PENGEMBANGAN FITUR SCHEDULE PADA APLIKASI MOBILE I'M UII

Disusun Oleh:

N a m a	: Alif Maulana Rizqi
NIM	: 19523193

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2023

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENGEMBANGAN FITUR SCHEDULE PADA APLIKASI MOBILE I'M UII

TUGAS AKHIR JALUR MAGANG

Disusun Oleh:

N a m a : Alif Maulana Rizqi

NIM : 19523193

Yogyakarta, 10 Juli 2023

Pembimbing,

(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PENGESAHAN DOSEN PENGUJI

PENGEMBANGAN FITUR SCHEDULE PADA APLIKASI MOBILE I'M UII

TUGAS AKHIR JALUR MAGANG

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 25 Juli 2023

Tim Penguji

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. _____

Anggota 1

Ari Sujarwo, S.Kom., M.I.T. _____

Anggota 2

Kholid Haryono, S.T., M.Kom. _____

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Alif Maulana Rizqi

NIM : 19523193

Tugas akhir dengan judul:

PENGEMBANGAN FITUR SCHEDULE PADA APLIKASI MOBILE I'M UII

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 10 Juli 2023

(Alif Maulana Rizqi)

HALAMAN PERSEMBAHAN

Laporan akhir ini saya persembahkan untuk kedua orang tua yang selalu memberikan dukungan baik berupa doa, moral, dan material. Saya juga ingin berterima kasih kepada teman-teman yang sudah memberikan bantuan dan semangat sehingga saya dapat menyelesaikan laporan akhir ini.

HALAMAN MOTO

“Orang kuat itu bukanlah orang yang menang bergulat, namun orang kuat ialah orang yang dapat menahan dirinya ketika marah.” (HR. Al-Bukhari dan Muslim).

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Puji dan syukur saya panjatkan kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan laporan akhir magang ini sebagai salah satu syarat kelulusan pada penjaluran magang di Program Studi Informatika Universitas Islam Indonesia. Laporan akhir magang ini menjelaskan berbagai macam kegiatan yang penulis lakukan dalam kurung waktu yang ditentukan di Badan Sistem Informasi UII.

Penulis dibantu oleh berbagai pihak dalam menyelesaikan laporan akhir magang ini, oleh karena itu tidak lupa penulis mengucapkan terima kasih banyak kepada:

1. Bapak Dhomas Hatta Fudholi, selaku dosen pembimbing yang telah memberikan arahan dalam menyelesaikan laporan akhir.
2. Mas Elbho Shindi Pangestu, Mas Yudhistira Arysa Yudha, dan semua orang yang terlibat dalam pembuatan aplikasi I'm UII yang telah memberikan masukan kepada penulis saat pelaksanaan magang.
3. Seluruh pihak manajemen di Badan Sistem Informasi Universitas Islam Indonesia yang telah menyambut dan menerima penulis sebagai pemagang.
4. Orang tua, keluarga, serta teman-teman penulis yang senantiasa memberikan dukungan kepada penulis.

Pada laporan akhir ini sangat dimungkinkan masih banyak kekurangan yang harus diperbaiki. Segala bentuk kritik dan saran akan dengan senang hati penulis terima. Penulis berharap bahwa laporan ini dapat bermanfaat bagi yang membacanya.

Yogyakarta, 10 Juli 2023

(Alif Maulana Rizqi)

SARI

Menurut Kamus Besar Bahasa Indonesia, waktu adalah seluruh rangkaian saat ketika proses, perbuatan, atau keadaan sedang berlangsung. Supaya bisa memanfaatkan dan tidak menyia-nyiakan waktu yang dimiliki, kita perlu konsep yang bernama manajemen waktu. Manajemen waktu merupakan cara yang bisa digunakan untuk menyeimbangkan waktu secara efektif dan efesien. Manajemen waktu melibatkan perencanaan, pengalokasian, menentukan prioritas, serta mengawasi seberapa produktif dalam menjalankan tugas-tugas.

Masalah manajemen waktu merupakan masalah yang umum dan sering dijumpai di mahasiswa. Salah satu masalah yang membuat mahasiswa kesulitan dalam mengatur waktunya adalah terlalu banyak mengikuti kegiatan semisal berorganisasi. Hal ini menyebabkan mahasiswa kesulitan dalam mengatur kegiatan belajarnya. Sehingga manajemen waktu merupakan keterampilan yang harus dimiliki mahasiswa. Masalah serupa juga terjadi pada mahasiswa Universitas Islam Indonesia (UII).

Melihat masalah tersebut, Badan Sistem Informasi (BSI) UII mengembangkan sebuah fitur pada aplikasi *mobile* yang dapat menangani masalah tersebut. Aplikasi tersebut adalah I'm UII. I'm UII merupakan aplikasi *mobile* yang memiliki target pengguna mahasiswa Universitas Islam Indonesia (UII) yang memiliki tujuan untuk mempermudah mahasiswa UII dalam menjalani masa kuliahnya. Salah satu fitur yang dapat membantu mahasiswa khususnya mahasiswa UII dalam manajemen waktu adalah fitur *schedule*. Dengan fitur ini mahasiswa dapat mengatur jadwal kegiatannya seperti melihat, membuat, mengubah, dan menghapus jadwal. Pada fitur *schedule* juga terdapat fitur yang memungkinkan mahasiswa mengundang mahasiswa lain ke jadwal yang dibuatnya. Selain itu pada fitur ini terdapat notifikasi yang mengingatkan kepada mahasiswa bahwa mereka mempunyai jadwal sehingga diharapkan mahasiswa tidak terlewat jadwal atau kegiatan tersebut.

Selama magang di BSI, penulis berperan sebagai *software engineer* yang memiliki tugas untuk mengembangkan fitur *schedule* aplikasi *mobile* I'm UII menggunakan Flutter dan mengembangkan *back-end* fitur *schedule* aplikasi I'm UII menggunakan Lumen. Proses pengembangan fitur *schedule* I'm UII menggunakan metode pengembangan *waterfall*. Selama kegiatan magang, penulis mendapatkan banyak ilmu baru seperti mengembangkan aplikasi *mobile* menggunakan Flutter dan *back-end* menggunakan Lumen, karena penulis tidak mendapatkan itu saat kuliah.

Kata kunci: Waterfall, Aplikasi Mobile, Flutter, Penjadwalan, Notifikasi.

GLOSARIUM

Android	Sistem operasi yang dikembangkan oleh Google untuk perangkat *mobile*.
API	Kumpulan aturan dan protokol yang memungkinkan komunikasi antara perangkat lunak.
Aplikasi Mobile	Perangkat lunak yang dirancang khusus untuk dijalankan pada perangkat *mobile* seperti *smartphone* atau *tablet.*
Back-end	Bagian dari sebuah aplikasi yang bertanggung jawab untuk pemrosesan logika, dan penanganan database.
Callback	Fungsi atau metode yang diberikan sebagai argumen ke fungsi lain.
Deploy	Proses mengunggah dan menjalankan aplikasi atau sistem ke *server* atau lingkungan produksi atau pengembangan.
Developer	Seseorang yang terlibat dalam pengembangan perangkat lunak atau aplikasi.
Form	Bagian antarmuka pengguna yang digunakan untuk mengumpulkan masukan dari pengguna, seperti isian teks atau pilihan.
Framework	Kerangka kerja perangkat lunak yang menyediakan struktur dan alat bantu untuk membangun aplikasi.
Front-end	Bagian dari sebuah aplikasi atau situs web yang berhubungan langsung dengan pengguna, termasuk tampilan dan interaksi antarmuka.
iOS	Sistem operasi yang dikembangkan oleh Apple untuk perangkat mobile seperti iPhone dan iPad.
Native	Mengacu pada aplikasi atau perangkat lunak yang dikembangkan untuk digunakan secara spesifik pada sistem operasi tertentu.
Package	Kumpulan modul, fungsi, atau kelas yang terorganisir dalam satu unit dalam pengembangan perangkat lunak.
Pop-up Dialog	Jendela kecil yang muncul di atas antarmuka pengguna untuk menampilkan pesan, permintaan, atau informasi tambahan.
Prototype	Rancangan dari sebuah produk atau aplikasi yang digunakan untuk tujuan pengujian atau demonstrasi.
Push Notification	Pesan atau pemberitahuan yang dikirim langsung ke perangkat *mobile* pengguna.

Query Parameter	Bagian dari URL yang digunakan untuk mengirimkan data atau informasi tambahan ke *server* melalui permintaan HTTP.
Request	Permintaan yang dikirim dari aplikasi ke API untuk mendapatkan atau mengirim data.
Response	Balasan yang diterima dari API setelah mengirim permintaan, biasanya berisi data atau informasi yang diminta.
Server	Komputer atau sistem yang menyediakan layanan atau sumber daya kepada komputer lainnya dalam jaringan, seperti menyimpan data atau menjalankan aplikasi.
Statement	Bagian dalam kode program yang melakukan tindakan atau operasi tertentu.
Wireframe	Sketsa atau representasi visual dasar dari tata letak atau antarmuka pengguna sebuah aplikasi atau situs web.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

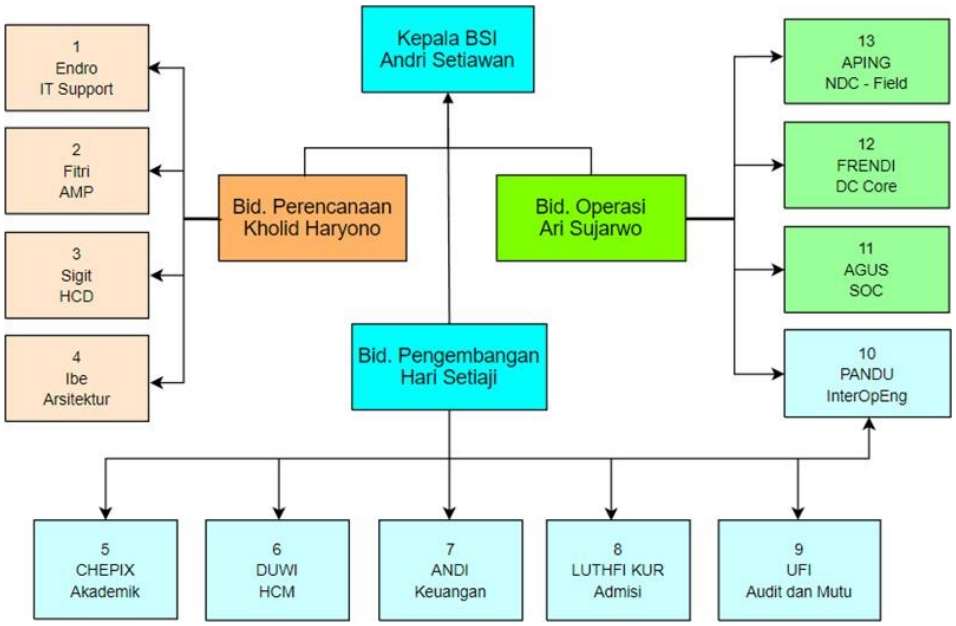
1.1 Latar Belakang

Badan Sistem Informasi (BSI) adalah organisasi yang bergerak di bidang teknologi informasi untuk melayani kebutuhan sistem informasi di lingkungan kampus Universitas Islam Indonesia. Untuk dapat bersaing di masa mendatang, Universitas Islam Indonesia harus mengembangkan kapasitasnya secara aktif. Peningkatan layanan sistem dan teknologi juga termasuk dalam pengembangan kapasitas ini.

BSI memiliki tiga peranan dalam menjalankan tugasnya, yang pertama yaitu melayani, BSI akan berperan sebagai pelayan untuk layanan yang sudah ada sebelumnya seperti koneksi internet dan ketersediaan sistem informasi yang mendukung proses bisnis yang ada di UII. Peran BSI yang kedua adalah mendampingi, BSI bisa menjadi sahabat diskusi perihal sistem dan teknologi informasi yang belum ada sebelumya. Peran BSI yang ketiga yaitu mengakselerasi, BSI sebagai katalis perubahan yang terjadi universitas, fakultas, dan program studi perihal sistem dan teknologi informasi yang dapat mendukung proses bisnis di UII.

BSI memiliki kantor operasional yang berada di Gedung GBPH Prabuningrat lantai 4, seluruh ruangan yang berada di lantai 4 adalah kantor BSI. BSI juga memiliki kontak yang bisa dihubungi, yaitu nomer WhatsApp 08124441414 atau bisa melalui surel itsupport@uii.ac.id.

Pengguna dari layanan BSI mencapai kurang lebih 30.000 pengguna. Pengguna dari layanan BSI sendiri mencakup mahasiswa, dosen, tenaga kependidikan, dan para pemangku kepentingan lainnya. Untuk memberikan layanan terbaik BSI didukung oleh sumber daya manusia yang mumpuni dalam bidangnya. Sumber daya manusia tersebut terdiri dari analis, *programmer, engineer,* dan konsultan yang berpengalaman dalam memenuhi kebutuhan layanan sistem dan teknologi informasi di UII. Ada beberapa bidang yang menjadi tanggung jawab BSI, yaitu bidang perencanaan sistem informasi, bidang pengembangan sistem informasi, bidang operasi sistem informasi, bidang layanan pengguna, dan bidang administrasi dan manajemen proyek (Badan Sistem Informasi, n.d.). Gambar 1.1 menunjukan struktur organisasi BSI.



Gambar 1.1 Struktur organisasi BSI UII

BSI memiliki tiga sasaran mutu dalam melaksanakan tugasnya yaitu memiliki kualitas layanan yang baik (koneksi internet dan sistem informasi) dengan nilai yang diterapkan yaitu 4 dari skala 1 sampai 5. Sasaran mutu kedua adalah layanan infrastruktur teknologi handal yang dapat bekerja dengan baik tanpa banyak mengalami gangguan, dengan nilai yang menjadi sasaran adalah *uptime* hingga 98% setiap tahunnya. Sasaran mutu yang ketiga ialah mengembangkan sistem informasi yang ada di UII yang mendukung 90% proses bisnis yang terdefinisi.

Badan Sistem Informasi UII memiliki karyawan lebih dari 80 orang yang terbagi ke dalam beberapa tim. Tim tersebut diantaranya adalah Tim Intero yang menangani *interoperability*, Tim Admisi yang menangani pendaftaran atau penerimaan penerimaan mahsiswa baru, Tim Keuangan yang menangani keuangan seperti tagihan, Tim Human Capital Management (HCM) yang menangani masalah sumber daya manusia, Tim Akademik yang menangani akademik mahasiswa, Tim Security Operation Center (SOC) yang menangani keamanan jaringan, dan Tim Network dan Data Center yang menangani jaringan.

Saat proses pelaksanaan magang, penulis ditempatkan di kantor BSI yang berada di lantai 4 Gedung GBPH Prabuningrat – Rektorat UII dan bergabung dalam tim percepatan atau tim tambahan yang memiliki tanggung jawab untuk mengembangkan aplikasi *mobile* I’m UII. Tim ini beranggotakan sembilan orang, dengan supervisor Bapak Hari Setiaji, Dian Sigit

Prastowodan selaku *product owner*, dan Elbo Shindi Pangestu sebagai *team-lead*. Tim ini terdiri dari dua divisi, yaitu divisi produk yang terdiri dari dua orang yaitu Anwaruddin Ridho Novianto dan Muhammad Habib Izdhihar Syaf yang memiliki tugas mengerjakan desain UI/UX aplikasi dan divisi pengembangan dengan anggota empat orang yaitu Alif Maulana Rizqi, Rio Risqi Akbar Herlambang, Fahrudin Nasikh Az Zuhdu, dan Yudhistira Arsya Yudha yang memiliki tugas mengembangkan aplikasi *mobile* I'm UII.

Dalam melaksanakan masa kuliahnya, mahasiswa seringkali menghadapi beberapa permasalahan. Salah satu permasalahan yang umum adalah kesulitan dalam mengatur waktunya sehingga dapat menyebabkan stres dan hidupnya kurang tertata dengan baik (Zega & Kurniawati, 2022). Menurut Kamus Besar Bahasa Indonesia, waktu adalah seluruh rangkaian saat ketika proses, perbuatan, atau keadaan sedang berlangsung (KBBI Daring, n.d.). Banyak orang yang mengibaratkan waktu sama dengan uang. Mereka benar bahwa waktu sama pentingnya dengan uang. Tapi perlu diingat bahwa jika seseorang kehilangan uang, maka dapat mencarinya lagi. Sedangkan jika membuang-buang waktu maka waktu tersebut tidak dapat didapatkan kembali (Grafiani, 2021). Maka dari itu kita perlu menggunakan waktu dengan baik supaya tidak menyia-nyiakan sisa waktu yang dimiliki.

Supaya bisa memanfaatkan waktu agar tidak menjadi sia-sia, kita perlu konsep yang bernama manajemen waktu. Manajemen waktu merupakan cara yang bisa digunakan untuk menyimbangkan waktu secara efektif dan efesien. Manajemen waktu melibatkan perencanaan, pengalokasian, menentukan prioritas, serta mengawasi seberapa produktif dalam menjalankan tugas-tugas (Hidayanto, 2021). Dalam manajemen waktu yang baik, mengharuskan seseorang untuk memahami tentang pentingnya prioritas tugas, mengatur jadwal, dan penggunaan teknik yang tepat agar dapat meningkatkan produktifitas. Manajemen waktu tidak hanya soal perkara mengurangi waktu untuk menyelesaikan suatu tugas saja, tetapi perlu juga untuk memperhatikan kualitas tugas yang dikerjakan. Seseorang yang memiliki manajemen waktu yang baik harus mampu menyelesaikan tugas dengan kualitas yang baik. Perlu diingat bahwa manajemen waktu tidak hanya soal menyelesaikan banyak perkerjaan atau tugas saja tetapi bisa menyeimbangkan antara pekerjaan dan kehidupan pribadi.

Masalah manajemen waktu merupakan masalah yang umum dan sering dijumpai di masyarakat, tidak terkecuali mahasiswa. Salah satu masalah yang membuat mahasiswa kesulitan dalam mengatur waktunya adalah terlalu banyak mengikuti kegiatan semisal berorganisasi. Hal ini menyebabkan mahasiswa kesulitan dalam mengatur kegiatan belajarnya. Sehingga manajemen waktu merupakan keterampilan yang harus dimiliki mahasiswa (Pratiwi,

2017). Masalah serupa juga terjadi pada mahasiswa UII. Banyak dari mereka yang terlupa tentang jadwal kuliah dan kesulitan dalam mengatur jadwal.

Mahasiswa yang memiliki manajemen waktu yang baik dapat memanfaatkan waktu yang dimiliki untuk menyelesaikan tugas yang harus dikerjakan dengan baik dan tepat waktu, sedangkan mahasiswa yang mempunyai manajemen waktu yang buruk akan kesulitan untuk menyelesaikan tugas yang perlu dikerjakan, dan walaupun mahasiswa tersebut dapat menyelesaikan tugasnya, tetapi hasilnya tidak maksimal karena dikerjakan secara tegesa-gesa (Erikson, 2017).

Kunci agar mahasiswa dapat menerapkan manajemen waktu yang baik adalah perencanaan. Perlu sebuah perencaan untuk perubahan mengatur waktu menjadi lebih baik. Rencana yang dapat diterapkan adalah sebagai berikut: menulis kegiatan dalam buku atau kalender, memberikan prioritas kepada kegiatan yang hendak dilakukan, membuat jadwal secara rutin semisal kegiatan untuk setiap hari atau setiap minggu, dan yang terakhir adalah melakukan kegiatan sesuai jadwal yang ditetapkan (Wilson, 2019).

Untuk mengatasi masalah yang dihadapi mahasiswa tersebut, BSI UII berinisiatif untuk mengembangkan sebuah fitur pada aplikasi *mobile* I'm UII. Fitur tersebut adalah fitur *schedule* yang memiliki tujuan untuk membantu mahasiswa dalam manajemen waktu. Dengan fitur ini mahasiswa dapat mengatur jadwal kegiatannya seperti melihat, membuat, mengubah, dan menghapus jadwal. Pada fitur *schedule* juga terdapat fitur yang memungkinkan mahasiswa mengundang mahasiswa lain ke jadwal yang dibuatnya. Selain itu pada fitur ini terdapat notifikasi yang mengingatkan kepada mahasiswa bahwa mereka mempunyai jadwal sehingga mahasiswa tidak terlupa jadwal tersebut. Yang membedakan fitur *schedule* aplikasi I'm UII dengan fitur penjadwalan pada aplikasi lain yaitu pada desain awal fitur *schedule* mahasiswa secara otomatis mendapatkan jadwal kuliahnya ketika *login* aplikasi, sedangkan fitur penjadwalan pada aplikasi lain, mahasiswa harus membuat jadwal kuliahnya secara manual.

1.2 Ruang Lingkup

Program magang di BSI dilaksanakan selama delapan bulan, yang dimulai pada bulan September 2022 dan berakhir pada bulan April 2023. Selama magang, penulis bertanggung jawab untuk membuat fitur *schedule* pada aplikasi I'm UII. Posisi penulis saat magang adalah sebagai pengembang *mobile* dan *back-end*. Ruang lingkup penulis sebagai pengembang *mobile* adalah sebagi berikut:

1. Menerapkan desain UI/UX pada aplikasi *mobile*

2. Mengimplementasikan API untuk fitur *schedule*
3. Melakukan pengujian *black box* untuk fitur *schedule*

Sedangkan untuk ruang lingkup penulis sebagai pengembang *back-end* diantaranya:

1. Membuat struktur *request* dan *response* API
2. Membuat tabel basis data
3. Mengembangkan API untuk fitur *schedule*

Selain mengembangkan aplikasi *mobile* dan juga *back-end*, penulis juga mengembangkan sistem notifikasi pada aplikasi I'm UII untuk fitur *schedule, reminder*, dan notifikasi aplikasi. Dalam mengembangkan sistem notifikasi, penulis menggunakan layanan Firebase Cloud Messaging (FCM) dan fitur *task scheduling* pada Lumen.

1.3 Tujuan

Tujuan dari pengembangan fitur *schedule* pada aplikasi I'm UII adalah memudahkan mahasiswa dalam mengelola jadwal pribadinya, karena di fitur *schedule* terdapat fungsionalitas seperti melihat jadwal, membuat jadwal, mengundang pengguna lain di jadwal yang dibuat, mengubah jadwal, serta menghapus jadwal.

1.4 Manfaat

Manfaat yang diperoleh dari pengembangan fitur *schedule* pada aplikasi I'm UII antara lain:

1. Pengguna dapat mengelola jadwal pribadinya dengan mudah. Pengguna bisa melihat, membuat, mengubah, dan menghapus jadwal.
2. Pengguna tidak lagi melupakan jadwalnya karena pada fitur *schedule* terdapat notifikasi pengingat yang akan muncul ketika jadwal akan dimulai.

1.5 Sistematika Penulisan

Supaya memberikan gambaran yang jelas tentang struktur dari laporan akhir, maka dibuatlah sistem penulisan yang berisikan informasi yang dijelaskan pada setiap bab-bab. Ada pun sistematika penulisan laporan akhir adalah sebagai berikut:

1. **BAB I PENDAHULUAN**

   Bab ini berisikan latar belakang, ruang lingkup magang, tujuan, manfaat, dan sistematika penulisan.

2. **BAB II KAJIAN PUSTAKA**

Bab ini menjelaskan prinsip-prinsip teoritis yang menjadi dasar dalam proyek yang dilakukan selama periode magang.

3. **BAB III PELAKSANAAN MAGANG**

   Pada bab ini berisikan kegiatan yang penulis lakukan selama magang beserta metode pengembangan yang digunakan.

4. **BAB IV REFLEKSI PELAKSANAAN MAGANG**

   Bab ini menjelaskan tentang refleksi yang penulis dapatkan setelah melaksanakan magang beserta relevansi akademik.

5. **BAB V PENUTUP**

   Pada bab ini berisikan kesimpulan dari laporan akhir dan juga saran untuk penelitian selanjutnya.

BAB II
LANDASAN TEORI DAN TINJAUAN PUSTAKA

2.1 Aplikasi *Mobile*

Aplikasi *mobile* merujuk pada aplikasi yang dikembangkan khusus untuk perangkat *mobile*, seperti *smartphone* atau *tablet* serta memiliki sistem operasi yang mendukung aplikasi atau perangkat lunak yang berjalan di perangkat *mobile*. Aplikasi *mobile* dapat diunduh dari tempat distribusi aplikasi masing-masing sistem operasi, seperti Play Store untuk sistem operasi Android dan App Store untuk pengguna iOS (Wang, Liao, & Yang, 2013). Aplikasi *mobile* memiliki beragam kegunaan seperti berkomunikasi, bersosialisasi, berbelanja, navigasi jalan, sampai permainan. Salah satu keuntungan dari aplikasi *mobile* adalah dapat memanfaatkan sensor-sensor yang terdapat pada perangkat *mobile*, seperti sensor posisi dan navigasi, gerak dan percepatan, serta sidik jari.

2.2 Flutter

Dikutip dari situs web resmi Flutter, Flutter merupakan *framework open-source* yang dikembangkan oleh Google untuk membuat tampilan yang menarik dengan mudah. Flutter juga di-*compile* secara *native* sehingga performanya mirip seperti aplikasi *native* (Flutter, n.d.). Proses *compile* kode Flutter ke kode *native* (Android NDK, LLVM, AOT-compiled) dilakukan tanpa ada interpreter sehingga proses *compile* menjadi lebih cepat (Suhendro, Sudarma, & Khrisne, 2021). Pada pengembangan aplikasi menggunakan Flutter, bahasa pemrograman yang digunakan adalah Dart. Dart pada awalnya dikembangkan untuk mengoptimalkan klien untuk pengembangan aplikasi web dan *mobile*. Salah satu fitur yang membuat Flutter populer adalah penggunaan konsep *widget* yang mudah dikustomisasi dalam pembuatan antarmuka pengguna. Pengembangan aplikasi menggunakan Flutter juga bisa lebih cepat karena terdapat fitur *hot reload* yang memungkinkan pengembang untuk melihat perubahan atas kode yang diubah tanpa menjalankan ulang aplikasi sehingga dapat menghemat waktu.

2.3 Aplikasi I’m UII

Aplikasi I’m UII merupakan aplikasi *mobile* yang dikembangkan oleh Badan Sistem Informasi UII untuk mengatasi masalah yang terjadi di kalangan mahasiswa. Masalah-masalah tersebut adalah kesulitan dalam mencari informasi, terlupa jadwal kuliah, dan tertinggal kegiatan-kegiatan penting seperti pendaftaran KKN.

2.4 Waterfall

Metode pengembangan waterfall merupakan suatu model proses pengembangan perangkat lunak yang berurutan dan linear, yang menggambarkan aliran pekerjaan seperti air terjun, di mana setiap tahap dalam proses hanya dapat dimulai setelah tahap sebelumnya diselesaikan sepenuhnya. Metode pengembangan waterfall terdiri dari beberapa tahap, yaitu analisis kebutuhan, desain, implementasi atau pengembangan, pengujian, dan pemeliharaan.

2.5 Fitur *Schedule* atau Penjadwalan

Fitur *schedule* atau penjadwalan merupakan salah satu fitur yang sangat berguna dalam mengatur dan mengelola waktu dengan efisien. Dengan fitur ini, pengguna dapat dengan mudah membuat jadwal kegiatan, mengatur alarm, dan mengingatkan untuk tugas-tugas yang penting. Fitur penjadwalan memungkinkan pengguna untuk merencanakan dan mengatur waktu dengan lebih terstruktur, sehingga memungkinkan mereka untuk tetap produktif dan menghindari keterlambatan.

2.6 Pengujian Black Box

Terdapat dua bagian yang dilakukan pada tahap pengujian perangkat lunak, yaitu *black box* dan *white box*. Pengujian *black box* merupakan metode pengujian yang hanya berfokus pada spesifikasi kebutuhan tanpa memeriksa rincian implementasinya. Pengujian *black box* bergantung pada analisis kebutuhan perangkat lunak untuk membuat kasus uji yang mencakup skenario yang mungkin terjadi. Perbedaan pengujian *black box* dengan pengujian *white box* terletak pada fokus pengujian. Pengujian *black box* berfokus pada pengujian dari sudut pandang pengguna, sedangkan pengujian *white box* berfokus pada pengujian untuk menemukan *bug* yang terdapat pada kode (Arfan & Hendrik, 2022).

2.7 Tinjauan Pustaka

Sebuah makalah telah mengkaji tentang desain dan implementasi sistem penjadwalan agenda berbasis Android (Rahmah, 2017). Pada makalah ini membahas tentang mengatur jadwal agenda pribadi dan dapat mengirimkan agenda ke pengguna lain. Jika pengguna satu memberikan agenda ke pengguna dua, maka pengguna satu dan dua akan menerima pengingat sesuai agenda yang dikirimkan. Pengingat untuk melakukan jadwal atau kegiatan dalam sistem penjadwalan ini berupa *alert dialog* dan bunyi alarm. *Alert dialog* berisikan nama kegiatan,

kata kunci, dan *form* input kata kunci dari jadwal. Bunyi alarm akan berhenti jika pengguna berhasil memasukan kata kunci di *form* yang tersedia.

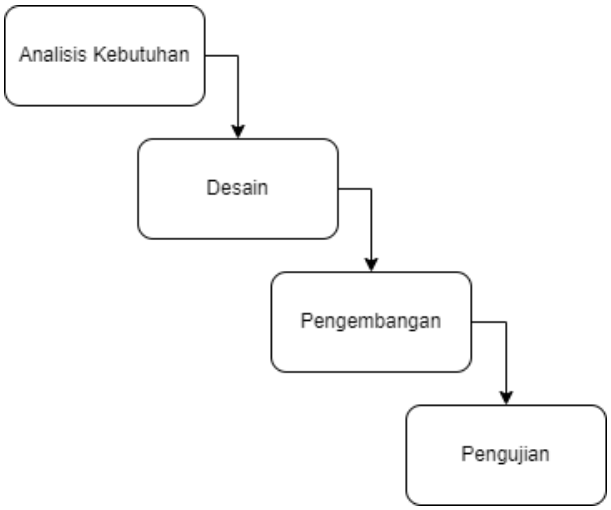
Terdapat sebuah makalah yang membahas tentang pembuatan aplikasi *reminder* jadwal perkuliahan yang berbasis Android untuk Jurusan Elektronika Universitas Negeri Padang (Edde & Budayawan, 2021). Pada makalah ini membahas tentang aplikasi pembuatan jadwal kuliah berbasis Android yang terdapat notifikasi pengingat untuk jadwal kuliah yang dibuat. Aktor atau pengguna yang dapat membuat dan mengubah jadwal kuliah hanya lah dosen, sedangkan mahasiswa hanya bisa melihat, mengatur waktu notifikasi, dan mendapatkan notifikasi jadwal kuliah.

Terdapat aplikasi *mobile* yang serupa dengan fitur *schedule* pada aplikasi I'm UII yaitu Microsoft To Do. Aplikasi ini dirancang untuk mengatasi masalah umum yang dialami oleh pengguna sehari-hari, seperti kesulitan dalam mengorganisir tugas-tugas yang perlu dilakukan (Microsoft Corporation, 2023). Pada Microsoft To Do, pengguna dapat menambahkan pengguna lain ke jadwal atau kegiatan yang dibuat dengan cara membuat *list* terlebih dahulu, lalu membuat kegiatan yang perlu dilakukan serta mengundang pengguna lain ke *list* tersebut, dan menambahkan pengguna lain ke kegiatan yang dibuat. Pada aplikasi ini juga pengguna yang berpartisipasi pada suatu kegiatan akan mendapatkan notifikasi pengingat kegiatan.

Selain Microsoft To Do, ada aplikasi yang mirip dengan fitur *schedule* pada aplikasi I'm UII yaitu Google Calendar. Aplikasi *mobile* Google Calendar adalah sebuah aplikasi yang memungkinkan pengguna untuk mengelola jadwal, acara, dan pengingat secara efisien langsung dari perangkat seluler mereka. Pengguna dapat mengundang orang lain ke acara mereka dengan memasukkan alamat email tamu (Google LLC, 2023).

BAB III
PELAKSANAAN MAGANG

Dalam pengembangan fitur *schedule* aplikasi I'm UII pada tim percepatan menggunakan metode pengembangan *waterfall*. *Waterfall* adalah metode pengembangan aplikasi yang prosesnya dilakukan secara berurutan dari satu tahap ke tahap lain. Ada lima tahap dalam pengembangan menggunakan metode *waterfall* yaitu analisis kebutuhan, desain, implementasi, pengujian, dan pemeliharaan (Sommerville, 2010). Tahapan metode pengembangan *waterfall* pada fitur *schedule* aplikasi I'm UII bisa dilihat pada Gambar 3.1. Tahapan pengembangan fitur *schedule* meliputi analisis kebutuhan, desain, pengembangan, dan pengujian. Karena fitur *schedule* aplikasi I'm UII masih dalam tahap pengujian, maka pengembangan belum masuk ke dalam tahap pemeliharaan.



Gambar 3.1 Tahapan metode pengembangan *waterfall* untuk fitur *schedule*

3.1 Analisis Kebutuhan

Pada tahap ini dilakukan pengumpulan data terkait kebutuhan calon pengguna sehingga dapat menspesifikasikan kebutuhan aplikasi agar dapat menyelesaikan masalah yang dihadapi pengguna dengan tepat. Di tahap analisis kebutuhan juga ditetapkan *service* atau fitur, batasan, serta tujuan pengembangan aplikasi (Sommerville, 2010). Pada tahap analisis kebutuhan terdapat beberapat beberapa proses yang dilakukan, misal deskripsi sistem dan perangkat lunak, analisis kebutuhan, dan pemodelan analisis. Beberapa proses akan dijelaskan pada sub bab bahasan berikutnya.

3.1.1 Deskripsi Sistem dan Perangkat Lunak

I’m UII merupakan aplikasi *mobile* yang memiliki tujuan untuk membantu mahasiswa dalam menjalani masa kuliahnya. Pada aplikasi *mobile* I’m UII terdapat fitur *schedule* yang berfungsi untuk mengelola jadwal mahasiswa. Laporan akhir ini hanya memiliki domain yaitu pengembangan fitur *schedule* pada aplikasi I’m UII.

3.1.2 Analisis Kebutuhan

Sebelum mengembangkan aplikasi, penulis dan rekan magang melakukan wawancara langsung dan survei daring kepada mahasiswa UII tentang masalah yang sering dihadapi saat kuliah. Mahasiswa narasumber berasal dari berbagai tahun angkatan dan fakultas di UII. Hal ini dilakukan untuk mendapatkan pemahaman tentang masalah yang dihadapi oleh mahasiswa UII secara umum saat menjalani kuliah. Analisis data hasil wawancara dan survei dilakukan oleh tim produk untuk menentukan fitur yang akan dikembangkan. Pengembangan fitur *schedule* didasarkan pada masalah sulitnya mengatur waktu bagi mahasiswa yang didapatkan ketika melakukan wawancara dan survei. Adapun hasil wawancara dan survei terkait dengan masalah sulitnya mengatur waktu di mahasiswa ditunjukan pada Tabel 3.1. Pada tabel tersebut terdapat kolom fakultas, angkatan, masalah yang dihadapi, serta tujuan mahasiswa.

Tabel 3.1 Deskripsi diagram *usecase*

Fakultas	**Angkatan**	**Masalah**	**Tujuan/Keinginan**
FPSB	2020	Kesulitan mengatur waktu organisasi dan perkuliahan	Dapat mengatur waktu lebih efektif lagi
FTI	2020	Kesulitan dalam mengatur waktu dan prioritas	Dapat mengatur waktu lebih baik supaya lebih produktif
FIAI	2017	Sering terlupa jadwal kuliah atau kegiatan	Mendapatkan notifikasi terkait jadwal kuliah atau kegiatan
FTSP	2022	Untuk melihat jadwal, prosesnya terbilang susah. Mahasiswa harus masuk ke website Gateway	Kemudahan dalam melihat jadwal kuliah agar tidak harus membuka website setiap ingin melihat jadwal
FTSP	2018	Kesulitan untuk mengatur waktu kegiatan akademik	Mendapatkan pengingat lini masa

Dari hasil wawancara dan survei yang dilakukan, dapat disimpulkan bahwa beberapa mahasiswa UII masih kesulitan dalam mengatur waktunya. Untuk dapat menyelesaikan masalah yang dihadapi, maka perlu dirancang kebutuhan fungsionalitas yang harus terdapat pada fitur *schedule* aplikasi I'm UII. Kebutuhan fungsionalitas dapat dilihat pada Tabel 3.2. Pada tabel ini terdapat dua kolom yaitu kode dan deskripsi kebutuhan fungsionalitas.

Tabel 3.2 Kebutuhan fungsionalitas fitur *schedule*

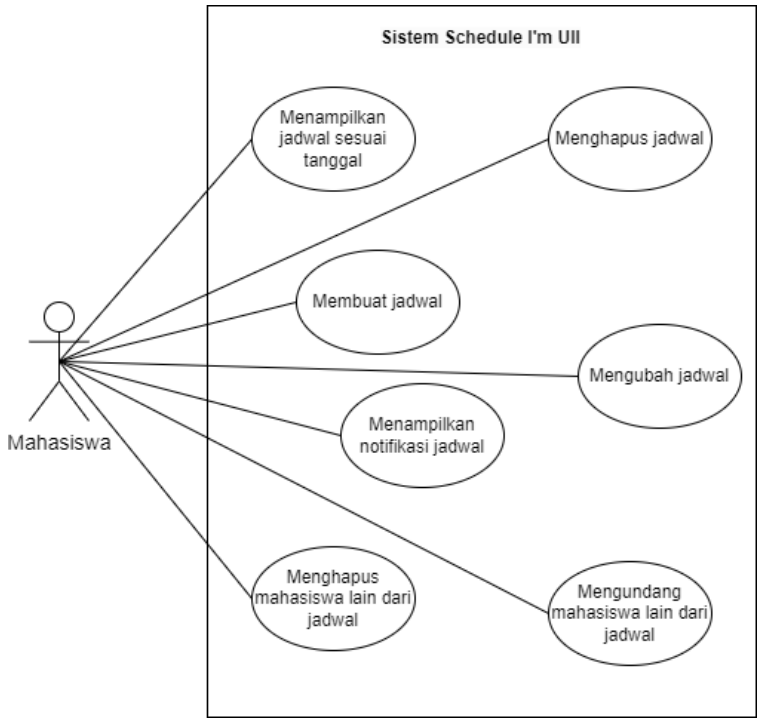
Kode	**Deskripsi**
IKF-01	Perangkat lunak dapat menampilkan jadwal pribadi mahasiswa sesuai dengan tanggal.
IKF-02	Perangkat lunak dapat membuat jadwal
IKF-03	Perangkat lunak dapat mengubah jadwal
IKF-04	Perangkat lunak dapat menghapus jadwal
IKF-05	Perangkat lunak dapat menampilkan notifikasi jadwal
IKF-06	Perangkat lunak dapat mengundang pengguna ke jadwal
IKF-07	Perangkat lunak dapat menghapus pengguna dari jadwal

Dalam konteks fitur *schedule* atau aplikasi I'm UII secara keseluruhan hanya memiliki satu aktor atau pengguna yaitu mahasiswa Universitas Islam Indonesia. Tabel 3.3 secara rinci menjelaskan tujuh *use case* yang terkait dengan fitur *schedule* pada aplikasi I'm UII. Tabel tersebut mengandung dua kolom yang mencakup kode dan nama serta deskripsi dari setiap *use case* yang memberikan informasi terperinci mengenai masing-masing *use case*. Dalam analisis kebutuhan fungsionalitas, ketujuh *use case* telah teridentifikasi dan dipetakkan dengan cermat untuk memenuhi kebutuhan pengguna.

Dalam upaya untuk menggambarkan dengan jelas dan memvisualisasikan fungsionalitas perangkat lunak yang sedang dibuat, penulis membuat sebuah diagram *use case* untuk fitur *schedule* aplikasi I'm UII. Ada dua komponen utama yang dihadirkan dalam diagram *use case* tersebut, yaitu aktor dan *use case* atau kegiatan yang dapat dilakukan oleh aktor. Diagram *use case* ini berfungsi sebagai alat bantu untuk memahami dan menggambarkan interaksi antara aktor dengan *use case* dalam sistem perangkat lunak yang sedang dikembangkan. Gambar 3.2 yang ditampilkan dibawah merupakan diagram *use case* dari fitur *schedule* pada aplikasi *mobile* I'm UII. Pada diagram *use case* tersebut terdapat satu aktor yaitu mahasiswa beserta tujuh *use case* yang telah terdefinisi.

Tabel 3.3 Daftar *use case* fitur *schedule*

Kode	**Nama & Deskripsi**
IUC-01	**Nama**: Menampilkan jadwal sesuai dengan tanggal **Deskripsi:** Mahasiswa UII dapat melihat jadwal kegiatan yang dimilikinya dengan cara memilih bulan dan tanggal di kalender.
IUC-02	**Nama**: Membuat jadwal **Deskripsi**: Mahasiswa dapat membuat jadwal dengan mengisi *form* buat jadwal dan mahasiswa juga bisa mengundang mahasiswa lain ke dalam jadwal yang akan dibuat
IUC-03	**Nama**: Mengubah jadwal **Deskripsi**: Mahasiswa dapat mengubah jadwal yang sudah dibuat seperti judul, tanggal, pengulangan, atau pun mahasiswa yang diundang. Mahasiswa yang dapat mengubah jadwal adalah mahasiswa yang membuat jadwal, sedangkan mahasiswa yang diundang dalam suatu jadwal tidak bisa mengubah jadwal
IUC-04	**Nama**: Menghapus jadwal **Deskripsi**: Mahasiswa dapat menghapus jadwal, baik itu jadwal yang dipilih sesuai dengan tanggal tertentu saja atau jadwal yang termasuk pengulangannya. Baik mahasiswa yang membuat atau yang diundang di suatu jadwal dapat menghapus jadwal
IUC-05	**Nama**: Menampilkan notifikasi jadwal **Deskripsi**: Mahasiswa mendapatkan notifikasi ketika terdapat jadwal yang akan segera dimulai. Waktu pengiriman notifikasi bisa diatur saat membuat atau mengubah jadwal. Waktu notifikasi yang bisa dipilih adalah 5, 15, 30, 60 menit sebelum jadwal dimulai
IUC-06	**Nama**: Mengundang pengguna lain ke jadwal **Deskripsi**: Mahasiswa bisa mengundang pengguna lain ke jadwal yang dibuatnya. Mahasiswa yang diundang akan memiliki jadwal tersebut. *Use case* ini bisa dilakukan ketika pengguna membuat atau mengubah jadwal
IUC-07	**Nama**: Menghapus pengguna lain dari jadwal **Deskripsi**: Mahasiswa bisa menghapus pengguna lain ke jadwal yang dibuatnya. Sistem akan menghapus jadwal mahasiswa yang dihapus dari jadwal. *Use case* ini bisa dilakukan ketika pengguna mengubah jadwal



Gambar 3.2 Diagram *use case* fitur *schedule*

3.1.3 Pemodelan Analisis

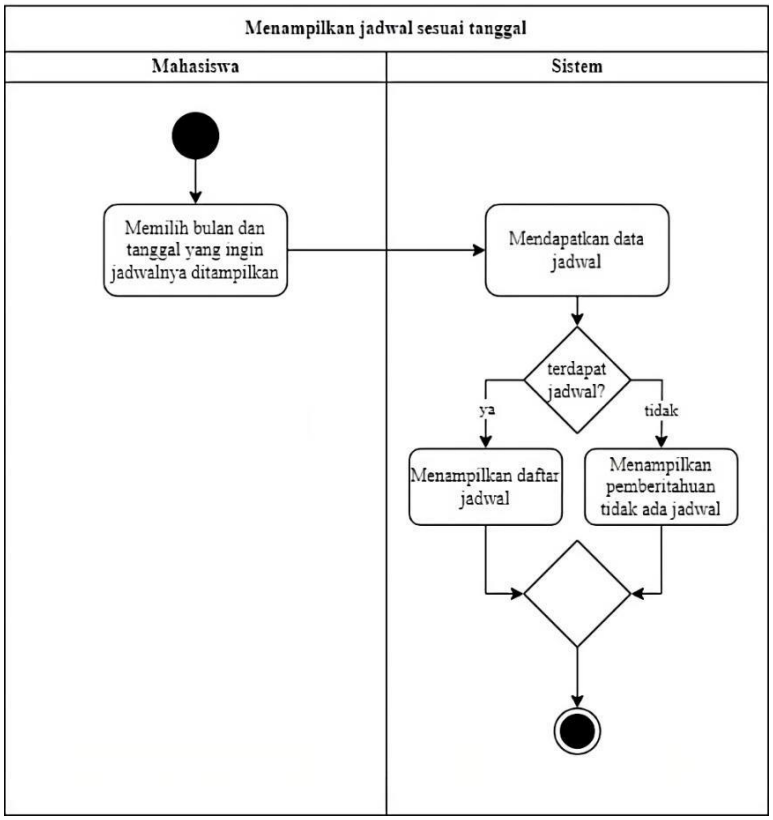
Untuk memvisualisasikan urutan aktivitas, pengambilan keputusan, dan aliran kontrol yang terjadi pada suatu proses bisnis dibuatlah *activity diagram*. Berikut adalah *activity diagram* dari fitur *schedule* beserta penjelasannya.

a. IUC-01 Menampilkan jadwal sesuai dengan tanggal

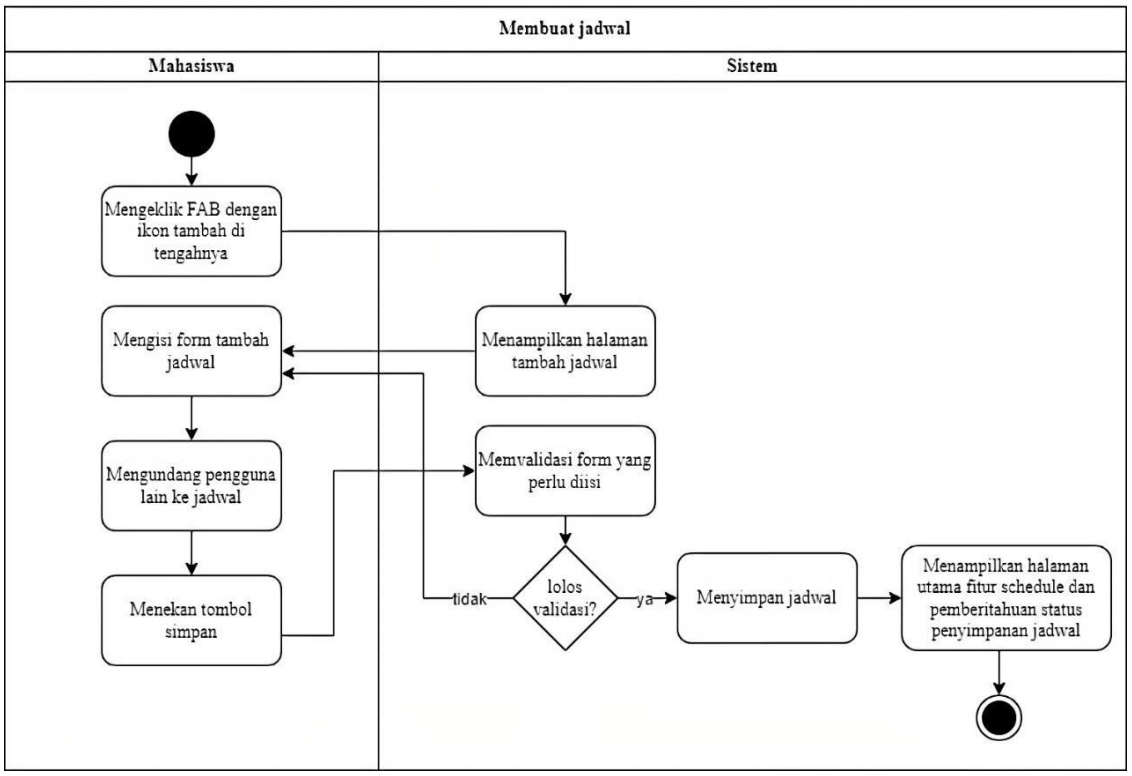
Pada Gambar 3.3 menunjukan *activity diagram* untuk menunjukan urutan aktifitas pada *use case* menampilkan jadwal sesuai tanggal. Mahasiswa sebagai aktor dapat melihat jadwal yang dimilikinya pada hari tertentu dengan cara memilih bulan lalu mengeklik tanggal yang ingin jadwalnya ditampilkan, setelah itu sitem akan mencoba untuk mendapatkan data jadwal.

b. IUC-02 Membuat jadwal

Mahasiswa dapat membuat jadwal pribadinya dengan cara mengisi *form* pembuatan jadwal dengan mengisikan detail jadwal seperti nama jadwal, tanggal dimulai dan berakhir, pengulangan, waktu notifikasi, lokasi, deskripsi, dan dapat mengundah mahasiswa lain di jadwal yang dibuat. Alur dari *use case* membuat jadwal divisualisasikan menggunakan *activity diagram* yang ditunjukan pada Gambar 3.4.



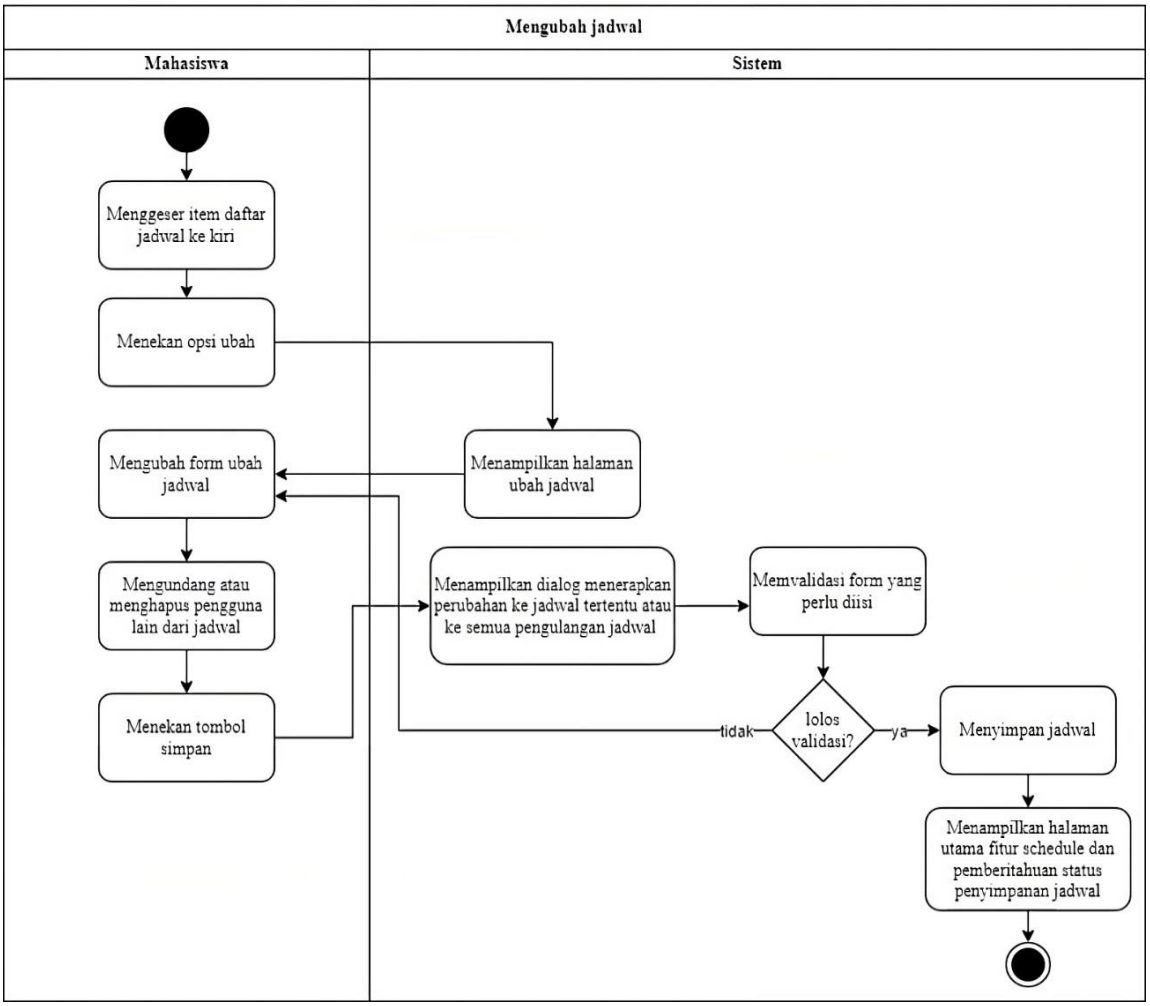
Gambar 3.3 Melihat jadwal sesuai tanggal



Gambar 3.4 Membuat jadwal

c. IUC-03 Mengubah jadwal

Mahasiswa dapat mengubah jadwal yang dibuatnya sendiri tetapi tidak bisa mengubah jadwal yang mana pada jadwal tersebut dia hanya berperan sebagai tamu yang diundang. Form yang terdapat pada *use case* ini sama dengan *use case* membuat jadwal. Pengguna dapat mengundang atau menghapus pengguna lain dari jadwal. Mahasiswa juga bisa mengubah jadwal tertentu saja atau semua yang termasuk pengulangan dari jadwal yang diubah dengan cara aplikasi akan menampilkan *alert dialog* yang menanyakan kepada pengguna terkait penerapan perubahan sebelum menyimpan jadwal tersebut ke sistem. *Activity diagram* untuk mengubah jadwal ditampilkan pada Gambar 3.5.

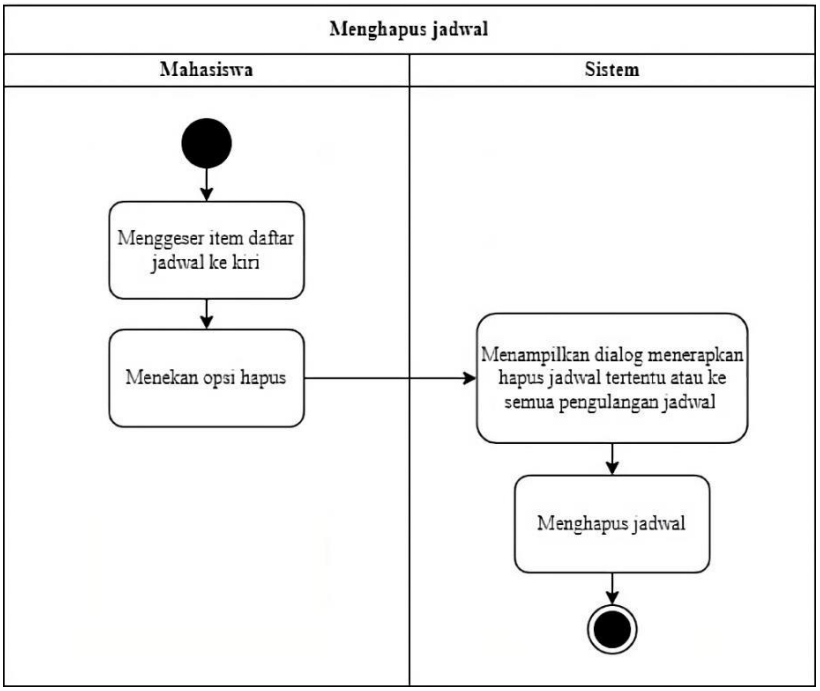


Gambar 3.5 Mengubah jadwal

d. IUC-04 Menghapus jadwal

Apabila terdapat kesalahan pada jadwal yang dibuat, mahasiswa bisa melakukan aksi menghapus jadwal. Mahasiswa yang diundang pada suatu jadwal juga dapat melakukan aksi

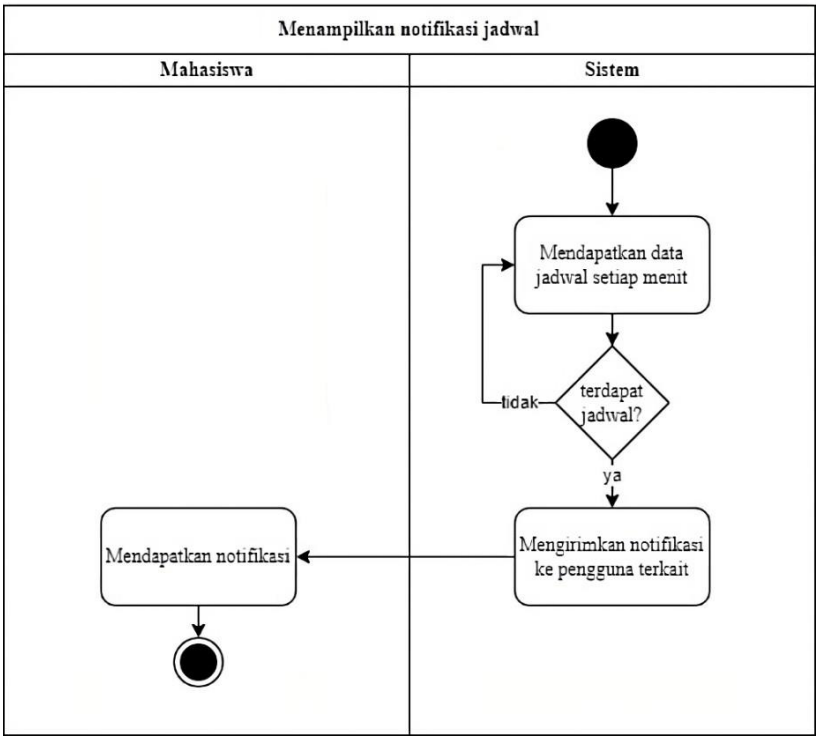
hapus jadwal, karena mungkin saja pembuat jadwal salah mengundang orang. Saat melakukan aksi hapus jadwal, aplikasi akan memunculkan *alert dialog* yang menanyakan kepada pengguna untuk menghapus jadwal tertentu saja atau menghapus semua jadwal yang merupakan pengulangan dari jadwal tersebut. *Activity diagram* yang menggambarkan urutan aktifitas hapus jadwal ditampilkan pada Gambar 3.6.



Gambar 3.6 Menghapus jadwal

e. IUC-05 Menampilkan notifikasi jadwal

Jika waktu jadwal akan dimulai, maka mahasiswa akan menerima notifikasi pemberitahuan bahwa jadwal akan dimulai. Pengiriman notifikasi dari sistem ke gawai pengguna berdasarkan waktu jadwal dan pengiriman notifikasi. Waktu pengiriman notifikasi dapat diatur saat membuat atau mengubah jadwal. Waktu pengiriman notifikasi yang bisa dipilih adalah 5, 15, 30, 60 menit sebelum jadwal dimulai. *Activity diagram* yang menunjukan urutan proses untuk menampilkan notifikasi jadwal dapat dilihat pada Gambar 3.7. Pada *activity diagram* tersebut terlihat bahwa sistem mendapatkan data jadwal setiap menitnya, jika terdapat jadwal pada menit tertentu maka sistem akan mengirimkan notifikasi ke gawai pengguna, sedangkan jika tidak terdapat jadwal, maka sistem akan mencoba untuk mendapatkan jadwal pada menit selanjutnya.



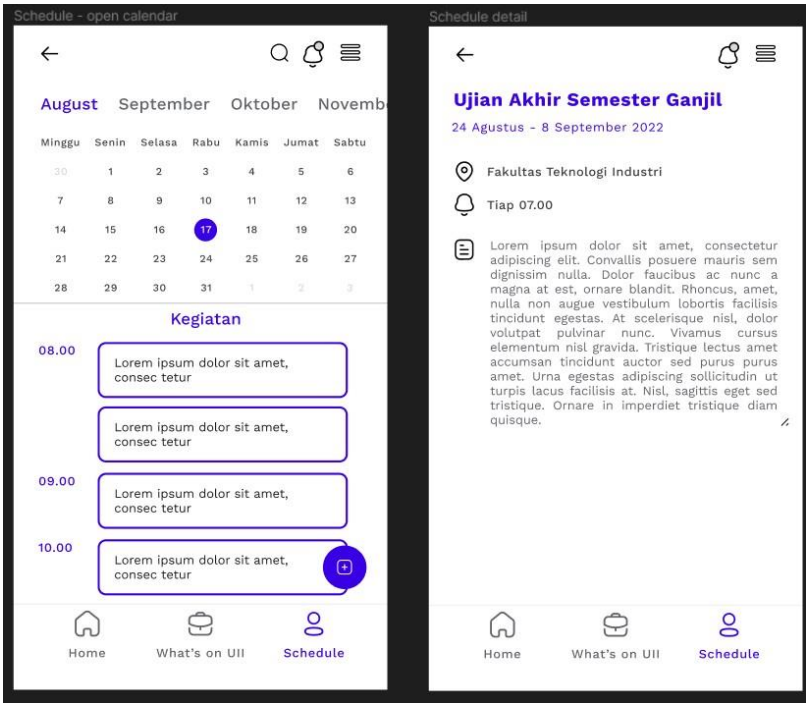
Gambar 3.7 Menampilkan notifikasi jadwal

3.2 Desain

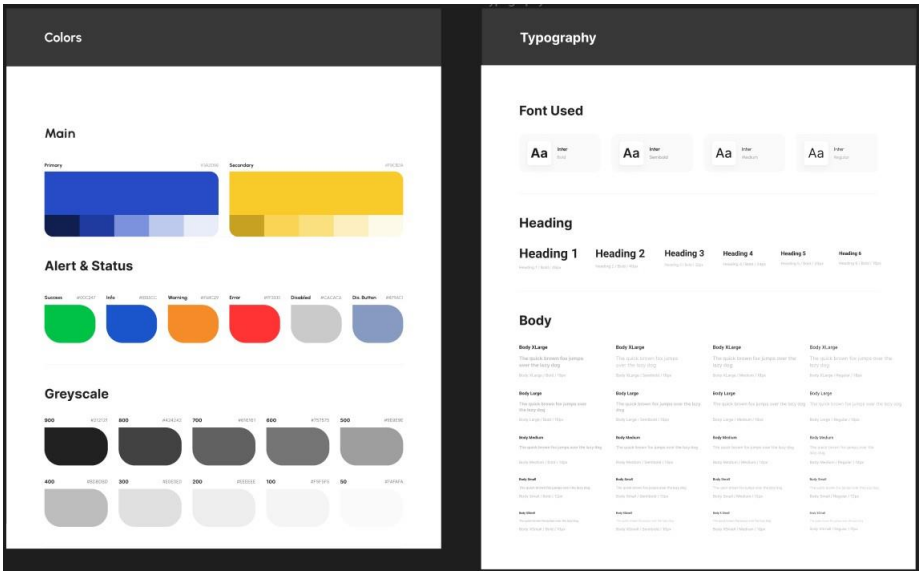
Keluaran dari tahap desain berupa *prototype* aplikasi I'm UII. Tahap desain ini dikerjakan oleh tim produk atau tim UI/UX yang terdiri dari Anwaruddin Ridho Novianto dan Muhammad Habib Izdhihar Syaf. Tahap desain ini dilakukan setelah tahap analisis kebutuhan karena pada tahap desain diperlukan data yang didapatkan dari tahap sebelumnya untuk dapat memahami kebutuhan dan preferensi pengguna. Setelah mendapatkan kebutuhan pengguna melalui wawancara dan survei, barulah tim produk membuat *wireframe* atau kerangka dasar aplikasi untuk merancang tata letak elemen-elemen utama yang meliputi navigasi, tombol, dan konten. *Wireframe* untuk halaman kalender dan detail jadwal bisa dilihat pada Gambar 3.8. Pada *wireframe* halaman kalender terdapat beberapa komponen seperti kalender, daftar jadwal, dan tombol tambah jadwal.

Tahap berikutnya dalam desain adalah membuat panduan desain seperti penggunaan warna, tipografi, ikon, dan elemen grafis lainnya akan ditentukan untuk menciptakan identitas visual aplikasi yang konsisten dan menarik. Setelah membuat panduan desain, tim produk akan melanjutkan ke tahap *prototyping*, di mana mereka akan membuat model interaktif aplikasi menggunakan alat seperti Adobe XD dan Figma. Dalam *prototyping* aplikasi I'm UII menggunakan Figma. *Prototyping* memungkinkan mereka untuk menguji dan memvalidasi

desain secara *real-time*, mendapatkan umpan balik dari pengguna, dan membuat perubahan yang diperlukan sebelum memasuki tahap pengembangan. Tahapan desain tampilan aplikasi ini memastikan bahwa pengguna dapat memiliki pengalaman yang baik dan efektif saat menggunakan aplikasi I'm UII. Panduan desain tipografi dan warna dapat dilihat pada Gambar 3.9. Pada gambar tersebut tersedia warna dan tipografi yang digunakan pada aplikasi I'm UII.



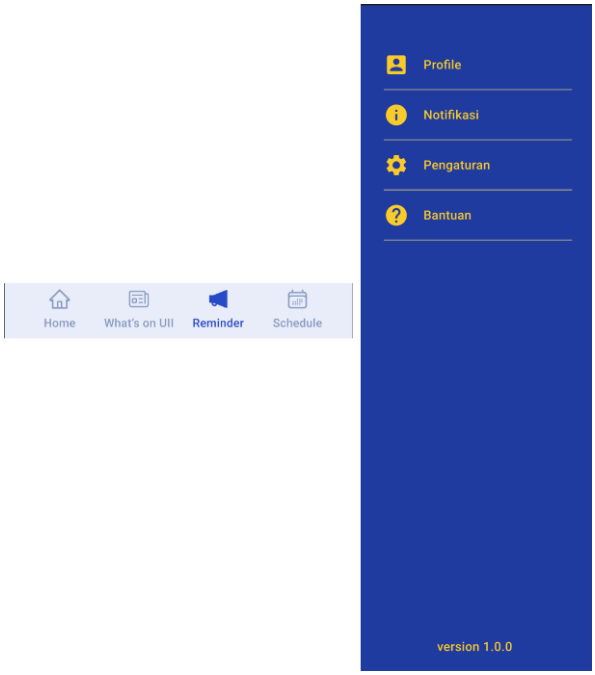
Gambar 3.8 Wireframe halaman *schedule* dan detail jadwal



Gambar 3.9 Panduan desain tipografi dan warna

Tema warna yang digunakan pada aplikasi *mobile* I'm UII adalah tema warna dari logo UII yang sesuai dengan pedoman penggunaan merek Universitas Islam Indonesia (Universitas Islam Indonesia, 2018). Supaya memberikan pengalaman pengguna yang menarik, digunakan juga variasi warna yang merupakan turunan dari tema warna dasar aplikasi. Penggunaan variasi warna dapat memudahkan mahasiswa memahami hierarki, memahami pola, dan navigasi antar halaman. Selain warna turunan dari warna dasar, digunakan juga warna untuk peringatan dan status yang dipilih berdasarkan efek psikologi dari warna tersebut, semisal warna merah menandakan bahaya atau terjadi kesalahan dan warna hijau menandakan sukses atau suatu proses berjalan dengan baik. Untuk memenuhi kebutuhan warna netral pada aplikasi, digunakan warna *greyscale*.

Menu navigasi terletak pada bagian dasar layar. Menu navigasi ini terdiri dari fitur utama aplikasi I'm UII, seperti *what's on* berita, *reminder*, *schedule*, dan juga *home* atau beranda. Posisi menu navigasi yang berada di dasar layar memiliki fungsi untuk memudahkan pengguna dalam mengakses fitur utama karena mudah dijangkau oleh ibu jari ketika menggunakan *smartphone* dengan satu tangan. Sedangkan untuk menu yang sifatnya sekunder seperti menu profil, notifikasi, pengaturan, dan bantuan, diletakan pada *sidebar* yang akan muncul jika menekan ikon *hamburger* yang berada di pojok kanan atas layar. Menu navigasi dan menu *sidebar* dapat dilihat pada Gambar 3.10.



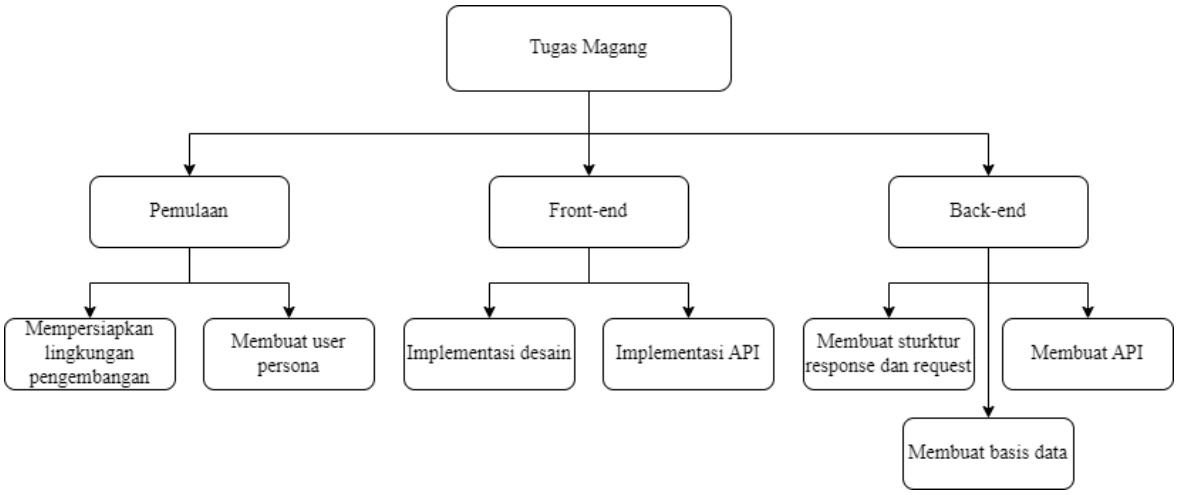
Gambar 3.10 Menu navigasi dan *sidebar* aplikasi I'm UII

3.3 Pengembangan

Selama proses pelaksanaan magang, penulis sudah menyelesaikan beberapa tugas yang diberikan. Dalam mengerjakan tugas, penulis menggunakan beberapa *tools* seperti berikut:

1. IDE dan *code editor*: Android Studio dan Visual Studio Code
2. *Version control:* GitLab
3. Pengujian API: Postman
4. Pengembangan aplikasi *mobile*: Flutter
5. Basis data manajemen: Dbeaver

Tugas-tugas yang penulis kerjakan selama kegiatan magang dapat dibagi menjadi tiga bagian, yaitu, pemulaan, *front-end*, dan *back-end.* Tugas-tugas yang terdapat di setiap bagian bisa dilihat pada Gambar 3.11.



Gambar 3.11 Tugas magang

Tahap pengembangan ini merupakan tahap penerjemahan desain yang yang telah dibuat menjadi aplikasi yang bisa dieksekusi. Pada tahap ini penulis mengerjakan aplikasi *mobile* dan juga *back-end* untuk aplikasi I'm UII. Tugas yang penulis kerjakan pada tahap ini adalah menerapkan desain antarmuka, membuat struktur *request* dan *response* API, merancang tabel basis data, membuat API untuk semua fungsionalitas pada fitur *schedule*, mengimplementasikan API yang telah dibuat ke aplikasi *mobile*, dan membuat sistem notifikasi untuk fitur *schedule*, *reminder*, dan notifikasi aplikasi.

Tugas pertama yang penulis lakukan pada tahap ini adalah menerapkan desain antarmuka yang telah dibuat oleh tim produk ke aplikasi *mobile*. Desain aplikasi I'm UII yang dibuat oleh tim produk, terdapat tiga fitur utama yaitu fitur *schedule*, *reminder*, dan *what's on* berita.

Penulis dan rekan tim pengembang mencoba untuk menerapkan desain, dengan fokus penulis ke penerapan desain untuk fitur *schedule.*

Tugas selanjutnya yang penulis kerjakan adalah membuat struktur *response* dan *request* API agar pembuatan API nanti hanya perlu mengikuti struktur *response* dan *request* yang telah dibuat. Setalah membuat struktur *response* dan *request*, penulis membuat tabel basis data yang sesuai dengan struktur *response* API. Jika tabel basis data untuk fitur *schedule* sudah tersedia, tugas penulis selanjutnya adalah membuat API menggunakan Lumen untuk fungsionalitas yang ada di fitur *schedule*. Supaya mahasiswa mendapatkan pengingat dari jadwal yang dimilikinya, maka diperlukan notifikasi. Penulis mengembangkan sistem notifikasi untuk fitur *schedule, reminder*, dan notifikasi aplikasi menggunakan layanan Firebase Cloud Messaging (FCM) untuk mengirimkan *push notification* dan *task scheduling* pada Lumen untuk mendapatkan jadwal setiap menit dan jika terdapat jadwal maka akan mengirimkan notifikasi FCM dengan bantuan *package* firebase-php yang dibuat oleh kreait di Lumen.

3.3.1 Implementasi Desain Antarmuka

Terdapat tiga fitur utama dalam aplikasi I'm UII yaitu berita atau *what's on*, *reminder,* dan *schedule.* Penulis mendapatkan tugas untuk mengembangkan fitur *schedule*. Fitur *schedule* ini berfungsi untuk membantu pengguna dalam mengelola jadwal karena pada fitur *schedule* pengguna dapat menyimpan, mengubah, menghapus, dan berbagi jadwal harian, mingguan, bulanan, dan tahunan. Pengguna dapat menambahkan jadwal dan terhubung dengan pengguna lain untuk berbagi jadwal.

Setelah tim produk selesai membuat *prototype* dari fitur *schedule* dan penulis sudah memahami tentang dasar-dasar Flutter maka tugas selanjutnya yang penulis lakukan adalah menerapkan desain antarmuka pengguna menggunakan *framework* Flutter. Untuk fitur *schedule* ini memiliki lima halaman yaitu halaman kalender jadwal, detail jadwal, tambah jadwal, ubah jadwal, dan tambah orang.

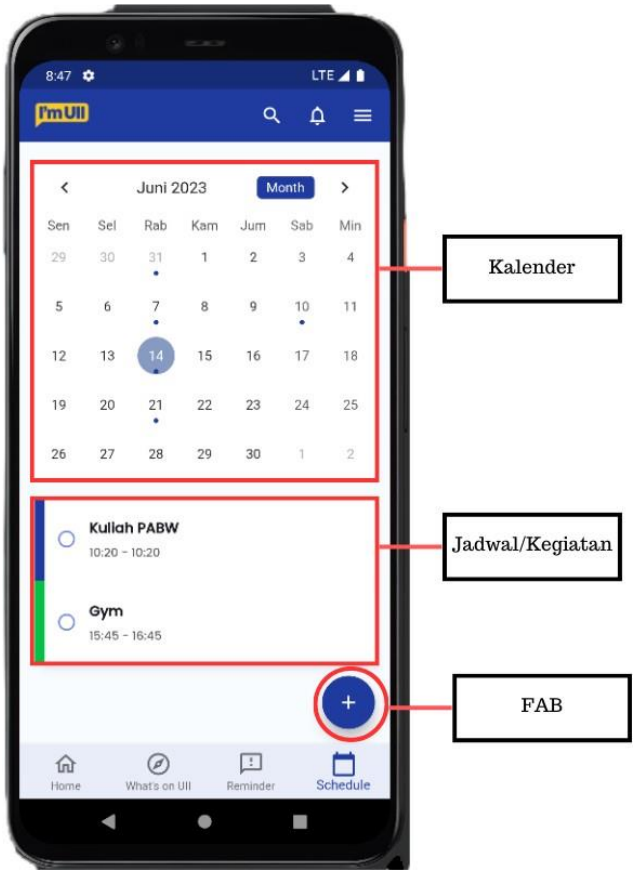
Halaman pertama yang penulis kerjakan adalah halaman kalender jadwal yang merupakan halaman pertama yang ditampilkan saat pengguna membuka fitur *schedule*. Fungsi utama dari halaman ini adalah menampilkan daftar jadwal yang relevan jika pengguna mengeklik bulan dan tanggal tertentu pada komponen kalender yang disediakan. Pada halaman ini terdapat beberapa komponen, yaitu kalender, daftar jadwal untuk tanggal tertentu, dan tombol FAB (Floating Action Button) untuk mengarahkan pengguna ke halaman tambah jadwal.

Untuk membuat halaman kalender jadwal ada 4 langkah yang harus dilakukan. Langkah pertama adalah menampilkan kalender. Komponen kalender berfungsi ganda sebagai alat untuk menampilkan daftar jadwal dengan memilih tanggal, sekaligus sebagai nilai awal ketika pengguna ingin membuat jadwal baru. Penulis menggunakan *package* Flutter TableCalendar. Dipilihnya *package* ini untuk menampilkan kalender karena tampilan dari *package* ini mirip dengan tampilan desain antarmuka yang dibuat. Selain mirip dengan desain, dipilihnya *package* ini karena dapat dikustomisasi sesuai kebutuhan seperti mengganti nama bulan atau hari yang awalnya dalam bahasa Inggris menjadi nama bulan atau hari dalam bahasa Indonesia.

Langkah kedua adalah menampilkan daftar jadwal saat pengguna memilih tanggal tertentu pada komponen kalender. Karena ini hanya menerapkan tampilan antarmuka pengguna saja, maka penulis hanya menggunakan data *sample* saja. Setiap *item* jadwal dibuat dari sebuah *card.* Pada *card* tersebut berisikan judul dari jadwal, jam dimulai hingga jam selesai, indikator warna untuk membedakan jadwal, *slideable* yang akan muncul ketika pengguna menggeser *item* daftar jadwal ke kiri yang digunakan untuk menampilkan opsi ubah dan hapus, serta komponen *checkbox* untuk menandai jika jadwal sudah selesai dilaksanakan atau belum, pengguna dapat mencentang untuk menandakan bahwa jadwal tersebut sudah selesai dilaksanakan. Jika pengguna mengeklik tanggal tertentu dan pada tanggal tersebut tidak ada jadwal maka akan ditampilkan gambar yang memberitahukan pengguna bahwa pengguna tidak memiliki jadwal pada tanggal tersebut. Bagian tersusah saat menampilkan daftar jadwal untuk tanggal tertentu adalah menyusun tampilan *item* jadwal karena harus responsif untuk setiap layar *device mobile*.

Langkah ketiga melibatkan perubahan *style* dari *highlight* tanggal yang berfungsi untuk menyoroti tanggal, baik itu tanggal hari ini atau tanggal yang dipilih yang awalnya memiliki bentuk segiempat menjadi bentuk lingkaran. Untuk *hightlight* tanggal hari ini, penulis memilih warna biru gelap yang terinspirasi dari warna biru yang digunakan dalam logo UII, sedangkah untuk tanggal yang diklik diberi warna abu-abu.

Langkah terakhir dalam membuat halaman kalender jadwal yaitu menambahkan tanda titik di bawah tanggal yang menandakan terdapat jadwal pada tanggal tersebut. Menurut penulis ini adalah bagian tersusah dalam membuat halaman kalender jadwal karena belum banyak yang membahas ini di internet sehingga penulis harus membaca dokumentasi dari *package* TableCalendar. Gambar 3.12 merupakan hasil implementasi desain dari halaman kalender.



Gambar 3.12 Implementasi halaman kalender

Langkah berikutnya dalam implementasi desain antarmuka adalah membuat halaman detail jadwal yang memberikan informasi lebih rinci tentang suatu jadwal. Halaman detail jadwal ini terbuka jika pengguna mengeklik salah satu *item* pada daftar jadwal. Halaman detail jadwal ini mencakup beberapa komponen yang memberikan rincian yang lebih lengkap. Pertama, terdapat judul jadwal, selanjutnya, ada warna indikator yang membantu membedakan jenis jadwal tersebut. Informasi tanggal dimulai dan berakhirnya jadwal juga disajikan dengan jelas. Selain itu, waktu mulai dan selesai jadwal juga ditampilkan, memungkinkan pengguna untuk mengetahui durasi keseluruhan acara. Lokasi pelaksanaan jadwal juga dicantumkan dalam halaman detail yang dapat mempermudah pengguna mengetahui lokasi pelaksanaan jadwal. Informasi mengenai pengulangan jadwal juga tersedia. Pada halaman ini juga ditampilkan waktu pengiriman notifikasi, seperti menerima notifikasi 15 menit sebelum jadwal dimulai. Terdapat kolom deskripsi dari jadwal yang berisikan informasi tambahan mengenai jadwal. Berikutnya ada komponen pengguna yang terlibat dalam jadwal yang berisikan pembuat jadwal dan pengguna yang diundang. Serta yang terakhir terdapat komponen tombol

untuk mengubah status jadwal, apakah sudah selesai dilaksanakan atau belum. Gambar 3.13 merupakan hasil implementasi desain untuk halaman detail jadwal yang di dalamnya terdapat beberapa komponen.

Gambar 3.13 Implementasi halaman detail jadwal

Setelah menerapkan desain antarmuka untuk halaman detail jadwal, halaman selanjutnya yang penulis implementasikan adalah halaman membuat jadwal. Seperti namanya, halaman ini memiliki fungsi utama untuk membuat jadwal baru. Untuk membuat jadwal terdapat beberapa langkah. Pertama pengguna mengeklik tombol FAB (Floating Action Button) yang memiliki ikon tambah di tengahnya pada halaman kalender. Setelah menakan tombol FAB, pengguna akan diarahkan ke halaman buat jadwal yang menyediakan formulir untuk membuat jadwal. Pada formulir tersebut pengguna dapat memasukan informasi jadwal yang relevan seperti judul, tanggal dimulai dan berakhirnya jadwal, pengulangan, waktu ditampilkan notifikasi, lokasi tempat pelaksanaan jadwal, dan deskripsi dari jadwal. Pengguna juga dapat menambahkan pengguna lain ke jadwal yang dibuat dengan menekan tombol Tambahkan Orang. Setelah memastikan bahwa informasi jadwal yang dibuat sudah benar, pengguna

mengeklik tombol Simpan yang berada di kanan *app bar*, selanjutnya akan muncul dialog peringatan yang memberitahukan apakah pengguna akan menyimpan jadwal atau tidak. Gambar 3.14 adalah hasil implementasi untuk halaman buat jadwal.

Gambar 3.14 Implementasi halaman buat jadwal

Selanjutnya penulis menerapkan desain antarmuka untuk halaman ubah jadwal. Pengguna akan diarahkan ke halaman ini jika pengguna mengeklik pilihan tombol Ubah saat menggeser *item* jadwal ke kiri. Halaman ubah jadwal ini memiliki formulir yang sama dengan formulir yang ada di halaman buat jadwal, yang membedakan hanya *form* tersebut langsung terisi dengan nilai dari jadwal yang hendak diubah sehingga pengguna tinggal mengganti nilai yang ingin diubah saja. Setelah pengguna menekan tombol Simpan, terdapat *pop-up dialog* dengan dua pilihan yang dapat pengguna pilih saat mengubah jadwal yaitu mengubah jadwal tertentu saja atau mengubah seluruh jadwal yang berkaitan dengan jadwal yang diubah. Jika pengguna memilih opsi “Ubah jadwal ini saja” maka jadwal yang akan diubah adalah jadwal tersebut saja, dan apabila pengguna memilih “Ubah seluruh jadwal” maka aplikasi akan

mengubah seluruh jadwal yang berkaitan, yaitu jadwal yang merupakan pengulangan dari jadwal tersebut seperti jadwal pengulangan setiap minggu atau setiap hari. Gambar 3.15 merupakan hasil implementasi desain antarmuka dari halaman ubah jadwal.

Gambar 3.15 Implementasi halaman ubah jadwal

Halaman terakhir yang terdapat pada fitur *schedule* adalah halaman tambah orang. Halaman ini berfungsi untuk menambahkan orang pada suatu jadwal sehingga pengguna dapat mengundang pengguna lain ke jadwal yang dibuat. Pada halaman ini juga pengguna dapat menghapus pengguna lain dari jadwal yang dibuatnya. Halaman ini hanya berisikan masukan untuk mencari nama atau nim pengguna, daftar pengguna baik itu saat proses pencarian atau saat sudah ditambahkan ke jadwal, serta tombol Simpan. Jika pengguna mencari nama atau nim dirinya sendiri, maka hasil pencarian tidak muncul. Setelah pengguna menambahkan atau menghapus pengguna lain, untuk menyimpan daftar pengguna yang akan diundang, pengguna menekan tombol Simpan di samping komponen pencarian. Gambar 3.16 adalah hasil implementasi desain untuk halaman tambah orang.

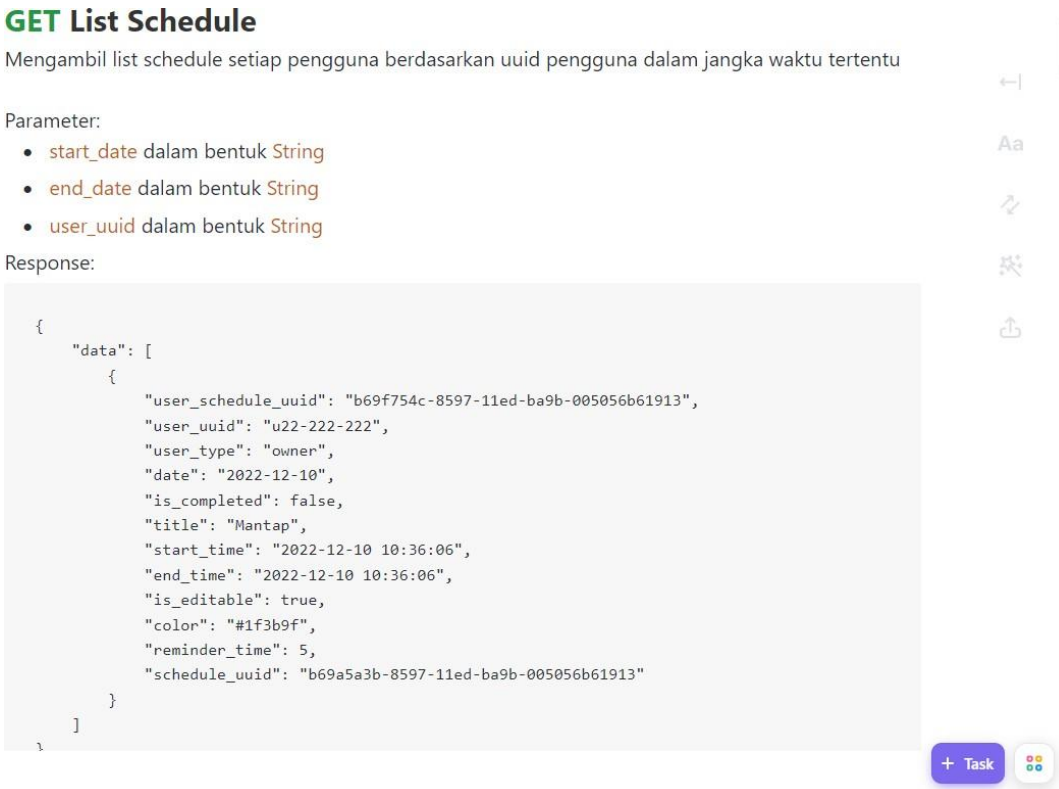
Gambar 3.16 Implementasi halaman tambah orang

3.3.2 Membuat Stuktur Request dan Response Body API

Tujuan dari pembuatan struktur *request* dan *response* API adalah pengembang bisa mengetahui struktur dari *request* maupun *response* API sehingga dapat mempermudah dalam pembuatan API nantinya karena pengembang tinggal menyesuaikan seperti struktur yang sudah dibuat. Selain itu, tujuan pembuatan struktur API adalah pengembang dapat mengetahui *response* yang diberikan API sehingga dapat membuat model atau *entity* yang sesuai struktur API untuk diisi data *dummy*.

Pembuatan API kontrak ini didokumentasikan di aplikasi ClickUp. Di Badan Sistem Informasi (BSI) sendiri untuk dokumentasi menggunakan aplikasi Confluence, tapi karena penulis dan pemagang lain yang mengerjakan proyek I'm UII tidak mendapatkan akun Confluence maka dipilihlah ClickUp untuk monitoring pekerjaan dan dokumentasi. Sebelum membuat struktur *request* dan *response* dari API, penulis membuat daftar API yang akan digunakan untuk fitur *schedule*. Ada tujuh API yang diperlukan untuk fitur *schedule* yaitu mendapatkan daftar jadwal dalam jangka waktu tertentu, mendapatkan detail jadwal, mencari pengguna berdasarkan nama atau nim, menambahkan jadwal, mengubah jadwal, mengubah

status jadwal, dan hapus jadwal. Gambar 3.17 menunjukan kontrak API untuk mendapatkan daftar jadwal dalam jangka waktu tertentu.



Gambar 3.17 Kontrak API mendapatkan daftar jadwal

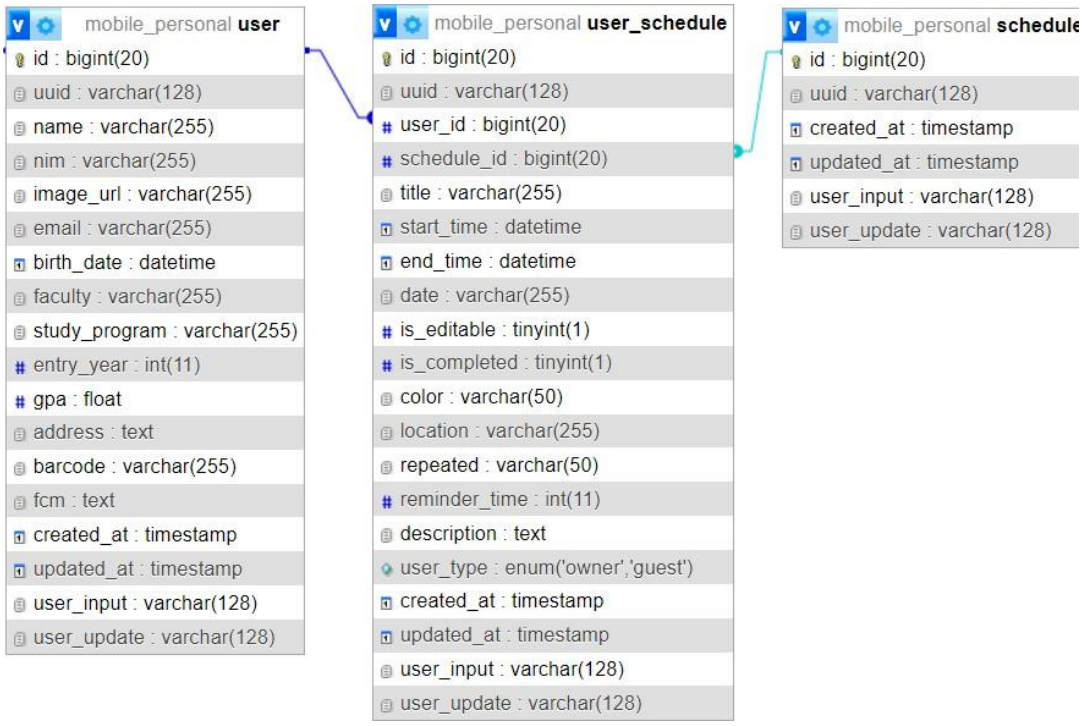
Setelah membuat kontrak API, dilaksanakan reviu kontrak API oleh Elbo Shindi Pangestu sebagai *team lead* yang dilaksanakan secara luring melalui Zoom. Setelah direviu ada API yang dinilai kurang seperti API mendapatkan daftar jadwal yang kurang parameter user_uuid, karena jika tidak ada parameter ini maka pengguna akan mendapatkan jadwal dari pengguna lain. Setelah mendapatkan umpan balik dari *team lead*, penulis merevisi API yang dinilai masih kurang.

3.3.3 Merancang Tabel Basis Data

Setelah selesai mengerjakan kontrak API yang direvisi, selanjutnya penulis mulai mengerjakan *back-end* aplikasi I'm UII, dimulai dari membuat basis datanya. Basis data sendiri adalah kumpulan data yang disimpan secara sistematis dan diorganisir dalam sistem komputer (Oracle, t.thn.). Fungsi dari basis data adalah untuk menyimpan, mengelola, dan mengambil data yang digunakan oleh aplikasi atau sistem. Ada beberapa jenis basis data yang digunakan

dalam dunia teknologi informasi, seperti basis data relasional, dokumen, graf, dll. Dalam mengembangkan aplikasi I'm UII ini menggunakan basis data relasional dengan MySQL sebagai RDBMS (Relational Database Management System).

Dalam membuat basis data, penulis dan pemagang lainnya dibantu oleh Elbo Shindi Pangestu karena dalam tim akselerasi yang memegang proyek I'm UII ini, beliau lah yang paling paham tentang *back-end* sebab pekerjaan utamanya adalah sebagai *back-end developer* pada tim keuangan. Kontrak API menjadi panduan untuk membuat basis data seperti tabel yang diperlukan, kolom data, dan relasi antar tabel. Dengan mengikuti kontrak API diharapkan tidak ada data yang redudansi dan membuat aplikasi lebih mudah untuk digunakan dan dikelola. Untuk fitur *schedule* didapatkan tiga tabel yang diperlukan yaitu tabel schedule, user_schedule, dan user. Penulis membuat basis data di server lokal yang dibantu aplikasi XAMPP. Untuk detail dari kolom untuk masing-masing tabel bisa dilihat pada Gambar 3.18.



Gambar 3.18 Basis data fitur *schedule*

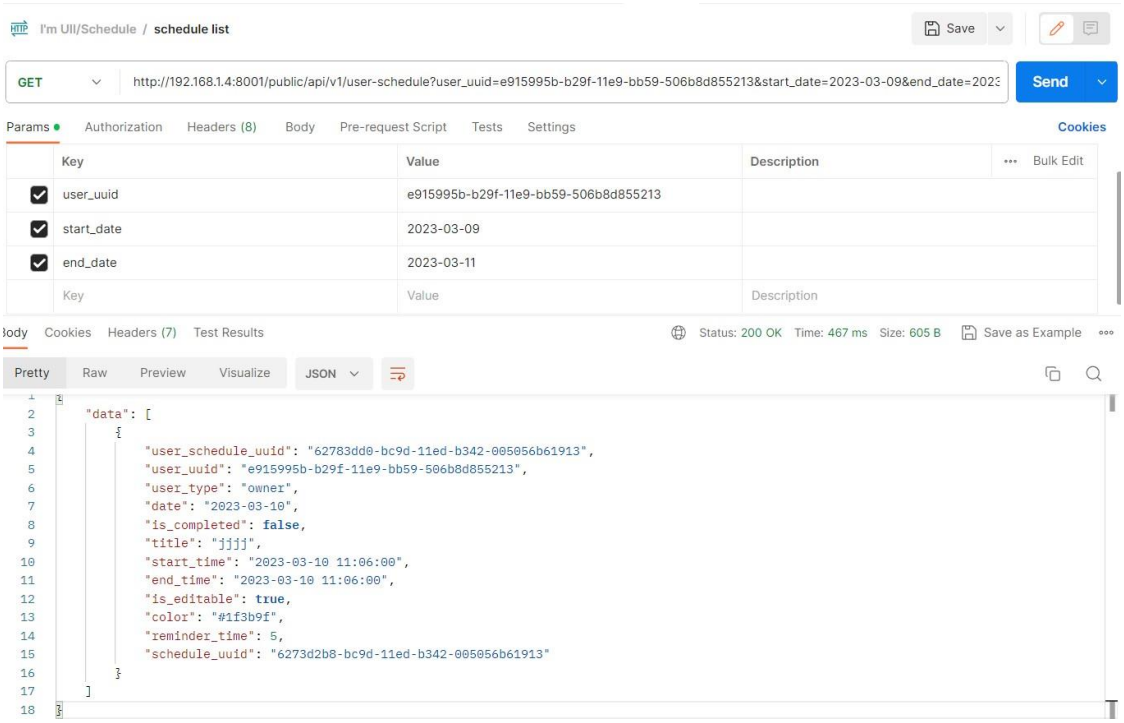
Relasi tabel *user* dengan tabel *schedule* adalah *many to many*. Setiap *user* atau pengguna memiliki satu atau lebih *schedule* atau jadwal dan setiap jadwal dapat dimiliki oleh satu atau lebih pengguna. Karena tabel *user* dan *schedule* memiliki relasi *many to many*, sehingga perlu dibuat tabel baru untuk mewakili relasi tersebut, yaitu tabel user_schedule.

Kemudian setelah membuat basis data di server lokal, penulis melakukan *export* basis data guna diserahkan ke *team lead* untuk disatukan dengan basis data yang dibuat pemagang lain, yang selanjutnya *team lead* akan melakukan *deploy* basis data ke lingkungan pengembangan BSI UII. Pada lingkungan pengembangan ini terdapat semua basis data proyek yang dikerjakan BSI namun khusus untuk tahap pengembangan saja. Untuk bisa mengakses dan mengelola basis data tersebut digunakan aplikasi manajemen basis data yaitu DBeaver, tapi agar bisa mengakses basis data yang ada di lingkungan pengembangan maka diperlukan koneksi internet yang tersambung dengan VPN Nutanix Fortinet khusus untuk karyawan BSI.

3.3.4 Membuat API

Langkah terakhir dalam pengembangan *back-end* adalah membuat API. Pada aplikasi I'm UII dibutuhkan API untuk mengakses dan melakukan operasi ke basis data. API merupakan singkatan dari Application Programming Interface yang merupakan sekumpulan peraturan dan spesifikasi yang menentukan cara bagaimana aplikasi atau sistem lain dapat berinteraksi dengannya. API menyediakan mekanisme untuk aplikasi mengirimkan *request* dan menerima *response*, sehingga dapat mengakses fitur atau data yang disediakan oleh aplikasi atau sistem lain.

Seperti yang disebutkan pada sub bab membuat struktur *request* dan *response* API, ada tujuh API yang diperlukan untuk fitur *schedule* yaitu mendapatkan daftar jadwal dalam jangka waktu tertentu, mendapatkan detail jadwal, mencari pengguna berdasarkan nama atau nim, menambahkan jadwal, mengubah jadwal, mengubah status jadwal, dan menghapus jadwal. Pada awalnya pembuatan API untuk aplikasi I'm UII menggunakan bahasa Golang karena dinilai lebih cepat memproses *request* atau permintaan dibanding menggunakan bahasa atau *framework* lain. Dalam pengembangan API menggunakan Golang, penulis sudah berhasil membuat tiga API yaitu API untuk mendapatkan daftar jadwal, detail jadwal, dan pencarian pengguna. Gambar 3.19 menunjukan salah satu respons API yang dibuat menggunakan Golang yaitu API untuk mendapatkan detail jadwal.



Gambar 3.19 API detail jadwal dengan Golang

Tetapi pengembangan API menggunakan Golang tidak dilanjutkan lagi dikarenakan pengembangan API dengan Golang dinilai sangat sulit sehingga pengembangan API diganti menggunakan *framework* Lumen. Lumen adalah sebuah *framework* PHP dari Laravel yang dirancang khusus untuk membuat aplikasi *microservices* dan API. Penulis merasa nyaman ketika mengembangkan API menggunakan Lumen karena penulis pernah membuat situs web menggunakan *framework* Laravel.

Dalam melakukan permintaan ke API digunakan protokol HTTP. Terdapat beberapa metode pada HTTP yang digunakan pada aplikasi I'm UII, yaitu:

1. GET: metode yang digunakan untuk mengambil data dari *server*, contoh dalam aplikasi I'm UII adalah API mendapatkan daftar jadwal, mendapatkan detail jadwal, dan pencarian pengguna.
2. POST: metode yang digunakan untuk mengirimkan data ke *server*, contoh penerapan pada aplikasi I'm UII adalah API untuk membuat jadwal.
3. PUT: digunakan untuk mengubah data di *server*, contoh pada aplikasi I'm UII adalah API untuk mengubah jadwal dan mengganti status selesai jadwal.
4. DELETE: metode HTTP untuk menghapus data dari *server*, contohnya adalah API untuk menghapus jadwal.

Pembuatan API ini dilakukan terus menerus karena jika ada *bug* atau tidak sesuai dengan standar yang digunakan di BSI maka API harus diperbaiki, contohnya yaitu penulis membuat API yang menampilkan id (*primary key*) pada *respons* API sedangkan ketentuan di BSI tidak boleh menampilkan id kepada pengguna, sehingga penulis harus memperbaiki API yang telah dibuat dengan mengganti id menggunakan uuid.

3.3.5 Mengimplementasikan API ke Aplikasi Flutter

Agar fitur yang terdapat pada API dapat digunakan maka langkah selanjutnya adalah menerapkan API ke aplikasi Flutter. Untuk menerapkan API yang sudah dibuat ke aplikasi Flutter diperlukan HTTP *client*. Pada aplikasi I'm UII penulis menggunakan *package* Flutter yang bernama dio sebagai HTTP *client* untuk berkomunikasi dengan API yang sudah dibuat.

Hal pertama yang penulis lakukan dalam menerapkan API ke aplikasi Flutter adalah menginstall *package* dio ke proyek aplikasi Flutter I'm UII dengan cara menambahkannnya di dalam berkas pubspec.yaml kemudian jalankan perintah "flutter pub get" pada terminal untuk menginstall *package* dio. Langkah selanjutnya penulis membuat permintaan HTTP ke *server* dengan memanfaatkan fungsi-fungsi yang ada di *package* dio. Dalam mengirimkan permintaan HTTP ke *server* dilakukan secara *asynchronous*, sehingga penulis menggunakan fitur bahasa pemrograman Dart yaitu *async/await*. *Async/await* digunakan untuk menjalankan kode yang beroperasi secara *asynchronous* dalam sintkas yang sama dengan kode *synchronous*. Ada beberapa fungsi pada *package* dio yang penulis gunakan dalam mengirim permintaan HTTP pada server yaitu:

1. Get()

   Fungsi get() pada *package* dio digunakan untuk melakukan permintaan HTTP GET ke sebuah API. Method ini menerima satu parameter wajib yaitu path yang berupa URL (Uniform Resource Locator) API yang akan dikonsumsi. Selain parameter path, fungsi dio.get() juga bisa menerima paremeter lain untuk mengatur permintaan HTTP ke server seperti queryParameters yang berisi *query parameter* yang akan dikirimkan dalam *request*, options yang berisi pengaturan *headers*, cancelToken yang berfungsi untuk membatalkan *request* yang sedang berlangsung, dan onReceiveProgress merupakan fungsi callback yang akan dijalankan saat menerima data dari *server*.

   Sebelum melakukan *request* ke server, penulis membuat kelas model untuk mem-*parsing* JSON ke bahasa pemrograman Dart. Untuk membuat kelas model ini, penulis memanfaatkan aplikasi web quicktype. Cara mengkonversi JSON ke bahasa pemrograman

Dart di quicktype sangatlah mudah, penulis hanya perlu meng-*copy* respons API dan mem-*paste* ke situs web quicktype, lalu tinggal menunggu hasil konversi.

Dalam menerapkan API ke aplikasi *mobile* I'm UII, fungsi get() digunakan untuk menangani *request* ke API untuk mendapatkan daftar jadwal dalam jangka waktu tertentu, mendapatkan detail jadwal, dan pencarian pengguna. Dalam menerapkan ketiga API tersebut, penulis hanya menggunakan parameter path, queryParameters, dan options untuk mengatur *header request*.

Dalam, lalu memanggil fungsi get() dari object yang dibuat dengan memanfaatkan fitur *async/await* karena fungsi get() memiliki kembalian atau *return* yang berupa Future, selanjutnya jika berhasil mendapatkan respons dari API maka konversi respons yang berupa JSON tersebut ke bahasa Dart menggunakan kelas model yang telah dibuat lalu me-*return* hasil konversi tersebut. Dalam memanggil fungsi yang menangani *request* ke *server*, penulis memanfaatkan *widget* FutureBuilder. Widget ini memungkinkan untuk menampilkan data yang dihasilkan dari suatu proses yang memakan waktu, seperti pengambilan data dari server atau pemrosesan data yang kompleks. FutureBuilder akan memantau Future yang diproses dan menampilkan hasilnya ketika proses selesai. Contoh penggunaan fungsi dio.get() dapat dilihat pada Gambar 3.20. Fungsi dio.get() memiliki beberapa *parameter* yaitu path yang berisikan URL dari API, options yang berisikan *header* dari *request*, dan queryParameters yang berisikan parameter dari API. Apabila sudah mendapatkan *response* yang berupa objek JSON dengan kode 200, maka akan diubah menjadi objek Dart.

```
Future<ScheduleList> getListSchedule(String userUuid, String startDate,
String endDate) async {
    if (await Utils.checkConnection()) {
      try {
        final response = await dio.get(
            'url',
            options: Options(headers: {'x-app': 'x-app', 'x-menu': 'x-
menu',}),
            queryParameters: {'user_uuid': userUuid, 'start_date':
startDate, 'end_date': endDate,});
        if(response.statusCode == 200) {
          return ScheduleList.fromJson(response.data);
        }
      }
    }
    ...
}
```

Gambar 3.20 Fungsi untuk mendapatkan daftar jadwal

Sedangkan penanganan Future menggunakan FutureBuilder dapat dilihat pada Gambar 3.21. Pada contoh penerapan FutureBuilder menggunakan dua *parameter* yaitu future yang diisi dengan proses penanganan Future, dan builder untuk menampilkan suatu *widget* ketika proses Future selesai.

```
_todaySchedule(DateTime selectedDate) {
    return Container(
      margin: const EdgeInsets.fromLTRB(16, 0, 16, 17),
      child: FutureBuilder(
        future: _schedule,
        builder: (context, AsyncSnapshot<ScheduleList> snapshot) {
          var state = snapshot.connectionState;
          if (state != ConnectionState.done) {
            ...
          }
          else {
            var isThereSchedule = false;
            if(snapshot.hasData) {
              ...
              return ListView.builder(
                primary: false,
                shrinkWrap: true,
                itemCount: snapshot.data?.data.length,
                itemBuilder: (context, index) {
                  var userSchedule = snapshot.data?.data[index];
                  _scheduleList.add(userSchedule!);
if(_dateFormat.format(DateTime.parse(_scheduleList[index].date))==_dateForm
at.format(selectedDate)) {
                    isThereSchedule = true;
                    return ScheduleWidget(
                      userUuid: userSchedule.userUuid,
                      scheduleUuid: userSchedule.scheduleUuid,
                      userScheduleUuid: userSchedule.userScheduleUuid,
                      title: userSchedule.title,
                      subtitle:
'${DateFormat.Hm().format(userSchedule.startTime)}                          -
${DateFormat.Hm().format(userSchedule.endTime)}',
                      isCompleted: userSchedule.isCompleted,
                      isEditable: userSchedule.isEditable,
                      onRefresh: _onRefresh,
                      refreshAfterDelete: _refreshAfterDelete,
                    );
                  }
                  ...
                },
              );
            }
            ...
          }
        },
      ),
    );
  }
```

Gambar 3.21 Penggunaan FutureBuilder untuk menangani proses Future

2. Post()

Fungsi post() pada *package* dio digunakan untuk melakukan permintaan HTTP POST ke sebuah *server*. Fungsi post() ini memiliki *parameter* yang sama dengan fungsi get(), yang membedakan hanyalah dua parameter tambahan yaitu data dan fungsi *callback* onSendProgress. Parameter data digunakan untuk mengirimkan data saat melakukan permintaan HTTP POST, bisa dalam bentuk berupa object, map, atau String. Sedangkan parameter fungsi *callback* onSendProgress berfungsi untuk melacak pengiriman data.

Sama seperti fungsi get(), jika *request* menggunakan fungsi post() berhasil maka akan menerima *response* oleh karena itu perlu membuat kelas model yang akan mengkonversi respons yang berupa JSON ke bahasa Dart menggunakan aplikasi web quicktype. Pembuatan fungsi yang meng-*handle* HTTP POST menggunakan *method* atau fungsi post() mirip dengan get(), yang membedakan hanyalah penggunaan parameter data dan tidak menggunakan parameter queryParameters. Gambar 3.22 adalah penerapan fungsi post() untuk membuat jadwal. Pada contoh penggunaan fungsi dio.post() di sini menggunakan *parameter* path, options, dan data yang berisikan informasi jadwal yang akan dibuat seperti judul, waktu mulai dan berakhir, lokasi, dan lain-lain.

```
Future<BaseResponseInfo> insertSchedule(String color, String title, String
startTime, String endTime, String repeated, int reminderTime, String
location, String description, String userInput, String userUpdate,
List<String> guests) async {
    if (await Utils.checkConnection()) {
      try {
        final response = await dio.post(
            'url',
            options: Options(
                headers: {'x-app': 'x-app', 'x-menu': 'x-menu',}
            ),
            data: {
              "title": title, "start_time": startTime, "end_time": endTime,
              "is_completed": false, "is_editable": true, "color": color,
              "location": location, "repeated": repeated,
              "reminder_time": reminderTime, "description": description,
              "guests_uuid": guests,
            },
        );
        if(response.statusCode == 201) {
          return BaseResponseInfo.fromJson(response.data);
        }
      }
      ...
    }
}
```

Gambar 3.22 Fungsi untuk membuat jadwal

3. Put()

Fungsi put() yang ada di *package* dio berfungsi untuk mengirimkan *request* HTTP PUT yang digunakan untuk mengubah data yang ada pada *server*. Fungsi put() ini digunakan untuk implementasi API untuk mengubah jadwal dan mengubah status selesai jadwal. Fungsi put() memiliki parameter yang sama dengan fungsi post().

Sama seperti fungsi get() dan post(), *request* HTTP yang menggunakan fungsi put() akan mendapatkan respons dari *server* baik respons yang menunjukan bahwa *request* yang dilakukan berhasil atau tidak berhasil, sehingga perlu dibuat kelas model yang akan mengkonversi JSON ke bahasa Dart menggunakan aplikasi *website* quicktype. Pembuatan fungsi yang meng-*handle* HTTP PUT menggunakan *method* atau fungsi put() sama dengan post(), jadi penulis tinggal menyalin fungsi tersebut dan mengganti URL parameter path. Penerapan fungsi put() untuk mengubah informasi jadwal dapat dilihat pada Gambar 3.23. Pada contoh penggunaan fungsi dio.put() di sini menggunakan *parameter* path, options, dan data yang berisikan informasi jadwal yang akan dibuat seperti judul, waktu mulai dan berakhir, lokasi, uuid mahasiswa yang dihapus dari jadwal, dan lain-lain.

```
Future<BaseResponseInfo> updateSchedule(String userScheduleUuid, String
color, String title, String startTime, String endTime, String repeated, int
reminderTime, String location, String description, String userInput, String
userUpdate,List<String> guests, List<String> deletedGuest, String updateType
) async {
    if (await Utils.checkConnection()) {
      try {
        final response = await dio.put(
          'url',
          options: Options(headers: {x-app': 'x-app', 'x-menu': 'x-menu',}),
          data: {
            "update_type": updateType, title": title,
            "start_time": startTime, end_time": endTime,
            "is_completed": false, "is_editable": true,
            "color": color, "location": location,
            "repeated": repeated, reminder_time": reminderTime,
            "description": description, "user_input": userInput,
            "user_update": userUpdate, "guests_uuid": guests,
            "deleted_uuid": deletedGuest,
          },
        );
        if(response.statusCode == 200) {
          return BaseResponseInfo.fromJson(response.data);
        }
      }
    }
}
```

Gambar 3.23 Fungsi untuk mengubah jadwal

4. Delete()

Fungsi delete() pada *package* dio Flutter berfungsi untuk mengirimkan *request* HTTP DELETE. *Request* ini berfungsi untuk menghapus data yang ada di *server*. Fungsi delete() digunakan untuk menerapkan API menghapus jadwal. Fungsi delete() ini memiliki *parameter* yang sama dengan fungsi get().

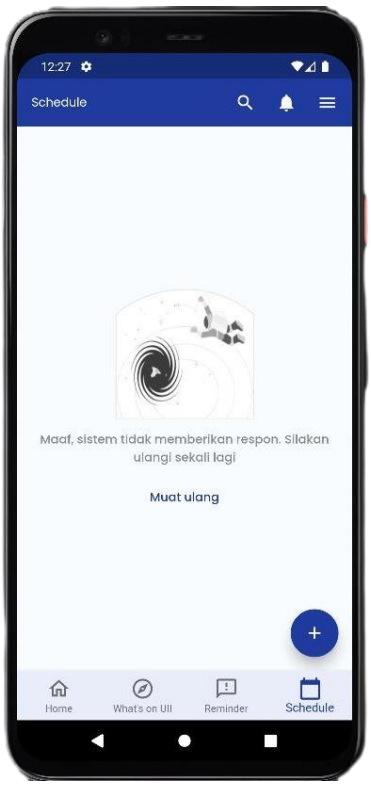
Setelah melakukan *request* menggunakan fungsi delete(), akan menerima respons jika *request* yang dilakukan berhasil atau tidak. Sehingga untuk menerima respons tersebut diperlukan kelas model yang akan mengkonversi JSON dari respons ke bahasa Dart. Dalam membuat kelas model ini, penulis menggunakan aplikasi web quicktype. Pembuatan fungsi yang meng-*handle* HTTP DELETE menggunakan *method* atau fungsi delete() sama dengan get(), jadi penulis tinggal *copy-paste* fungsi tersebut dan mengganti nilai parameter path dengan URL API menghapus jadwal. Penerapan fungsi delete() untuk menghapus data jadwal dapat dilihat pada Gambar 3.24 di bawah. Pada contoh penggunaan fungsi dio.delete(), penulis menggunakan *parameter* path, options, dan queryParameters yang berisikan user_schedule_uuid, user_uuid, user_type, dan delete_type.

```
Future<BaseResponseInfo> deleteSchedule(String userUuid,
String scheduleUuid, String userScheduleUuid, String deleteType,
String userType) async {
    if (await Utils.checkConnection()) {
      try {
        final response = await dio.delete(
            'url',
            options: Options(
                headers: {
                  'x-app': 'x-app',
                  'x-menu': 'x-menu',
                }
            ),
            queryParameters: {
              'user_schedule_uuid': userScheduleUuid,
              'user_uuid': userUuid,
              'user_type': userType,
              'delete_type': deleteType,
            }
        );
        if(response.statusCode == 200) {
          return BaseResponseInfo.fromJson(response.data);
        }
      }
      ...
    }
}
```

Gambar 3.24 Fungsi untuk menghapus jadwal

Saat mengkonsumsi API ada kalanya *request* yang dikirimkan gagal karena banyak hal seperti tidak ada koneksi internet, *time out*, dan kesalahan di sisi *server*. Oleh karena itu, penting untuk menangani kesalahan atau *error* tersebut untuk menjamin kestabilan dan keandalan dari aplikasi yang dibuat. Jika kesalahan tidak ditangani maka aplikasi akan tertutup secara paksa. Dalam menangani *error*, penulis menanganinya di sisi klien yaitu dengan menampilkan halaman yang berisi informasi tentang *error* yang terjadi. Saat menerapkan API, penulis menaruh kode yang mengkonsumsi API di dalam *statement* try sehingga jika terdapat kesalahan maka *statement* catch yang akan dieksekusi. Di dalam *statement* catch ini terdapat tiga penanganan *error* yaitu, *error* yang masih mendapat respons contohnya saat melakukan konversi dari JSON ke Dart terdapat nilai yang tipe datanya berbeda, kesalahan yang tidak mendapatkan respons seperti koneksi *time out*, dan *error* saat terjadi kesalahan pada *server* misalnya *server* sedang tidak berfungsi. Gambar 3.25 dan Gambar 3.26 yang merupakan penerapan penangan eror untuk eror tidak ada koneksi dan kesalahan *server*. Pada kedua halaman tersebut terdapat tulisan yang menandakan tidak ada koneksi internet atau terjadi kesalahan pada *server* dan tombol untuk memuat ulang halaman.

Gambar 3.25 Halaman eror tidak ada koneksi internet



Gambar 3.26 Halaman eror saat server bermasalah

3.3.6 Mengembangkan Sistem Notifikasi

Pengiriman pesan atau notifikasi ke pengguna aplikasi *mobile* menjadi semakin penting dalam pengembangan aplikasi. Hal ini disebabkan karena pengguna aplikasi *mobile* cenderung menginginkan informasi terbaru tanpa harus membuka aplikasi tersebut. Salah satu layanan yang dapat dimanfaatkan untuk mengirimkan notifikasi adalah Firebase Cloud Messaging atau biasa disebut FCM yang dikembangkan oleh Google.

Dalam pengembangan aplikasi *mobile*, penggunaan FCM sangat penting untuk mengirimkan *push notification* kepada pengguna. FCM dapat membantu pengembang aplikasi *mobile* untuk mengirimkan notifikasi yang dapat memicu tindakan dari pengguna, meningkatkan interaksi pengguna, dan memberikan informasi yang relevan kepada pengguna. Dengan menggunakan Firebase Cloud Messaging, pengembang dapat mengoptimalkan aplikasi *mobile* Flutter untuk meningkatkan pengalaman pengguna dan memberikan nilai tambah pada proses bisnis aplikasi.

Pada Aplikasi I'm UII, pengiriman notifikasi dibutuhkan untuk fitur *schedule* dan *reminder* sebagai pengingat dan notifikasi aplikasi sebagai pemberitahuan semisal *server* sedang dalam perbaikan. Untuk fitur *schedule*, notifikasi akan dikirimkan ke pengguna sesuai

dengan waktu pada jadwal. Dalam aplikasi I'm UII pengiriman notifikasi ke gawai pengguna menggunakan layanan FCM dan fitur *task scheduling* pada Lumen.

Pada sistem notifikasi ini, FCM digunakan untuk mengirimkan *push notification* ke gawai pengguna. Alasan menggunakan FCM karena layanan ini bersifat gratis dan 95% dari notifikasi yang dikirimkan akan diterima oleh pengguna dengan rata-rata kecepatan 250ms (Google, 2023). Sedangkan penggunaan Lumen di sistem notifikasi ini adalah untuk menjalankan kode yang berfungsi untuk mendapatkan data jadwal setiap menit, lalu jika terdapat jadwal pada menit tersebut maka akan dikirimkan notifikasi ke gawai pengguna berdasarkan token FCM dari gawai pengguna tersebut.

Proses pengiriman notifikasi diawali saat pengguna pertama kali memasang aplikasi di gawai mereka. Saat pengguna berhasil *login* menggunakan akun Gateway, di latar belakang aplikasi akan mengecek penyimpanan local atau *shared preference* untuk mendapatkan token FCM dan nilai untuk mengecek apakah gawai pengguna sudah berlangganan topik notifikasi aplikasi. Jika aplikasi pertama kali dipasang, maka aplikasi akan membuat token FCM untuk gawai pengguna dan berlangganan ke topik notifikasi aplikasi supaya dapat mendapatkan notifikasi dari sistem seperti pemberitahuan pemeliharaan *server*.

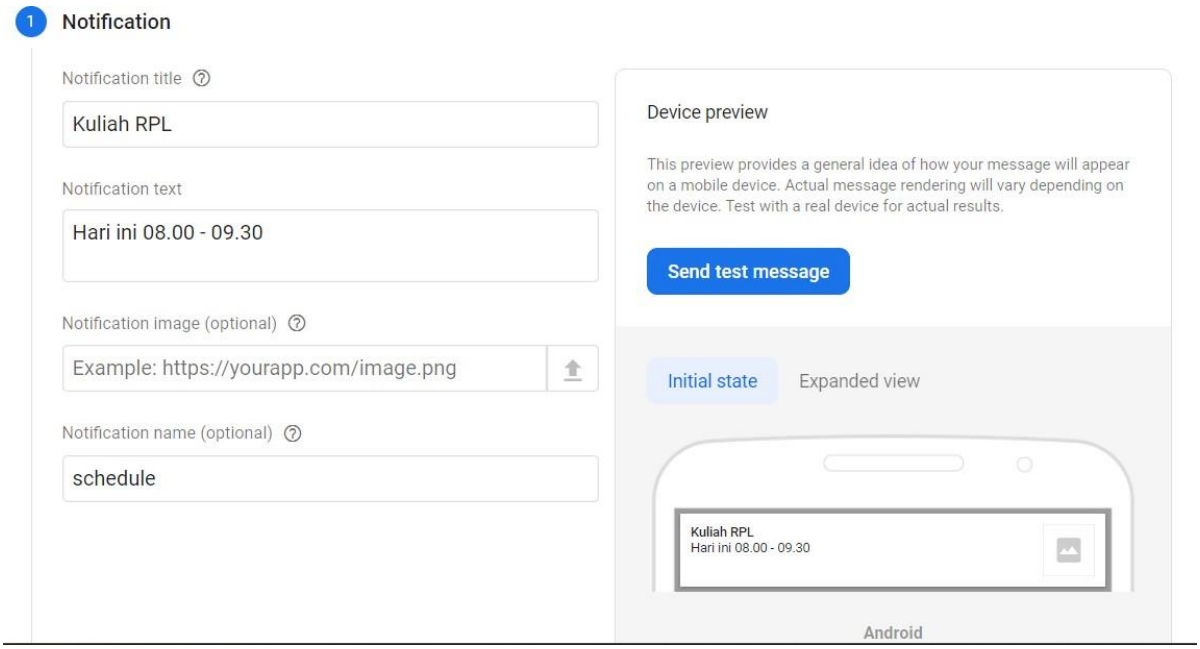
Untuk mengirimkan notifikasi jadwal ke pengguna, diperlukan data pengguna di basis data aplikasi I'm UII seperti token FCM untuk setiap *device* pengguna. Sehingga setelah berhasil mendapatkan token FCM gawai pengguna, maka aplikasi I'm UII akan melakukan operasi *insert* data pengguna ke basis data I'm UII.

Seperti yang sudah disebutkan sebelumnya, *server* I'm UII akan mencoba mendapatkan data jadwal setiap menitnya, apabila terdapat data jadwal pada menit tersebut, maka *server* akan mengirimkan notifikasi ke gawai pengguna yang bersangkutan. Untuk mengirimkan notifikasi ke gawai pengguna dari Lumen, penulis menggunakan *package* laravel-firebase yang dibuat oleh kreait. Sedangkan untuk mengirim notifikasi aplikasi yang merupakan pemberitahuan seperti terdapat pembaruan aplikasi, penulis hanya menggunakan FCM dengan mengisi *form* pada Firebase Console untuk membuat notifikasi. Kode untuk mengirim notifikasi *schedule* dan *form* FCM di Firebase Console dapat dilihat pada Gambar 3.27 dan Gambar 3.28. Pada Gambar 3.27, fungsi dari baris kode `$response = $client->request()` adalah untuk mendapatkan data jadwal di menit tertentu. Proses selanjutnya adalah penambahan *payload* pada notifikasi yang akan dikirimkan. Payload ini berfungsi untuk membuka halaman detail jadwal ketika notifikasi ditekan. Langkah selanjutnya adalah pembuatan notifikasi dengan kode `$message = CloudMessage::withTarget('token',`

`$notification->fcm)...`, pada kode tersebut penulis menambahkan token FCM pengguna yang mempunyai jadwal terkait, judul dan jam jam notifikasi, dan *payload*. Dan langkah terakhir adalah mengirimkan notifikasi ke gawai pengguna dengan kode `$messaging->send($message)`.

```
protected function schedule(Schedule $schedule) {
   $schedule->call(function () {

       // get schedule data
       $client = new Client(['base_uri' => 'url, 'timeout'  => 5,]);

       $response = $client->request('GET', 'path', [
          'headers' => [
             'x-app' => 'x-app',
             'x-menu' => 'x-menu',
          ]
       ]);

       $body = $response->getBody();
       $body_array = json_decode($body);

       // send notification
       $messaging = app('firebase.messaging');

       foreach ($body_array->data as $notification) {

          // add payload data
          $payloadData = [
             'user_schedule_reminder_uuid'=>$notification-
>user_schedule_uuid,
              'schedule_reminder_uuid'=>$notification-
>schedule_or_reminder_uuid,
              'user_uuid' => $notification->user_uuid,
              'user_type' => $notification->user_type,
              'is_editable' => $notification->is_editable,
              'is_completed' => $notification->is_completed,
              'color' => $notification->color,
              'notification_from' => $notification->notification_from,
          ];

          $unixTime = strtotime($notification->start_time);
          $scheduleTime = date('H:i', $unixTime);
          $body = 'Hari ini '.$scheduleTime;

          $message = CloudMessage::withTarget('token', $notification->fcm)
          ->withNotification(Notification::create($notification->title,
$body))->withData($payloadData);

          $messaging->send($message);
       }
    })->everyMinute();
}
```

Gambar 3.27 Kode mengirimkan notifikasi di Lumen



Gambar 3.28 Form FCM pada Firebase Console

3.4 Pengujian Aplikasi

Pengujian aplikasi merupakan tahapan dalam pengembangan aplikasi yang dilakukan untuk memastikan bahwa aplikasi berjalan sesuai apa yang diharapkan dan memastikan tidak terdapat cacat (Rambe & Prihantoro, 2022). Pengujian aplikasi dapat dibagi menjadi dua yaitu *white box testing* dan *black box testing*. Pengujian yang dilakukan untuk fitur *schedule* aplikasi I'm UII adalah *black box testing*. *Black box testing* adalah pengujian yang dilakukan untuk mengamati hasil *output* dari suatu *input* pada perangkat lunak tanpa mengetahui struktur kode dari aplikasi (Setiawan, 2021).

Pengujian fitur *schedule* dilakukan oleh tim pengembang, yaitu Alif Maulana Rizqi, Fahrudin Nasikh Az Zuhdu, dan Rio Risqi Akbar Herlambang. Pengujian dilakukan dengan manual atau menjalankan *test case* yang telah dibuat pada aplikasi secara langsung. *Test case* untuk pengujian fitur *schedule* dapat dilihat pada Tabel 3.4.

Tabel 3.4 Hasil pengujian *black box*

No.	**Use case**	**Skenario**	**Hasil yang diharapkan**	**Hasil pengujian**
1.	IUC-01 Menampilkan daftar	Menampilkan daftar jadwal sesuai tanggal	Pengguna menekan menu *schedule*, lalu mendapatkan daftar	Berhasil

No.	Use case	Skenario	Hasil yang diharapkan	Hasil pengujian
	jadwal sesuai tanggal		jadwal sesuai tanggal yang dipilih, jika tidak terdapat jadwal pada tanggal tersebut maka ditampilkan gambar yang menandakan tidak terdapat jadwal	
2.	IUC-02 Menampilkan detail jadwal	Membuka halaman detail jadwal	Pengguna dapat melihat detail dari jadwal dengan mengelik *item* daftar jadwal pada halaman *schedule*	Berhasil
3.	IUC-03 Mengubah jadwal	Mengubah satu jadwal	Pengguna menggeser *item* daftar jadwal ke kiri, lalu memilih opsi **Ubah**, akan muncul detail berita dan pengguna bisa mengubahnya, tekan tombol **Simpan**, pilih opsi **Ubah jadwal ini saja**, maka detail jadwal tersebut akan berubah	Berhasil
		Mengubah semua jadwal yang berkaitan	Pengguna menggeser *item* daftar jadwal ke kiri, lalu memilih opsi **Ubah**, akan	Berhasil

No.	Use case	Skenario	Hasil yang diharapkan	Hasil pengujian
			muncul detail berita dan pengguna bisa mengubahnya, tekan tombol **Simpan**, pilih opsi **Ubah seluruh jadwal**, maka detail jadwal tersebut dan jadwal yang berkaitan akan berubah	
4.	IUC-04 Menghapus jadwal	Menghapus satu jadwal	Pengguna menggeser *item* daftar jadwal ke kiri, lalu memilih opsi **Hapus**, pilih opsi **Hapus jadwal ini saja**, maka jadwal tersebut akan dihapus	Berhasil
		Menghapus semua jadwal yang berkaitan	Pengguna menggeser *item* daftar jadwal ke kiri, lalu memilih opsi **Hapus**, pilih opsi **Hapus seluruh jadwal**, maka jadwal tersebut dan jadwal yang berkaitan akan dihapus	Berhasil
5.	IUC-05 Mengirimkan notifikasi jadwal	Mengirimkan notifikasi jadwal	Pengguna yang membuat dan yang diundang ke jadwal mendapat notifikasi	Berhasil

No.	Use case	Skenario	Hasil yang diharapkan	Hasil pengujian
			tentang jadwal yang hendak dimulai	

Berdasarkan Tabel 3.4 dapat disimpulkan bahwa semua *use case* yang ada di fitur *schedule* aplikasi I'm UII telah berhasil melewati pengujian dan dapat berjalan sesuai yang diharapkan sehingga pengguna mendapatkan pengalaman yang baik saat menggunakan fitur *schedule*.

3.5 Perbandingan dengan Studi dan Aplikasi Terdahulu

Semua fungsionalitas yang harus terdapat pada fitur *schedule* berhasil dikembangkan dan dapat berjalan dengan baik setelah dilakukan pengujian menggunakan pengujian *black box*. Untuk bisa mengetahui kekurangan dan kelebihan dari fitur *schedule* aplikasi I'm UII, maka dilakukan perbandingan dengan studi dan aplikasi terdahulu.

Merujuk pada studi dan aplikasi yang dibahas pada bagian tinjauan pustaka di bab dua, makalah yang menjadi pembanding pertama membahas tentang desain dan implementasi sistem penjadwalan agenda berbasis Android (Rahmah, 2017). Sistem yang dikembangkan pada makalah ini memiliki kemiripan dengan fitur *schedule* aplikasi I'm UII, yaitu terdapat pengingat untuk melakukan jadwal. Sedangkan perbedaannya adalah bentuk pengingat jadwalnya, pada sistem penjadwalan makalah yang diacu, pengingat berupa *alert dialog* yang dibarengi dengan bunyi alarm, sedangkan pengingat pada fitur *schedule* I'm UII berupa notifikasi.

Makalah acuan yang kedua membahas tentang pembuatan aplikasi *reminder* jadwal perkuliahan yang berbasis Android untuk Jurusan Elektronika Universitas Negeri Padang (Edde & Budayawan, 2021). Aplikasi *reminder* yang dikembangkan pada makalah yang menjadi acuan memiliki kemiripan dengan fitur *schedule*. Kemiripannya yaitu terletak pada pengiriman notifikasi ke mahasiswa terkait jadwal. Sedangkan perbedaannya adalah pada fitur *schedule* I'm UII, mahasiswa dapat membuat jadwal pribadinya sendiri dan dapat mengundang orang lain ke jadwal tersebut. Sedangkan aplikasi *reminder* pada makalah acuan, yang memiliki wewenang untuk membuat jadwal kuliah adalah dosen, sedangkan mahasiswa hanya dapat melakukan mengatur waktu notifikasi dan mendapatkan notifikasi.

Terdapat aplikasi *mobile* yang serupa dengan fitur *schedule* pada aplikasi I'm UII yaitu Microsoft To Do. Yang membedakan antara Microsoft To Do dengan fitur *schedule* I'm UII adalah proses bisnis dalam mengundang pengguna lain ke suatu jadwal atau kegiatan. Pada Microsoft To Do, pengguna harus membuat *list* terlebih dahulu, lalu membuat kegiatan yang perlu dilakukan serta mengundang pengguna lain ke *list* tersebut, dan menambahkan pengguna lain ke kegiatan yang dibuat. Sedangkan untuk mengundang orang ke jadwal atau kegiatan pada fitur schedule I'm UII hanya perlu membuat jadwal lalu menambahkan pengguna lain ke jadwal tersebut.

Selain Microsoft To Do, ada aplikasi yang mirip dengan fitur *schedule* pada aplikasi I'm UII yaitu Google Calendar. Yang membedakan antara Google Calendar dengan fitur *schedule* I'm UII adalah pada Google Calendar tidak terdapat jadwal kuliah pengguna secara bawaan, pengguna harus menambahkan secara manual jadwal kuliah. Sedangkan pada fitur *schedule*, saat pengguna *log in* ke aplikasi I'm UII, maka aplikasi akan secara otomatis mendapatkan jadwal kuliah pengguna. Selain jadwal kuliah, integrasi jadwal mahasiswa juga dapat dikembangkan lebih lanjut, seperti integrasi dengan tanggal tagihan, seminar atau webinar yang dilaksanakan di lingkungan UII, bahkan jadwal pesantrenisasi.

BAB IV
REFLEKSI PELAKSANAAN MAGANG

4.1 Relevansi Akademik

Selama magang penulis berfokus pada pengembangan fitur *schedule* aplikasi I'm UII. Penulis mengembangkan aplikasi *mobile* dan *back-end* untuk fitur *schedule*. Pengembangan fitur *schedule* pada aplikasi *mobile* menggunakan *framework* Flutter. Sedangkan pengembangan *back-end* fitur *schedule* menggunakan *framework* Lumen. Pengembangan aplikasi I'm UII pada tim percepatan menggunakan metode pengembangan *waterfall*.

Terdapat beberapa perbedaan yang penulis temukan antara teori yang dipelajari di perkuliahan dengan praktik lapangan di BSI UII. Perbedaannya yaitu, metode pengembangan aplikasi, pengembangan aplikasi *mobile*, dan pengembangan API. Pada subbab ini akan membahas tentang perbedaan antara teori yang dipelajari selama kuliah dengan praktik lapangan yang diterapkan BSI UII.

4.1.1 Metode Pengembangan

Metode pengembangan aplikasi yang digunakan oleh BSI adalah metode pengembangan *agile* dengan *framework scrum*. Pada awalnya dalam mengembangkan aplikasi, tim percepatan menggunakan metode pengembangan *scrum,* tetapi karena di tim percepatan terdapat karyawan BSI yang sudah masuk ke dalam tim inti, sehingga akan memberatkan mereka jika menerapkan metode pengembangan *scrum* karena karyawan tersebut harus memprioritaskan pengerjaan tugas yang diserahkan pada mereka di tim inti BSI sehingga tidak bisa menerapkan *event scrum* seperti *daily meeting* atau *sprint retrospective* pada tim percepatan, sehingga metode pengembangan diganti menjadi *waterfall*. Langkah ini diambil saat berdiskusi dengan *product owner*.

Di BSI UII untuk monitoring dan dokumentasi tugas *sprint* menggunakan Jira, tetapi karena pemagang tidak mendapatkan akun Jira sehingga monitoring *sprint* untuk tim percepatan menggunakan platform ClickUp. Selain untuk dokumentasi tugas *sprint*, ClickUp juga digunakan untuk dokumentasi dari API kontrak yang berfungsi sebagai pedoman dalam pembuatan API. Dipilihnya ClickUp sebagai alat untuk monitoring *sprint* dikarenakan fitur yang dimilikinya mirip dengan Jira, yaitu bisa membuat *card* lalu *assign* anggota tim yang akan mengerjakannya, menambahkan *story point* pada *card*, dan bisa menggeser *card* sesuai dengan status pengerjaannya seperti *in progress* atau *accepted*.

Terdapat perbedaan antara teori dan praktik saat magang terkait metode pengembangan aplikasi. Pada teori yang dipelajari di kuliah, saat menggunakan *waterfall* sebagai metode pengembangan dibutuhkan pembuatan dokumen sebagai keperluan untuk tahap yang lain, tetapi yang penulis alami tidak demikian. Pada saat magang, penulis mendapati tidak adanya dokumentasi, semisal tidak adanya dokumen SRS (Software Requirement Spesification). Metode pengembangan *waterfall* pada tim percepatan terlihat dari pengerjaan tugas sesuai dengan tahapan *waterfall*. Pada pengembangan fitur *schedule* aplikasi I'm UII tahapan yang terlaksana hanya tahap analisis kebutuhan, desain, pengembangan, dan pengujian. Untuk tahap pemeliharaan belum terlaksana karena tugas penulis hanya sampai tahap pengujian saja.

4.1.2 Pengembangan Aplikasi *Mobile*

Selama melaksanakan magang di BSI UII, penulis mendapati perbedaan dengan yang dipelajari di perkuliahan. Di perkuliahan terdapat mata kuliah yang mempelajari pengembangan aplikasi *mobile* untuk platform Android menggunakan bahasa pemrograman Kotlin. Sedangkan selama pelaksanaan magang dengan mengembangkan fitur *schedule* aplikasi I'm UII, pengembangan aplikasi *mobile* menggunakan *framework* Flutter. Dipilihnya Flutter untuk pengembangan aplikasi I'm UII karena dengan menggunakan Flutter bisa mempercepat waktu pengembangan untuk aplikasi *multi-platform* seperti Android dan iOS hanya dengan satu basis kode saja (Flutter, n.d.).

Perbedaan yang besar dalam mengembangkan aplikasi *mobile* menggunakan Kotlin dengan Flutter adalah fitur *hot reload* pada Flutter. Fitur ini berfungsi untuk melihat perubahan pada kode aplikasi tanpa perlu melakukan proses *rebuild* atau *restart* aplikasi secara manual, sehingga dapat mempercepat pengembangan aplikasi. Sedangkan jika ingin melihat perubahan pada aplikasi yang dibangun menggunakan Kotlin, pengembang harus menjalankan ulang aplikasi yang dapat memakan banyak waktu karena harus mengulang proses *build* aplikasi walaupun perubahan yang dilakukan hanya sedikit.

4.1.3 Pengembangan API

Di perkuliahan juga terdapat mata kuliah Pengembangan Aplikasi Berbasis Web yang mempelajari pengembangan web. Pada mata kuliah ini penulis mempelajari pengembangan *website* secara keseluruhan atau *fullstack*. Tetapi pengembangan aplikasi web pada mata kuliah ini belum menggunakan API untuk berkomunikasi dan berbagi data sehingga kode untuk *front-end* dan *back-end* dijadikan satu projek. Saat magang penulis berkesempatan untuk

mengembangkan API sebagai *back-end* aplikasi menggunakan *framework* Lumen. Untuk bisa mengambil atau mengubah data pada basis data, dilakukan pemanggilan API pada *front-end* atau aplikasi *mobile*. Dengan mengembangkan API juga dapat memudahkan integrasi dengan aplikasi lain, seperti pada fitur *schedule* aplikasi I'm UII terdapat integrasi dengan jadwal kuliah mahasiswa, penulis hanya menggunakan API untuk mendapatkan jadwal mahasiswa yang sebelumnya sudah dibuat oleh tim BSI.

Selama di perkuliahan, penulis tidak pernah mempelajari pengembangan sistem notifikasi untuk sebuah aplikasi. Pada kegiatan magang ini penulis berhasil mengembangkan sistem notifikasi untuk aplikasi I'm UII yang meliputi notifikasi *schedule* atau jadwal, notifikasi *reminder,* dan notifikasi aplikasi yang memberi informasi seperti pemberitahuan pemeliharaan *server*.

Pada Tabel 4.1 menjelaskan tentang perbedaan antara teori yang didapatkan saat mengikuti kuliah dengan praktik lapangan. Pada tabel tersebut terdapat tiga kolom, yaitu kolom topik sebagai bahasan perbedaan antara perkuliahan dan praktik lapangan, kuliah yang membahas tentang yang dipelajari penulis sela akuliah, dan kolom praktik lapangan yang membahas tentang kenyataan yang diterapkan di saat magang oleh perusahaan.

Tabel 4.1 Relevansi akademik

Topik	**Kuliah**	**Praktik Lapangan**
Metode pengembangan	Membuat dokumen sebagai kebutuhan untuk tahap selanjutnya.	Kurangnya dokumentasi, semisal tidak adanya dokumen SRS (Software Requirement Spesification) dan SDD (Software Design Document), sehingga jika ada yang kurang dimengerti dengan *flow* aplikasi maka harus menanyakan langsung ke *team lead*.
Pengembangan aplikasi *mobile*	Saat mengikuti kuliah, bahasa pemrograman yang digunakan untuk mengembangkan aplikasi *mobile* adalah Kotlin.	Pada tim percepatan yang mengembangkan aplikasi I'm UII, pengembangan aplikasi I'm UII menggunakan Flutter.
Pengembangan API	Terdapat matakuliah tentang pengembangan web, tetapi	Membuat API untuk untuk logika aplikasi dan komunikasi dengan

Topik	Kuliah	Praktik Lapangan
	untuk logika dan komunikasi dengan basis data tidak menggunakan API.	basis data menggunakan *framework* Lumen.
Membuat sistem notifikasi	Penulis tidak pernah mendapatkan ilmu atau pun mengembangkan sistem notifikasi pada sebuah aplikasi.	Membuat sistem notifikasi untuk fitur aplikasi I'm UII seperti fitur *reminder, schedule*, dan notifikasi aplikasi. Pengembangan notifikasi menggunakan fitur *task scheduling* pada Lumen dan FCM.

4.2 Pembelajaran Magang

Setelah melaksanakan magang, penulis mengetahui bahwa metode pengembangan aplikasi bisa berubah di tengah pengembangan dikarenakan berbagai hal. Perubahan metode pengembangan dalam tim percepatan BSI dikarenakan untuk mengurangi beban karyawan BSI yang masuk ke dalam tim percepatan karena memiliki kesibukan di tim inti BSI. Menurut saya metode pengembangan *waterfall* yang diterapkan di tim percepatan kurang memadai, karena kurangnya dokumentasi sehingga jika ada yang tidak dimengerti maka harus bertanya langsung ke *team lead*.

Penulis mendapatkan kesempatan untuk mencoba teknologi yang belum penulis gunakan sebelumnya. Penulis baru pertama kali menggunakan *framework* Flutter untuk pengembangan aplikasi *mobile* dan mengembangkan API untuk *back-end* aplikasi menggunakan Lumen. Sebelum mengikuti magang, dalam pengembangan aplikasi *mobile* penulis hanya bisa menggunakan Kotlin, tetapi karena pengembangan aplikasi I'm UII menggunakan Flutter maka penulis dituntut untuk belajar hal baru yaitu pengembangan aplikasi menggunakan Flutter. Pada saat magang juga penulis mendapatkan pengetahuan baru yaitu cara mengkonsumsi API pada aplikasi *mobile*. Dengan menggunakan API di fitur *schedule* pada aplikasi *mobile* I'm UII dapat memudahkan penulis untuk mengembangkan aplikasi *mobile* karena lebih berfokus pada pembuatan tampilan aplikasi ketimbang logika dan komunikasi data dengan basis data. Dengan menggunakan Flutter penulis dapat dengan cepat membuat

tampilan aplikasi karena Flutter menyediakan *widget* atau komponen yang mudah dikustomisasi.

Setelah mengikuti kegiatan magang ini, penulis mendapatkan pengetahuan baru tentang pengembangan API sebagai *back-end* dan cara mengkonsumsinya di aplikasi *mobile*. Saat mengembangkan API, terdapat standar yang digunakan BSI seperti tidak boleh menampilkan id dari suatu tabel pada respons API. Pemanggilan API pada aplikasi *mobile* Flutter menggunakan bantuan *package* dio. Dengan menerapkan pemisahan antara kode *front-end* dan *back-end* dapat mempermudah penulis dalam mengembangkan aplikasi karena aplikasi menjadi lebih terfokus. Semisal proyek aplikasi untuk *front-end* hanya berfokus pada tampilan dan interaksi pengguna saja, sedangkan untuk proyek aplikasi *back-end* berfokus pada logika bisnis dan pengolahan data.

Penulis juga mempelajari tentang pengiriman *push notification* menggunakan layanan Firebase Cloud Messaging (FCM). Penulis berhasil mengirimkan notifikasi ke gawai pengguna melalui Firebase Console FCM dan Lumen. Penulis juga mengetahui bahwa terdapat beberapa cara untuk mengirimkan notifikasi jadwal, yang pertama yaitu menjalankan suatu *service back-end* setiap menitnya untuk mengecek apakah terdapat jadwal pada menit tertentu, jika terdapat jadwal di menit tersebut maka akan dikirim notifikasi jadwal ke pengguna. Cara yang kedua yaitu menjalankan *background process* pada aplikasi Flutter setiap beberapa menit untuk mendapatkan data jadwal dari *back-end*. Data jadwal tersebut akan didaftarkan menjadi notifikasi sesuai dengan waktu jadwal ke aplikasi *mobile* menggunakan *package* Flutter Local Notification pada Flutter. Sebelum mendaftarkan jadwal menggunakan Flutter Local Notification, penulis harus menghapus notifikasi yang sudah terdaftar pada aplikasi supaya menghindari notifikasi yang ganda. Pada awalnya penulis menerapkan cara yang kedua untuk pengiriman notifikasi aplikasi I'm UII, tetapi setelah berdiskusi dengan *team lead*, penulis mendapatkan pengetahuan baru yaitu apabila menerapkan *background process* pada sisi klien yaitu aplikasi *mobile* maka akan memberatkan *smartphone* pengguna karena selalu *request* ke API setiap menitnya dan melakukan proses mendaftarkan dan menghapus notifikasi. Hal ini dapat mengakibatkan aplikasi I'm UII mengkonsumsi sumber daya seperti RAM saat *background process* berjalan sehingga bisa mengakibatkan terganggunya pengalaman pengguna saat menggunakan *smartphone* untuk tugas yang membutuhkan sumber daya yang besar seperti bermain gim (Google Developers, 2021).

Setelah mengikuti kegiatan magang, penulis merasakan bahwa lebih baik dalam berkomunikasi karena penulis harus bisa berkomunikasi dengan rekan tim terutama jika

menghadapi suatu masalah. Adanya rapat progres juga membantu penulis dalam meningkatkan kemampuan dalam mengorganisasi pikiran karena penulis harus bisa menyampaikan dengan jelas apa yang ingin disampaikan. Saat magang juga penulis pernah mengisi acara TechTalk BSI sebagai pemateri dan berbagi pengetahuan penulis tentang pengembangan aplikasi *mobile* menggunakan Flutter.

BAB V
PENUTUP

5.1 Kesimpulan

Selama delapan bulan, penulis telah melaksanakan kegiatan magang di BSI UII. Pada kegiatan magang ini, penulis memiliki peran sebagai *software engineer* dengan tugas mengembangkan fitur *schedule* pada aplikasi *mobile* I'm UII. Berdasarkan pelaksanaan magang yang telah dilaksanakan, dapat disimpulkan sebagai berikut:

1. Berdasarkan hasil wawancara dan survei, didapatkan bahwa beberapa mahasiswa masih memiliki kesulitan dalam mengelola jadwalnya, sehingga diperlukan sebuah fitur untuk menangani masalah tersebut.
2. Dengan menggunakan metode pengembangan *waterfall*, fitur *schedule* aplikasi I'm UII berhasil dikembangkan. Karena aplikasi masih dalam tahap pengujian, sehingga siklus pengembangan belum masuk ke tahap pemeliharaan.
3. Fitur *schedule* dapat membantu mahasiswa dalam mengatur jadwalnya karena memiliki fungsionalitas seperti melihat, membuat, mengubah, dan menghapus jadwal. Mahasiswa juga dapat mengundang mahasiswa lain di jadwal yang dibuatnya. Fitur *schedule* juga menyediakan pengingat yang bnerupa notifikasi supaya mahasiswa tidak terlupa dengan jadwal yang dimilikinya.
4. Berdasarkan hasil pengujian menggunakan pengujian *black box* bahwa fitur *schedule* berjalan dengan baik tanpa ada cacat atau *bug*.
5. Fitur *schedule* I'm UII dapat berjalan dengan baik di gawai dengan sistem operasi Android.
6. Penulis dapat menerapkan ilmu yang didapatkan dari perkuliahan seperti implementasi pengembangan aplikasi menggunakan metode *waterfall*. Penulis juga mendapatkan ilmu yang tidak diajarkan selama kuliah, seperti pengembangan aplikasi *mobile* menggunakan Flutter, pengembangan API, dan pengiriman *push notification* menggunakan layanan FCM.

5.2 Saran

Penulis memiliki beberapa saran yang bisa diterapkan untuk pengembangan lebih lanjut fitur *schedule* aplikasi I'm UII, yaitu:

1. Dapat dilakukan pengujian lebih lanjut terkait fitur *schedule* karena pengujian yang penulis lakukan menggunakan pengujian *black box* besar kemungkinan belum mencakup semua skenario pengujian.
2. Walaupun fitur *schedule* berjalan baik di sistem operasi Android, terdapat kemungkinan bahwa tidak dengan sistem operasi lain. Penulis hanya menjalankan dan menguji aplikasi untuk platform Android saja dikarenakan kendala perangkat pengembangan, sehingga diperlukan pengujian lebih untuk platform lain terutama iOS untuk mengetahui apakah fitur *schedule* dapat berjalan dengan baik.
3. Menggunakan metode pengiriman notifikasi yang berbeda seperti pengiriman notifikasi yang didorong suatu *event*. Event tersebut bisa berupa membuat, mengubah, dan menghapus jadwal yang ada di basis data. Jika terdapat *event* maka akan memicu proses pengiriman notifikasi.
4. Mengembangkan aplikasi untuk semua unit kerja yang ada di UII untuk memasukan data terkait jadwal. Jadwal yang dimaksudkan seperti jadwal pesantrenisasi, tagihan, KKN, S3D, seminar atau webinar jurusan, lomba, beasiswa, dan kegiatan lainnya, baik kegiatan wajib atau pun kegiatan yang bisa diikuti mahasiswa. Jadwal kegiatan yang diinput oleh unit kerja UII akan dikirimkan ke mahasiswa yang berlangganan topik tertentu semisal topik webinar jurusan sehingga mahasiswa memiliki jadwal webinar yang diinput oleh jurusan.
5. Membuat halaman pengaturan berlangganan topik terkait jadwal. Pada halaman ini mahasiswa dapat memilih berlanggan topik jadwal sesuai keiingan. Topik jadwal pada halaman ini hanya untuk kegiatan yang bersifat tidak wajib diikuti mahasiswa seperti seminar dan lomba. Untuk kegiatan yang bersifat wajib seperti pembayaran tagihan, KKN, dan pesantrenisasi akan otomatis berlangganan ketika pengguna berhasil *login* ke aplikasi I’m UII.
6. Mengembangkan *back-end* untuk memasukan jadwal yang dibuat oleh unit kerja UII ke tabel jadwal pada aplikasi I’m UII sesuai dengan pengguna yang berlangganan topik tersebut.

DAFTAR PUSTAKA

Arfan, A., & Hendrik. (2022). Penerapan STLC dalam Pengujian Automation Aplikasi Mobile (Studi kasus: LMS Amikom Center PT. GIT Solution). *Automata*.

Badan Sistem Informasi UII. (n.d.). *Manajemen BSI*. Diperoleh dari https://bsi.uii.ac.id/manajemen-bsi/

Edde, G. P., & Budayawan, K. (2021). Pembuatan Aplikasi Reminder Jadwal Perkuliahandi JurusanTeknik Eletronika Berbasis Android. *Vocational Teknik Elektronika dan Informatika*.

Erikson, G. (2017). *Analisis Manajemen Waktu Terhadap Penulisan Laporan Akhir Pada Mahasiswa Semester 6 Angkatan 2014 Program Studi Administrasi Bisnis Jurusan Administrasi Bisnis Politeknik Negeri Sriwijaya.*

Flutter. (n.d.). *Build apps for any screen*. Diperoleh dari https://flutter.dev/

Google. (2023, Juli 07). *Firebase Cloud Messaging*. Diperoleh dari https://firebase.google.com/docs/cloud-messaging

Google Developers. (2021, Maret 11). *Background Execution Limits.* Diperoleh dari https://developer.android.com/about/versions/oreo/background

Google LLC. (2023, Juli 05). *Google Calendar*. Diperoleh dari https://play.google.com/store/apps/details?id=com.google.android.calendar

Grafiani, C. P. (2021). *Seni Manajemen Waktu: Rahasia Bagaimana Orang-orang Sukses Mengatur Waktu Mereka.* Kabupaten Bantul: Anak Hebat Indonesia.

Hidayanto, D. N. (2021). *Manajemen Waktu: Filosofi, Teori, Implementasi - Rajawali Pers.* Bantul Regency: PT. RajaGrafindo Persada.

KBBI Daring. (n.d.). *Waktu*. Diperoleh dari https://kbbi.kemdikbud.go.id/entri/waktu

Microsoft Corporation. (2023, Mei 30). *Microsoft To Do: Lists & Tasks*. Diperoleh dari PlayStore: https://play.google.com/store/apps/details?id=com.microsoft.todos

Oracle. (n.d.). *What Is a Database?* Diperoleh dari https://www.oracle.com/id/database/what-is-database/

Pratiwi, S. S. (2017). Pengaruh Keaktifan Mahasiswa Dalam Organisasi dan Motivasi. *Jurnal Pendidikan dan Ekonomi*, 54-64.

Rahmah, M. (2017). Desain dan implementasi sistem penjadwalan agenda berbasis android. *Jurnal Teknologi Informasi & Komunikasi Digital*, 196-206.

Rambe, A. R., & Prihantoro, H. (2022). Pengujian Otomatis Aplikasi Mobile dengan Teknik Black-box Menggunakan Appium (Studi Kasus: Pengembangan Aplikasi Jala Mobile). *Automata*.

Setiawan, R. (2021, November 17). *Black Box Testing Untuk Menguji Perangkat Lunak.* Diperoleh dari https://www.dicoding.com/blog/black-box-testing/

Sommerville, I. (2010). *Software Engineering, 9th Edition.* New York: Pearson.

Suhendro, J. M., Sudarma, M., & Khrisne, D. C. (2021). Rancang Bangun Aplikasi Seluler Penyedia Jasa Perawatan dan Kecantikan Menggunakan Framework Flutter. *Spektrum*.

Universitas Islam Indonesia. (2018, Desember 03). *Pedoman Penggunaan Merek Universitas Islam Indonesia*. Diperoleh dari https://drive.google.com/file/d/18noncN10wtL1PO5TY3SBb6XHyWIWG6ZI/view

Wang, H.-Y., Liao, C., & Yang, L.-H. (2013). What Affects Mobile Application Use? *International Journal of Marketing Studies*, 11-22.

Wilson, R. (2019). Perancangan Informasi Manajemen Waktu Melalui Media Buku Ilustrasi. *elibrary UNIKOM*.

Zega, Y. G., & Kurniawati, G. E. (2022). Pentingnya Manajemen Waktu Bagi Mahasiswa Dalam Meningkatkan Prestasi Belajar Di Sekolah Tinggi Teologi Duta Panisal Jember. *Metanoia*, 58-70.

LAMPIRAN