

Pengembangan Aplikasi Monitoring Kegiatan Kuliah Kerja Nyata Universitas Hasanuddin Menggunakan Metode *Waterfall*

Bulqis Ramadani¹, Hendra², Muhammad Sadno³, Edy Saputra Rusdi⁴

^{1,2,3,4}Sistem Informasi, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin

¹bulqisramadani.4@gmail.com, ²hendra@unhas.ac.id, ³muh.sadno@unhas.ac.id, ⁴edy_saputra@sci.unhas.ac.id

Abstract

The Community Service program is an academic initiative aimed at providing students with direct experience in applying their knowledge in society while developing social skills. At Universitas Hasanuddin, students are required to report their activities to their supervising lecturers as part of monitoring and evaluation. However, the manual reporting system is inefficient and hinders effective evaluation. This research aims to develop an Android-based monitoring application for Community Service activities to improve reporting efficiency and accelerate the evaluation process. The application is developed using the Waterfall model, with Java for Android and Node.js for the API connected to a MySQL database. System testing is conducted using the User Acceptance Testing method to evaluate user acceptance and determine if the system is ready for operational use. The User Acceptance Testing results show an average score of 3.83 from the supervising lecturers and 3.47 from the students, indicating that the application is acceptable and meets user needs in supporting the Community Service program. Therefore, the monitoring application for Community Service at Universitas Hasanuddin was successfully developed, functioning according to the designed specifications, and is accepted by the users.

Keywords: Android, Waterfall, Monitoring, Community Service, User Acceptance Testing

Abstrak

Kuliah Kerja Nyata merupakan program akademik yang bertujuan untuk memberikan pengalaman langsung bagi mahasiswa dalam menerapkan ilmu di masyarakat serta mengembangkan keterampilan sosial. Dalam pelaksanaan Kuliah Kerja Nyata di Universitas Hasanuddin, mahasiswa diwajibkan untuk melaporkan setiap kegiatan kepada dosen pembimbing sebagai bagian dari proses monitoring dan evaluasi. Namun, sistem pelaporan yang masih dilakukan secara manual dinilai kurang efisien dan menghambat efektivitas evaluasi. Penelitian ini bertujuan untuk mengembangkan sebuah aplikasi monitoring kegiatan Kuliah Kerja Nyata berbasis Android guna meningkatkan efisiensi dalam pelaporan dan mempercepat proses evaluasi. Aplikasi ini dikembangkan menggunakan metode *Waterfall*, dengan bahasa pemrograman Java untuk android serta Node.js untuk layanan API yang terhubung ke database MySQL. Pengujian sistem dilakukan dengan metode *User Acceptance Testing* untuk mengevaluasi tingkat penerimaan pengguna dan menilai apakah sistem siap digunakan dalam lingkungan operasional yang sebenarnya. Hasil *User Acceptance Testing* menunjukkan skor rata-rata 3,83 dari dosen pembimbing dan 3,47 dari mahasiswa, yang menunjukkan bahwa aplikasi dapat diterima dan telah memenuhi kebutuhan pengguna dalam mendukung pelaksanaan Kuliah Kerja Nyata. Dengan demikian, aplikasi monitoring kegiatan Kuliah Kerja Nyata Universitas Hasanuddin berbasis android berhasil dikembangkan dan berfungsi sesuai dengan spesifikasi yang dirancang dan dapat diterima oleh pengguna.

Kata kunci: Android, *Waterfall*, Monitoring, Kuliah Kerja Nyata, *User Acceptance Testing*

©This work is licensed under a Creative Commons Attribution -ShareAlike 4.0 International License

1. Pendahuluan

Dalam pendidikan tinggi, salah satu aspek penting adalah Kuliah Kerja Nyata (KKN), sebuah program akademik yang menggabungkan pendidikan, penelitian, dan pengabdian kepada masyarakat. KKN bertujuan untuk memberikan pengalaman langsung kepada mahasiswa dalam menerapkan ilmu yang telah dipelajari di bangku perkuliahan ke dalam kehidupan masyarakat [1].

Dalam pelaksanaan KKN, mahasiswa diwajibkan untuk melaporkan setiap kegiatan yang telah dilakukan selama masa pengabdian. Laporan ini menjadi dasar bagi Dosen Pembimbing KKN (DPK) dalam mengevaluasi progres serta pencapaian mahasiswa. Proses monitoring ini sangat penting untuk memastikan bahwa tujuan dan sasaran dari program KKN dapat berjalan sesuai rencana. Namun, saat ini proses monitoring kegiatan KKN di Universitas Hasanuddin

masih dilakukan secara manual. Mahasiswa harus menyerahkan laporan tertulis atau mengirimkan dokumentasi melalui pesan singkat kepada Dosen Pembimbing KKN (DPK).

Menurut laporan dari Direktur Transformasi Pendidikan dan Inovasi Pembelajaran Universitas Hasanuddin, jumlah peserta KKN Universitas Hasanuddin pada tahun 2024 mencapai 4.289 mahasiswa [2]. Dengan skala peserta KKN yang besar dan lokasi penempatan yang tersebar di berbagai daerah, sistem manual ini menjadi semakin tidak efisien. Mahasiswa seringkali menghadapi kendala dalam menyampaikan laporan tepat waktu akibat keterbatasan akses komunikasi, sedangkan dosen pembimbing mengalami kesulitan dalam memantau perkembangan mahasiswa secara langsung. Selain itu, keterlambatan dalam pengiriman laporan juga

menghambat proses evaluasi dan pemberian umpan balik dari dosen pembimbing kepada mahasiswa.

Seiring dengan perkembangan teknologi, berbagai institusi pendidikan mulai beralih ke sistem berbasis digital untuk meningkatkan efisiensi dan efektivitas dalam berbagai aspek akademik. Dalam konteks KKN, pemanfaatan teknologi juga dapat dipertimbangkan untuk membantu mengatasi permasalahan yang ada, terutama dalam aspek monitoring dan evaluasi kegiatan mahasiswa. Dengan adanya sistem yang lebih modern, berbagai kendala yang muncul dalam proses pelaksanaan KKN berpotensi untuk diminimalkan, sehingga program ini dapat berjalan lebih efektif dan terstruktur.

Salah satu teknologi yang berpotensi untuk diterapkan dalam sistem monitoring KKN adalah aplikasi berbasis Android. Sebagai sistem operasi mobile terbesar di dunia, Android kini menjadi platform utama dalam pengembangan aplikasi mobile yang mendukung berbagai kebutuhan pengguna. Data terbaru menunjukkan bahwa pada November 2024, pengguna Android mencapai 93,45% dari total pengguna smartphone di Indonesia [3]. Dominasi Android ini disebabkan oleh sifatnya yang *open-source*, fleksibilitas yang tinggi, serta dukungan ekosistem aplikasi yang luas.

Berbagai penelitian sebelumnya telah mengembangkan sistem monitoring KKN atau kegiatan sejenis berbasis Android dengan fokus dan pendekatan yang berbeda. Penelitian oleh Hidayanti et al, memanfaatkan Firebase Realtime Database untuk memungkinkan pelaporan kegiatan secara langsung, namun kinerjanya sangat bergantung pada koneksi internet stabil sehingga efektivitasnya menurun di wilayah dengan jaringan terbatas [4]. Sementara itu, penelitian Syah et al di Universitas Lampung berhasil meningkatkan kemudahan akses informasi KKN, tetapi belum mengintegrasikan fitur pemantauan progres secara real-time maupun fasilitas komunikasi langsung antara mahasiswa dan dosen pembimbing [5]. Adapun penelitian Akib & Muin, telah menghadirkan sistem yang terintegrasi dengan sistem akademik universitas dan mampu beroperasi di lokasi dengan konektivitas terbatas, namun fitur komunikasi interaktif serta notifikasi instan yang mendukung respons cepat masih belum dioptimalkan [6].

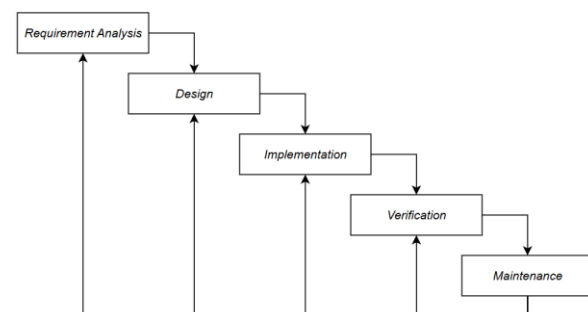
Berdasarkan kondisi tersebut, penelitian ini mengisi kekosongan dengan mengembangkan aplikasi monitoring KKN berbasis Android yang mengintegrasikan pelaporan kegiatan terstruktur, fitur komunikasi dua arah antara mahasiswa dan dosen pembimbing, notifikasi real-time, serta optimasi kinerja pada area dengan konektivitas terbatas. Pendekatan ini diharapkan mampu menjadikan pelaksanaan KKN di Universitas Hasanuddin lebih efektif, efisien, dan responsif dibandingkan sistem monitoring sebelumnya. Melalui sistem ini, proses pelaporan kegiatan dapat dilakukan secara lebih cepat

dan akurat, evaluasi dapat dipercepat, serta komunikasi antara mahasiswa dan dosen pembimbing menjadi lebih mudah dan terkoordinasi.

2. Metode Penelitian

Metode pengembangan sistem yang digunakan dalam penelitian ini adalah metode *waterfall*, salah satu model dalam *System Development Life Cycle (SDLC)* yang menggunakan pendekatan sistematis dan berurutan dalam pengembangan perangkat lunak. Model ini terdiri dari beberapa tahap utama, yaitu *requirement analysis* (analisis kebutuhan), *design* (perancangan sistem), *implementation* (pengkodean), *verification* (verifikasi dan pengujian sistem), dan *maintenance* (pemeliharaan sistem). Setiap fase harus diselesaikan sepenuhnya sebelum beralih ke fase berikutnya, yang meminimalkan kemungkinan kesalahan dan memastikan proses pengembangan yang terstruktur [7].

Metode waterfall dipilih karena alur pengembangan sistem monitoring KKN ini memiliki kebutuhan yang relatif jelas sejak awal berdasarkan hasil observasi dan wawancara dengan pihak terkait. Selain itu, metode ini memberikan dokumentasi yang lengkap di setiap tahap, memudahkan pelacakan apabila terjadi kesalahan.



Gambar 1. Metode *Waterfall*

1) *Requirement Analysis* (Analisis Kebutuhan): Tahap ini merupakan langkah awal yang bertujuan untuk mengidentifikasi dan mendefinisikan kebutuhan-kebutuhan dari sistem yang akan dibangun. Pada fase ini, dilakukan pengumpulan informasi mengenai fungsi-fungsi yang dibutuhkan oleh pengguna dan bagaimana sistem diharapkan bekerja. Tahapan ini dilakukan dengan mengumpulkan kebutuhan pengguna melalui observasi terhadap proses monitoring KKN yang berjalan selama ini dan wawancara dengan pihak-pihak yang terkait dengan pelaksanaan KKN.

2) *Design* (Perancangan Sistem): Setelah kebutuhan sistem terdefinisi, tahap selanjutnya adalah membuat rancangan sistem yang meliputi desain arsitektur, antarmuka pengguna, basis data, dan alur proses dalam sistem. Desain ini digunakan sebagai cetak biru untuk pengembangan perangkat selanjutnya.

3) *Implementation* (Pengkodean): Tahap implementasi merupakan proses penerjemahan rancangan sistem ke dalam kode program sesuai dengan spesifikasi yang telah ditentukan. Aplikasi dibangun menggunakan

bahasa pemrograman Java untuk platform Android. *Backend* API dibangun menggunakan Node.js, serta *database* MySQL sebagai penyimpanan data. Untuk mengantisipasi risiko kegagalan pada tahap implementasi, digunakan strategi *modular coding* agar setiap fitur dapat dibangun, diuji, dan diperbaiki secara terpisah sebelum digabungkan.

4) *Verification* (Verifikasi dan Pengujian): Pada tahap ini, pengujian dilakukan untuk memastikan bahwa sistem yang telah diimplementasikan berjalan sesuai dengan spesifikasi yang telah ditetapkan dan memenuhi kebutuhan pengguna. Pengujian dilakukan dengan dua pendekatan utama, yaitu *User Acceptance Testing* (UAT) untuk mengevaluasi fungsionalitas, kemudahan penggunaan, dan pengalaman pengguna aplikasi dalam skenario nyata, serta *load testing* untuk mengukur kinerja sistem saat menerima banyak permintaan secara bersamaan. Strategi pengujian meliputi penentuan *baseline* performa normal, simulasi beban bertahap untuk mengamati respons sistem, pemantauan *response time*, serta identifikasi *bottleneck* yang dapat menyebabkan penurunan performa. Jika ditemukan potensi masalah, dilakukan penyesuaian pada konfigurasi server, optimasi *query database*, atau perbaikan kode untuk meningkatkan efisiensi.

5) *Maintenance* (Pemeliharaan): Pada tahap ini, sistem yang telah dikembangkan akan dipantau dan diperbarui sesuai kebutuhan. Pemeliharaan dapat meliputi perbaikan terhadap kesalahan yang ditemukan setelah implementasi, peningkatan fitur, serta penyesuaian terhadap perubahan lingkungan atau kebutuhan pengguna.

3. Hasil dan Pembahasan

3.1. Analisis Kebutuhan

Berdasarkan permasalahan yang telah diidentifikasi, sistem monitoring KKN berbasis Android perlu dirancang dengan fitur-fitur utama yang mendukung pelaporan kegiatan secara *real-time* serta komunikasi yang lebih efektif antara mahasiswa dan dosen pembimbing. Beberapa kebutuhan sistem yang harus dipenuhi antara lain:

3.1.1. Kebutuhan Fungsional

- 1) Mahasiswa dapat membuat dan mengunggah laporan kegiatan KKN secara berkala.
- 2) Dosen pembimbing dapat memantau laporan mahasiswa dan memberikan umpan balik.
- 3) Sistem mendukung notifikasi *real-time* untuk memastikan laporan dan komentar dapat diterima tepat waktu.
- 4) Tersedia fitur manajemen akun untuk mahasiswa dan dosen pembimbing.

3.1.2. Kebutuhan Non-Fungsional

- 1) Aplikasi dirancang dengan tampilan yang sederhana dan intuitif agar mudah dipahami oleh mahasiswa dan dosen.

- 2) Tombol dan menu navigasi jelas serta mudah diakses.

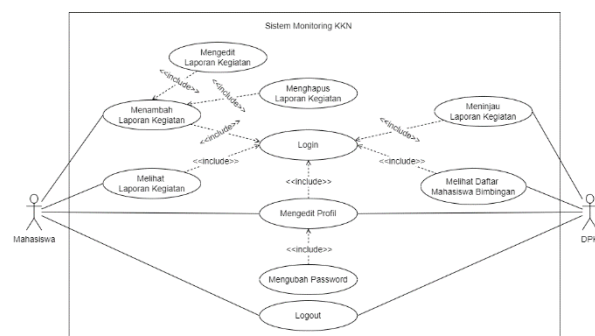
- 3) Aplikasi dapat berjalan pada perangkat Android dengan versi minimal Android 8.0 (Oreo) atau lebih baru.

- 4) Mahasiswa dapat mengunggah *logbook* meskipun perangkat tidak terhubung ke internet.

3.2. Perancangan Sistem

3.2.1. Use case Diagram

Use case Diagram digunakan untuk memodelkan interaksi antara pengguna (aktor) dan sistem. Diagram ini memberikan gambaran tentang bagaimana sistem beroperasi dengan menunjukkan fungsi utama yang disediakan serta peran pengguna yang berinteraksi dengannya. Diagram ini sering digunakan dalam tahap awal pengembangan sistem untuk mendokumentasikan persyaratan fungsional, membantu komunikasi dengan pengguna, serta menjadi dasar untuk desain dan pengujian sistem [8]. *Use case* diagram dapat dilihat pada Gambar 2.



Gambar 2. *Use case* Diagram Sistem Monitoring KKN

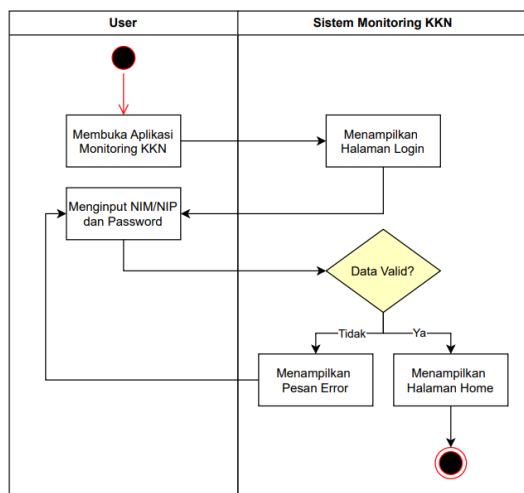
Perancangan sistem dilakukan dengan mempertimbangkan dua aktor utama yang memiliki hak akses dan fungsi yang berbeda. Mahasiswa dapat menambahkan, mengedit, menghapus, dan melihat laporan kegiatan, serta mengunggah foto dokumentasi sebagai bukti aktivitas. Sementara itu, DPK dapat meninjau laporan mahasiswa, menyetujui laporan, serta memberikan ulasan atau catatan sebagai umpan balik. Selain itu, sistem juga menyediakan fitur *login* bagi kedua aktor untuk mengamankan akses, serta fitur pengelolaan profil dan *logout*.

3.2.2. Activity Diagram

Activity diagram digunakan untuk memodelkan alur kerja atau proses bisnis dalam suatu sistem. Diagram ini menggambarkan urutan aktivitas yang terjadi dalam suatu proses, termasuk aliran keputusan, percabangan, dan kondisi yang mempengaruhi jalannya aktivitas [9]. Berikut adalah *activity* diagram dari sistem yang dikembangkan.

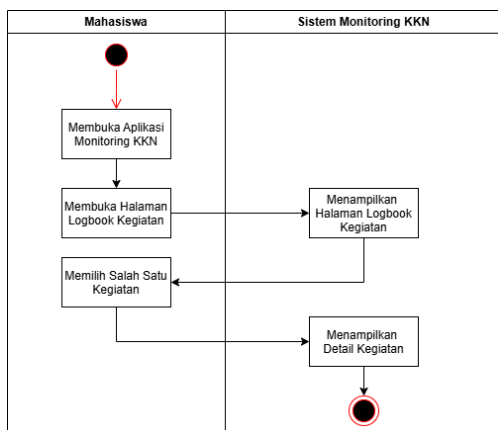
Saat pertama membuka aplikasi pengguna diarahkan ke halaman *login*. Pengguna menginput NIM atau NIP beserta kata sandi. Jika kredensial valid, sistem memberikan akses ke halaman beranda. Namun jika tidak, maka sistem menampilkan pesan kesalahan dan

kembali ke halaman *login*. *Activity* diagram *login* dapat dilihat pada gambar 3.



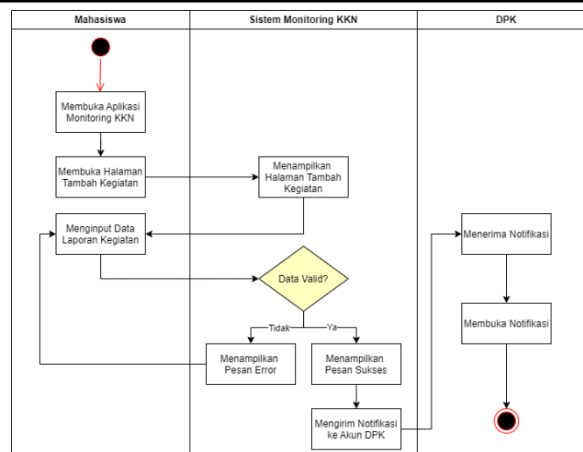
Gambar 3. *Activity* Diagram *Login*

Activity diagram lihat *logbook* menunjukkan proses ketika mahasiswa mengakses daftar *logbook* yang telah dibuat. Sistem menampilkan *logbook* yang dikelompokkan berdasarkan tanggal. Mahasiswa dapat memilih salah satu entri *logbook* untuk melihat detailnya. *Activity* diagram lihat *logbook* dapat dilihat pada gambar 4.



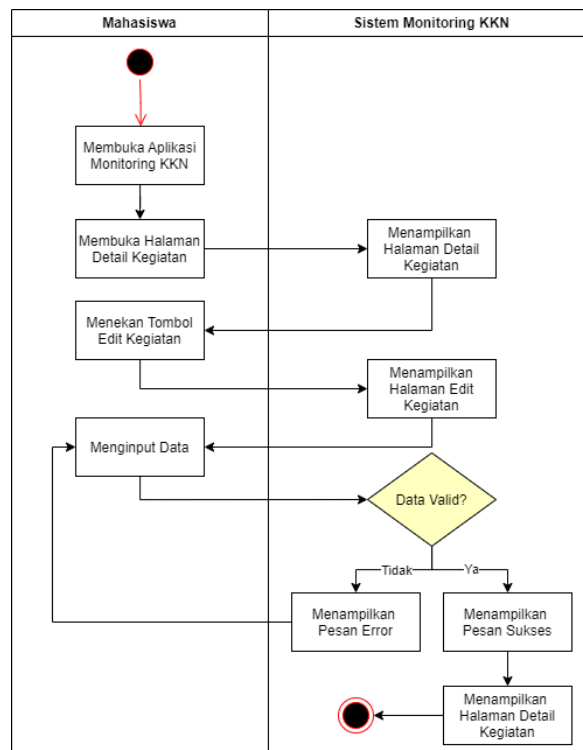
Gambar 4. *Activity* Diagram Lihat *Logbook* Kegiatan

Pada *activity* diagram tambah *logbook* kegiatan, mahasiswa memilih opsi tambah *logbook*, kemudian mengisi formulir yang terdiri dari tanggal, dokumentasi, dan uraian kegiatan. Setelah itu, data dikirim ke server untuk disimpan dalam basis data dan notifikasi dikirim ke dosen pembimbing. *Activity* diagram tambah *logbook* kegiatan dapat dilihat pada gambar 5.



Gambar 5. *Activity* Diagram Tambah *Logbook*

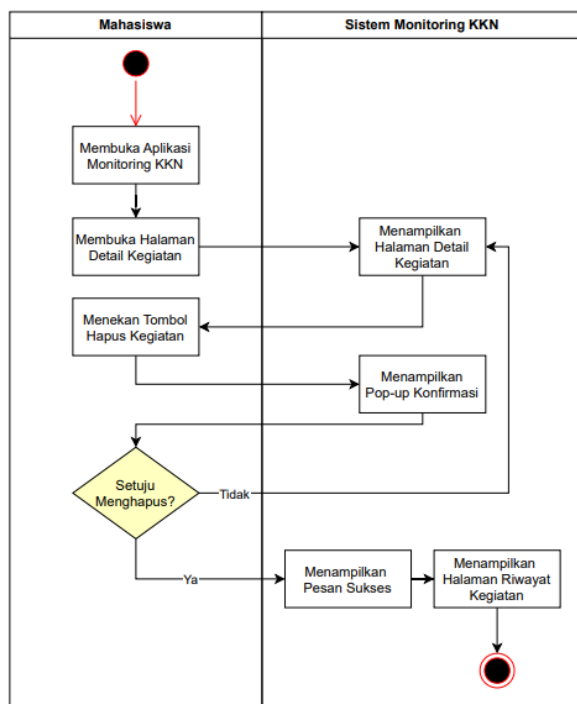
Activity diagram edit *logbook* kegiatan menjelaskan proses ketika mahasiswa mengedit *logbook* yang telah dibuat. Pengguna memilih entri *logbook* yang ingin di edit kemudian memilih menu edit pada halaman detail kegiatan, mengubah informasi yang diperlukan, lalu menyimpan perubahan. Saat perubahan berhasil maka akan diarahkan kembali ke halaman detail kegiatan. *Activity* diagram edit *logbook* kegiatan dapat dilihat pada gambar 6.



Gambar 6. *Activity* Diagram Edit *Logbook* kegiatan

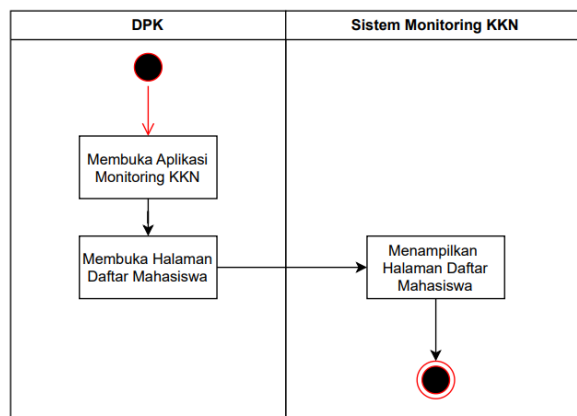
Activity diagram hapus *logbook* kegiatan menunjukkan proses ketika mahasiswa menghapus *logbook* tertentu. Setelah memilih entri *logbook* dan masuk ke halaman detail kegiatan, mahasiswa dapat memilih menu hapus, dan sistem meminta konfirmasi sebelum mengirim permintaan penghapusan ke server. Jika berhasil, *logbook* akan dihapus dan kembali ke halaman *logbook*

kegiatan. *Activity* diagram hapus *logbook* kegiatan dapat dilihat pada gambar 7.



Gambar 7. *Activity* Diagram Hapus *Logbook* kegiatan

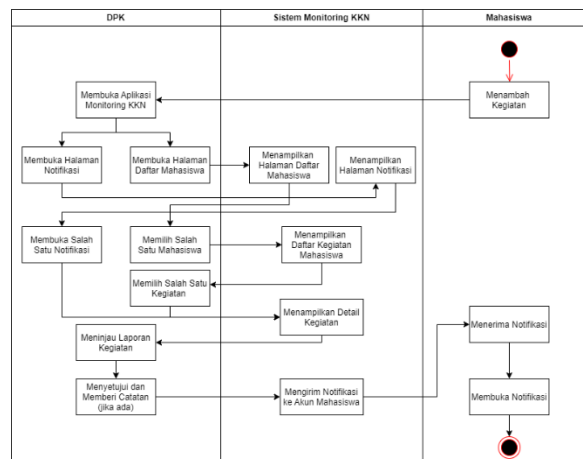
Pada *activity* diagram lihat mahasiswa bimbingan, dosen memilih menu mahasiswa bimbingan untuk menampilkan daftar mahasiswa yang berada di bawah bimbingannya. Sistem mengambil data dari server berdasarkan NIP dosen dan menampilkannya dalam bentuk daftar. *Activity* diagram lihat mahasiswa bimbingan dapat dilihat pada gambar 8.



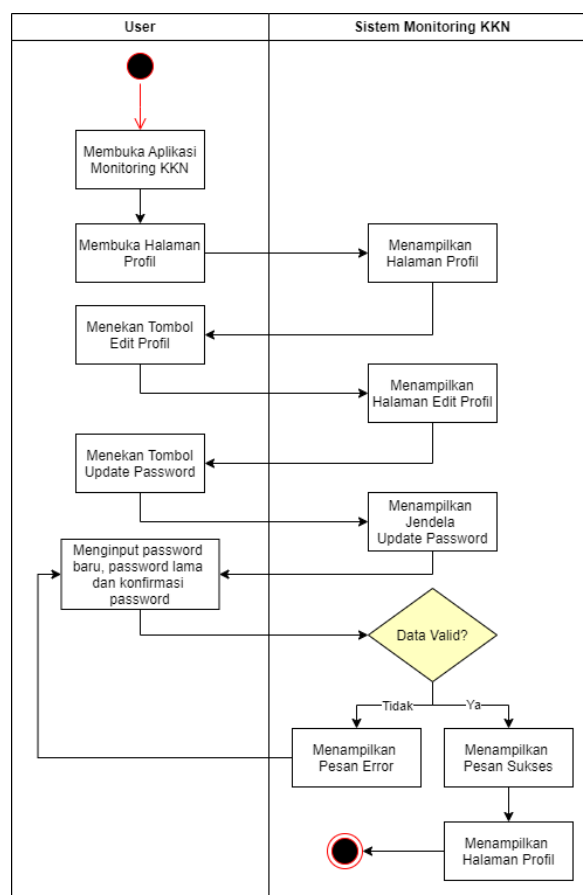
Gambar 8. *Activity* Diagram Lihat Mahasiswa Bimbingan

Activity diagram tinjau *logbook* kegiatan menggambarkan alur aktivitas dosen pembimbing dalam meninjau *logbook* yang dikirimkan oleh mahasiswa. Proses dimulai ketika dosen mengakses daftar mahasiswa bimbingan, kemudian memilih salah satu mahasiswa untuk melihat *logbook* yang telah dikirimkan. Dosen memiliki opsi untuk memberikan komentar sebagai bentuk evaluasi atau umpan balik. Selanjutnya, dosen dapat menyetujui entri *logbook* tersebut berdasarkan penilaian terhadap isi kegiatan.

Setelah dosen menyetujui *logbook* sistem akan mengirim notifikasi ke mahasiswa bahwa *logbook* mereka telah disetujui. *Activity* diagram tinjau *logbook* kegiatan dapat dilihat pada gambar 9.



Gambar 9. *Activity* Diagram Tinjau *Logbook* Kegiatan

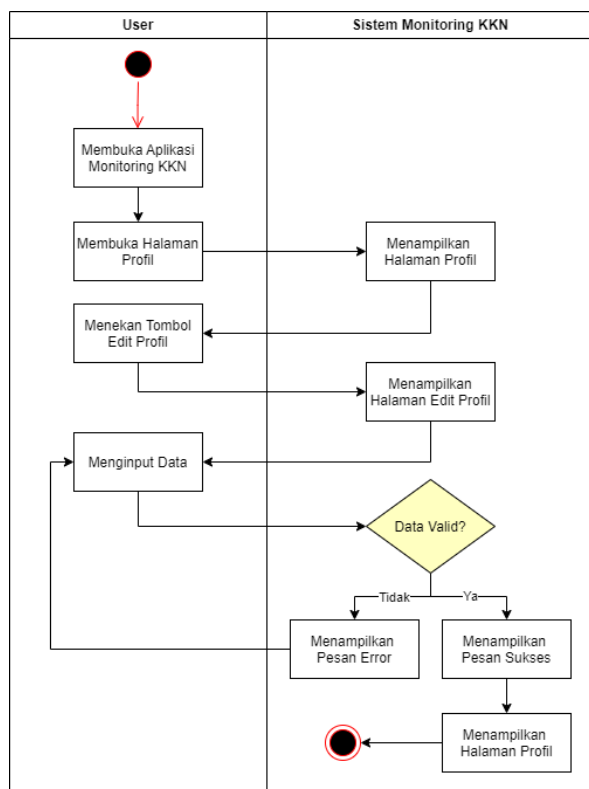


Gambar 10. *Activity* Diagram Ubah *Password*

Activity diagram ubah *password* menggambarkan alur proses perubahan kata sandi. Aktivitas dimulai saat pengguna mengakses halaman ubah kata sandi melalui menu edit profil. Pada halaman ini, pengguna diminta untuk mengisi kata sandi lama sebagai bentuk verifikasi identitas, serta memasukkan kata sandi baru yang diinginkan. Sistem kemudian akan melakukan validasi terhadap kata sandi lama. Jika validasi berhasil

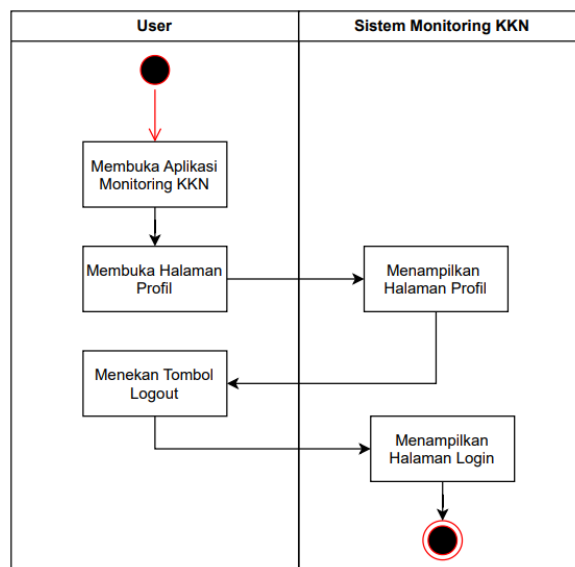
dan kata sandi baru memenuhi syarat keamanan, sistem akan memproses perubahan dan memberikan notifikasi bahwa kata sandi telah berhasil diperbarui. Sebaliknya, jika terjadi kesalahan, sistem akan menampilkan pesan kegagalan yang sesuai agar pengguna dapat memperbaiki input. *Activity* diagram ubah *password* dapat dilihat pada gambar 10.

Activity diagram edit profil menggambarkan alur saat pengguna memperbarui data pribadinya seperti nama, alamat, nomor telepon, atau wilayah administrasi. Setelah perubahan berhasil dilakukan, pengguna diarahkan kembali ke halaman profil. *Activity* diagram edit profil dapat dilihat pada gambar 11.



Gambar 11. *Activity* Diagram Edit Profil

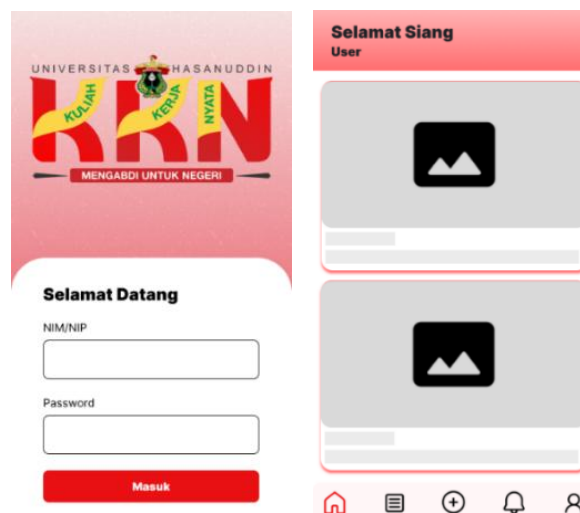
Terakhir, *activity* diagram *logout* menampilkan proses pengguna keluar dari akun mereka. Setelah menekan tombol *logout*, sistem menghapus sesi pengguna dan mengarahkan kembali ke halaman *login*. *Activity* diagram *logout* dapat dilihat pada gambar 12.



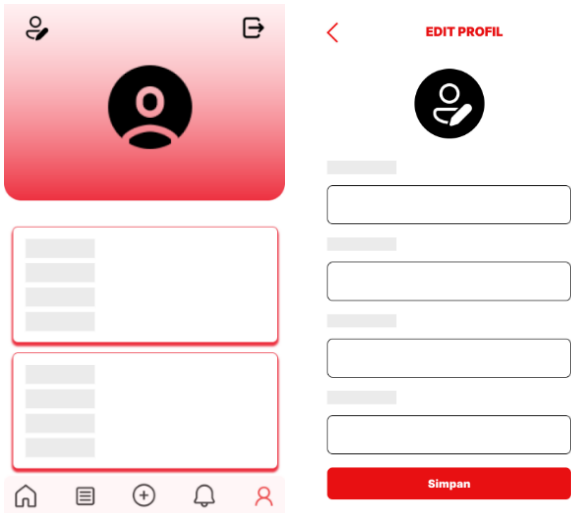
Gambar 12. *Activity* Diagram Logout

3.2.3. Rancangan Antarmuka

Rancangan antarmuka merupakan tahap perancangan visual dari sistem yang memperlihatkan tampilan dan susunan elemen-elemen interaktif yang akan digunakan oleh pengguna. Rancangan antarmuka disusun berdasarkan fungsionalitas utama sistem yang telah didefinisikan sebelumnya, dengan mempertimbangkan kebutuhan dua jenis pengguna, yaitu mahasiswa dan dosen pembimbing lapangan (DPK). Antarmuka dirancang agar setiap fitur dapat diakses dengan mudah melalui navigasi yang jelas dan terstruktur. Berikut merupakan hasil rancangan antarmuka dari sistem monitoring kegiatan KKN yang dikembangkan.



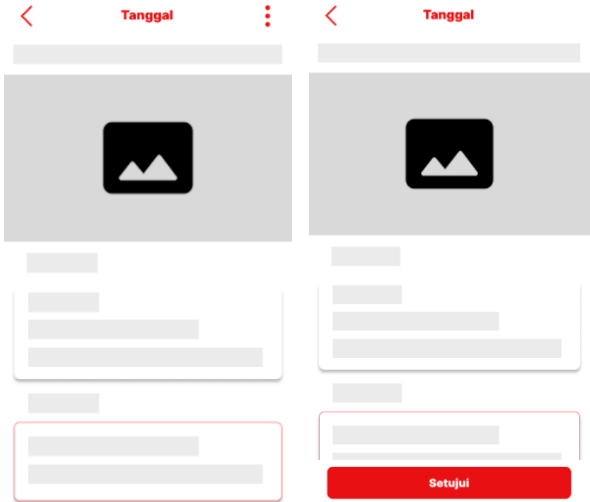
Gambar 13. Desain Antarmuka Halaman Login dan Beranda



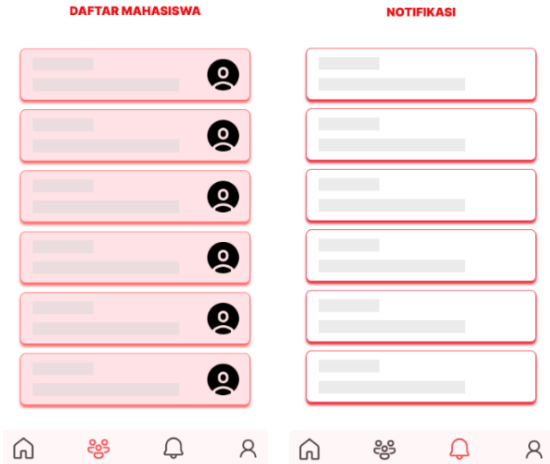
Gambar 14. Desain Antarmuka Halaman Profil dan Edit Profil



Gambar 15. Desain Antarmuka Halaman Logbook dan Tambah Kegiatan



Gambar 16. Desain Antarmuka Halaman Detail Kegiatan dan Tinjau Kegiatan



Gambar 17. Desain Antarmuka Halaman Daftar Mahasiswa Bimbingan dan Notifikasi

3.3. Implementasi

Implementasi sistem dilakukan dengan pengembangan layanan *backend* melalui serangkaian *endpoint API* untuk komunikasi data dan integrasi fitur-fitur aplikasi sesuai kebutuhan pengguna, baik mahasiswa maupun dosen pembimbing.

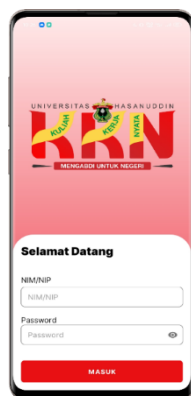
Backend sistem dikembangkan menggunakan pendekatan *REST API (Representational State Transfer Application Programming Interface)* yang merupakan arsitektur yang digunakan dalam pengembangan layanan web, dan memanfaatkan protokol HTTP untuk pertukaran data [9]. *REST API* sering digunakan karena fleksibilitasnya dalam mendukung berbagai format data dan kemampuannya untuk diintegrasikan dengan berbagai platform dan bahasa pemrograman [10]. API yang dikembangkan menyediakan akses ke data pengguna, *logbook* kegiatan, notifikasi, dan informasi lokasi administratif. Adapun daftar endpoint yang digunakan ditunjukkan pada Tabel 1.

Tabel 1. Daftar Endpoint API

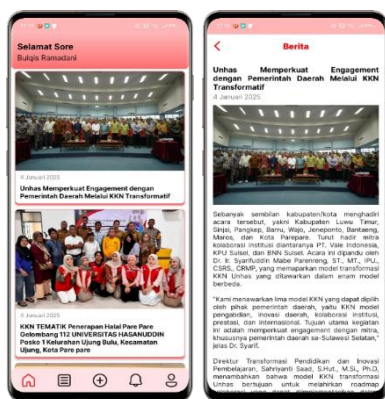
Method	Endpoint	Deskripsi
POST	/login	Login pengguna
PATCH	/updatePassword/{userId}	Memperbarui password pengguna
GET	/mahasiswa/{nim}	Mendapatkan data mahasiswa berdasarkan NIM
GET	/mahasiswa/dpk/{nip}	Mendapatkan data mahasiswa daftar berdasarkan NIP dosen pembimbing
PATCH	/mahasiswa/{id}	Memperbarui data mahasiswa
GET	/dosen/{nip}	Mendapatkan data dosen berdasarkan NIP
PATCH	/dosen/{nip}	Memperbarui data dosen
POST	/logbook	Menambahkan entri logbook
GET	/logbook/{parameter}	Mendapatkan data logbook berdasarkan parameter (nim/id logbook)
PATCH	/logbook/{id}	Mengedit logbook oleh mahasiswa

<i>PATCH</i>	<i>/notifikasi/{userId}</i>	Menyetujui dan mengomentari <i>logbook</i> oleh dosen
<i>DELETE</i>	<i>/logbook/{id}</i>	Menghapus <i>logbook</i>
<i>GET</i>	<i>/notifikasi/{userId}</i>	Mendapatkan daftar notifikasi pengguna
<i>PATCH</i>	<i>/notifikasi/{id}</i>	Menandai notifikasi sebagai sudah dibaca
<i>GET</i>	<i>/berita</i>	Mendapatkan daftar berita
<i>GET</i>	<i>/berita/{id}</i>	Mendapatkan detail berita berdasarkan id
<i>GET</i>	<i>/provinsi</i>	Mendapatkan daftar provinsi
<i>GET</i>	<i>/kabupaten/{provId}</i>	Mendapatkan daftar kabupaten berdasarkan id provinsi
<i>GET</i>	<i>/kecamatan/{kabId}</i>	Mendapatkan daftar kecamatan berdasarkan id kabupaten
<i>GET</i>	<i>/desa/{kecId}</i>	Mendapatkan daftar desa berdasarkan id kecamatan

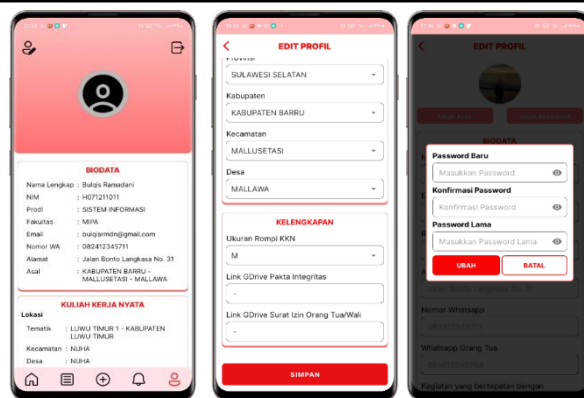
Dari sisi antarmuka aplikasi, pengguna pertama kali diarahkan ke halaman *login* yang digunakan untuk melakukan proses autentikasi seperti pada Gambar 17. Setelah berhasil *login*, pengguna dibawa ke halaman beranda yang menyajikan informasi dan berita terkait pelaksanaan seperti pada Gambar 18. Kemudian terdapat halaman profil, pada halaman ini pengguna dapat melihat data pribadinya, terdapat pula fitur untuk mengedit profil dan mengganti kata sandi melalui halaman edit profil seperti pada gambar 19.



Gambar 17. Implementasi *Login*

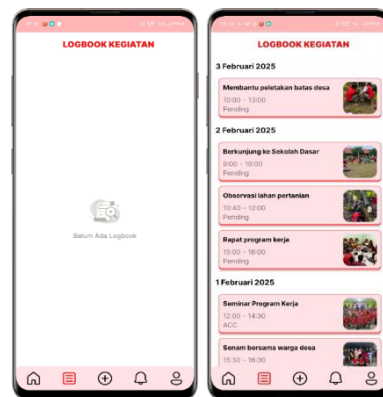


Gambar 18. Implementasi Beranda

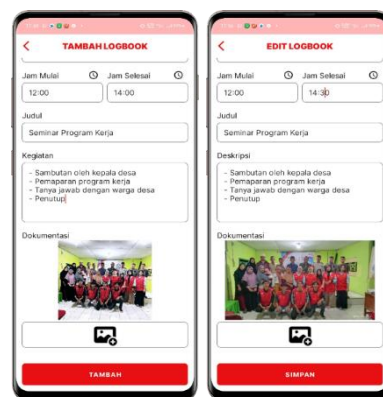


Gambar 19. Implementasi Profil, Edit Profil dan Ubah *Password*

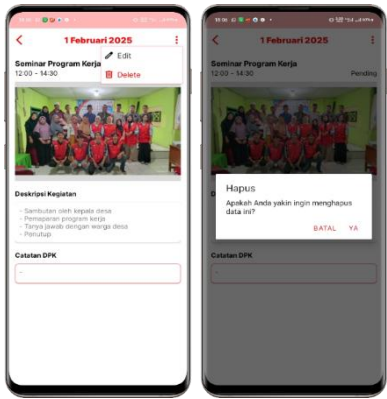
Fitur *logbook* merupakan bagian inti bagi mahasiswa dalam mencatat aktivitas harian KKN. Mahasiswa dapat melihat daftar *logbook* yang telah dibuat melalui halaman *logbook* seperti pada Gambar 20. Mahasiswa dapat menambahkan entri baru menggunakan formulir yang tersedia di halaman tambah *logbook*, *logbook* yang telah dibuat dapat diedit jika diperlukan seperti yang terlihat pada Gambar 21, dan setiap entri dapat dibuka untuk melihat detail kegiatan pada halaman detail terdapat menu untuk edit dan hapus *logbook*, jika memilih "edit" maka akan diarahkan ke halaman edit *logbook* sedangkan jika memilih "delete" maka akan muncul dialog konfirmasi seperti pada Gambar 22.



Gambar 20. Implementasi *Logbook* Kegiatan



Gambar 21. Implementasi Tambah dan Edit *Logbook*



Gambar 22. Implementasi Detail Kegiatan

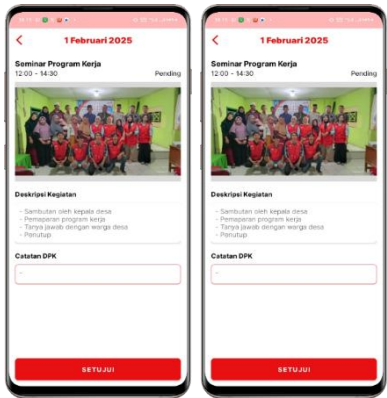


Gambar 25. Implementasi Notifikasi

Bagi dosen pembimbing, sistem menyediakan halaman daftar mahasiswa bimbingan yang menampilkan seluruh mahasiswa yang berada dalam tanggung jawabnya seperti pada Gambar 23. Dosen juga memiliki akses untuk meninjau kegiatan mahasiswa, memberikan komentar, dan menyetujui *logbook* melalui halaman tinjau kegiatan seperti pada gambar 24. Komentar dan persetujuan ini kemudian akan dikirim sebagai notifikasi ke mahasiswa. Semua notifikasi yang diterima pengguna, baik mahasiswa maupun dosen, ditampilkan dalam halaman notifikasi yang dapat dilihat pada Gambar 25. Aplikasi menyediakan fitur *logout* yang berfungsi untuk mengakhiri sesi pengguna seperti pada Gambar 26.



Gambar 23. Implementasi Daftar Mahasiswa Bimbingan



Gambar 24. Implementasi Tinjau Kegiatan



Gambar 26. Implementasi Logbook Kegiatan

3.4. Pengujian

3.4.1. User Acceptance Testing (UAT)

User Acceptance Testing merupakan tahap krusial dalam pengembangan perangkat lunak, yang melibatkan pengguna akhir dalam menguji sistem untuk memastikan bahwa sistem sesuai dengan kebutuhan dan ekspektasi mereka [11]. Pengujian ini dilakukan untuk menilai apakah sistem siap digunakan dalam lingkungan operasional yang sebenarnya. Dalam penelitian ini, UAT dilakukan oleh 2 Dosen Pembimbing Kegiatan (DPK) dan 30 mahasiswa. Pengujian dilakukan dengan menggunakan kuesioner yang berisi pertanyaan evaluatif, dan jawaban dinilai sebagaimana ditampilkan dalam Tabel 2 berikut.

Tabel 2. Pilihan Jawaban dan Bobot Nilai

Jawaban	Bobot
Sangat Setuju	4
Setuju	3
Kurang Setuju	2
Tidak Setuju	1

Untuk menilai tingkat penerimaan aplikasi, hasil dari kuesioner dianalisis dengan menghitung rata-rata skor keseluruhan. Skor ini kemudian dibandingkan dengan kriteria evaluasi untuk menentukan apakah aplikasi memenuhi kebutuhan pengguna atau tidak. Kriteria penilaian pada Tabel 3.

Tabel 3. Kriteria Skor Evaluasi

Rentang Skor Keseluruhan	Persentase	Keterangan
<1,00	< 25%	Aplikasi tidak memenuhi kebutuhan
1,00 – 1,99	25% - 49,99%	Aplikasi kurang memenuhi kebutuhan
2,00 – 2,99	50% - 74,99%	Aplikasi diterima dengan beberapa perbaikan
3,00 – 4,00	75% - 100%	Aplikasi diterima dan memenuhi kebutuhan

Pengujian oleh Dosen Pembimbing Kegiatan (DPK) dilakukan untuk menilai sejauh mana aplikasi dapat membantu mereka dalam memantau dan memberikan bimbingan kepada mahasiswa KKN. Hasil dari UAT yang dilakukan oleh DPK disajikan dalam Tabel 4 berikut.

Tabel 4. Hasil Kuesioner Dosen

No	Pertanyaan	Skor				Total Skor	Rata-rata Skor (Total ÷ 2)
		S	S	K	T		
		S	×	S	S		
		×	3	×	×		
		4		2	1		
1.	Apakah aplikasi mudah digunakan untuk memantau <i>logbook</i> mahasiswa?	2	0	0	0	8	4
2.	Apakah notifikasi <i>real-time</i> berfungsi dengan baik saat mahasiswa menambahkan <i>logbook</i> ?	2	0	0	0	8	4
3.	Apakah fitur komentar/catatan pada <i>logbook</i> memudahkan dalam memberikan <i>feedback</i> kepada mahasiswa?	2	0	0	0	8	4
4.	Apakah data mahasiswa dan kegiatan KKN mereka dapat diakses dengan mudah?	2	0	0	0	8	4
5.	Apakah pengubahan data profil, termasuk ubah <i>password</i> , dapat dilakukan dengan mudah?	0	2	0	0	6	3
6.	Apakah antarmuka aplikasi jelas dan nyaman digunakan?	2	0	0	0	8	4

Untuk memperoleh hasil evaluasi keseluruhan, rata-rata skor dihitung menggunakan rumus berikut:

$$\begin{aligned} \text{Rata - rata Skor} &= \frac{\sum \text{Rata-rata Skor per Pertanyaan}}{\text{Jumlah Pertanyaan}} \\ &= \frac{4 + 4 + 4 + 4 + 3 + 4}{6} = \frac{23}{6} = 3,83 \end{aligned}$$

Berdasarkan hasil pengujian UAT yang dilakukan oleh DPK, diperoleh rata-rata skor keseluruhan sebesar 3,83 dari skala maksimal 4,0 atau sebesar 95,83%. Berdasarkan kriteria skor evaluasi, hasil ini menunjukkan bahwa aplikasi berada dalam kategori aplikasi diterima dan memenuhi kebutuhan, yang berarti aplikasi secara umum dapat diterima dan telah memenuhi kebutuhan DPK dalam memantau dan memberikan bimbingan kepada mahasiswa KKN.

Pengujian oleh mahasiswa dilakukan untuk menilai sejauh mana aplikasi dapat membantu mereka dalam mencatat *logbook*, mengunggah dokumentasi, serta mengakses fitur lainnya. Hasil dari UAT yang dilakukan oleh mahasiswa disajikan dalam Tabel 5 berikut.

Tabel 5. Hasil Kuesioner Mahasiswa

No	Pertanyaan	Skor				Total Skor	Rata-rata Skor (Total ÷ 30)
		SS	S	K	T		
		×	×	×	×		
		×	3	×	×		
				2	1		
1.	Apakah aplikasi mudah digunakan untuk mencatat <i>logbook</i> KKN?	17	13	0	0	107	3,56
2.	Apakah notifikasi <i>real-time</i> berfungsi dengan baik saat dosen menyetujui <i>logbook</i> ?	10	18	2	0	98	3,26
3.	Apakah dokumentasi kegiatan dapat diunggah dengan mudah?	19	11	0	0	109	3,63
4.	Apakah aplikasi memudahkan pelacakan dan pendokumentasian kegiatan KKN?	14	16	0	0	104	3,46
5.	Apakah pengubahan data profil, termasuk ubah <i>password</i> , dapat dilakukan dengan mudah?	14	14	2	0	102	3,4

6.	Apakah antarmuka aplikasi jelas dan nyaman digunakan?	15	15	0	0	105	3,5
----	---	----	----	---	---	-----	-----

Untuk memperoleh hasil evaluasi keseluruhan, rata-rata skor dihitung menggunakan rumus berikut:

$$\begin{aligned} \text{Rata - rata Skor} &= \frac{\sum \text{Rata-rata Skor per Pertanyaan}}{\text{Jumlah Pertanyaan}} \\ &= \frac{3,56 + 3,26 + 3,63 + 3,46 + 3,4 + 3,5}{6} \\ &= \frac{20,81}{6} = 3,47 \end{aligned}$$

Berdasarkan hasil pengujian UAT yang dilakukan oleh mahasiswa diperoleh rata-rata skor keseluruhan sebesar 3,47 dari skala maksimal 4,0 atau sebesar 86,75%. Berdasarkan kriteria skor evaluasi, hasil ini menunjukkan bahwa aplikasi berada dalam kategori aplikasi diterima dan memenuhi kebutuhan, yang berarti aplikasi secara umum dapat diterima dan telah memenuhi kebutuhan mahasiswa dalam mencatat *logbook*, mengunggah dokumentasi, serta mengakses fitur lainnya dengan baik.

3.4.2. Load testing

Load testing merupakan teknik pengujian performa untuk mengukur respon sistem di bawah berbagai kondisi beban, guna mengetahui perilaku perangkat lunak ketika diakses oleh banyak pengguna secara simultan [12]. Pengujian ini penting dilakukan guna memastikan bahwa aplikasi mampu menangani sejumlah besar pengguna aktif pada waktu yang bersamaan, tanpa mengalami penurunan performa yang signifikan.

Dalam pengujian ini, digunakan Apache JMeter sebagai alat bantu simulasi beban pengguna. Apache JMeter merupakan salah satu tool open-source yang banyak digunakan untuk melakukan uji kinerja sistem web secara otomatis dan terukur. JMeter mampu menguji beberapa modul sistem informasi seperti *login*, unggah data, dan pengelolaan profil dengan memberikan hasil berupa waktu respons rata-rata, *throughput*, dan tingkat kesalahan secara sistematis [13].

Skenario pengujian dibuat dengan memvariasikan jumlah *thread* (yang merepresentasikan pengguna simultan) dari 4000 hingga 8000, dengan waktu *ramp-up* selama 600 detik untuk setiap skenario. *Ramp-up period* ini memberikan waktu kepada sistem untuk membagi jumlah pengguna secara bertahap, sehingga mencerminkan kondisi nyata ketika pengguna masuk secara bertahap dalam kurun waktu tertentu. Parameter yang dianalisis dalam pengujian meliputi rata-rata *response time* (waktu rata-rata yang dibutuhkan sistem untuk merespon permintaan pengguna), *throughput* (jumlah permintaan yang berhasil diproses per detik), dan *error percentage* (persentase kegagalan proses permintaan). Hasil *load testing* dapat dilihat pada Tabel 6.

Tabel 6. Hasil *Load testing*

<i>Number of Threads (users)</i>	<i>Ramp-up Period (s)</i>	<i>Rata-rata Response Time (ms)</i>	<i>Throughput</i>	<i>Error (%)</i>
4000	600	1551	6,6/sec	0,00%
5000	600	7687	8,1/sec	0,00%
6000	600	39643	8,8/sec	3,73%
7000	600	46785	10,3/sec	18,16%
8000	600	47801	12,2/sec	27,69%

Hasil pengujian menunjukkan bahwa sistem masih dapat merespon dengan baik hingga beban mencapai 5000 pengguna secara bersamaan. Pada skenario ini, waktu respon rata-rata masih berada dalam kisaran yang dapat diterima yaitu 7687 ms (7,6 detik) dan tidak terdapat kesalahan (*error*) dalam proses permintaan. *Throughput* juga meningkat secara bertahap dari 6,6 hingga 8,1 permintaan per detik, menandakan sistem masih dapat memproses permintaan secara efisien.

Namun, ketika jumlah pengguna ditingkatkan menjadi 6000 hingga 8000, performa sistem mulai menunjukkan penurunan. Terjadi peningkatan waktu respon secara drastis hingga mencapai lebih dari 47 detik pada 8000 pengguna, disertai dengan peningkatan *error percentage* hingga 27,69%. Hal ini mengindikasikan bahwa sistem mulai mengalami *bottleneck* akibat keterbatasan sumber daya atau arsitektur *backend* yang belum optimal dalam menangani beban besar. Meskipun *throughput* masih meningkat hingga 12,2 permintaan per detik, hal ini tidak sebanding dengan lonjakan *error* dan lamanya waktu respon.

Analisis mendalam menunjukkan bahwa penurunan performa ini disebabkan oleh keterbatasan sumber daya server (CPU dan RAM pada konfigurasi AWS EC2 t2.micro) dan bandwidth jaringan yang tidak mampu menampung lonjakan trafik. Akibatnya, jika aplikasi digunakan pada skala di atas 5000 pengguna simultan tanpa optimisasi, maka pengguna akan mengalami keterlambatan signifikan dan potensi kegagalan akses. Untuk mengatasinya, beberapa langkah dapat diterapkan, seperti peningkatan spesifikasi server (CPU, RAM), penggunaan load balancer untuk membagi beban ke beberapa server, serta pengujian stres lanjutan untuk memvalidasi perbaikan. Dengan demikian, sistem dapat dipastikan mampu mempertahankan performa yang stabil meskipun beban pengguna meningkat.

Dengan demikian, dapat disimpulkan bahwa aplikasi mampu menangani beban hingga 5000 pengguna secara bersamaan dengan performa yang stabil. Untuk pengguna di atas jumlah tersebut, diperlukan optimalisasi server atau infrastruktur tambahan untuk menjaga performa aplikasi tetap optimal.

4. Kesimpulan

Pengembangan aplikasi monitoring kegiatan Kuliah Kerja Nyata berbasis Android memberikan solusi atas permasalahan pelaporan manual yang kurang efisien

dalam pelaksanaan KKN di Universitas Hasanuddin. Aplikasi ini mampu memfasilitasi pelaporan kegiatan secara sistematis dan mempercepat proses evaluasi oleh dosen pembimbing. Berdasarkan hasil pengujian, aplikasi telah memenuhi fungsionalitas yang dirancang dan diterima dengan baik oleh pengguna. Implementasi aplikasi ini memberikan kontribusi terhadap peningkatan efisiensi administrasi kegiatan KKN dan memperkuat pemanfaatan teknologi dalam pengelolaan program akademik di perguruan tinggi. Penggunaan sistem ini juga dapat menjadi contoh penerapan transformasi digital pada kegiatan pembelajaran berbasis lapangan. Untuk pengembangan lebih lanjut aplikasi dapat diperluas untuk *platform* iOS agar dapat diakses oleh lebih banyak pengguna dengan berbagai jenis perangkat *mobile*. Selain itu, penambahan fitur pendaftaran, pendataan program kerja, dan laporan akhir juga disarankan guna meningkatkan kelengkapan administrasi KKN secara digital serta mendukung pengelolaan data yang lebih efektif dan terstruktur. Dari sisi teknis, disarankan menerapkan arsitektur client-server yang lebih scalable, penggunaan kompresi data untuk mempercepat sinkronisasi, serta optimisasi performa server dengan load balancing dan database connection pooling guna menjaga stabilitas saat trafik meningkat.

Daftar Rujukan

- [1] Kementerian Pendidikan Tinggi, Sains, dan Teknologi, 2024. *Panduan Kuliah Kerja Nyata (KKN) Kebangsaan 2024*. [Online]. Available: <https://kemdiktisaintek.go.id/epustaka/panduan-kuliah-kerja-nyata-kkn-kebangsaan-2024>. [Accessed 12 Oktober 2024].
- [2] I. Basri, 2024. *Unhas Lepaskan 4289 Mahasiswa KKN Gelombang 112*. [Online]. Available: <https://identitasunhas.com/unhas-lepaskan-4289-mahasiswa-kkn-gelombang-112/>. [Accessed 3 November 2024].
- [3] Statcounter, 2024. *Mobile Operating System Market Share Indonesia*. [Online]. Available: <https://gs.statcounter.com/os-market-share/mobile/indonesia/2024>. [Accessed 20 Oktober 2024].
- [4] Hidayanti, N., Widyawati, W., Fatullah, R., & Budiono, B. 2020. Rancang bangun aplikasi monitoring kegiatan kuliah kerja mahasiswa berbasis android di Universitas Banten Jaya. *Teknika: Jurnal Sains Dan Teknologi*, 16(2), 267. <https://doi.org/10.36055/tjst.v16i2.8680>
- [5] Syah, F., Lukito, W., Aristoteles, A., Efendi, N., & Eka Febriansyah, F. 2018. Sistem Informasi Kuliah Kerja Nyata (Kkn) Berbasis Android Universitas Lampung. *Jurnal Komputasi*, 6(2), 1–10. <https://doi.org/10.23960/komputasi.v6i2.1655>
- [6] Akib, F., & Muin, A. 2024. Implementasi Sistem Monitoring Pelaksanaan Kuliah Kerja Nyata Berbasis Aplikasi Mobile Pada Universitas Islam Negeri Alauddin Makassar. *Jurnal Insypro (Information System and Processing)*, 9(2). <https://doi.org/10.24252/insypro.v9i2.52738>
- [7] Aceng, A. W. 2020. Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi. *Jurnal Ilmu-Ilmu Informatika Dan Manajemen STMIK*, <https://www.researchgate.net/publication/346397070>
- [8] Santoso, J. T. & Migunani, 2021. Desain & Analisis Sistem Berorientasi Obyek dengan UML. Semarang: Yayasan Prima Agus Teknik.
- [9] Novianto, M. A., & Munir, S. 2022. Analisis dan Implementasi Restful API guna Pengembangan Sistem Informasi Akademik pada Perguruan Tinggi. *Jurnal Informatika Terpadu*, 8(1), 47–61. <https://doi.org/10.54914/jit.v8i1.409>
- [10] Wulandari, N., Wibowo, A., & Susanto, B. 2021. Penerapan Restful API untuk Membangun Program Pembayaran Piutang Menggunakan Otentikasi OAuth 2.0. *Jurnal Terapan Teknologi Informasi*, 5(1), 1–10. <https://doi.org/10.21460/jutei.2021.51.230>
- [11] Vanesha, N. A., Rizky, R., & Purwanto, A. 2024. Comparison Between Usability and User Acceptance Testing on Educational Game Assessment. *Jurnal Sisfokom (Sistem Informasi Dan Komputer)*, 13(2), 210–215. <https://doi.org/10.32736/sisfokom.v13i2.2099>
- [12] Permatasari, D. I. 2020. Pengujian Aplikasi menggunakan metode Load Testing dengan Apache JMeter pada Sistem Informasi Pertanian. *Jurnal Sistem dan Teknologi Informasi (JUSTIN)*, vol. 8, no. 1, p. 135, doi: 10.26418/justin.v8i1.34452
- [13] Indrianto. 2023. Performance Testing On Web Information System Using Apache Jmeter And Blazemeter. *Jurnal Ilmiah Ilmu Terapan Universitas Jambi*, vol. 7, no. 2, pp. 138–149, doi: 10.22437/jiituj.v7i2.28440