggu yang rendah.

c) *Get perizinan*

Hasil pengujian pada gambar 4.27 dan 4.28 diketahui *Load Time* 0,088 detik, *Response Time* 0,139 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,1393 detik. Proses pengambilan data perizinan berjalan efisien karena *request* dapat diproses dengan cepat tanpa adanya kegagalan.

d) *Get pengumuman*

Hasil pengujian pada gambar 4.27 dan 4.28 diketahui *Load Time* 0,070 detik, *Response Time* 0,124 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,1247 detik. Pengambilan data pengumuman memiliki performa yang sangat cepat. Nilai *Load Time*, *Response Time*, dan *Latency* yang rendah menunjukkan bahwa fitur ini dapat diakses dengan baik oleh pengguna.

e) *Post perizinan*

Hasil pengujian pada gambar 4.27 dan 4.28 diketahui *Load Time* 0,150 detik, *Response Time* 0,532 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,5322 detik. Pengiriman data perizinan berjalan lancar, meskipun *Response Time* dan *Latency* lebih tinggi dibandingkan beberapa request *GET*. Hal ini wajar karena proses *Post Perizinan* melibatkan pengiriman data, validasi, dan penyimpanan data ke *server*.

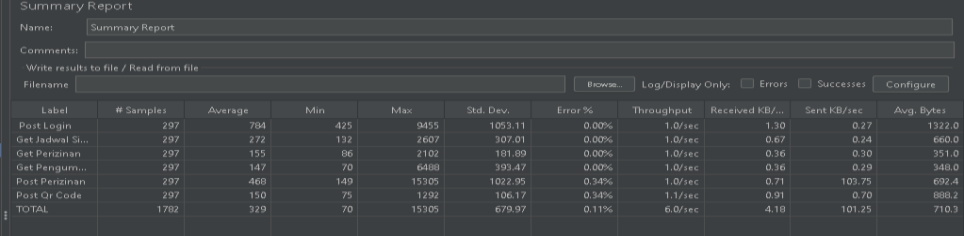
f) *Post QR Code*

Hasil pengujian pada gambar 4.27 dan 4.28 diketahui *Load Time* 0,103 detik, *Response Time* 0,161 detik, *Throughput* 1 request/s, *Error Rate* 1,09%, dan *Latency* 0,1578 detik. Proses *Post Qr Code* memiliki waktu respon yang cukup cepat. Namun, terdapat sebagian kecil request yang gagal, sehingga fitur *QR Code* perlu dievaluasi lebih lanjut agar prosesnya lebih stabil.

Pengujian dilakukan terhadap 92 virtual *user* yang menjalankan 6 *test case*, sehingga menghasilkan total 552 request selama proses *load testing*, *Load Time* terendah adalah 0,070 detik, rata-rata *Response Time* sebesar 0,308 detik, total *Throughput* sebesar 6 request/s, *Error Rate* total sebesar 0,18%, dan rata-rata *Latency* sebesar 0,3073 detik. Hasil ini menunjukkan bahwa sistem mampu menangani 92 siswa dengan performa yang baik dan stabil. Aplikasi *Mobile* siswa memenuhi harapan karena rata-rata *Respons Time* dan *Latency* masih berada di bawah satu detik, serta *Error Rate* total tergolong rendah. Meskipun demikian, fitur *Post Qr Code* masih perlu diperhatikan karena memiliki *Error Rate* sebesar 1,09%.

2) Hasil Pengujian *Mobile* Siswa 297 Sampel

Pengujian performa dilakukan pada aplikasi *Mobile* siswa dengan jumlah sampel sebanyak 297 pengguna. Hasil pengujian *mobile* siswa 297 sampel ada pada gambar 4.29 dan gambar 4.30.



Summary Report

Name: Summary Report

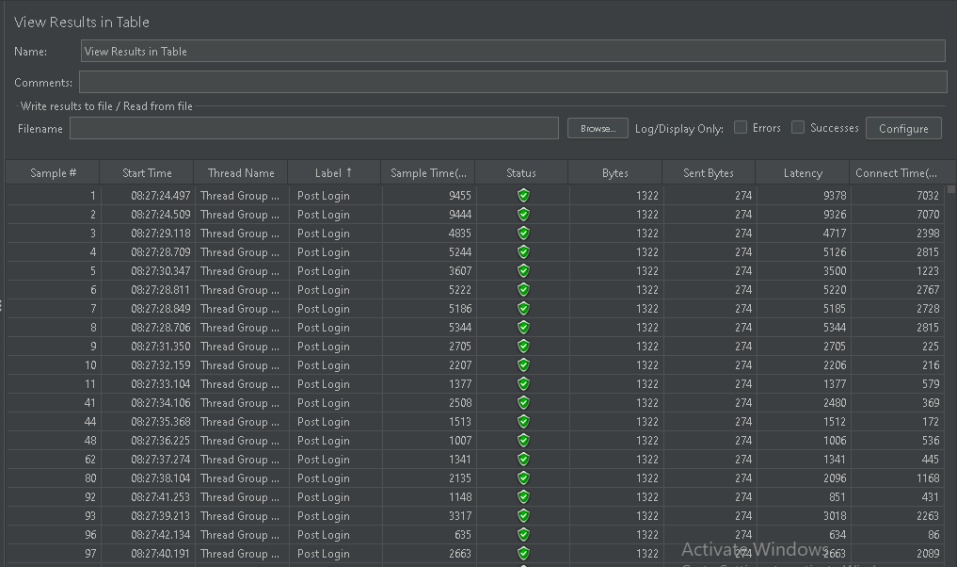
Comments:

Write results to file / Read from file

Filename: Log/Display Only: ☐ Errors ☐ Successes

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
Post Login	297	794	425	9455	1053.11	0.00%	1.0/sec	1.30	0.27	1322.0
Get Jadwal S...	297	272	132	2607	307.01	0.00%	1.0/sec	0.67	0.24	660.0
Get Penzina...	297	155	86	2102	181.89	0.00%	1.0/sec	0.36	0.30	351.0
Get Pengum...	297	147	70	6488	393.47	0.00%	1.0/sec	0.36	0.29	348.0
Post Penzina...	297	468	149	15305	1022.95	0.34%	1.0/sec	0.71	103.75	692.4
Post Qr Code	297	150	75	1292	106.17	0.34%	1.1/sec	0.91	0.70	888.2
TOTAL	1782	329	70	15305	679.97	0.11%	6.0/sec	4.18	101.25	710.3

Gambar 4. 29 *Summary Reports Load testing* pada *mobile* siswa 297 sampel



View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label 1	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	08:27:24.497	Thread Group ...	Post Login	9455	✓	1322	274	9378	7032
2	08:27:24.509	Thread Group ...	Post Login	9444	✓	1322	274	9326	7070
3	08:27:29.118	Thread Group ...	Post Login	4835	✓	1322	274	4717	2398
4	08:27:28.709	Thread Group ...	Post Login	5244	✓	1322	274	5126	2815
5	08:27:30.347	Thread Group ...	Post Login	3607	✓	1322	274	3500	1223
6	08:27:28.811	Thread Group ...	Post Login	5222	✓	1322	274	5220	2767
7	08:27:28.849	Thread Group ...	Post Login	5186	✓	1322	274	5185	2728
8	08:27:28.706	Thread Group ...	Post Login	5344	✓	1322	274	5344	2815
9	08:27:31.350	Thread Group ...	Post Login	2705	✓	1322	274	2705	225
10	08:27:32.159	Thread Group ...	Post Login	2207	✓	1322	274	2206	216
11	08:27:33.104	Thread Group ...	Post Login	1377	✓	1322	274	1377	579
41	08:27:34.106	Thread Group ...	Post Login	2508	✓	1322	274	2480	369
44	08:27:35.368	Thread Group ...	Post Login	1513	✓	1322	274	1512	172
48	08:27:36.225	Thread Group ...	Post Login	1007	✓	1322	274	1006	536
62	08:27:37.274	Thread Group ...	Post Login	1341	✓	1322	274	1341	445
80	08:27:38.104	Thread Group ...	Post Login	2135	✓	1322	274	2096	1168
92	08:27:41.253	Thread Group ...	Post Login	1148	✓	1322	274	851	431
93	08:27:39.213	Thread Group ...	Post Login	3317	✓	1322	274	3018	2263
96	08:27:42.134	Thread Group ...	Post Login	635	✓	1322	274	634	86
97	08:27:40.191	Thread Group ...	Post Login	2663	✓	1322	274	2663	2089

Gambar 4. 30 *View Results in table Load testing* pada *mobile* siswa 297 sampel

a) *Post Login*

Hasil pengujian pada gambar 4.29 dan 4.30 diketahui *Load Time* 0,425 detik, *Response Time* 0,788 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,7779 detik. Proses *login* siswa berjalan lancar tanpa adanya *error*. Nilai *Response Time* dan *Latency* pada *test case* ini relatif lebih tinggi dibandingkan beberapa *test case* lainnya karena proses *login* melibatkan validasi data pengguna dan *autentikasi* sebelum pengguna dapat masuk ke dalam aplikasi.

b) *Get Jadwal*

Hasil pengujian pada gambar 4.29 dan 4.30 diketahui *Load Time* 0,132 detik, *Response Time* 0,272 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,2712 detik. Pengambilan data jadwal siswa berjalan dengan cukup cepat dan stabil. Tidak adanya *error* menunjukkan bahwa sistem mampu menampilkan data jadwal dengan baik pada pengujian 297 sampel.

c) *Get Perizinan*

Hasil pengujian pada gambar 4.29 dan 4.30 diketahui *Load Time* 0,086 detik, *Response Time* 0,155 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,1548 detik. Proses pengambilan data perizinan berjalan efisien karena waktu respon dan *Latency* yang dihasilkan tergolong rendah.

d) *Get pengumuman*

Hasil pengujian pada gambar 4.29 dan 4.30 diketahui *Load Time* 0,070 detik, *Response Time* 0,147 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,1473 detik. Fitur pengumuman memiliki performa yang sangat baik karena nilai *Load Time*, *Response Time*, dan *Latency* tergolong rendah.

e) *Post Perizinan*

Hasil pengujian pada gambar 4.29 dan 4.30 diketahui *Load Time* 0,149 detik, *Response Time* 0,468 detik, *Throughput* 1 request/s, *Error Rate* 0,34%, dan *Latency* 0,4675 detik. Pengiriman data perizinan masih dapat berjalan dengan cukup baik, meskipun terdapat sedikit *error*. Nilai *Response Time* dan *Latency* lebih tinggi dibandingkan *request GET* karena proses ini melibatkan pengiriman data, validasi, dan penyimpanan data ke *server*.

f) *Post QR Code*

Hasil pengujian pada gambar 4.29 dan 4.30 diketahui *Load Time* 0,075 detik, *Response Time* 0,150 detik, *Throughput* 1,1 request/s, *Error Rate* 0,34%, dan *Latency* 0,1501 detik. Proses *Post Qr Code* memiliki waktu respon yang cukup cepat. Namun, masih terdapat sedikit kegagalan *request* yang ditunjukkan melalui *Error Rate* sebesar 0,34%, sehingga fitur ini tetap perlu diperhatikan pada tahap evaluasi sistem.

Pengujian dilakukan terhadap 297 virtual *user* dengan 6 *test case* yang dijalankan pada setiap virtual *user*, sehingga total *request* yang diproses selama

pengujian sebanyak 1.782 *request*. *Load Time* terendah adalah 0,070 detik, rata-rata *Respons Time* keseluruhan sebesar 0,329 detik, total *Throughput* sebesar 6 *request/s*, *Error Rate* total sebesar 0,11%, dan rata-rata *Latency* sebesar 0,3281 detik. Hasil ini menunjukkan bahwa sistem mampu menangani 297 siswa dengan performa yang cukup baik dan stabil. Aplikasi *Mobile* siswa dapat dinyatakan memenuhi harapan karena rata-rata *Respons Time* dan *Latency* masih berada di bawah satu detik. Selain itu, *Error Rate* total hanya sebesar 0,11%, sehingga tingkat kegagalan *request* tergolong rendah. Fitur *Post Perizinan* dan *Post Qr Code* masih perlu dilakukan pemantauan karena kedua *test case* tersebut memiliki *Error Rate* sebesar 0,34%.

3) Hasil Pengujian *Mobile* Siswa 1034 Sampel

Pengujian performa dilakukan pada aplikasi *Mobile* siswa dengan jumlah sampel sebanyak 1034 pengguna. Hasil pengujian *mobile* siswa 1034 sampel ada pada gambar 4.31 dan gambar 4.32.

Summary Report

Name:

Summary Report

Comments:

Write results to file / Read from file

Filename

Browse...

Log/Display Only:

☐ Errors

☐ Successes

Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
Post Login	1034	703	426	6199	523.69	0.00%	1.0/sec	1.29	0.27	1322
Get Jadwal Si...	1034	329	131	3545	403.54	0.00%	1.0/sec	0.65	0.23	660
Get Perizinan	1034	158	4	1352	114.17	0.10%	1.0/sec	0.34	0.29	352
Get Pengum...	1034	134	1	794	98.27	0.29%	1.0/sec	0.35	0.28	353
Post Perizinan	1034	412	148	4095	328.75	0.00%	1.0/sec	0.67	99.54	604
Post Qr Code	1034	188	2	2070	138.49	0.10%	1.0/sec	0.86	0.67	881
TOTAL	6204	321	1	6199	369.75	0.08%	6.0/sec	4.15	101.16	709

Gambar 4. 31 *Summary Reports Load testing* pada *mobile* siswa 1034 sampel

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	13:35:55.413	Thread Group ...	Post Login	609	✓	1322	274	609	136
2	13:35:56.025	Thread Group ...	Get Jadwal Sis...	175	✓	660	233	175	70
3	13:35:56.200	Thread Group ...	Get Perizinan	120	✓	351	294	120	0
4	13:35:56.320	Thread Group ...	Get Pengumu...	88	✓	348	287	88	0
5	13:35:56.409	Thread Group ...	Post Perizinan	280	✓	684	101875	280	0
6	13:35:56.234	Thread Group ...	Post Login	534	✓	1322	274	534	102
7	13:35:56.690	Thread Group ...	Post Qr Code	169	✓	880	686	169	0
8	13:35:56.772	Thread Group ...	Get Jadwal Sis...	711	✓	660	233	710	627
9	13:35:57.483	Thread Group ...	Get Perizinan	109	✓	351	294	109	0
10	13:35:57.593	Thread Group ...	Get Pengumu...	79	✓	348	287	79	0
11	13:35:57.225	Thread Group ...	Post Login	820	✓	1322	274	819	77
12	13:35:58.047	Thread Group ...	Get Jadwal Sis...	177	✓	660	233	177	73
13	13:35:58.224	Thread Group ...	Get Perizinan	116	✓	351	294	116	0
14	13:35:58.340	Thread Group ...	Get Pengumu...	109	✓	348	287	109	0
15	13:35:57.673	Thread Group ...	Post Perizinan	820	✓	684	101833	820	0
16	13:35:58.495	Thread Group ...	Post Qr Code	162	✓	880	686	162	0
17	13:35:58.450	Thread Group ...	Post Perizinan	227	✓	684	101875	227	0
18	13:35:58.233	Thread Group ...	Post Login	584	✓	1322	274	584	57
19	13:35:58.677	Thread Group ...	Post Qr Code	162	✓	880	686	162	0
20	13:35:58.818	Thread Group ...	Get Jadwal Sis...	152	✓	660	233	152	57

Gambar 4. 32 *View Results in table Load testing pada mobile siswa 1034 sampel*

a) *Post Login*

Hasil pengujian pada gambar 4.31 dan 4.32 diketahui *Load Time* 0,426 detik, *Response Time* 0,703 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,6973 detik. Proses *login* siswa berjalan dengan baik tanpa *error*. Nilai *Response Time* dan *Latency* pada *test case* ini relatif lebih tinggi dibandingkan beberapa fitur lainnya karena proses *login* melibatkan validasi data pengguna dan *autentikasi* sebelum siswa dapat masuk ke dalam aplikasi.

b) *Get Jadwal*

Hasil pengujian pada gambar 4.31 dan 4.32 diketahui *Load Time* 0,131 detik, *Response Time* 0,329 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,3293 detik. Pengambilan data jadwal siswa berjalan stabil tanpa *error*. Hal ini menunjukkan bahwa sistem mampu menampilkan data jadwal siswa dengan baik pada pengujian 1034 sampel.

c) *Get Perizinan*

Hasil pengujian pada gambar 4.31 dan 4.32 diketahui *Load Time* 0,004 detik, *Response Time* 0,158 detik, *Throughput* 1 request/s, *Error Rate* 0,10%, dan *Latency* 0,1588 detik. Proses pengambilan data perizinan berjalan cukup cepat dengan waktu respon yang rendah. Meskipun terdapat sedikit *error*, nilai

tersebut masih tergolong kecil sehingga tidak terlalu memengaruhi performa keseluruhan sistem.

d) *Get* pengumuman

Hasil pengujian pada gambar 4.31 dan 4.32 diketahui *Load Time* 0,001 detik, *Response Time* 0,134 detik, *Throughput* 1 request/s, *Error Rate* 0,29%, dan *Latency* 0,1344 detik. Pengambilan data pengumuman memiliki waktu respon yang cepat. Namun, terdapat *Error Rate* sebesar 0,29%, sehingga fitur ini tetap perlu dipantau untuk memastikan seluruh *request* dapat diproses secara stabil.

e) *Post* Perizinan

Hasil pengujian pada gambar 4.31 dan 4.32 diketahui *Load Time* 0,148 detik, *Response Time* 0,412 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,4127 detik. Pengiriman data perizinan berjalan dengan baik tanpa *error*. Nilai *Response Time* dan *Latency* lebih tinggi dibandingkan *request GET* karena proses ini melibatkan pengiriman data, validasi, dan pemrosesan data pada *server*.

f) *Post QR Code*

Hasil pengujian pada gambar 4.31 dan 4.32 diketahui *Load Time* 0,002 detik, *Response Time* 0,188 detik, *Throughput* 1 request/s, *Error Rate* 0,10%, dan *Latency* 0,1871 detik. Proses *Post Qr Code* memiliki waktu respon yang cukup cepat. Meskipun terdapat sedikit *error*, nilai *Error Rate* masih rendah sehingga tidak terlalu memengaruhi hasil pengujian secara keseluruhan.

Pengujian dilakukan oleh 1.034 virtual *user* yang menjalankan 6 *test case*, sehingga menghasilkan total 6.204 *request* selama proses *load testing*. *Load Time* terendah adalah 0,001 detik, rata-rata *Response Time* keseluruhan sebesar 0,321 detik, total *Throughput* sebesar 6 request/s, *Error Rate* total sebesar 0,08%, dan rata-rata *Latency* sebesar 0,3199 detik. Hasil ini menunjukkan bahwa sistem mampu menangani 1034 sampel dengan performa yang cukup baik dan stabil.

Fitur *Post Login*, *Get* Jadwal Siswa, *Get* Perizinan, *Get* Pengumuman, *Post* Perizinan, dan *Post Qr Code* dapat diproses dengan baik. Nilai rata-rata

Response Time dan *Latency* masih berada di bawah satu detik, sedangkan *Error Rate* total hanya sebesar 0,08%.

c. *Mobile Guru*

Pengujian dilakukan pada aplikasi *Mobile guru* dengan jumlah pengguna secara bersamaan sebanyak 38, 52, dan 59 *user*, sesuai dengan tingkat kelonggaran *error* yang telah ditentukan. Hasil pengujian diperoleh dari tiga tampilan utama pada JMeter, yaitu *Summary Report*, *View Results in Table*, dan *View Results Tree*. *Summary Report* digunakan untuk melihat nilai *Load Time*, *Response Time*, *Throughput*, dan *Error Rate*. *View Results in Table* digunakan sebagai dasar perhitungan nilai *Latency*. Perhitungan lengkap *Latency* dicantumkan pada RTM melalui link <https://bit.ly/DokumenRTMPengujian>, sedangkan *View Results Tree* digunakan untuk melihat detail *request* dan *respons*, terutama pada *test case* yang mengalami *error*. Melalui *View Results Tree*, penyebab kegagalan *request* dapat dianalisis berdasarkan *Response message*, *Response Code*, atau informasi *error* yang ditampilkan oleh *server*.

Hasil pengujian pada *test case* ini menunjukkan performa sistem dalam menangani berbagai aktivitas guru, seperti mengirim fungsi *login*, mengakses data jadwal mengajar, mengakses data kelas, mengakses data absensi siswa, mengirim data kehadiran siswa, mengakses data perizinan siswa, serta mengirim atau memproses data yang berkaitan dengan aktivitas pembelajaran. Setiap aktivitas diuji dengan mengirim *request* secara bersamaan untuk melihat kemampuan sistem dalam merespons permintaan dari pengguna *Mobile guru*.

1) Hasil Pengujian *Mobile Guru* 38 Sampel

Pengujian performa dilakukan pada aplikasi *Mobile guru* dengan jumlah sampel sebanyak 38 pengguna. Hasil pengujian *mobileI guru* 38 sampel ada pada gambar 4.33 sampai 4.35.

Summary Report

Name: Summary Report

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes ☐ Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
Post Login	38	986	527	2594	437.43	0.00%	1.0/sec	1.02	0.30	1039.9
Get Jadwal Pe...	38	564	136	3220	622.90	0.00%	1.0/sec	0.51	0.27	522.0
Get jadwal Ke...	38	166	78	528	107.11	0.00%	1.0/sec	0.34	0.26	344.0
Get Riwayat ...	38	208	92	731	138.28	0.00%	1.0/sec	0.31	0.28	313.0
Get Riwayat P...	38	9743	1393	22442	6547.55	2.63%	41.7/min	378.09	0.20	556473.8
Post Laporkan	38	602	140	5271	890.22	100.00%	42.7/min	0.38	0.30	547.0
Post perizinan	38	191	96	1051	161.79	100.00%	42.9/min	0.24	0.24	350.0
Post buat pe...	38	691	484	1023	154.95	0.00%	42.5/min	0.42	0.30	612.8
TOTAL	342	1533	78	22442	3678.64	22.51%	6.0/sec	365.38	1.94	62299.2

Gambar 4. 33 Summary Reports Load testing pada mobile guru 38 sampel

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes ☐ Configure

Sample #	Start Time	Thread Name	Label	Sample Time...	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	23:31:01.677	Thread Group ...	Post Login	696	✓	1039	309	695	226
2	23:31:02.374	Thread Group ...	Get Jadwal Per...	173	✓	522	273	173	62
3	23:31:02.547	Thread Group ...	Get jadwal Kelas	97	✓	344	267	97	0
4	23:31:02.644	Thread Group ...	Get Riwayat A...	238	✓	313	288	237	0
5	23:31:02.582	Thread Group ...	Post Login	586	✓	1039	309	585	50
6	23:31:03.168	Thread Group ...	Get Jadwal Per...	152	✓	522	273	152	56
7	23:31:03.320	Thread Group ...	Get jadwal Kelas	106	✓	344	267	106	0
8	23:31:03.426	Thread Group ...	Get Riwayat A...	144	✓	313	288	143	0
9	23:31:03.503	Thread Group ...	Post Login	531	✓	1039	309	530	76
10	23:31:04.034	Thread Group ...	Get Jadwal Per...	170	✓	522	273	170	78
11	23:31:02.882	Thread Group ...	Get Riwayat A...	1404	✓	571457	307	755	77
12	23:31:04.205	Thread Group ...	Get jadwal Kelas	96	✓	344	267	89	0
13	23:31:04.286	Thread Group ...	Get Riwayat Pe...	166	✓	490	331	165	56
14	23:31:04.302	Thread Group ...	Get Riwayat A...	431	✓	313	288	431	0
15	23:31:04.452	Thread Group ...	Post Laporkan	403	✗	547	415	403	291
16	23:31:04.656	Thread Group ...	Post perizinan	100	✗	350	342	100	0
17	23:31:03.570	Thread Group ...	Get Riwayat A...	1393	✓	571457	307	751	55
18	23:31:04.864	Thread Group ...	Get Riwayat Pe...	301	✓	489	331	301	50
19	23:31:04.509	Thread Group ...	Post Login	878	✓	1040	309	878	382
20	23:31:05.266	Thread Group ...	Post Laporkan	226	✗	547	415	226	58

Gambar 4. 34 View Results in table Load testing pada mobile guru 38 sampel

Text

Sampler result Request Response data

Response Body Response headers

Find Case sensitive Regular exp.

```
{
  "message": "App\\Services\\NotificationHelperService::tokenTemplate(): Argument #1 ($token) must be of type string, null given, called in /hor
  \"exception\": \"TypeError\",
  \"file\": \"/home/pres2446/sas/app/Services/NotificationHelperService.php\",
  \"line\": 50,
  \"trace\": [
    {
      \"file\": \"/home/pres2446/sas/app/Services/NotificationHelperService.php\",
      \"line\": 199,
      \"function\": \"tokenTemplate\",
      \"class\": \"App\\Services\\NotificationHelperService\",
      \"type\": \"->\"
    },
    {
      \"file\": \"/home/pres2446/sas/app/Http/Controllers/api/AttendanceController.php\",
      \"line\": 942,
      \"function\": \"templatePermissionApproval\",
      \"class\": \"App\\Services\\NotificationHelperService\",
      \"type\": \"->\"
    }
  ]
}
```

Gambar 4. 35 View Results Tree Load testing pada mobile guru 38 sampel

a) Post Login

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,527 detik, *Response Time* 0,986 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,9758 detik. Proses login guru berjalan dengan baik tanpa adanya

error. Nilai *Response Time* dan *Latency* pada proses *login* relatif lebih tinggi karena sistem perlu melakukan validasi data pengguna dan *autentikasi* sebelum guru dapat masuk ke dalam aplikasi.

b) *Get Jadwal Personal*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,136 detik, *Response Time* 0,564 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,5643 detik. Pengambilan data jadwal personal guru berjalan cukup stabil. Tidak adanya *error* menunjukkan bahwa sistem mampu menampilkan data jadwal personal dengan baik pada pengujian 38 sampel.

c) *Get Jadwal Kelas*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,078 detik, *Response Time* 0,166 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,1661 detik. Proses pengambilan jadwal kelas berjalan cepat dan stabil. Nilai *Load Time*, *Response Time*, dan *Latency* yang rendah menunjukkan bahwa fitur ini dapat diproses dengan baik oleh sistem.

d) *Get Riwayat Absensi Mapel*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,092 detik, *Response Time* 0,208 detik, *Throughput* 1 request/s, *Error Rate* 0%, dan *Latency* 0,092 detik. Pengambilan data riwayat absensi mata pelajaran berjalan dengan baik. Nilai *error* 0% menunjukkan bahwa seluruh *request* pada *test case* ini berhasil diproses.

e) *Get Riwayat Absensi Kelas*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 1,393 detik, *Response Time* 9,743 detik, *Throughput* 41,7 request/min, *Error Rate* 2,63%, dan *Latency* 1,5156 detik. *Test case* ini memiliki *Response Time* paling tinggi dibandingkan *test case* lainnya. Hal ini menunjukkan bahwa proses pengambilan riwayat absensi siswa membutuhkan waktu lebih lama, kemungkinan karena data yang diakses lebih besar atau proses *query* lebih berat. Terdapat *Error Rate* sebesar 2,63%, sehingga fitur ini perlu menjadi perhatian pada tahap evaluasi sistem.

f) *Get Riwayat Perizinan*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,156 detik, *Response Time* 0,643 detik, *Throughput* 42,7 request/min, *Error Rate* 0%, dan *Latency* 0,6434 detik. Pengambilan data riwayat perizinan berjalan cukup baik dan tidak terdapat *error*. Meskipun nilai *Latency* lebih tinggi dibandingkan beberapa *request GET* lainnya, sistem masih mampu memproses *request* dengan stabil.

g) *Post Laporkan*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,096 detik, *Response Time* 0,191 detik, *Throughput* 42,9 request/min, *Error Rate* 100%, dan *Latency* 0,1941 detik. Waktu respon pada *test case* ini tergolong cepat, namun *Error Rate* sebesar 100% muncul karena proses *Post Perizinan* membutuhkan token FCM untuk mengirimkan notifikasi ke perangkat siswa dapat dilihat pada gambar 4.35. Pengujian menggunakan JMeter, *request* dikirim melalui virtual *user* sehingga tidak memiliki token FCM dari perangkat asli. Akibatnya, server gagal mengirimkan notifikasi dan *request* tercatat sebagai *error*. Keterangan *error* ini diperkuat melalui hasil *View Results Tree*.

h) *Post Perizinan*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,140 detik, *Response Time* 0,602 detik, *Throughput* 42,7 request/min, *Error Rate* 100%, dan *Latency* 0,6134 detik. *Test case* ini juga mengalami *Error Rate* sebesar 100% karena proses *Laporkan* membutuhkan token FCM untuk mengirimkan notifikasi kepada siswa dapat dilihat pada gambar 4.35. Karena pengujian dilakukan secara virtual melalui JMeter dan tidak menggunakan perangkat *Mobile* siswa yang memiliki token FCM, server tidak dapat mengirimkan notifikasi. *Error* pada *test case* ini lebih disebabkan oleh keterbatasan kondisi pengujian virtual, bukan karena lambatnya performa sistem.

i) *Post Pengumuman*

Hasil pengujian pada gambar 4.33 dan 4.34 diketahui *Load Time* 0,484 detik, *Response Time* 0,484 detik, *Throughput* 42,5 request/min, *Error Rate* 0%, dan *Latency* 0,6950 detik. Proses pembuatan pengumuman berjalan dengan baik

tanpa adanya *error*. Meskipun *Response Time* dan *Latency* lebih tinggi dibandingkan beberapa *request GET*, fitur ini tetap dapat dikatakan stabil karena seluruh *request* berhasil diproses.

Pengujian melibatkan 38 virtual *user* yang menjalankan 9 *test case*, sehingga menghasilkan total 342 *request* selama proses *load testing*. Rata-rata *Response Time* keseluruhan sebesar 1,533 detik, total *Throughput* sebesar 6,0 *request/s*, *Error Rate* total sebesar 22,51%, dan rata-rata *Latency* sebesar 1,8570 detik. Hasil ini menunjukkan bahwa sebagian besar fitur *Mobile* guru masih dapat berjalan dengan cukup baik, terutama fitur dengan *Error Rate* 0%. Namun, nilai *Error Rate* total cukup tinggi karena terdapat dua *test case* dengan *Error Rate* 100%, yaitu *Post Laporkan* dan *Post Perizinan*. Selain itu, fitur *Get Riwayat Absensi Siswa* juga perlu diperhatikan karena memiliki *Response Time* dan *Latency* paling tinggi dibandingkan fitur lainnya.

2) Hasil Pengujian *Mobile* Guru 52 Sampel

Pengujian performa dilakukan pada aplikasi *Mobile* guru dengan jumlah sampel sebanyak 52 sampel. Hasil pengujian *mobile* guru 52 sampel ada pada gambar 4.36 sampai 4.38.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Post Login	52	3005	481	6066	1643.65	0.00%	55.4/min	0.94	0.28	1039.9
Get Jadwal Pe...	52	825	143	1993	465.10	0.00%	54.7/min	0.46	0.24	522.0
Get Jadwal Ke...	52	710	81	1823	437.80	0.00%	54.0/min	0.30	0.23	344.0
Get Riwayat ...	52	892	121	2291	503.70	0.00%	53.0/min	0.27	0.25	313.0
Get Riwayat ...	52	8950	1525	16115	4524.96	0.00%	43.6/min	405.43	0.22	571457.3
Get Riwayat P...	52	1031	175	5370	843.35	0.00%	44.6/min	0.36	0.24	490.0
Post Laporkan	52	953	184	3457	740.01	100.00%	44.6/min	0.57	0.32	790.4
Post perizinan	52	714	103	1892	533.52	100.00%	44.8/min	0.26	0.25	350.0
Post buat pe...	52	1498	500	3214	814.38	0.00%	44.6/min	0.44	0.31	612.9
TOTAL	468	2064	81	16115	3046.66	22.22%	64/sec	397.67	2.06	63991.1

Gambar 4. 36 *Summary Reports Load testing* pada *mobile* guru 52 sampel

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes

Sample # ↑	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	00:03:43.872	Thread Group ...	Post Login	481	✓	1039	309	481	67
2	00:03:44.354	Thread Group ...	Get Jadwal Per...	189	✓	522	273	189	72
3	00:03:44.544	Thread Group ...	Get jadwal Kelas	101	✓	344	267	101	0
4	00:03:44.644	Thread Group ...	Get Riwayat A...	123	✓	313	288	122	0
5	00:03:44.808	Thread Group ...	Post Login	618	✓	1039	309	617	71
6	00:03:45.426	Thread Group ...	Get Jadwal Per...	143	✓	522	273	142	55
7	00:03:45.569	Thread Group ...	Get jadwal Kelas	111	✓	344	267	111	0
8	00:03:45.681	Thread Group ...	Get Riwayat A...	211	✓	313	288	210	0
9	00:03:45.825	Thread Group ...	Post Login	516	✓	1039	309	516	58
10	00:03:44.767	Thread Group ...	Get Riwayat A...	1769	✓	571457	307	879	55
11	00:03:46.342	Thread Group ...	Get Jadwal Per...	196	✓	522	273	196	55
12	00:03:46.538	Thread Group ...	Get jadwal Kelas	81	✓	344	267	81	0
13	00:03:46.619	Thread Group ...	Get Riwayat A...	184	✓	313	288	184	0
14	00:03:46.537	Thread Group ...	Get Riwayat Pe...	271	✓	490	331	271	49
15	00:03:46.808	Thread Group ...	Post Laporan	426	✗	547	435	426	51
16	00:03:47.236	Thread Group ...	Post perizinan	176	✗	350	342	176	0
17	00:03:45.892	Thread Group ...	Get Riwayat A...	1525	✓	571457	307	931	55
18	00:03:46.828	Thread Group ...	Post Login	666	✓	1040	309	666	56
19	00:03:47.418	Thread Group ...	Get Riwayat Pe...	351	✓	490	331	351	57
20	00:03:47.495	Thread Group ...	Get Jadwal Per...	298	✓	522	298	298	61

Activate Windows
Go to Settings to activate Windows.

Gambar 4. 37 View Results in table Load testing pada guru siswa 52 sampel

View Results Tree

Name: View Results Tree

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes

Search: Case sensitive ☐ Regular exp. ☐ Search

Text	Sampler result	Request	Response data
Post perizinan	✓		
Post buat pengumuman	✓		
Post buat pengumuman	✓		
Post buat pengumuman	✓		
Post buat pengumuman	✓		
Post buat pengumuman	✓		
Post buat pengumuman	✓		
Post Laporan	✗		
Post perizinan	✗		
Post buat pengumuman	✓		
Get Riwayat Absensi Guru	✓		
Get Riwayat Perizinan	✓		
Post Laporan	✗		
Post buat pengumuman	✓		
Get Riwayat Absensi Guru	✓		
Get Riwayat Perizinan	✓		
Post Laporan	✗		

Thread Name: Thread Group 1-41
Sample Start: 2026-05-29 23:51:07 GMT+07:00
Load time: 239
Connect Time: 75
Latency: 239
Size in bytes: 13205
Sent bytes: 436
Header size in bytes: 475
Body size in bytes: 12730
Sample Count: 1
Error Count: 1
Data type ("text"/"bin"/""): text
Response code: 500
Response message: Internal Server Error
HTTPSampleResult fields:
ContentType: application/json
DataEncoding: null

Gambar 4. 38 View Results Tree Load testing pada mobile guru 52 sampel

a) Post Login

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,481 detik, *Response Time* 3,005 detik, *Throughput* 55,4 request/min, *Error Rate* 0%, dan *Latency* 3,0007 detik. Proses *login* guru berjalan tanpa adanya *error*. Namun, nilai *Response Time* dan *Latency* pada *test case* ini cukup tinggi karena proses *login* melibatkan validasi data pengguna dan autentikasi sebelum guru dapat masuk ke dalam aplikasi.

b) Get Jadwal Personal

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,143 detik, *Response Time* 0,825 detik, *Throughput* 54,7 request/min, *Error Rate* 0%, dan *Latency* 0,8250 detik. Pengambilan data jadwal personal guru berjalan cukup baik dan tidak mengalami *error*. Hal ini menunjukkan bahwa sistem masih mampu menampilkan data jadwal personal dengan stabil pada pengujian 52 sampel.

c) *Get Jadwal Kelas*

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,081 detik, *Response Time* 0,710 detik, *Throughput* 54,0 request/min, *Error Rate* 0%, dan *Latency* 0,7110 detik. Proses pengambilan jadwal kelas berjalan dengan baik. Nilai *error* 0% menunjukkan bahwa seluruh *request* pada *test case* ini berhasil diproses oleh sistem.

d) *Get Riwayat Absensi Mapel*

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,121 detik, *Response Time* 0,892 detik, *Throughput* 53,0 request/min, *Error Rate* 0%, dan *Latency* 0,8919 detik. Pengambilan data riwayat absensi mata pelajaran berjalan stabil tanpa adanya kegagalan *request*. Nilai *Response Time* dan *Latency* masih tergolong dapat diterima karena tetap berada di bawah satu detik.

e) *Get Riwayat Absensi Kelas*

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 1,525 detik, *Response Time* 8,950 detik, *Throughput* 43,6 request/min, *Error Rate* 0%, dan *Latency* 7,0878 detik. *Test case* ini memiliki nilai *Response Time* dan *Latency* paling tinggi dibandingkan *test case* lainnya. Hal ini menunjukkan bahwa proses pengambilan riwayat absensi siswa membutuhkan waktu lebih lama, karena jumlah data yang diakses lebih besar atau proses *query* lebih berat. Meskipun demikian, nilai *Error Rate* 0% menunjukkan bahwa *request* tetap berhasil diproses.

f) *Get Riwayat Perizinan*

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,175 detik, *Response Time* 1,031 detik, *Throughput* 44,6 request/min, *Error Rate* 0%, dan *Latency* 1,0310 detik. Pengambilan data riwayat perizinan berjalan tanpa

error. Namun, nilai *Respons Time* dan *Latency* sudah berada di atas satu detik, sehingga fitur ini perlu diperhatikan apabila jumlah data yang diproses semakin besar.

g) *Post Laporkan*

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,103 detik, *Response Time* 0,714 detik, *Throughput* 44,8 request/min, *Error Rate* 100%, dan *Latency* 0,7143 detik. Waktu respon pada *test case* ini masih tergolong cukup cepat, tetapi *Error Rate* sebesar 100% menunjukkan bahwa seluruh *request* tercatat gagal. Berdasarkan keterangan *error* pada *View Results Tree*, kegagalan ini terjadi karena proses *Post Perizinan* membutuhkan token FCM untuk mengirimkan notifikasi ke perangkat siswa dapat dilihat pada gambar 4.38. Pada pengujian menggunakan JMeter, *request* dikirim melalui virtual *user* sehingga tidak memiliki token FCM dari perangkat asli. Akibatnya, server gagal mengirimkan notifikasi kepada siswa dan *request* tercatat sebagai *error*.

h) *Post Perizinan*

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,184 detik, *Response Time* 0,953 detik, *Throughput* 44,6 request/min, *Error Rate* 100%, dan *Latency* 0,9528 detik. *Test case* ini juga mengalami *Error Rate* sebesar 100% karena proses *Laporkan* membutuhkan token FCM untuk mengirimkan notifikasi kepada siswa dapat dilihat pada gambar 4.38. Karena pengujian dilakukan secara virtual melalui JMeter dan tidak menggunakan perangkat *Mobile* siswa yang memiliki token FCM, server tidak dapat mengirimkan notifikasi. Oleh karena itu, *error* pada *test case* ini lebih disebabkan oleh keterbatasan kondisi pengujian virtual, bukan karena lambatnya performa sistem.

i) *Post Pengumuman*

Hasil pengujian pada gambar 4.36 dan 4.37 diketahui *Load Time* 0,500 detik, *Response Time* 1,498 detik, *Throughput* 44,6 request/min, *Error Rate* 0%, dan *Latency* 1,4986 detik. Proses pembuatan pengumuman berjalan tanpa *error*. Namun, nilai *Response Time* dan *Latency* berada di atas satu detik karena proses

ini melibatkan pengiriman data dan pemrosesan pada server sebelum pengumuman berhasil dibuat.

Pengujian dilakukan menggunakan 52 virtual user dengan 9 test case, sehingga menghasilkan total 468 request selama proses pengujian. *Load Time* terendah adalah 0,710 detik pada *Get Jadwal Kelas*, rata-rata *Respons Time* keseluruhan sebesar 2,064 detik, total *Throughput* sebesar 6,4 request/s, *Error Rate* total sebesar 22,22%, dan rata-rata *Latency* sebesar 0,6195 detik. Nilai *Error Rate* total cukup tinggi terutama disebabkan oleh *test case Post Laporkan* dan *Post Perizinan* yang memiliki *Error Rate* 100%. *Error* tersebut tidak sepenuhnya menunjukkan kegagalan performa server, tetapi lebih disebabkan oleh keterbatasan pengujian virtual JMeter yang tidak menggunakan perangkat asli atau token tertentu yang dibutuhkan oleh sistem.

3) Hasil Pengujian *Mobile* Guru 59 Sampel

Pengujian performa dilakukan pada aplikasi *Mobile* guru dengan jumlah sampel sebanyak 59 pengguna. Hasil pengujian *mobile* guru 59 sampel ada pada gambar 4.39 dan gambar 4.40.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Post Login	59	1275	483	2644	557.76	0.00%	58.7/min	0.99	0.30	1039.9
Get Jadwal Pe...	59	544	135	3665	575.23	0.00%	59.0/min	0.50	0.26	522.0
Get jadwal Ke...	59	166	73	478	84.43	0.00%	59.0/min	0.33	0.26	344.0
Get Riwayat ...	59	219	97	626	124.88	0.00%	59.0/min	0.30	0.28	313.0
Get Riwayat ...	59	12055	1353	56073	12898.17	6.78%	39.7/min	344.49	0.19	532814.8
Get Riwayat P...	59	715	149	2649	685.51	0.00%	40.2/min	0.32	0.22	490.0
Post Laporkan	59	892	149	11348	1606.58	100.00%	40.2/min	0.38	0.28	573.3
Post perizinan	59	207	86	491	113.07	100.00%	40.5/min	0.23	0.23	353.0
Post buat pe...	59	735	462	1122	185.46	0.00%	40.2/min	0.40	0.28	612.9
TOTAL	531	1867	73	56073	5656.45	22.98%	5.8/sec	338.41	1.86	59673.7

Gambar 4. 39 *Summary Reports Load testing pada mobile guru 59 sampel*

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: ☐ Errors ☐ Successes

Sample # ↑	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	00:30:38.003	Thread Group ...	Post Login	891	✓	1039	309	888	491
2	00:30:38.919	Thread Group ...	Get Jadwal Per...	135	✓	522	273	135	52
3	00:30:39.054	Thread Group ...	Get jadwal Kelas	93	✓	344	267	93	0
4	00:30:39.148	Thread Group ...	Get Riwayat A...	98	✓	313	288	98	0
5	00:30:39.929	Thread Group ...	Post Login	532	✓	1039	309	531	67
6	00:30:39.463	Thread Group ...	Get Jadwal Per...	145	✓	522	273	145	51
7	00:30:39.608	Thread Group ...	Get jadwal Kelas	82	✓	344	267	82	0
8	00:30:39.690	Thread Group ...	Get Riwayat A...	113	✓	313	288	113	0
9	00:30:39.919	Thread Group ...	Post Login	524	✓	1039	309	522	65
10	00:30:39.247	Thread Group ...	Get Riwayat A...	1353	✓	571457	307	717	55
11	00:30:40.445	Thread Group ...	Get Jadwal Per...	189	✓	522	273	189	88
12	00:30:40.634	Thread Group ...	Get jadwal Kelas	85	✓	344	267	85	0
13	00:30:40.602	Thread Group ...	Get Riwayat Pe...	199	✓	490	331	199	62
14	00:30:40.720	Thread Group ...	Get Riwayat A...	116	✓	313	288	116	0
15	00:30:40.801	Thread Group ...	Post Laporan	547	✗	547	435	547	421
16	00:30:40.926	Thread Group ...	Post Login	483	✓	1040	309	482	83
17	00:30:39.804	Thread Group ...	Get Riwayat A...	1607	✓	571457	307	826	78
18	00:30:41.349	Thread Group ...	Post perizinan	90	✗	350	342	90	0
19	00:30:41.410	Thread Group ...	Get Jadwal Per...	172	✓	522	273	172	58
20	00:30:41.413	Thread Group ...	Get Riwayat Pe...	207	✓	490	331	207	53

Active Windows 206
Go to Settings to activate Windows

Gambar 4. 40 View Results in table Load testing pada mobile guru 59 sampel

a) *Post Login*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,483 detik, *Response Time* 1,275 detik, *Throughput* 58,7 request/min, *Error Rate* 0%, dan *Latency* 1,2592 detik. Proses *login* guru berjalan dengan baik tanpa *error*. Nilai *Response Time* dan *Latency* relatif lebih tinggi karena sistem perlu melakukan validasi dan autentikasi pengguna sebelum dapat masuk

b) *Get Jadwal Personal*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,135 detik, *Response Time* 0,544 detik, *Throughput* 59,0 request/min, *Error Rate* 0%, dan *Latency* 0,5442 detik. Pengambilan data jadwal personal guru berjalan stabil tanpa *error*.

c) *Get Jadwal Kelas*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,073 detik, *Response Time* 0,166 detik, *Throughput* 59,0 request/min, *Error Rate* 0%, dan *Latency* 0,1661 detik. Proses pengambilan jadwal kelas berjalan cepat dan stabil, seluruh *request* berhasil diproses.

d) *Get Riwayat Absensi Mapel*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,097 detik, *Response Time* 0,219 detik, *Throughput* 59,0 request/min, *Error Rate* 0%,

dan *Latency* 0,2191 detik. Pengambilan data riwayat absensi mata pelajaran berjalan lancar dan seluruh *request* berhasil.

e) *Get Riwayat Absensi Kelas*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 1,353 detik, *Response Time* 12,055 detik, *Throughput* 39,7 request/min, *Error Rate* 6,78%, dan *Latency* 1,9373 detik. *Test case* ini memiliki *Response Time* dan *Latency* paling tinggi. *Error Rate* sebagian kecil tercatat karena jumlah data yang diakses lebih besar atau proses *query* lebih berat.

f) *Get Riwayat Perizinan*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,149 detik, *Response Time* 0,715 detik, *Throughput* 40,2 request/min, *Error Rate* 0%, dan *Latency* 0,7226 detik. Proses pengambilan data perizinan berjalan baik dan tidak mengalami *error*.

g) *Post Laporkan*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,086 detik, *Response Time* 0,207 detik, *Throughput* 40,5 request/min, *Error Rate* 100%, dan *Latency* 0,2097 detik. Semua *request* gagal diproses karena fitur ini membutuhkan token FCM untuk mengirim notifikasi ke siswa. Pengujian menggunakan virtual *user* tidak menyediakan token FCM dari perangkat asli, sehingga server gagal mengirim notifikasi. Keterangan *error* diperkuat dari *View Results Tree*.

h) *Post Perizinan*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,149 detik, *Response Time* 0,892 detik, *Throughput* 40,2 request/min, *Error Rate* 100%, dan *Latency* 0,7093 detik. Semua *request* gagal karena alasan sama dengan *Post Perizinan*; server tidak menerima token FCM. *Error* ini bersifat akibat kondisi pengujian virtual, bukan kegagalan sistem.

i) *Post Pengumuman*

Hasil pengujian pada gambar 4.39 dan 4.40 diketahui *Load Time* 0,462 detik, *Response Time* 0,735 detik, *Throughput* 44,6 request/min, *Error Rate* 0%, dan *Latency* 0,7377 detik. Pembuatan pengumuman berjalan stabil, seluruh

request berhasil diproses meskipun *Response Time* dan *Latency* sedikit lebih tinggi.

Pengujian melibatkan 59 virtual *user* yang menjalankan 9 *test case*, sehingga menghasilkan total 531 *request* selama proses *load testing*. *Load Time* terendah adalah 0,073 detik, rata-rata *Response Time* keseluruhan sebesar 1,867 detik, total *Throughput* 5,8 *request/s*, *Error Rate* total 22,22%, dan rata-rata *Latency* 0,7228 detik. Nilai *Error Rate* total tinggi terutama disebabkan oleh *test case Post Perizinan* dan *Post Laporkan* yang membutuhkan token FCM, sehingga kegagalan *request* tercatat sebagai *error* pada *Summary Report*. *Error* ini bukan murni kegagalan performa sistem, melainkan keterbatasan pengujian virtual JMeter yang tidak menggunakan perangkat asli.

3.10.1. Gorilla testing

Pengujian *gorilla testing* dilakukan untuk menguji kestabilan sistem dengan memberikan *request* secara berulang pada fitur tertentu dalam jangka waktu tertentu. Pengujian difokuskan pada fitur utama yang memiliki aktivitas tinggi dan berhubungan langsung dengan proses absensi pada sistem Presensiku. Pada pengujian ini menggunakan nilai *loop count* sebesar 100 untuk mensimulasikan pengiriman *request* secara terus-menerus pada *endpoint API* yang diuji. Hasil pengujian diperoleh dari tiga tampilan utama pada JMeter, yaitu *Summary Report*, *View Results in Table*, dan *View Results Tree*. *Summary Report* digunakan untuk melihat nilai *Load Time*, *Response Time*, *Throughput*, dan *Error Rate*. *View Results in Table* digunakan sebagai dasar perhitungan nilai *Latency*. Perhitungan lengkap *Latency* dicantumkan pada RTM melalui link <https://bit.ly/DokumenRTMPengujian>, sedangkan *View Results Tree* digunakan untuk melihat detail *request* dan *respons*, terutama pada *test case* yang mengalami *error*. Melalui *View Results Tree*, penyebab kegagalan *request* dapat dianalisis berdasarkan *Response message*, *Response Code*, atau informasi *error* yang ditampilkan oleh *server*.

Hasil pengujian diperoleh dari tiga tampilan utama pada JMeter, yaitu *Summary Report*, *View Results in Table*, dan *View Results Tree*. *Summary Report* digunakan untuk melihat nilai *Load Time*, *Response Time*, *Throughput*,

dan *Error Rate*. *View Results in Table* digunakan sebagai dasar perhitungan nilai *Latency*. Perhitungan lengkap *Latency* dicantumkan pada RTM melalui link <https://bit.ly/DokumenRTMPengujian>, sedangkan *View Results Tree* digunakan untuk melihat detail *request* dan *response*, terutama pada *test case* yang mengalami *error*. Melalui *View Results Tree*, penyebab kegagalan *request* dapat dianalisis berdasarkan *Response message*, *Response Code*, atau informasi *error* yang ditampilkan oleh *server*.

a. *Scan QR Code Absensi Mapel (Mobile Siswa)*

Pengujian performa pada fitur *QR Code* dilakukan menggunakan metode *Gorilla testing*, yaitu pengujian berulang pada fitur tertentu untuk mengetahui konsistensi *respons* sistem ketika menerima *request* secara terus-menerus. Pengujian dilakukan menggunakan 1 virtual *user* dengan 100 kali looping pada setiap *test case*. Hasil pengujian *Scan QR Code* absensi mapel (*mobile siswa*) ada pada gambar 4.41 dan gambar 4.42.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
Post Login	100	395	371	740	52.89	0.00%	1.6/sec	1.76	0.42	1145.8
Post Qr Code	100	233	150	2311	272.92	0.00%	1.6/sec	1.63	1.05	1057.2
TOTAL	200	314	150	2311	212.70	0.00%	3.1/sec	3.37	1.46	1101.5

Gambar 4. 41 *Summary Reports golilla testing* pada *scan qr Code* absensi mapel (*mobile siswa*)

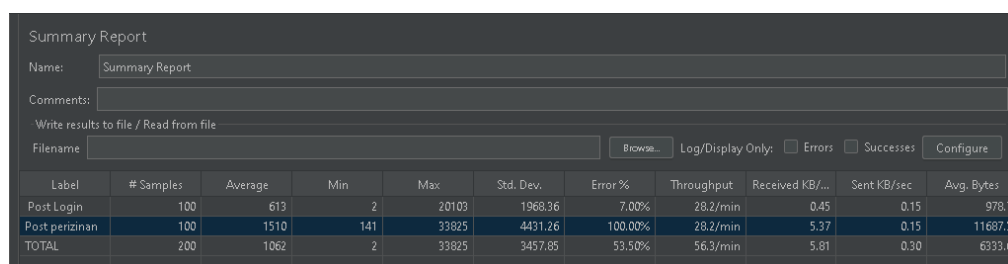
Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	23:56:24.359	Thread Group ...	Post Login	617	✓	1322	274	616	210
2	23:56:24.991	Thread Group ...	Post Qr Code	304	✓	1057	682	304	56
3	23:56:25.296	Thread Group ...	Post Login	377	✓	1144	274	376	0
4	23:56:25.673	Thread Group ...	Post Qr Code	161	✓	1056	682	161	54
5	23:56:25.895	Thread Group ...	Post Login	421	✓	1144	274	420	0
6	23:56:26.256	Thread Group ...	Post Qr Code	162	✓	1056	682	162	50
7	23:56:26.418	Thread Group ...	Post Login	380	✓	1144	274	380	0
8	23:56:26.799	Thread Group ...	Post Qr Code	154	✓	1059	682	154	49
9	23:56:26.993	Thread Group ...	Post Login	382	✓	1144	274	382	0
10	23:56:27.336	Thread Group ...	Post Qr Code	208	✓	1058	682	208	71
11	23:56:27.544	Thread Group ...	Post Login	385	✓	1144	274	385	0
12	23:56:27.950	Thread Group ...	Post Qr Code	248	✓	1057	682	248	106
13	23:56:28.178	Thread Group ...	Post Login	391	✓	1144	274	391	0
14	23:56:28.569	Thread Group ...	Post Qr Code	330	✓	1056	682	330	183
15	23:56:28.899	Thread Group ...	Post Login	449	✓	1144	274	387	0
16	23:56:29.348	Thread Group ...	Post Qr Code	185	✓	1056	682	185	70
17	23:56:29.533	Thread Group ...	Post Login	386	✓	1144	274	385	0
18	23:56:29.919	Thread Group ...	Post Qr Code	159	✓	1059	682	159	49
19	23:56:30.078	Thread Group ...	Post Login	384	✓	1144	274	384	0
20	23:56:30.462	Thread Group ...	Post Qr Code	155	✓	1058	682	155	49

Gambar 4. 42 *View Results in table gorilla testing* pada scan QR Code absensi mapel (*mobile siswa*)

Hasil pengujian pada gambar 4.41 dan 4.42 diketahui *Load Time* 0,233 detik, *Response Time* 0,233 detik, *Throughput* 1,6 request/s, *Error Rate* 0%, dan *Latency* 0,2331 detik. Pengujian pada fitur *Post QR Code* berjalan dengan baik karena seluruh *request* berhasil diproses tanpa *error*. Nilai *Error Rate* sebesar 0% menunjukkan bahwa sistem mampu menerima dan memvalidasi data QR Code, id siswa, id kelas, latitude, dan longitude secara stabil selama proses pengujian berulang. Selain itu, nilai *Response Time* dan *Latency* yang berada di bawah 1 detik menunjukkan bahwa proses pemindaian atau pengiriman data QR Code pada aplikasi *Mobile* memiliki waktu *respons* yang cepat. Fitur QR Code dapat dikatakan berjalan stabil dan layak digunakan dalam proses absensi siswa.

b. Kelola Perizinan (*Mobile Guru*)

Pengujian performa pada fitur Perizinan dilakukan menggunakan metode *Gorilla testing*, yaitu pengujian berulang pada fitur tertentu untuk mengetahui konsistensi *respons* sistem ketika menerima *request* secara terus-menerus. Pengujian dilakukan menggunakan 1 virtual *user* dengan 100 kali *looping* pada *test case Post Perizinan*. Hasil pengujian kelola perizinan (*mobile siswa*) ada pada gambar 4.3 dan gambar 4.45



Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
Post Login	100	613	2	20103	1968.36	7.00%	28.2/min	0.45	0.15	978.7
Post perizinan	100	1510	141	33825	4431.26	100.00%	28.2/min	5.37	0.15	11687.3
TOTAL	200	1062	2	33825	3457.85	53.50%	56.3/min	5.81	0.30	6333.0

Gambar 4. 43 *Summary Reports gorilla testing* pada post perizinan (*mobile siswa*)

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	202630524	Thread Group 1-1	Post Login	465	✓	1039	309	465	59
2	202630990	Thread Group 1-1	Post perizinan	216	✗	528	344	216	114
3	202631206	Thread Group 1-1	Post Login	375	✓	861	347	375	0
4	202631581	Thread Group 1-1	Post perizinan	141	✗	528	344	141	51
5	202631723	Thread Group 1-1	Post Login	400	✓	861	347	399	0
6	202632126	Thread Group 1-1	Post perizinan	177	✗	528	344	177	73
7	202632303	Thread Group 1-1	Post Login	384	✓	861	347	383	0
8	202632687	Thread Group 1-1	Post perizinan	604	✗	13175	344	190	74
9	202633291	Thread Group 1-1	Post Login	952	✓	861	347	952	0
10	202634243	Thread Group 1-1	Post perizinan	155	✗	13175	344	154	53
11	202634398	Thread Group 1-1	Post Login	383	✓	861	347	383	0
12	202634781	Thread Group 1-1	Post perizinan	194	✗	13175	344	191	79
13	202634975	Thread Group 1-1	Post Login	378	✓	861	347	377	0
14	202635353	Thread Group 1-1	Post perizinan	250	✗	13175	344	250	113
15	202635604	Thread Group 1-1	Post Login	419	✓	861	347	411	0
16	202636023	Thread Group 1-1	Post perizinan	160	✗	13175	344	158	60
17	202636183	Thread Group 1-1	Post Login	394	✓	861	347	393	0
18	202636577	Thread Group 1-1	Post perizinan	179	✗	13175	344	178	54
19	202636756	Thread Group 1-1	Post Login	395	✓	861	347	394	0
20	202637151	Thread Group 1-1	Post perizinan	199	✗	13175	344	195	79

Go to Settings to activate Windows

Gambar 4. 44 *View Results in table gorilla testing pada post perizinan (mobile guru)*

View Results Tree

Name: View Results Tree

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes

Search: Case sensitive ☐ Regular exp.

Test

- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan
- Post Login
- Post perizinan

Sampler result Request Response data

Response Body Response headers

```
{
  "message": "App\\Services\\NotificationHelperService::tokenTemplate(): Argument #1 ($token) must be of type string, null given, called in /home/pres2446/sis/app/exception", "type": "TypeError",
  "file": "/home/pres2446/sis/app/Services/NotificationHelperService.php",
  "line": 50,
  "trace": [
    {
      "file": "/home/pres2446/sis/app/Services/NotificationHelperService.php",
      "line": 189,
      "function": "tokenTemplate",
      "class": "App\\Services\\NotificationHelperService",
      "type": "function"
    },
    {
      "file": "/home/pres2446/sis/app/Http/Controllers/api/AttendanceController.php",
      "line": 942,
      "function": "templatePermissionApproval",
      "class": "App\\Services\\NotificationHelperService",
      "type": "function"
    }
  ]
}
```

Go to Settings to activate Windows

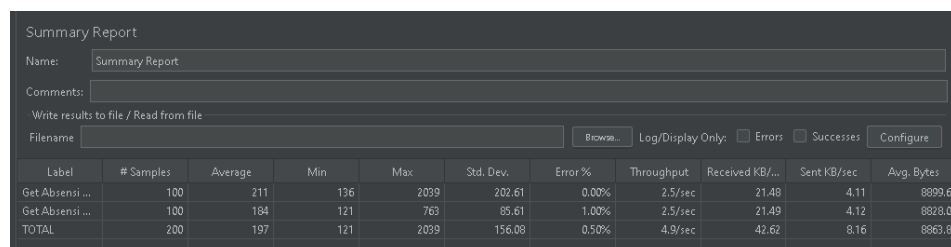
Gambar 4. 45 *View Results Tree gorilla testing pada post perizinan (mobile guru)*

Hasil pengujian pada gambar 4.43 dan gambar 4.44 menunjukkan nilai *Load Time* sebesar 0,141 detik detik detik, *Response Time* sebesar 1,510 detik, *Throughput* sebesar 28,2 request/min, *Error Rate* sebesar 100%, dan *Latency* sebesar 0,4820 detik. Nilai *Error Rate* sebesar 100% muncul karena proses *Post Perizinan* memerlukan token *Firebase Cloud Messaging* (FCM) untuk mengirimkan notifikasi ke perangkat siswa dapat di lihat oada gambar 4.45. Pengujian yang dilakukan menggunakan Apache JMeter memanfaatkan virtual

user sehingga tidak memiliki token FCM dari perangkat asli. Kondisi tersebut menyebabkan server gagal mengirimkan notifikasi dan setiap *request* tercatat sebagai *error* pada hasil pengujian. Nilai *Response Time* dan *Latency* yang berada di bawah 1 detik menunjukkan bahwa proses pengiriman *request* ke server berlangsung dengan cepat. Hasil *View Results Tree* juga menunjukkan bahwa *error* yang terjadi disebabkan oleh kegagalan pengiriman notifikasi FCM, bukan karena kegagalan proses pengajuan perizinan. *Error Rate* yang diperoleh tidak sepenuhnya mencerminkan kegagalan fungsi utama fitur Perizinan, melainkan keterbatasan lingkungan pengujian yang tidak menggunakan perangkat *Mobile* asli.

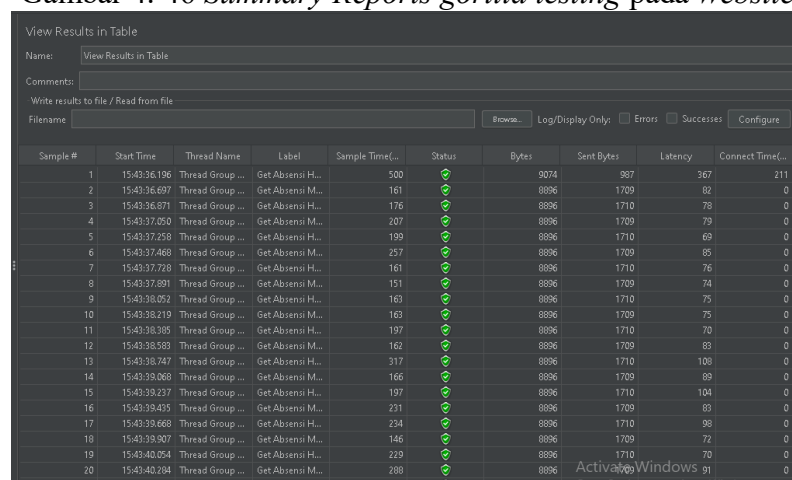
c. Website

Pengujian ini digunakan 1 virtual *user* dengan 100 kali *looping* pada masing-masing *test case*, sehingga total sampel pengujian berjumlah 200 *request*. Fitur yang diuji meliputi Akses Absensi Kehadiran dan Akses Absensi Mapel. Hasil pengujian ada pada gambar 4.46 dan gambar 4.47.



Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Get Absensi ...	100	211	136	2039	202.61	0.00%	2.5/sec	21.48	4.11	8899.6
Get Absensi ...	100	184	121	763	85.61	1.00%	2.5/sec	21.49	4.12	8828.0
TOTAL	200	197	121	2039	156.08	0.50%	4.9/sec	42.62	8.16	8863.8

Gambar 4. 46 *Summary Reports gorilla testing pada Website*



Sample #	Start Time	Thread Name	Label	Sample Time...	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	15:43:36.196	Thread Group ...	Get Absensi H...	500	✓	9074	987	367	211
2	15:43:36.697	Thread Group ...	Get Absensi M...	161	✓	8896	1709	82	0
3	15:43:36.871	Thread Group ...	Get Absensi H...	176	✓	8896	1710	78	0
4	15:43:37.050	Thread Group ...	Get Absensi M...	207	✓	8896	1709	79	0
5	15:43:37.258	Thread Group ...	Get Absensi H...	199	✓	8896	1710	69	0
6	15:43:37.468	Thread Group ...	Get Absensi M...	257	✓	8896	1709	85	0
7	15:43:37.728	Thread Group ...	Get Absensi H...	161	✓	8896	1710	76	0
8	15:43:37.891	Thread Group ...	Get Absensi M...	151	✓	8896	1709	74	0
9	15:43:38.052	Thread Group ...	Get Absensi H...	163	✓	8896	1710	75	0
10	15:43:38.219	Thread Group ...	Get Absensi M...	163	✓	8896	1709	75	0
11	15:43:38.385	Thread Group ...	Get Absensi H...	197	✓	8896	1710	70	0
12	15:43:38.583	Thread Group ...	Get Absensi M...	162	✓	8896	1709	83	0
13	15:43:38.747	Thread Group ...	Get Absensi H...	317	✓	8896	1710	108	0
14	15:43:39.068	Thread Group ...	Get Absensi M...	166	✓	8896	1709	89	0
15	15:43:39.237	Thread Group ...	Get Absensi H...	197	✓	8896	1710	104	0
16	15:43:39.435	Thread Group ...	Get Absensi M...	231	✓	8896	1709	83	0
17	15:43:39.668	Thread Group ...	Get Absensi H...	234	✓	8896	1710	98	0
18	15:43:39.907	Thread Group ...	Get Absensi M...	146	✓	8896	1709	72	0
19	15:43:40.054	Thread Group ...	Get Absensi H...	229	✓	8896	1710	70	0
20	15:43:40.204	Thread Group ...	Get Absensi M...	288	✓	8896	1709	91	0

Gambar 4. 47 *View Results in table gorilla testing pada website*

1) Akses Absensi Kehadiran

Hasil pengujian pada gambar 4.46 dan 4.47 diketahui *Load Time* 0,211 detik, *Response Time* 0,211 detik, *Throughput* 2,5 request/s, *Error Rate* 0%, dan *Latency* 0,0868 detik. Proses akses data absensi kehadiran berjalan dengan baik tanpa adanya *request* yang gagal. Nilai *Load Time* dan *Response Time* tergolong rendah, sehingga menunjukkan bahwa sistem mampu memberikan *respons* dengan cepat ketika pengguna mengakses fitur absensi kehadiran.

2) Akses Absensi Mapel

Hasil pengujian pada gambar 4.46 dan 4.47 diketahui *Load Time* 0,184 detik, *Response Time* 0,184 detik, *Throughput* 2,5 request/s, *Error Rate* 1,00%, dan *Latency* 0,0856 detik. Proses akses data absensi mapel secara umum berjalan stabil, meskipun terdapat sebagian kecil *request* yang mengalami kegagalan. *Error Rate* sebesar 1,00% menunjukkan bahwa hampir seluruh *request* berhasil diproses oleh server, sedangkan satu *request* gagal kemungkinan disebabkan oleh *respons* server yang tidak stabil pada saat pengujian berlangsung.

Pengujian ini dilakukan pada 200 *request* pada 2 *test case*, *Load Time* rata-rata keseluruhan sebesar 0,197 detik, *Response Time* rata-rata sebesar 0,197 detik, total *Throughput* 4,9 request/s, *Error Rate* total 0,50%, dan rata-rata *Latency* sebesar 0,0862 detik. Hasil ini menunjukkan bahwa performa fitur *website* pada akses absensi kehadiran dan absensi mapel tergolong baik karena waktu *respons* berada di bawah 1 detik. *Error Rate* total yang hanya sebesar 0,50% menunjukkan bahwa sistem relatif stabil, meskipun masih terdapat satu kegagalan *request* yang dapat disebabkan oleh *respons* server yang tidak stabil pada saat pengujian.

4.8. Test Closure

a. Test Closure Website

Berdasarkan hasil pengujian *load testing* pada sistem berbasis *website* menggunakan Apache JMeter dokumen lengkap nya bisa diakses pada link <https://bit.ly/DokumenTestReport>, secara umum sistem telah berhasil menjalankan sebagian besar skenario pengujian sesuai dengan rancangan *test*

case. Seluruh fitur utama seperti *login*, akses *QR Code*, serta pengambilan data absensi harian dan absensi mata pelajaran menunjukkan hasil yang stabil dengan nilai *Response Time* dan *Load Time* yang relatif rendah serta *Error Rate* sebesar 0% pada sebagian besar *request*.

Fitur *Post Foto* menunjukkan *Error Rate* sebesar 100%. Hasil analisis menunjukkan bahwa kegagalan tersebut disebabkan oleh belum tersedianya VPS sebagai infrastruktur server untuk mendukung proses unggah dan pemrosesan data foto. Kondisi ini menyebabkan permintaan pengiriman foto tidak dapat diproses dengan baik sehingga menghasilkan *error*. Meskipun demikian, *error* yang muncul tidak menunjukkan kegagalan performa sistem secara keseluruhan, melainkan keterbatasan infrastruktur server yang digunakan pada saat pengujian. Oleh karena itu, implementasi VPS diperlukan untuk meningkatkan kemampuan *server* dalam menangani proses unggah foto secara optimal.

Berdasarkan hasil pengujian yang telah dilakukan, sistem *website* secara umum mampu menangani beban akses tunggal dengan baik. Namun, masih diperlukan penyesuaian pada skenario pengiriman *file* agar pengujian pada fitur *Post Foto* dapat merepresentasikan kondisi sistem yang sebenarnya secara lebih akurat.

b. *Test Closure Mobile Siswa*

Berdasarkan hasil pengujian pada aplikasi *Mobile* siswa dengan variasi jumlah pengguna 92, 297, dan 1034 *user*, sistem menunjukkan kemampuan yang cukup stabil dalam menangani beban akses secara bersamaan. Sebagian besar fitur seperti *login*, pengambilan jadwal, pengumuman, serta pengiriman perizinan dapat berjalan dengan baik dengan nilai *Response Time* dan *Latency* yang masih berada di bawah ambang batas satu detik pada sebagian besar skenario pengujian dokumen lengkap nya bisa diakses pada link <https://bit.ly/DokumenTestReport>.

Temuan utama dalam pengujian ini terdapat pada fitur *Post QR Code* yang pada beberapa skenario menghasilkan *Error Rate* kecil (0,10%–1,09%). Berdasarkan hasil analisis, *error* tersebut bersifat minor dan tidak mengganggu

keseluruhan proses sistem, sehingga masih dalam batas toleransi pengujian performa. Tidak ditemukan kegagalan sistem yang bersifat konsisten pada seluruh skenario pengujian.

Hasil pengujian menunjukkan bahwa aplikasi *Mobile* siswa memiliki performa yang baik dalam menangani peningkatan jumlah pengguna hingga lebih dari seribu *user*, dengan tingkat *error* yang sangat rendah serta waktu *respons* yang tetap stabil.

c. *Test Closure Mobile Guru*

Berdasarkan hasil pengujian *load testing* pada aplikasi *Mobile* guru dengan jumlah pengguna 38, 52, dan 59 *user*, sistem secara umum mampu menjalankan sebagian besar fitur dengan baik. Fitur seperti *login*, pengambilan jadwal, serta akses data perizinan dan pengumuman menunjukkan performa yang stabil dengan *Response Time* yang masih berada dalam rentang wajar dokumen lengkap nya bisa diakses pada link <https://bit.ly/DokumenTestReport>.

Hasil pengujian menunjukkan adanya temuan penting pada beberapa fitur yang menghasilkan *Error Rate* tinggi, yaitu *Post* Laporkan dan *Post* Perizinan yang mencapai 100% pada beberapa skenario pengujian. Berdasarkan hasil analisis melalui *View Results Tree*, *error* tersebut disebabkan oleh ketidak terpenuhinya kebutuhan token *Firebase Cloud Messaging* (FCM) pada lingkungan pengujian menggunakan virtual *user*. Kondisi ini menyebabkan server tidak dapat mengirimkan notifikasi, sehingga *request* tercatat sebagai gagal. Selain itu, pada fitur *Get Riwayat Absensi Kelas* juga ditemukan *Response Time* yang relatif tinggi dibandingkan fitur lainnya, yang menunjukkan bahwa proses pengambilan data memiliki tingkat kompleksitas yang lebih besar.

Hasil pengujian menunjukkan bahwa kegagalan yang terjadi pada beberapa fitur tidak disebabkan oleh performa *server* yang buruk, melainkan oleh keterbatasan infrastruktur yang digunakan selama pengujian. Secara keseluruhan, sistem tetap mampu mempertahankan stabilitas dalam menangani beban pengguna sesuai dengan skenario yang telah diuji.

d. *Test Closure Gorilla testing*

Berdasarkan hasil pengujian *Gorilla testing* yang dilakukan pada fitur utama sistem dengan pengulangan *request* sebanyak 100 kali, sistem menunjukkan kemampuan yang cukup baik dalam mempertahankan stabilitas pada sebagian besar fitur. Fitur seperti scan QR Code pada *Mobile* siswa serta akses halaman absensi pada *website* mampu berjalan dengan *Response Time* yang rendah dan *Error Rate* mendekati nol dokumen lengkap nya bisa diakses pada link <https://bit.ly/DokumenTestReport>.

Hasil pengujian menunjukkan temuan penting pada beberapa fitur yang menghasilkan *error*. Fitur Kelola Perizinan pada aplikasi mobile guru memiliki *Error Rate* sebesar 100%. Hasil analisis menunjukkan bahwa kegagalan tersebut disebabkan oleh ketergantungan sistem terhadap token *Firebase Cloud Messaging* (FCM) yang tidak tersedia pada lingkungan pengujian menggunakan *virtual user*. Kondisi tersebut menyebabkan proses pengiriman notifikasi tidak dapat dilakukan sehingga seluruh *request* tercatat sebagai gagal. Fitur *website* menunjukkan *Error Rate* sebesar 1% pada akses absensi mata pelajaran, yang mengindikasikan adanya ketidakstabilan minor pada sebagian kecil *request* selama pengujian berlangsung.

Berdasarkan hasil pengujian dapat disimpulkan bahwa sistem secara umum mampu menangani pengujian berulang dengan baik. *Error* yang muncul lebih banyak disebabkan oleh keterbatasan lingkungan pengujian, bukan oleh kegagalan utama pada sistem.

4.9. Hasil Analisis

Pengujian performa sistem absensi Presensiku dilakukan untuk mengetahui kemampuan sistem dalam menangani beban pengguna secara bersamaan serta mengukur kestabilan dan kecepatan respon sistem pada kondisi penggunaan nyata. Pengujian ini disusun berdasarkan hasil analisis kebutuhan, perancangan pengujian, hingga implementasi pengujian menggunakan Apache JMeter yang mencakup metode *load testing* dan *gorilla testing* pada sistem berbasis *website* dan *mobile*.

a. Hasil Pengujian Mobile

Tabel 4. x hasil pengujian *mobile*

<i>User</i>	<i>Load Time</i>	<i>Response Time</i>	<i>Throughput</i>	<i>Error Rate</i>	<i>Latency</i>
92	0,070	0,308	6	0,18%	0,3073
297	0,070	0,329	6	0,11%	0,3281
1034	0,001	0,321	6	0,08%	0,3199

Berdasarkan hasil total pengujian, peningkatan jumlah pengguna dari 92, 297, hingga 1.034 virtual *user* tidak memberikan perubahan yang signifikan terhadap performa aplikasi *mobile* siswa. Nilai *Load Time* tetap berada pada rentang 0,001–0,070 detik, sehingga menunjukkan bahwa halaman aplikasi dapat dimuat dengan cepat meskipun jumlah pengguna terus bertambah. Seluruh nilai *Load Time* masih berada di bawah standar 3 detik.

Nilai *Response Time* mengalami perubahan yang relatif kecil, yaitu dari 0,308 detik pada 92 virtual *user* menjadi 0,329 detik pada 297 virtual *user*, kemudian turun menjadi 0,321 detik pada 1.034 virtual *user*. Perbedaan tersebut menunjukkan bahwa peningkatan jumlah pengguna tidak memberikan pengaruh yang berarti terhadap waktu respons sistem karena seluruh nilai masih berada pada rentang yang hampir sama.

Nilai *Throughput* tetap sebesar 6 request/s pada setiap skenario pengujian. Hasil tersebut menunjukkan bahwa kemampuan sistem dalam memproses permintaan tetap stabil meskipun jumlah virtual *user* meningkat lebih dari sepuluh kali lipat.

Nilai *Error Rate* mengalami penurunan dari 0,18% pada 92 virtual *user* menjadi 0,11% pada 297 virtual *user*, kemudian kembali menurun menjadi 0,08% pada 1.034 virtual *user*. Seluruh nilai *Error Rate* berada di bawah batas 5%, sehingga menunjukkan bahwa sistem mampu memproses permintaan pengguna dengan tingkat kegagalan yang sangat rendah.

Nilai *Latency* mengalami perubahan yang relatif kecil, yaitu 0,3073 detik, 0,3281 detik, dan 0,3199 detik pada masing-masing skenario pengujian. Seluruh nilai tersebut masih berada di bawah 1 detik, sehingga menunjukkan

bahwa server mampu memberikan respons awal kepada pengguna dengan cepat meskipun jumlah virtual user meningkat. Secara keseluruhan, hasil pengujian menunjukkan bahwa aplikasi mobile siswa mampu mempertahankan performa yang stabil pada seluruh variasi beban pengguna, ditunjukkan oleh nilai Load Time, Response Time, Latency, dan Error Rate yang memenuhi standar, serta nilai Throughput yang tetap konstan selama proses pengujian.

b. Hasil Total Seluruh pengujian

Tabel 4. x hasil pengujian *mobile*

<i>User</i>	<i>Load Time</i>	<i>Response Time</i>	<i>Throughput</i>	<i>Error Rate</i>	<i>Latency</i>
38	0,078	1,533	6/s	22,5%	1,8570
52	0,081	2,064	6,4/s	22,22%	0,6195
59	0,073	1,867	5,8/s	22,22%	0,7228

Berdasarkan hasil total pengujian aplikasi mobile guru, peningkatan jumlah pengguna dari 38, 52, hingga 59 virtual user menunjukkan perubahan performa yang relatif kecil pada sebagian besar parameter. Nilai Load Time berada pada rentang 0,073–0,081 detik, sehingga seluruh skenario pengujian masih berada di bawah standar 3 detik. Hasil tersebut menunjukkan bahwa aplikasi mampu memuat halaman dengan cepat meskipun jumlah virtual user meningkat.

Nilai Response Time mengalami peningkatan dari 1,533 detik pada 38 virtual user menjadi 2,064 detik pada 52 virtual user. Peningkatan tersebut menunjukkan bahwa bertambahnya jumlah virtual user menyebabkan server memerlukan waktu yang lebih lama untuk memproses permintaan yang diterima secara bersamaan. Kondisi ini dapat terjadi karena sumber daya server, seperti prosesor, memori, maupun proses komunikasi dengan basis data, harus menangani lebih banyak request secara simultan sehingga waktu respons sistem meningkat. Nilai Response Time kemudian menurun menjadi 1,867 detik pada 59 virtual user. Penurunan tersebut menunjukkan bahwa sistem masih mampu mempertahankan kestabilan performa meskipun menerima beban yang lebih tinggi. Perbedaan nilai Response Time antar skenario relatif kecil sehingga

peningkatan jumlah virtual user tidak menyebabkan penurunan performa yang signifikan.

Nilai Throughput mengalami perubahan yang tidak terlalu signifikan, yaitu 6 request/s pada 38 virtual user, meningkat menjadi 6,4 request/s pada 52 virtual user, kemudian sedikit menurun menjadi 5,8 request/s pada 59 virtual user. Perubahan tersebut menunjukkan bahwa kemampuan sistem dalam memproses permintaan tetap relatif stabil pada setiap skenario pengujian.

Nilai Error Rate sebesar 22,5% pada 38 virtual user serta 22,22% pada 52 dan 59 virtual user menunjukkan bahwa tingkat kegagalan sistem masih berada di atas batas 5%. Nilai tersebut tidak disebabkan oleh penurunan performa sistem secara keseluruhan, melainkan dipengaruhi oleh kegagalan pada fitur Post Perizinan yang memerlukan Firebase Cloud Messaging (FCM) token. Virtual user yang digunakan selama pengujian tidak memiliki token tersebut sehingga proses pengiriman permintaan pada fitur tersebut tidak dapat diproses dengan baik.

Nilai Latency sebesar 1,8570 detik pada 38 virtual user menunjukkan waktu respons awal yang lebih tinggi dibandingkan standar 1 detik. Nilai tersebut menurun menjadi 0,6195 detik pada 52 virtual user dan 0,7228 detik pada 59 virtual user, sehingga kedua skenario terakhir telah memenuhi standar yang digunakan. Secara keseluruhan, aplikasi mobile guru mampu mempertahankan performa yang relatif stabil pada parameter Load Time, Response Time, dan Throughput. Nilai Error Rate yang masih tinggi dipengaruhi oleh kendala pada fitur Post Perizinan, sehingga hasil tersebut lebih mencerminkan keterbatasan lingkungan pengujian daripada penurunan performa sistem secara keseluruhan.

c. Hasil total seluruh pengujian *website*

Tabel 4. x hasil pengujian *website*

Label	<i>Load Timen</i>	<i>Respons e Time</i>	<i>Throughput</i>	<i>Error Rate</i>	<i>Latency</i>
Post Login	2,342	2,342	25,6/min	0%	0,565

Label	Load Timen	Respon s Time	Throughput	Error Rate	Latency
Get Qr Code	0,207	0,207	4,8/min	0%	0,165
Get Absensi Harian	4,528	4,528	13,3/min	0%	0,1393
Get Absensi Mapel	0, 166	0,166	6/min	0%	0,089
Post foto	0,037	0,037	27/min	100%	0,037
Total	0,037	1,456	76,7	20%	0,2938

Berdasarkan hasil pengujian pada website, nilai Load Time setiap fitur menunjukkan hasil yang bervariasi. Fitur Post Foto memiliki Load Time tercepat, yaitu 0,037 detik, diikuti Get Absensi Mapel sebesar 0,166 detik, Get QR Code sebesar 0,207 detik, Post Login sebesar 2,342 detik, dan Get Absensi Harian sebesar 4,528 detik. Nilai Load Time pada fitur Get Absensi Harian melebihi standar 3 detik, sehingga menunjukkan bahwa fitur tersebut memerlukan waktu pemuatan yang lebih lama dibandingkan fitur lainnya.

Nilai Response Time menunjukkan pola yang sama dengan Load Time. Fitur Post Foto memiliki waktu respons tercepat sebesar 0,037 detik, sedangkan Get Absensi Harian memiliki waktu respons tertinggi sebesar 4,528 detik. Nilai Response Time pada sebagian besar fitur masih tergolong cepat, namun fitur Get Absensi Harian memerlukan perhatian karena memiliki waktu respons yang lebih tinggi dibandingkan standar yang digunakan.

Nilai Throughput tertinggi diperoleh pada fitur Post Foto sebesar 27 request/menit, diikuti Post Login sebesar 25,6 request/menit, Get Absensi Harian sebesar 13,3 request/menit, Get Absensi Mapel sebesar 6 request/menit, dan Get QR Code sebesar 4,8 request/menit. Hasil tersebut menunjukkan bahwa fitur Post Foto mampu memproses permintaan lebih banyak dalam satuan waktu dibandingkan fitur lainnya.

Nilai Error Rate menunjukkan bahwa empat fitur, yaitu Post Login, Get QR Code, Get Absensi Harian, dan Get Absensi Mapel, tidak mengalami kegagalan selama proses pengujian dengan Error Rate sebesar 0%. Fitur Post

Foto memperoleh Error Rate sebesar 100%, sehingga menyebabkan Error Rate keseluruhan menjadi 20%. Nilai tersebut berada di atas batas 5%, sehingga menunjukkan bahwa performa website belum sepenuhnya optimal. Berdasarkan hasil pengujian, kegagalan pada fitur Post Foto disebabkan oleh server Flask yang tidak berjalan pada VPS, sehingga proses pengiriman foto tidak dapat diproses oleh sistem.

Nilai Latency seluruh fitur berada pada rentang 0,037–0,565 detik, sehingga masih berada di bawah standar 1 detik. Nilai Latency tertinggi terdapat pada fitur Post Login, yaitu 0,565 detik, sedangkan nilai terendah terdapat pada fitur Post Foto, yaitu 0,037 detik. Hasil tersebut menunjukkan bahwa server mampu memberikan respons awal kepada pengguna dengan cepat pada seluruh fitur yang diuji.

Secara keseluruhan, website memperoleh rata-rata Response Time sebesar 1,456 detik, Load Time terendah sebesar 0,037 detik, Throughput total sebesar 76,7 request/menit, Latency rata-rata sebesar 0,2938 detik, dan Error Rate total sebesar 20%. Hasil tersebut menunjukkan bahwa parameter Load Time, Throughput, dan Latency telah menunjukkan performa yang baik. Nilai Response Time rata-rata sebesar 1,456 detik masih berada di atas waktu respons optimal sebesar 0,1 detik, sehingga waktu respons sistem belum memenuhi kriteria yang ditetapkan dalam penelitian ini. Nilai Error Rate total sebesar 20% juga melebihi batas 5%, yang dipengaruhi oleh kegagalan pada fitur Post Foto, sedangkan fitur lainnya berhasil diproses tanpa kesalahan selama pengujian.

d. Hasil pengujian *gorilla testing*

Tabel 4.x hasil pengujian *gorilla testing*

Label	<i>Load Time</i>	<i>Response Time</i>	<i>Throughput</i>	<i>Error Rate</i>	<i>Latency</i>
Post Scan Qr Code (Mobile Siswa)	0,233	0,233	1,6/s	0%	2,233
Post Perizinan	0,141	1,510	28.2/min	100%	0,4820

Label	Load Time	Response Time	Throughput	Error Rate	Latency
(Mobile Guru)					
Get	0,211	0,211	2,5/s	0%	0,0868
Riwayat Harian Kelas					
(Website)					
Get	0,184	0,184	2,5/s	1,00%	0,0856
Riwayat Harian Mapel					
(Website)					

Berdasarkan hasil gorilla testing, fitur Post Scan QR Code pada aplikasi mobile siswa memperoleh Load Time dan Response Time sebesar 0,233 detik, Throughput sebesar 1,6 request/s, serta Error Rate sebesar 0%. Nilai Load Time telah memenuhi standar karena berada di bawah 3 detik, sedangkan Response Time masih berada di atas waktu respons optimal sebesar 0,1 detik. Nilai Latency sebesar 2,233 detik menunjukkan bahwa waktu respons awal server melebihi standar 1 detik, meskipun seluruh proses pengujian dapat diselesaikan tanpa kegagalan.

Fitur Post Perizinan pada aplikasi mobile guru memperoleh Load Time sebesar 0,141 detik, Response Time sebesar 1,510 detik, Throughput sebesar 28,2 request/menit, Latency sebesar 0,4820 detik, dan Error Rate sebesar 100%. Nilai Load Time dan Latency masih memenuhi standar yang digunakan dalam penelitian. Nilai Response Time belum memenuhi waktu respons optimal sebesar 0,1 detik, sedangkan Error Rate sebesar 100% menunjukkan bahwa seluruh permintaan gagal diproses. Kondisi tersebut disebabkan oleh fitur Post Perizinan yang memerlukan Firebase Cloud Messaging (FCM) token, sementara virtual user pada Apache JMeter tidak memiliki token tersebut sehingga proses pengiriman permintaan tidak dapat diproses oleh sistem.

Fitur Get Riwayat Harian Kelas pada website memperoleh Load Time dan Response Time sebesar 0,211 detik, Throughput sebesar 2,5 request/s, Error Rate sebesar 0%, serta Latency sebesar 0,0868 detik. Nilai Load Time berada

di bawah 3 detik, sedangkan Latency berada di bawah 1 detik, sehingga menunjukkan bahwa halaman dapat dimuat dengan cepat dan server mampu memberikan respons awal dengan baik. Nilai Response Time masih berada di atas waktu respons optimal sebesar 0,1 detik, namun seluruh permintaan berhasil diproses tanpa kesalahan selama pengujian berulang.

Fitur Get Riwayat Harian Mapel pada website memperoleh Load Time dan Response Time sebesar 0,184 detik, Throughput sebesar 2,5 request/s, Latency sebesar 0,0856 detik, dan Error Rate sebesar 1,00%. Nilai Load Time dan Latency telah memenuhi standar yang digunakan, sedangkan Response Time masih berada di atas waktu respons optimal sebesar 0,1 detik. Nilai Error Rate sebesar 1,00% masih berada di bawah batas 5%, sehingga menunjukkan bahwa fitur tetap mampu mempertahankan keandalannya selama pengujian berulang.

Secara keseluruhan, hasil gorilla testing menunjukkan bahwa sebagian besar fitur mampu mempertahankan performa yang stabil ketika dijalankan secara berulang. Hal tersebut ditunjukkan oleh nilai Load Time, Latency, dan Error Rate yang memenuhi standar pada sebagian besar fitur. Parameter Response Time pada seluruh fitur masih berada di atas waktu respons optimal sebesar 0,1 detik, sedangkan Error Rate sebesar 100% pada fitur Post Perizinan dipengaruhi oleh keterbatasan lingkungan pengujian, yaitu tidak tersedianya FCM token pada virtual user Apache JMeter, bukan disebabkan oleh penurunan performa sistem.

BAB 5 KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil penelitian dan pengujian performa sistem absensi Presensiku menggunakan metode *load testing* dan *gorilla testing* dengan Apache JMeter, maka dapat diambil kesimpulan sebagai berikut.

- a. Analisis performa sistem absensi berbasis *website* dan *Mobile* di SMK Taruna Bakti Kertosono dilakukan menggunakan metode *load testing* dan *gorilla testing* dengan Apache JMeter. Pengujian dilakukan pada 3 skenario *load testing* (*website* admin, *Mobile* siswa, *Mobile* guru) dan 1 skenario *gorilla testing* dengan total 48 *test case* (44 *load testing* dan 4 *gorilla testing*). Hasil pengujian menunjukkan bahwa sistem mampu menangani banyak pengguna secara bersamaan, namun tingkat performanya berbeda pada setiap fitur dan jenis *request*. Hal ini terlihat dari perbedaan nilai parameter seperti *Throughput*, *Response Time*, dan *Error Rate* pada tiap modul, di mana fitur berbasis *GET* cenderung lebih stabil dibandingkan fitur *POST* yang memiliki proses pengiriman data lebih kompleks.
- b. Sistem belum sepenuhnya mampu mempertahankan stabilitas dan kecepatan *respons* saat beban pengguna meningkat. Hal ini dibuktikan dari hasil pengujian yang menunjukkan *Load Time* berkisar antara 0,001 detik hingga 4,528 detik, serta *Response Time* tertinggi mencapai 9,743 detik pada fitur *Get Riwayat Absensi Kelas Mobile* guru. Selain itu, pada beberapa skenario ditemukan *Error Rate* sebesar 100% pada fitur *Post Perizinan*, *Post Laporkan*, dan *Post Foto*. Hasil analisis menunjukkan bahwa *error* tersebut tidak sepenuhnya disebabkan oleh kegagalan *server backend*, melainkan dipengaruhi oleh faktor pendukung lainnya. Fitur *Post Perizinan* dan *Post Laporkan*, *error* terjadi akibat kegagalan pengiriman notifikasi melalui *Firebase Cloud Messaging* (FCM) karena token perangkat tidak valid atau tidak tersedia pada beberapa request saat pengujian berlangsung. Sementara itu, *error* pada fitur *Post Foto* disebabkan oleh belum tersedianya VPS yang

mendukung proses unggah dan pemrosesan data foto sehingga permintaan yang dikirim tidak dapat diproses secara optimal. Meskipun demikian, sebagian besar fitur lainnya masih mampu mempertahankan *Error Rate* sebesar 0% dengan waktu *respons* yang relatif stabil. Hasil tersebut menunjukkan bahwa sistem memiliki performa yang cukup baik, namun masih diperlukan optimalisasi pada integrasi layanan notifikasi dan infrastruktur server agar stabilitas serta kecepatan respons dapat dipertahankan pada kondisi beban yang lebih tinggi.

5.2 Saran

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, terdapat beberapa saran yang dapat diberikan untuk pengembangan sistem, yaitu:

- a. Perlu penanganan dan validasi yang lebih baik pada *Firebase Cloud Messaging* (FCM), terutama dalam pengelolaan token perangkat agar tidak terjadi kegagalan pengiriman notifikasi saat sistem menerima banyak *request* secara bersamaan.
- b. Perlu dilakukan peningkatan pada stabilitas fitur yang memiliki *Response Time* tinggi, khususnya pada fitur *Get Riwayat Absensi Kelas Mobile* guru yang mencapai 9,743 detik, dengan melakukan optimasi *query database* dan pengurangan beban proses pada server.
- c. Mengurangi beban server saat pengiriman data seperti foto dan perizinan, disarankan menggunakan optimasi proses upload file, seperti kompresi *file* atau pemisahan proses *upload* dan penyimpanan data.
- d. Sistem disarankan untuk menerapkan *caching* pada data yang sering diakses, seperti jadwal, absensi, dan *QR Code*, agar dapat meningkatkan kecepatan respon dan mengurangi *request* langsung ke server.

DAFTAR PUSTAKA

- Abda'u, P. D., Susanto, A., Supriyono, A. R., & Prasetyanti, D. N. (2024). Perbandingan Kinerja Antara Gatling Dan Apache Jmeter Pada Uji Beban Restful Api. *Infotekmesin*, 15(1), 211–215.
- Alga Saputra. (2023). *Analisis Performa Website Sman 1 Rogojampi*.
- Amin, N. F., Garancang, S., & Abunawas, K. (2023). Konsep Umum Populasi Dan Sampel Dalam Penelitian. *Jurnal Pilar*, 14(1), 15–31.
- Arief Anastiar Suharsono. (2024). *Rancang Bangun Analisis Performa Website*.
- Armawati, Y., Siambaton, M. Z., & Santoso, H. (2023). Aplikasi Absensi Online Civitas Akademik Smk Swasta Abdi Sejati Kerasaan I Dengan Menggunakan Algoritma Sequential Searching. *Buletin Utama Teknik Vol*, 18(3). <https://core.ac.uk/download/pdf/578296415.pdf>
- Azizah, N., Jannah, R., Sudur, M., Rahman, Z., & Muhammad, J. (2024). Meningkatkan Efetifitas Penggunaan Absensi Digital Dalam Rekapitulasi Guru Di Sekolah Dasar (Sd) Desa Trebungan. *Jurnal Masyarakat Berdikari Dan Berkarya (Mardika)*, 2(1), 1–9. <https://doi.org/10.55377/Mardika.V2i1.9732>
- Barus, A. C., Harungguan, J., & Manulu, E. (2021a). Pengujian Api Website Untuk Perbaikan Performansi Aplikasi Ditenun. *Journal Of Applied Technology And Informatics Indonesia*, 1(2), 14–21.
- Barus, A. C., Harungguan, J., & Manulu, E. (2021b). Pengujian Api Website Untuk Perbaikan Performansi Aplikasi Ditenun. *Journal Of Applied Technology And Informatics Indonesia*, 1(2), 14–21.
- Dhaifullah, I. R., Salsabila, A. A., & Yaqin, M. A. (2022a). Survei Teknik Pengujian Software. *Journal Automation Computer Information System*, 2(1), 31–38.
- Dhaifullah, I. R., Salsabila, A. A., & Yaqin, M. A. (2022b). Survei Teknik Pengujian Software. *Journal Automation Computer Information System*, 2(1), 31–38.
- Dwinur Andrianto, L., & Fatrianto Suyatno, D. (2024). *Analisis Performa Load testing Antara Mysql Dan Nosql MongoDB Pada Restapi Nodejs Menggunakan Postman*. <https://ejournal.unesa.ac.id/index.php/Jeisbi/article/view/58157>

- Endra, R. Y., Aprilinda, Y., Dharmawan, Y. Y., & Ramadhan, W. (2022). Analisis Perbandingan Bahasa Pemrograman Php Laravel Dengan Php Native Pada Pengembangan *Website*. *Expert*, 11(1), 346061.
- Ginasari, N. L. A. S., Wibawa, K. S., Wirdiani, A., & Kadek, N. (2021a). Pengujian Stress Testing Api Sistem Pelayanan Dengan Apache Jmeter. *Jurnal Ilmiah Teknologi Dan Komputer*, 2(3), 552–557. <https://pdfs.semanticscholar.org/0eee/223247dd22f8ec17c74ddac57dd0f220f687.Pdf>
- Ginasari, N. L. A. S., Wibawa, K. S., Wirdiani, A., & Kadek, N. (2021b). Pengujian Stress Testing Api Sistem Pelayanan Dengan Apache Jmeter. *Jurnal Ilmiah Teknologi Dan Komputer*, 2(3), 552–557.
- Hadinata, W., & Stianingsih, L. (2024). Analisis Perbandingan Performa Restfull Api Antara Express. Js Dengan Laravel Framework. *Jurnal Informatika Dan Teknik Elektro Terapan*, 12(1).
- Hamidah, I., Haromain, I., & Drehem, I. M. (2025). Evaluasi Pengujian Kinerja Menggunakan Jmeter Untuk Menunjang Stabilitas Aplikasi Layanan Perbankan Pada Pt Bank Rakyat Indonesia Tbk. *Dbesti: Journal Of Digital Business And Technology Innovation*, 2(1), 114–126.
- Hasibuan, A. N., & Dirgahayu, T. (2021). Pengujian Dengan Unit Testing Dan *Test case* Pada Proyek Pengembangan Modul Manajemen Pengguna. *Automata*, 2(1).
- Hermansyah, H., Wijaya, R. F., & Wahyuni, S. (2024). Desain Aplikasi Cinta Mangrove Berbasis *Mobile* Di Desa Kota Pari Dengan Metode Waterfall. *Senashtek 2024*, 2(1), 42–48.
- Hidayat, A. M. N., Rizaldy, A., Hartono, N., & Harwalis, H. (2024). Pengujian Kinerja Web Server Elastic Cloud Compute (Ec2) Free Tier Pada Amazon Web Service (Aws) Menggunakan Jmeter. *Jurnal Sistem Informasi Dan Informatika (Simika)*, 7(1), 82–94. <https://www.lppm-unbaja.ac.id/ejournal/index.php/jsii/article/view/3208>
- Hidayat, M. A. R., Izzati, N. N., Prirhatama, A. R. P., Patria, Y., & Kurmilasari, S. (2025). Pengujian Perangkat Lunak Pada *Website* Ka'cake: Implementasi Unit Testing, Integration Testing, System Testing, Dan Validation Testing Untuk Menjamin Kualitas Dan Keandalan Sistem. *Jati (Jurnal Mahasiswa Teknik Informatika)*, 9(4), 6805–6811.

- Indrianto, I. (2023). Performance Testing On Web Information System Using Apache Jmeter And Blazemeter. *Jurnal Ilmiah Ilmu Terapan Universitas Jambi*, 7(2), 138–149.
- Indriyani, F., Anwar, S., Haidir, A., Irfiani, E., Sanjaya, Y. R., Firmansyah, N., & Khoirunnisa, T. R. (2025). Pelatihan Dan Pendampingan Penggunaan Aplikasi *Mobile* Dasawisma Untuk Pengolahan Data Mandiri Warga. *Pengabdian Kepada Masyarakat Indonesia Sean (Abdimas Sean)*, 3(01), 34–40.
- Mardiati, D., & Saputra, Y. (2025). Implementasi Sistem Informasi Manajemen Klinik Menggunakan Metode Black Box Testing. *Jurnal Informatika Dan Teknik Elektro Terapan*, 13(1).
- Marpaung, N. L., Hutabarat, S., & Izzi, M. (2022). Pembuatan Aplikasi Absensi Karyawan Menggunakan BarCode Berbasis *Website*. *Fordicate*, 1(2), 180–191.
<https://Jurnal.Mdp.Ac.Id/Index.Php/Fordicate/Article/View/2417/732>
- Mukti, S. (2024). Sistem Informasi Absensi Guru Dan Siswa Smk Muhammadiyah 2 Kota Tegal Berbasis *Website* Menggunakan Framework *Codeighiter*. *Jurnal Review Pendidikan Dan Pengajaran (Jrpp)*, 7(2), 5234–5238.
<https://Journal.Universitaspahlawan.Ac.Id/Index.Php/Jrpp/Article/View/27870>
- Mulana, L., Prihandani, K., & Rizal, A. (2022). Analisis Perbandingan Kinerja Framework *Codeigniter* Dengan Express. Js Pada Server Restful Api. *Jurnal Ilmiah Wahana Pendidikan*, 8(16), 316–326.
<https://Doi.Org/Https://Doi.Org/10.5281/Zenodo.7067707>
- Mutmainnah, K., & Ihsan, A. N. (2024). Perbandingan Kualitas Fitur Pembelian Tiket Konser Bilibli. Com Dan Tiket. Com Menggunakan Metode *Load testing*. *Jurnal Ilmiah Informatika Komputer*, 29(2), 103–116.
<https://Ejournal.Gunadarma.Ac.Id/Index.Php/Infokom/Article/View/11011>
- Nababan, P., Jamaluddin, J., Perangin-Angin, R., & Purba, E. N. (2022). Sistem Informasi Absensi Siswa Pada Smk Negeri 1 Pantai Labu Berbasis Web Dengan Whatsapp Gateway. *Tamika: Jurnal Tugas Akhir Manajemen Informatika & Komputerisasi Akuntansi*, 2(2), 61–67.
<https://Doi.Org/10.46880/Tamika.Vol2no2.Pp61-67>

- Noviana, R. (2022). Pembuatan Aplikasi Penjualan Berbasis Web Monja Store Menggunakan Php Dan Mysql. *Jurnal Teknik Dan Science*, 1(2), 112–124. <https://Journal.Admi.Or.Id/Index.Php/Jts/Article/View/128>
- Nurhasanah, A., Pribadi, R. A., & Nur, M. D. (2021). Analisis Kurikulum 2013. *Didaktik: Jurnal Ilmiah Pgsd Stkip Subang*, 7(02), 484–493.
- Prakasa, B. Y. (2024). Auto Bayar App: Software Quality Assurance At Pt. Xy. *Procedia Of Engineering And Life Science*, 5, 262–272.
- Raffles, S. A., & Nasution, M. I. P. (2024). Peran Penting Pengolahan Data Dalam Transformasi Bisnis Melalui Analisis. *Jurnal Rimba: Riset Ilmu Manajemen Bisnis Dan Akuntansi*, 2(1), 341–348.
- Raweyai, S. S., & Widiyari, I. R. (2024). Performance Testing Of Academic Website Using Load testing Method Supported By Apache Jmeter At Xyz University. *Jurnal Teknik Informatika (Jutif)*, 5(3), 721–730. <https://doi.org/10.52436/1.Jutif.2024.5.3.1796>
- Ruliansyah, R., Huda, B., & Hananto, A. L. (2023a). Penerapan Software Testing Life Cycle Pada Pengujian Otomatisasi Platform Dzikra. *Computer Science Research And Its Development Journal*, 15(1), 1–11.
- Ruliansyah, R., Huda, B., & Hananto, A. L. (2023b). Penerapan Software Testing Life Cycle Pada Pengujian Otomatisasi Platform Dzikra. *Computer Science Research And Its Development Journal*, 15(1), 1–11.
- Sari, R. N., Astuti, W., & Suminar, L. (2025). Dampak Industri Pt. Semen Indonesia Pabrik Tuban Terhadap Kondisi Permukiman Di Sekitarnya. *Desa-Kota: Jurnal Perencanaan Wilayah, Kota, Dan Permukiman*, 7(1), 149–161.
- Setiawan, H., & Enda, D. (2025). Pengujian Load testing Website Jurusan Teknik Informatika Politeknik Negeri Bengkalis. *Jekin-Jurnal Teknik Informatika*, 5(1), 1–12.
- Shalsabilla, S. Y., Wahyuni, E. D., & Wibowo, N. C. (2024). Implementasi Blackbox Automation Testing Pada Aplikasi Donor Menggunakan Framework Stlc Dalam Lingkup Pengembangan Agile Scrum. *Jati (Jurnal Mahasiswa Teknik Informatika)*, 8(2), 2261–2269.
- Susanto, P. C., Arini, D. U., Yuntina, L., Soehaditama, J. P., & Nuraeni, N. (2024). Konsep Penelitian Kuantitatif: Populasi, Sampel, Dan Analisis Data (Sebuah Tinjauan Pustaka). *Jurnal Ilmu Multidisplin*, 3(1), 1–12.

- Syifa, F. N., Nabil, T. N. A., Arifin, D. H., & Sidik, R. (2025). Uji Kualitas *Website* Admin Travel Booking Menggunakan Halstead's Metric Dan Equivalence Partitioning. *Jurnal Nasional Teknologi Dan Sistem Informasi*, 11(1), 67–72.
- Tejaya, W., Rahman, S., Munir, A., Informatika, T., & Kharisma Makassar, S. (2023). *Pengujian Website Invitees Menggunakan Metode Load testing Dengan Apache Jmeter*. <https://Jurnal.Kharisma.Ac.Id/Kharismatech/Article/View/305>
- Ushakova, I., Plokha, O., & Skorin, Y. (2022). Approaches To Web Application Performance Testing And Real-Time Visualization Of Results. *Bulletin Of Kharkov National AutoMobile And Highway University*, 96, 71. <https://doi.org/10.30977/Bul.2219-5548.2022.96.0.71>
- Wiyatnanto, E., & Haris, A. I. (2021). Kinerja Arsitektur Interoperabilitas Menggunakan Government Service Bus (Gsb) Dan Peer To Peer (P2p). *Ultimatics: Jurnal Teknik Informatika*, 13(1), 7–11.
- Zahra, N. R. D., Putri, A. W., Azzahra, K. S., Widhiwipati, D. R., Nurfajri, M. I., Mindara, G. P., & Wicaksono, A. (2025). Pengujian Pada *Website* Smartpetscare Untuk Layanan Grooming Hewan Menggunakan Metode Black Box Testing. *Jati (Jurnal Mahasiswa Teknik Informatika)*, 9(1), 378–383.

LAMPIRAN

Lampiran 1 Kegiatan Wawancara dengan admin



Lampiran 2 Kegiatan Wawancara dengan Siswa



Lampiran 3 Kegiatan Wawancara dengan Guru



Lampiran 4 Hasil Wawancara dengan Admin

Narasumber : Sita Elisa Permatasari, S.Pd.

Jabatan : Guru BK (Bimbingan Konseling)

Waktu : Senin, 17 Maret 2025

No	Pertanyaan	Jawaban
1	Bagaimana proses pencatatan absensi siswa dilakukan saat ini di sekolah?	Saat ini, pencatatan absensi masih dilakukan secara manual setiap hari. Setelah kegiatan belajar selesai, biasanya ketua kelas akan mengumpulkan buku absensi dan menyerahkannya ke guru BK untuk direkap.
2	Apa saja kendala yang sering Anda hadapi dalam mengelola data absensi siswa	Saat ini, pencatatan absensi masih dilakukan secara manual setiap hari. Setelah kegiatan belajar selesai, biasanya ketua kelas akan mengumpulkan buku absensi dan menyerahkannya ke guru BK untuk direkap.
3	Apakah Anda merasa sistem absensi saat ini sudah efisien dan mudah digunakan?	Belum, karena sistem absensi sekarang masih kurang akurat. Misalnya, ada siswa yang sebenarnya hadir tetapi dicatat alpa hingga jam pulang hanya karena tidak ada di kelas saat guru masuk. Siswa juga sering kali pasif dan tidak mengklarifikasi kesalahan tersebut.

4	Seberapa penting fitur rekap otomatis dan laporan absensi harian/bulanan bagi Anda?	Sangat penting, karena berhubungan dengan disiplin siswa. Tanpa sistem rekap otomatis, bisa jadi ada siswa yang tidak masuk selama beberapa hari tanpa terdeteksi. Dengan adanya rekap otomatis, kita dapat langsung mengetahui dan menindaklanjuti lebih cepat kepada siswa yang bersangkutan.
5	Bagaimana pandangan Anda terhadap pentingnya pengujian sistem sebelum sistem absensi diimplementasikan secara penuh?	Pengujian sangat penting dilakukan sebelum sistem digunakan secara penuh oleh seluruh pengguna. Tanpa pengujian, kami tidak bisa memastikan apakah sistem mampu berjalan dengan stabil saat diakses oleh banyak orang secara bersamaan. Terutama saat jam-jam sibuk.

Nganjuk, 17 Maret 2025

Peneliti,



Evita Nur Sianturi

Mengetahui,

Guru BK SMK Taruna Bakti

Kertosono



Siti Elisa Permatasari, S.Pd.

Lampiran 5 Hasil Wawancara dengan Siswa

Narasumber : Azzahra Putri Muyasyaroh

Jabatan : Siswa

Kelas : X TKJ 2

Waktu : Senin, 17 Maret 2025

No	Pertanyaan	Jawaban
1	Bagaimana alur absensi yang kamu jalani setiap hari di sekolah?	Proses absensi dilakukan secara manual oleh guru.
2	Di mana kamu melakukan absensi?	Proses absensi dilakukan secara manual oleh guru.
3	Pernahkah kamu atau temanmu titip absen?	Saya tidak pernah menitip absen, tetapi lebih sering memberi tahu izin melalui chat kepada teman.
4	apa saja kendala / kekurangan dari sistem absensi manual menurutmu?	Masalah dalam rekap absensi termasuk kesalahan pencatatan dan ketidakakuratan, serta buku rekap pribadi yang kadang hilang, dan izin lewat WhatsApp tanpa surat.
5	Apakah kamu punya HP untuk absensi digital?	Tidak ada masalah, semua siswa memiliki ponsel.

Nganjuk, 17 Maret 2025

Peneliti,



Evita Nur Sianturi

Mengetahui,

Siswa BK SMK Taruna Bakti

Kertosono



Azzahra Putri Muyasyaroh

Narasumber : Diva Maulida Agustina Putri

Jabatan : Siswa

Kelas : X AKL 1

Waktu : Senin, 17 Maret 2025

No	Pertanyaan	Jawaban
1	Bagaimana alur absensi yang kamu jalani setiap hari di sekolah?	Proses absensi dengan buku yang disediakan oleh sekolah dan guru mencatat kehadiran saat jam mata Pelajaran.
2	Di mana kamu melakukan absensi?	Absensi dilakukan di dalam kelas masing-masing.
3	Pernahkah kamu atau temanmu titip absen?	Tidak pernah menitip absen, tetapi pernah bolos tidak masuk bilanganya izin ke teman.
4	apa saja kendala / kekurangan dari sistem absensi manual menurutmu?	Ada kesalahan dalam pencatatan, seperti ketika guru mencatat siswa sebagai tidak hadir padahal terlambat.
5	Apakah kamu punya HP untuk absensi digital?	Iya, saya punya Hp

Nganjuk, 17 Maret 2025

Peneliti,



Evita Nur Sianturi

Mengetahui,

Siswa BK SMK Taruna Bakti

Kertosono



Divi Maulida Agustina Putri

Lampiran 6 Hasil Wawancara dengan Guru

Narasumber : Krismon Nuvi F., S.Pd.

Jabatan : Guru

Waktu : Senin, 17 Maret 2025

No	Pertanyaan	Jawaban
1	Apakah SMK taruna Bakti kertosono saat ini sudah memiliki sistem absensi digital?	Belum, saat ini SMK Taruna Bakti Kertosno masih menggunakan sistem absensi manual.
2	Apakah ada keinginan untuk menerapkan sistem absensi berbasis digital?	Belum, saat ini SMK Taruna Bakti Kertosno masih menggunakan sistem absensi manual.
3	Apa alasan utama sekolah ingin mengembangkan sistem absensi berbasis digital?	Alasan SMK Taruna Bakti Kertosono ingin mengembangkan sistem absensi digital karena proses absensi manual sering memakan waktu, rawan kesalahan pencatatan, dan sulit saat merekap data kehadiran. Dengan sistem digital, absensi bisa dilakukan lebih cepat, data langsung tercatat otomatis, dan laporan bisa dibuat dengan mudah.
4	Apakah sudah ada proses pengembangan sistem nya?	Ya, pengembangan sistem absensi digital sudah mulai direncanakan.
5	Saat ini sudah sampai pada pengembangan sistem absensinya?	Masih dalam tahap perancangan dan persiapan awal sistem absensi.
6	ada berapa jumlah murid, guru, staf admin?	Sekitar 1000 murid, 40 guru, dan 5 staf admin.
7	Siapa saja yang nantinya akan menggunakan sistem absensi ini?	Siswa, guru, dan staf admin.
8	Apakah diperlukan pengujian untuk mengetahui performance aplikasi sistem absensi?	Ya, sangat diperlukan agar sistem berjalan optimal saat digunakan.

9	Sejauh mana harapan sekolah terhadap keberhasilan sistem ini?	Sekolah berharap sistem dapat berjalan lancar dan digunakan jangka panjang
10	Kira-kira, berapa detik ideal yang dibutuhkan untuk 1 kali proses absensi yang sukses?	Sekitar 3 detik per orang.

Nganjuk, 17 Maret 2025

Peneliti,



Evita Nur Sianturi

Mengetahui,

Guru SMK Taruna Bakti Kertosono



Krismon Nuvi F., S.Pd.