

BAB 3

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Kegiatan magang di PT Andreo Friendly Persada meliputi penempatan pada posisi *Intern Developer* di bawah divisi IT Department. Penempatan tersebut merupakan hasil penugasan dari *supervisor* lapangan, Ibu Eni Suprapti.

Dalam alur koordinasi kerja sehari-hari, posisi ini berada di bawah arahan langsung *supervisor*. Namun, inisiasi untuk sebuah proyek baru atau penugasan berskala besar berasal dari perintah Direktur. Arahan strategis dari Direktur tersebut kemudian diteruskan kepada *supervisor* untuk diolah menjadi tugas-tugas yang lebih spesifik dan terperinci. Dengan demikian, *supervisor* berperan sebagai jembatan antara mandat pimpinan dan eksekusi teknis di lapangan. Pola koordinasi ini menegaskan bahwa segala bentuk pekerjaan yang dilakukan merupakan bagian dari alur komando yang jelas. Kedudukan sebagai pelaksana teknis memungkinkan adanya fleksibilitas tinggi, di mana *supervisor* dapat mengalihkan tugas pekerjaan dari satu proyek ke proyek lainnya sesuai dengan perubahan prioritas yang ditetapkan oleh Direktur.

3.1.1 Alur Kerja dan Koordinasi Magang

Alur kerja selama kegiatan magang di PT Andreo Friendly Persada disusun untuk memastikan adanya kontribusi yang efektif sambil menyediakan bimbingan yang terstruktur.

Alur kerja tersebut adalah sebagai berikut.

1. Penugasan

Penugasan diberikan secara langsung oleh *supervisor*, Ibu Eni Suprapti, sesuai dengan kebutuhan proyek yang sedang berjalan di perusahaan. Tugas yang diberikan berfokus pada bidang *Full Stack Web Development* dan *Android Development*, tergantung pada prioritas proyek.

2. Pembimbingan Awal

Setelah menerima penugasan, *supervisor* memberikan pengarahan awal (*briefing*). Sesi ini mencakup penjelasan mengenai tujuan tugas, hasil akhir

(*deliverable*) yang diharapkan, standar teknis yang digunakan, serta alat kerja (*tools*) yang relevan untuk pelaksanaan tugas.

3. Pelaksanaan Tugas

Pelaksanaan tugas dilakukan secara mandiri sesuai dengan arahan yang telah diberikan. Proses ini mencakup analisis masalah, perancangan solusi, pengkodean, dan pengujian awal. Upaya pencarian solusi secara mandiri sangat diutamakan sebelum berdiskusi lebih lanjut dengan *supervisor* jika menemui kendala.

4. Pemantauan dan Umpan Balik

Supervisor memantau kemajuan pekerjaan secara berkala melalui dua mekanisme utama:

- (a) Update Harian: Komunikasi singkat dan informal melalui *chat* atau rapat harian untuk melaporkan progres dan kendala.
- (b) Sesi Mingguan: Pertemuan evaluasi setiap akhir minggu untuk meninjau hasil kerja, memberikan umpan balik yang lebih mendalam, dan merencanakan target untuk minggu berikutnya.

5. Kolaborasi Tim

Kolaborasi aktif dengan anggota tim pengembang lainnya merupakan bagian penting dari alur kerja. Kolaborasi ini terjadi dalam berbagai kegiatan, seperti diskusi teknis, proses *code review* untuk memastikan kualitas kode, dan integrasi pekerjaan dengan bagian sistem yang lain demi mencapai tujuan proyek secara keseluruhan.

Terdapat alur koordinasi yang jelas untuk memastikan kelancaran komunikasi dan kerja sama, di antaranya sebagai berikut.

1. Koordinasi Langsung dengan *Supervisor*

Jalur koordinasi utama terjalin secara langsung dengan *supervisor*. Koordinasi ini mencakup semua aspek pekerjaan, mulai dari penerimaan tugas, permintaan klarifikasi, pelaporan kemajuan, hingga sesi untuk mendapatkan umpan balik.

2. Koordinasi dengan Tim Pengembang

Koordinasi juga dilakukan dengan sesama anggota tim pengembang, baik *front-end* maupun *back-end*, untuk membahas isu-isu teknis. Koordinasi ini

penting untuk memastikan integrasi sistem berjalan lancar, menyelesaikan masalah teknis bersama, dan saling berbagi pengetahuan.

3.2 Tugas yang Dilakukan

Pengembangan *website Eight*, sebuah situs *web* yang berfungsi sebagai profil perusahaan, mencakup spektrum tanggung jawab yang luas. Tahap awal pekerjaan meliputi *hosting* dan pengujian untuk memastikan kelayakan situs, hingga aspek desain visual seperti pembuatan logo dan templat untuk layanan (*gigs*). Proyek ini juga melibatkan pengembangan halaman-halaman inti seperti beranda (*homepage*) dan halaman portofolio (*works*). Selain itu, dilakukan pula penyempurnaan di sisi antarmuka, termasuk perbaikan konten dan penambahan animasi untuk meningkatkan pengalaman pengguna, serta pengembangan fitur fungsional seperti sistem akun.

Proyek kedua adalah *SportsBooking*, sebuah aplikasi *mobile* yang dirancang untuk memudahkan pengguna dalam memesan lapangan olahraga secara daring. Pengembangan aplikasi ini berfokus pada fitur-fitur fundamental yang esensial bagi pengguna. Cakupan tugasnya meliputi pembuatan alur autentikasi, seperti halaman *login* dan registrasi akun baru, serta perancangan dan implementasi antarmuka untuk menampilkan daftar lokasi (*venue*) dan proses pemesanan lapangan. Untuk memastikan keamanan dan pengelolaan data yang andal, aplikasi diintegrasikan dengan *Firebase*, yang mencakup penanganan proses autentikasi pengguna dan manajemen data pemesanan.

Selanjutnya, pengembangan *platform KartuDigital* berfokus pada pengerjaan sisi *frontend*. Proyek ini bertujuan untuk menyediakan layanan pembuatan undangan digital yang interaktif dan intuitif. Lingkup pekerjaan pada bagian ini meliputi perancangan berbagai templat undangan yang menarik serta pembangunan fitur *editor* yang memungkinkan pengguna untuk memodifikasi templat tersebut sesuai keinginan. Bagian krusial lainnya adalah integrasi sistem *frontend* dengan *backend* melalui *API*, untuk memastikan data undangan dapat dikelola dan ditampilkan dengan benar.

Sebagai pelengkap proyek *KartuDigital*, dikerjakan pula sisi *backend* yang bertanggung jawab atas logika, keamanan, dan pengelolaan data di belakang layar. Tanggung jawab utama dalam bagian ini adalah pembangunan sistem autentikasi pengguna yang aman menggunakan *JSON Web Tokens (JWT)* melalui implementasi *middleware*. Selain itu, dikembangkan pula mekanisme pengelolaan

session pengguna menggunakan *cookies* serta penerapan proteksi rute (*protected routes*). Hal ini memastikan bahwa hanya pengguna yang terautentikasi yang dapat mengakses halaman atau fitur tertentu, sehingga keamanan data pengguna tetap terjaga.

Proyek terakhir adalah pengerjaan *platform* Undangan Digital yang dibangun menggunakan kerangka kerja modern, *Next.js*. Proyek ini merupakan pengembangan lebih lanjut dari *platform* undangan digital dengan memanfaatkan keunggulan teknologi *Next.js* untuk performa dan *SEO* yang lebih baik. Pengerjaannya meliputi pengembangan templat undangan yang dinamis dan fungsional, lengkap dengan fitur *editor*. Di samping itu, diimplementasikan pula sistem keamanan melalui integrasi autentikasi berbasis *JWT*, yang memungkinkan pengelolaan akses pengguna yang aman dan efisien di dalam *platform*.

3.3 Uraian Pelaksanaan Magang

Seluruh rangkaian kegiatan selama masa magang dicatat dalam sebuah jurnal kerja mingguan. Pencatatan ini bertujuan untuk mendokumentasikan setiap proses, mulai dari penugasan awal, eksekusi teknis, hingga penyelesaian masalah. Jurnal ini berfungsi sebagai bukti pelaksanaan kerja serta rekam jejak kontribusi pada proyek-proyek yang ditangani. Rekapitulasi jurnal kerja mingguan tersebut dirangkum dalam Tabel 3.1 di bawah ini.

Tabel 3.1. Uraian Pelaksanaan Magang

Minggu	Kegiatan
1-2	Pengembangan Aplikasi <i>Mobile</i> untuk sistem pemesanan lapangan secara <i>online</i> menggunakan <i>Android Studio</i>. Pembuatan halaman depan dan halaman <i>login</i>. Perancangan desain dan alur pengguna (<i>user flow</i>) untuk aplikasi menggunakan <i>Figma</i>. Pengajuan prototipe <i>Figma</i> dan presentasi aplikasi <i>SportsBooking</i>.
3	Pembuatan proyek <i>Android Studio</i> baru dan perancangan tampilan <i>XML</i> untuk halaman <i>login</i> dan pendaftaran (<i>signup</i>). Melanjutkan dan finalisasi tampilan halaman <i>login</i> dan pendaftaran.

Tabel 3.1. Uraian Pelaksanaan Magang (Lanjutan)

Minggu	Kegiatan
4	Pembuatan halaman Beranda <i>Venue</i> dan bilah navigasi (<i>navbar</i>) untuk tampilan <i>mobile</i>. Pembuatan Halaman <i>Admin</i> untuk melakukan <i>input</i> data <i>venue</i>. Perancangan tata letak (<i>layout</i>) <i>XML</i> untuk halaman depan yang menampilkan rekomendasi <i>venue</i>. Pembuatan <i>MainActivity</i> untuk menampilkan daftar <i>venue</i> dari <i>database Firebase</i>.
5	Pembuatan <i>LoginActivity</i> dan <i>SignupActivity</i> yang terintegrasi dengan <i>Firebase Authentication</i>. Pengujian (<i>testing</i>) fungsi pembuatan akun dan proses autentikasi pengguna. Pembuatan halaman <i>DetailLapanganActivity</i> untuk mengambil (<i>fetching</i>) data detail <i>venue</i>.
6	Perancangan tata letak <i>DetailLapangan</i> dan implementasi <i>Google Maps API</i>. Pembuatan fungsi pemesanan dan perancangan skema data pemesanan (<i>booking</i>) di <i>Firebase</i>.
7	Melakukan format dan unggah data pemesanan ke <i>Firebase</i>. Pembuatan fitur slot pemesanan (<i>booking slot</i>) yang terikat pada <i>ID</i> pengguna (<i>userId</i>).
8	Perancangan tata letak <i>editor</i>, mencakup bilah sisi (<i>sidebar</i>) dan halaman pratinjau (<i>preview page</i>). Menjadikan <i>editor</i> responsif untuk perangkat <i>mobile</i> dan <i>tablet</i>. Perancangan tata letak untuk <i>template TheCharming.tsx</i>.
9	Memperbaiki fungsi <i>login</i> yang bermasalah. <i>Editor</i>: Implementasi pengambilan (<i>fetching</i>) data <i>template</i> milik pengguna. Implementasi proses <i>onboarding</i> untuk menerapkan <i>template</i> yang dipilih pengguna.
10	Pengambilan (<i>fetching</i>) data <i>template</i> untuk proses <i>onboarding</i>. Menjadikan dasbor pengguna dan <i>template</i> menjadi responsif. <i>Editor</i>: Implementasi <i>dynamic routes</i> untuk <i>template</i> pengguna. Perbaikan <i>bug</i> pada tata letak halaman <i>editor</i>.

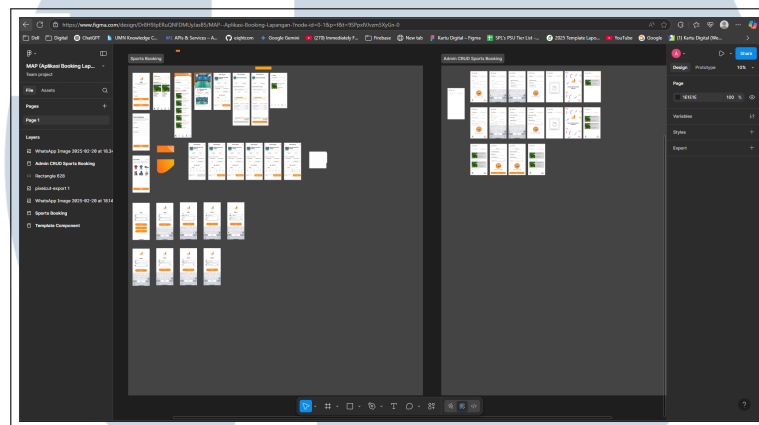
Tabel 3.1. Uraian Pelaksanaan Magang (Lanjutan)

Minggu	Kegiatan
11	Perbaikan <i>bug</i> pada proses <i>onboarding</i> di mana <i>template</i> tidak dapat ditampilkan. Implementasi sistem otentikasi <i>JWT</i> dengan menyimpan <i>token</i> di dalam sesi (<i>session</i>).
12	Pembuatan <i>middleware</i> untuk autentikasi pada rute terproteksi (<i>protected routes</i>). Konfigurasi <i>server</i> terkait masa berlaku <i>token</i> dan <i>cookies</i> (<i>expired token/cookies</i>). Memperbaiki <i>AuthContext</i> agar <i>cookies</i> tidak hilang pada sisi <i>klien</i> .
13	Implementasi fungsi <i>logout</i> : menghapus <i>cookies</i> dan memblokir akses dari <i>server</i> . Perbaikan <i>bug</i> pada proses <i>login</i> yang tidak mengarahkan ke dasbor. Perbaikan <i>bug</i> pada fungsi <i>logout</i> yang memerlukan dua kali klik.
14-16	Implementasi autentikasi sisi <i>server</i> (<i>server-side auth</i>): penggunaan <i>httpOnly</i> , <i>secure</i> , dan perbaikan <i>bug logout</i> . Pembuatan layar pembuka/layar <i>logout</i> (<i>splash screen/logout screen</i>) sebagai penanda saat menunggu <i>cookies</i> terhapus. Pembuatan <i>template TropicalParadise.tsx</i> dengan dua variasi tampilan. Perbaikan <i>bug</i> pada <i>editor</i> di mana teks tidak terganti secara otomatis saat diketik. Implementasi fitur unggah audio di dalam <i>editor</i> untuk fungsi pratinjau.

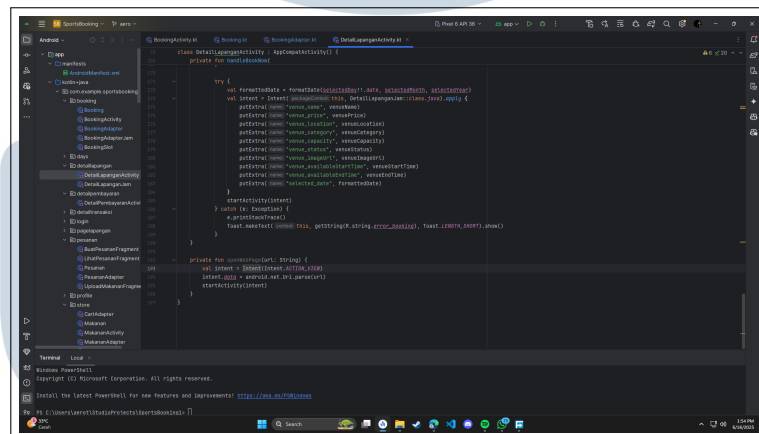
3.4 Perangkat dan *software* yang digunakan

Dalam merancang aplikasi *SportsBooking*, *software* yang digunakan adalah *Figma*. *Figma* digunakan untuk merancang *user interface* dari aplikasi ini dan membuat *wireframe* untuk bagaimana *user* akan berinteraksi dengan aplikasi ini. Pada gambar 3.1, terdapat dua *interface* yang berbeda. Satu untuk *user* dan satu khusus untuk *admin* agar dapat mengelola aplikasi. Untuk membangun

aplikasi berbasis *Android*, *software IDE* yang digunakan adalah *Android Studio*. *Android Studio* menggunakan bahasa *Kotlin*, yaitu sebuah bahasa pemrograman yang dikhususkan untuk aplikasi berbasis *Android*. *Android Studio* memiliki *virtual device Android* yang dapat menjalankan aplikasi selagi dibangun untuk dapat dijalankan secara langsung. Tampilan *Android Studio* dan Projek Aplikasi *SportsBooking* dapat dilihat pada gambar 3.2.



Gambar 3.1. Design User Interface Aplikasi *SportsBooking* melalui *software Figma*



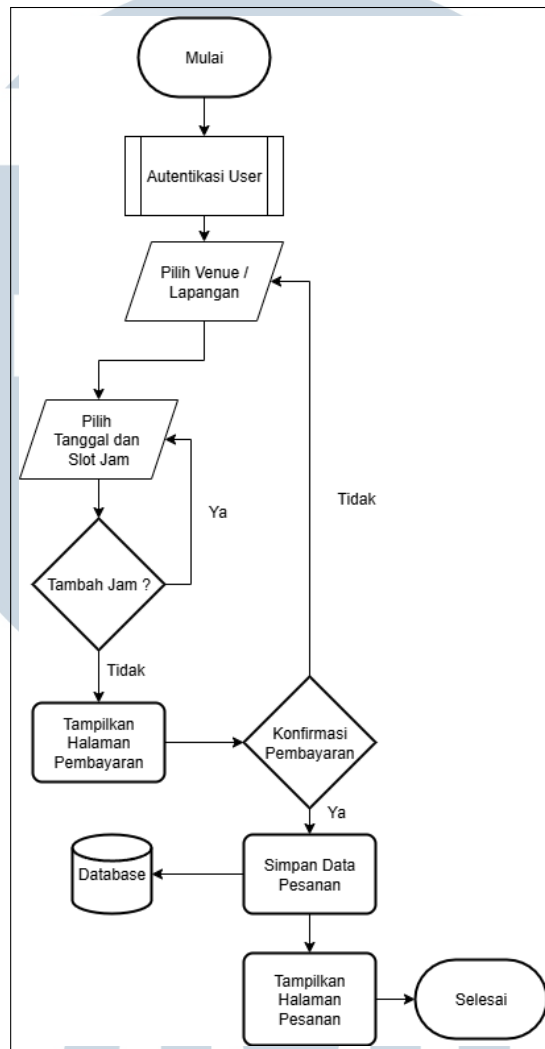
Gambar 3.2. Tampilan *software IDE* untuk develop Aplikasi *SportsBooking*, *Android Studio*

3.5 Perancangan aplikasi

3.5.1 Flowchart Aplikasi Booking Lapangan

Pembuatan aplikasi *booking* lapangan terdiri dari Autentikasi, Pemesanan Lapangan, dan Pembayaran. Tahapan yang dilakukan pada pra produksi berfokus untuk melakukan riset serta pembuatan *flowchart*. Tujuan dari pembuatan *flowchart*

adalah untuk memahami proses penggunaan aplikasi yang akan dibuat. *Flowchart* dari aplikasi *SportsBooking* dapat dilihat pada Gambar 3.3.



Gambar 3.3. *Flowchart* Aplikasi *booking* lapangan.

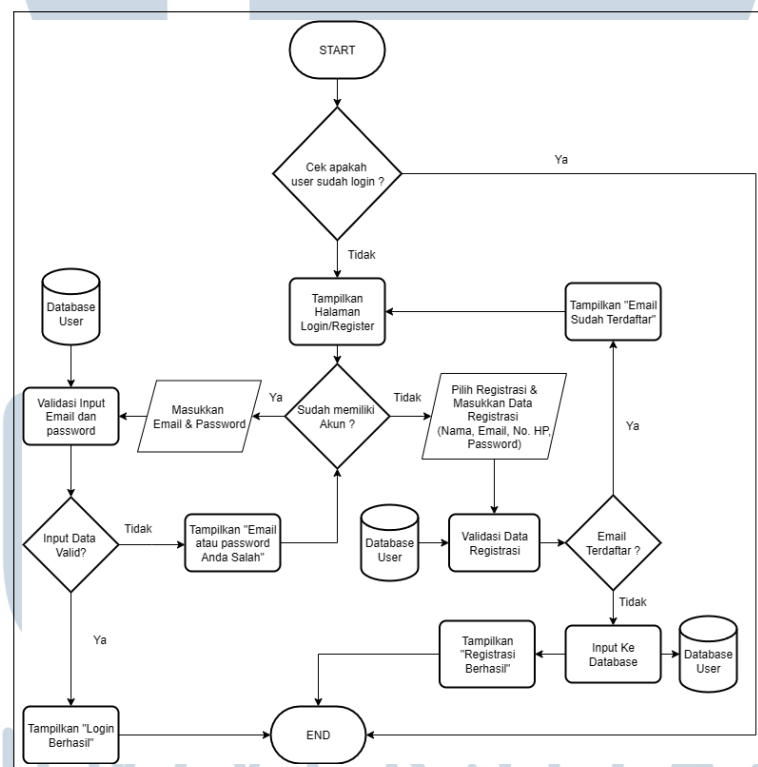
Pada gambar 3.3, *flowchart* ini menggambarkan alur aplikasi *booking* lapangan. Pertama, pengguna akan melalui tahap autentikasi untuk memastikan hanya pengguna yang terdaftar dapat mengakses aplikasi. Setelah berhasil masuk, pengguna bisa melihat informasi lapangan yang tersedia, seperti jenis lapangan, jam operasional, dan ketersediaan jadwal pemesanan.

Langkah berikutnya adalah proses pemesanan di mana pengguna diminta untuk memilih lapangan serta tanggal pemakaian, lalu dilanjutkan dengan memilih slot waktu. Sistem menyediakan opsi bagi pengguna untuk menambahkan lebih dari satu jam pemesanan. Jika pengguna memilih untuk menambahkan jam, maka

akan kembali ke proses pemilihan slot. Jika tidak, sistem melanjutkan ke tahap pembayaran.

Setelah pembayaran dilakukan, sistem melakukan verifikasi apakah pembayaran berhasil. Jika berhasil, data pesanan akan dikonfirmasi dan disimpan ke dalam *database*. Informasi ini kemudian ditampilkan di halaman pesanan sebagai bukti *booking* kepada pengguna. Namun, jika pembayaran gagal, sistem akan mengembalikan pengguna ke proses pemilihan lapangan untuk memulai ulang. *Flowchart* ditutup dengan langkah akhir yaitu "Selesai" yang menandai berakhirnya proses *booking*.

3.5.2 Flowchart Autentikasi User



Gambar 3.4. Flowchart Autentikasi User

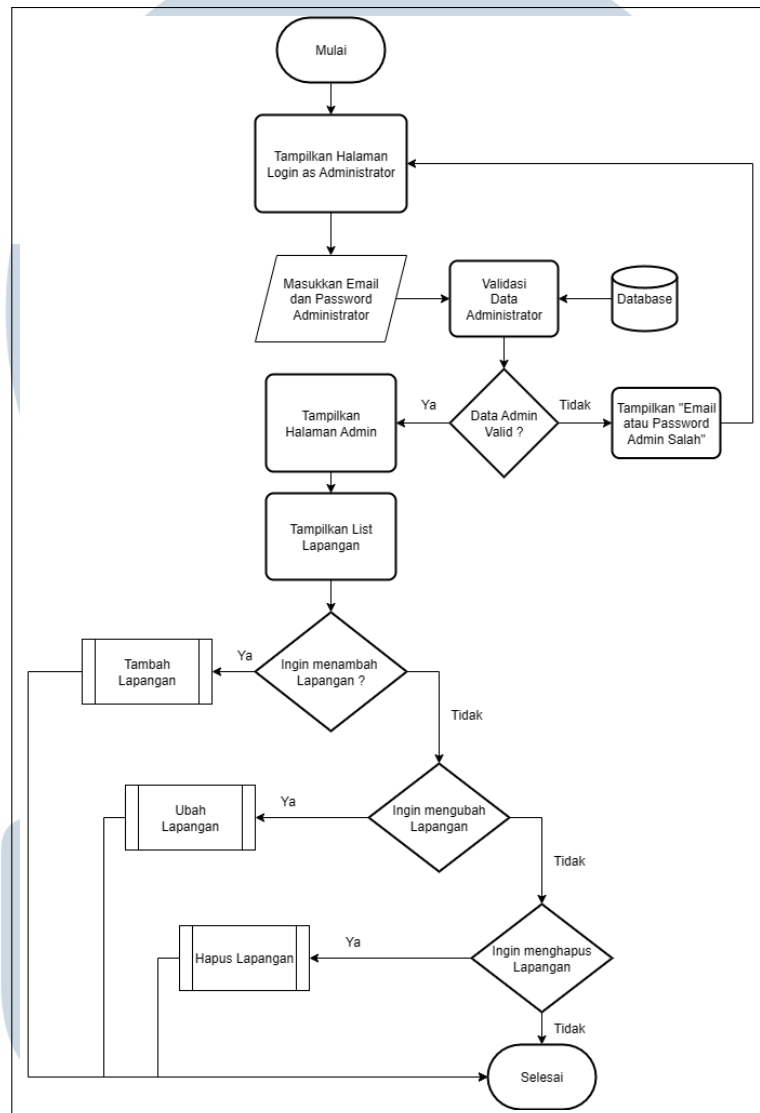
Flowchart pada gambar 3.4 menggambarkan alur proses autentikasi pengguna, meliputi login dan registrasi. Proses dimulai ketika pengguna membuka aplikasi. Sistem pertama-tama memeriksa apakah pengguna sudah login atau belum. Jika sudah, pengguna akan langsung diarahkan ke halaman utama (*Main*). Namun, jika belum, sistem akan menampilkan halaman *Login/Register*.

Pengguna memiliki dua pilihan: *Login* atau *Register*. Jika memilih *Login*, pengguna diminta memasukkan email dan *password*. Sistem kemudian memvalidasi input tersebut dengan data yang tersimpan di Database User. Jika valid, *login* berhasil dan pengguna diarahkan ke halaman utama. Jika tidak valid, sistem akan menolak akses dan pengguna harus mengulangi proses *login*.

Jika pengguna memilih *Register*, pengguna juga harus mengisi data registrasi, termasuk nama, email, nomor HP, dan *password*. Sistem akan memeriksa apakah email sudah terdaftar di database. Jika belum, data pengguna akan disimpan dan registrasi dinyatakan berhasil, kemudian pengguna diarahkan ke halaman utama. Namun, jika email sudah terdaftar, sistem akan memberi notifikasi dan pengguna diminta untuk menggunakan email lain atau melakukan *login*.



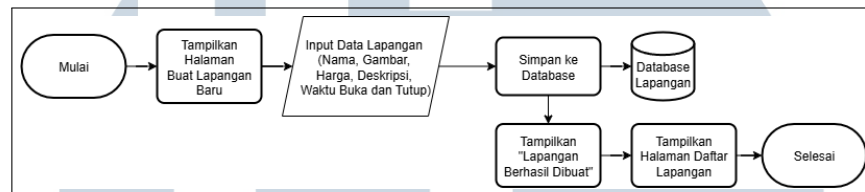
3.5.3 Flowchart Halaman Admin (Membuat, Mengubah dan Menghapus Lapangan)



Gambar 3.5. Flowchart Admin Page

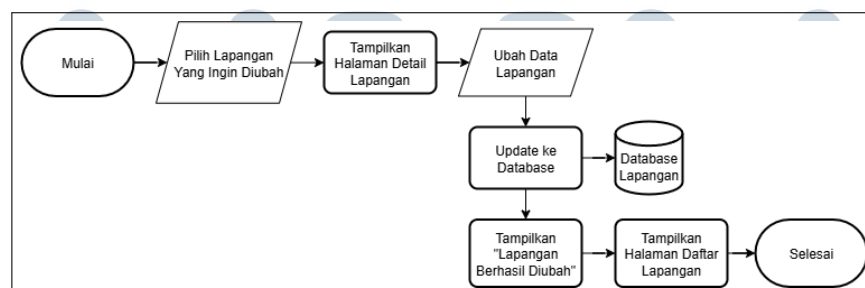
Flowchart 3.5 menggambarkan alur utama dari halaman admin dalam mengelola data lapangan. Proses dimulai dengan autentikasi sebagai admin untuk langkah keamanan awal. Setelah berhasil login, sistem akan menampilkan daftar lapangan yang telah tersedia. Dari sini, admin dapat memilih tiga opsi utama, yaitu menambahkan lapangan baru, mengubah lapangan yang sudah ada, atau menghapus lapangan. Jika admin memilih untuk menambahkan, maka sistem akan menampilkan *form* input untuk menambahkan data baru. Jika

memilih mengubah, maka admin akan diarahkan ke halaman detail lapangan untuk melakukan perubahan. Jika memilih menghapus, sistem akan menampilkan konfirmasi sebelum data benar-benar dihapus. Setelah setiap tindakan (tambah, ubah, hapus) selesai, sistem akan kembali ke daftar lapangan untuk memungkinkan tindakan lanjutan atau menyelesaikan sesi kerja.



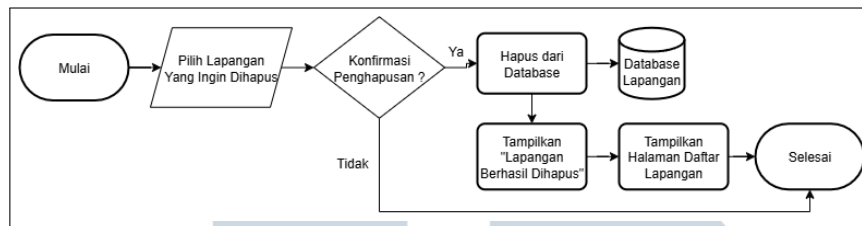
Gambar 3.6. *Flowchart Tambah Lapangan*

Flowchart 3.6 menjelaskan proses penambahan data lapangan baru. Proses ini dimulai dengan menampilkan halaman pembuatan lapangan baru, kemudian admin mengisi form yang terdiri dari nama lapangan, gambar, harga, deskripsi, serta waktu buka dan tutup. Setelah data diinput, sistem akan menyimpan informasi tersebut ke dalam database. Jika proses penyimpanan berhasil, sistem akan menampilkan notifikasi bahwa lapangan berhasil dibuat, lalu mengarahkan admin kembali ke halaman daftar lapangan. Proses berakhir setelah semua langkah tersebut selesai dilakukan.



Gambar 3.7. *Flowchart Ubah Lapangan*

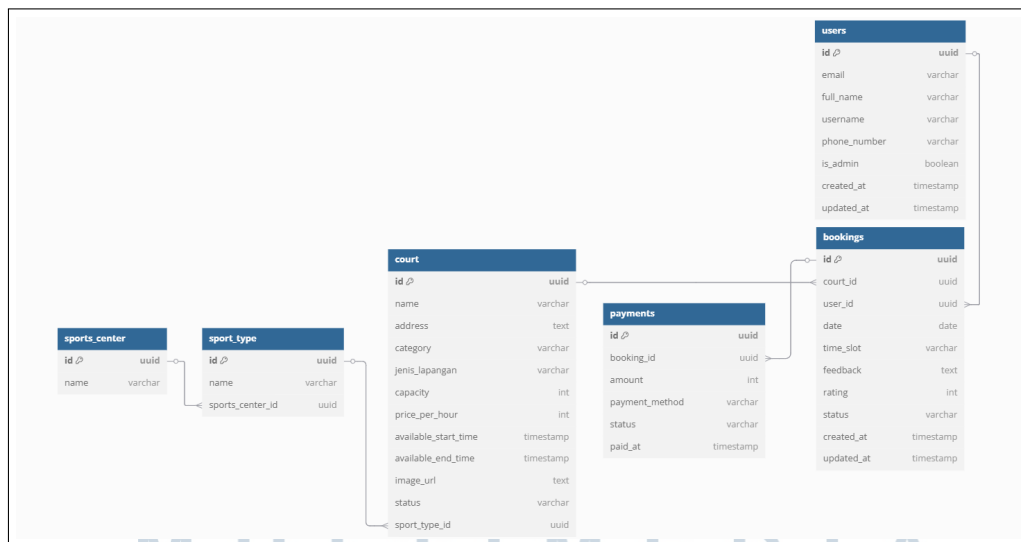
Flowchart 3.7 menggambarkan proses pengubahan data lapangan. Admin memulai dengan memilih lapangan yang ingin diubah dari daftar. Setelah itu, sistem akan menampilkan detail data lapangan tersebut untuk diedit. Admin kemudian dapat mengubah informasi seperti nama, harga, atau data lainnya sesuai kebutuhan. Setelah perubahan dilakukan, sistem akan menyimpan data baru ke dalam *database*. Jika *update* berhasil, akan muncul notifikasi bahwa lapangan berhasil diubah, dan admin akan diarahkan kembali ke daftar lapangan.



Gambar 3.8. *Flowchart* Hapus Lapangan

Flowchart 3.8 menjelaskan proses penghapusan data lapangan. Proses ini diawali dengan admin memilih lapangan yang ingin dihapus. Setelah itu, sistem akan menampilkan konfirmasi penghapusan untuk menghindari kesalahan. Jika admin membatalkan, maka tidak ada perubahan yang dilakukan. Namun jika admin mengonfirmasi penghapusan, maka data lapangan akan dihapus dari database. Setelah berhasil dihapus, sistem akan menampilkan notifikasi bahwa penghapusan berhasil, lalu kembali ke halaman daftar lapangan untuk melanjutkan pengelolaan data lainnya.

3.5.4 Data Modeling Firebase



Gambar 3.9. Diagram Data Modeling Firebase

Gambar 3.9 menunjukkan struktur data *modeling* yang digunakan dalam sistem pemesanan lapangan olahraga berbasis *Firebase*. Model ini dirancang untuk menggambarkan keterkaitan antar entitas utama seperti pengguna, pusat olahraga, jenis olahraga, lapangan, serta pemesanan.

Struktur ini terdiri dari lima tabel utama:

1. *users*: Menyimpan informasi akun pengguna aplikasi seperti *email*, *full_name*, *username*, *phone_number*, dan status *is_admin*. Tabel ini digunakan sebagai referensi utama dalam proses pemesanan.
2. *sports_center*: Mewakili daftar pusat olahraga atau GOR (Gelanggang Olahraga), seperti GOR Badminton Mega Kuningan. Setiap pusat olahraga dapat memiliki beberapa jenis olahraga.
3. *sport_type*: Menyimpan data jenis olahraga seperti *badminton*, *futsal*, dan *basket* yang tersedia di suatu pusat olahraga. Entitas ini terhubung dengan tabel *sports_center* melalui *foreign key*.
4. *court*: Berisi informasi detail mengenai lapangan yang tersedia. Informasi mencakup *name*, *address*, *category*, *jenis_lapangan*, *capacity*, *price_per_hour*, waktu ketersediaan (*available_start_time* dan *available_end_time*), serta status lapangan. Setiap lapangan dikaitkan dengan jenis olahraga melalui *foreign key sport_type_id*.
5. *bookings*: Menyimpan catatan seluruh transaksi pemesanan yang dilakukan oleh pengguna. Informasi yang dicatat meliputi *user_id*, *court_id*, tanggal (*date*), waktu pemesanan (*time_slot*), status pemesanan (*status*), serta *feedback* dan *rating* dari pengguna.

Model ini memudahkan proses pengelolaan data di *Firebase*, dengan memastikan keterhubungan antar dokumen menggunakan struktur seperti *foreign key*. Dengan adanya model ini, sistem mampu mencegah terjadinya pemesanan ganda, memungkinkan pengumpulan ulasan pengguna, dan membuka kemungkinan integrasi fitur tambahan seperti pembayaran dan manajemen *admin* secara terstruktur.

3.6 User Requirement

Berikut adalah daftar kebutuhan pengguna (*User Requirement*) untuk aplikasi booking lapangan olahraga:

1. Pengguna dapat melakukan pendaftaran akun menggunakan email dan password.

2. Pengguna dapat masuk ke dalam sistem menggunakan akun yang telah didaftarkan.
3. Pengguna dapat melihat daftar dan jadwal lapangan yang tersedia untuk dipesan.
4. Pengguna dapat memesan lapangan sesuai dengan jadwal yang tersedia.
5. Pengguna menerima notifikasi konfirmasi setelah melakukan pemesanan.
6. Pengguna dapat membatalkan pemesanan jika terjadi perubahan rencana.
7. Pengguna dapat melihat riwayat pemesanan yang telah dilakukan sebelumnya.
8. Pengguna dapat mengatur profil akun, seperti mengganti password atau email.

3.7 Implementasi

Aplikasi *SportsBooking* akan diawali dari *page Login/Register*, hal ini akan memudahkan *user* untuk mendaftarkan data diri untuk menggunakan aplikasi *booking* lapangan.

The image displays two side-by-side mobile application screens for the SportsBooking app. The left screen is the 'Login' page, featuring a logo at the top, a welcome message, and input fields for 'Email' and 'Password'. It includes a 'Masuk' (Login) button and a 'Daftar' (Register) button. The right screen is the 'Register' page, titled 'Selamat Bergabung' (Welcome), with a message about completing a profile. It contains input fields for 'Username', 'Nama Lengkap' (Full Name), and 'Nomer Telepon' (Phone Number), followed by a 'Daftar' button.

Gambar 3.10. Tampilan *login page* dan *register page* untuk aplikasi *SportsBooking*

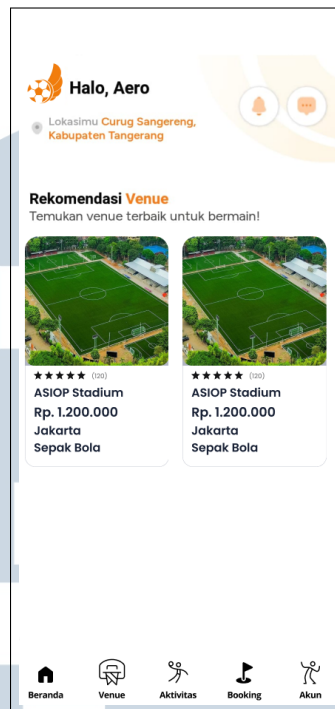
Pada gambar 3.10, *user* akan diarahkan langsung ke halaman *login* pertama kali membuka aplikasi. Jika belum memiliki akun, maka *user* dapat mengklik tombol "Daftar" untuk mendaftarkan akun baru yang kemudian akan mengarah ke halaman "Daftar". *User* dapat mengisi data diri seperti *username*, nama lengkap, dan nomor telepon yang dapat digunakan. Autentikasi dilakukan dengan memanggil *auth* dari *Firebase* untuk melakukan pencocokan apakah akun sudah dibuat di dalam tabel *Users* atau belum dan kemudian memeriksa *password* yang dimasukkan sesuai atau tidak untuk memastikan autentikasi *user*.

Fungsi autentikasi menginisialisasi *FirebaseAuth* dan *FirebaseFirestore* saat *onCreate()* dijalankan. Kemudian *onCreate()* dijalankan untuk mendapatkan referensi ke elemen *UI* seperti *EditText* untuk *email* dan *password*, serta *Button* untuk *login* dan *TextView* untuk pendaftaran.

Ketika tombol *login* (*btnLogin*) diklik, fungsi ini mengambil teks dari kolom *email* dan *password*. Kemudian, memeriksa apakah kolom *email* atau *password* kosong jika ya, pesan *Toast* akan muncul memberitahukan pengguna bahwa *email* dan *password* tidak boleh kosong.

Jika kedua kolom terisi, fungsi ini mencoba untuk masuk (*signInWithEmailAndPassword*) dengan *email* dan *password* yang diberikan menggunakan *FirebaseAuth*. Jika proses *login* berhasil (*task.isSuccessful*), fungsi akan memeriksa peran pengguna (*checkUserRole*) berdasarkan *ID* pengguna (*user.uid*) yang didapatkan dari *FirebaseAuth*. Namun, jika *login* gagal, sebuah pesan *Toast* akan ditampilkan kepada pengguna dengan detail kesalahan otentikasi.

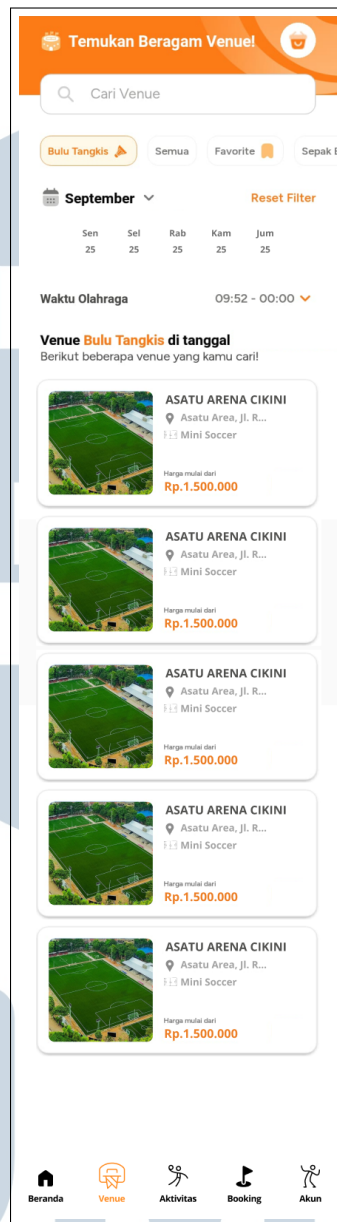




Gambar 3.11. Tampilan *Homepage* Dari Aplikasi *SportsBooking*

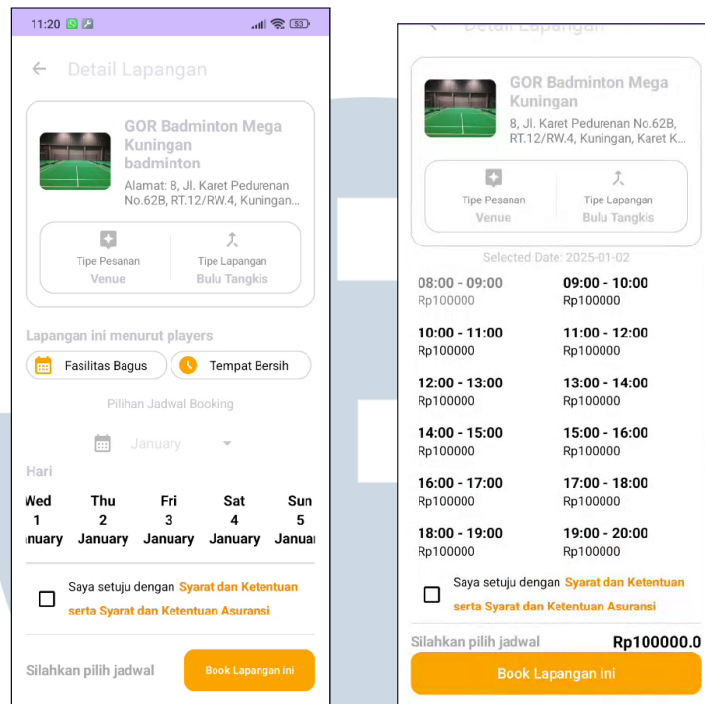
Pada gambar 3.11, terdapat tampilan *Homepage* dari aplikasi *SpotsBooking*. Setelah *user* sudah terdaftar dan *login*, maka *user* akan diarahkan ke *homepage* di mana terdapat rekomendasi pilihan untuk *venue* olahraga yang sedang ditampilkan. Lalu pada bagian bawah terdapat navigasi untuk berpindah-pindah halaman seperti *Venue*, *Aktivitas*, *Booking*, dan *Akun*.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



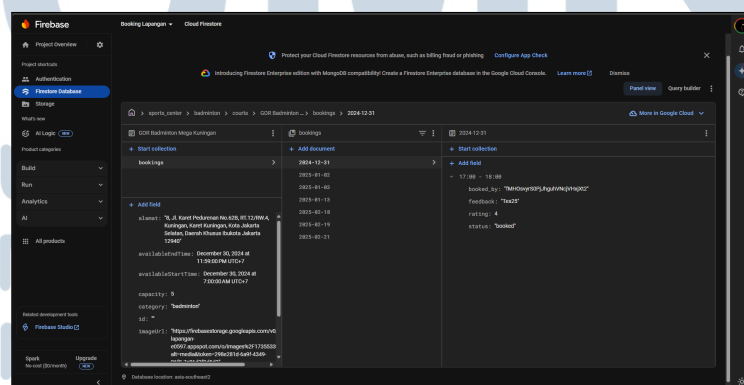
Gambar 3.12. Tampilan Halaman *Venue* Dari Aplikasi *SportsBooking*

Pada gambar 3.12 terdapat halaman *Venue*, yaitu halaman yang berisi *list* dari *venue* yang sudah terdaftar dalam aplikasi *SportsBooking*. Dikategorikan berdasarkan cabang olahraga yang sesuai, kemudian *user* juga dapat *filter* untuk *venue* yang tersedia pada tanggal tertentu. *Venue* diatur sesuai dengan kategori dalam *Firebase* dengan skema agar *venue* tidak memiliki kategori yang salah.



Gambar 3.13. Booking tanggal dan slot jam Dari Aplikasi SportsBooking

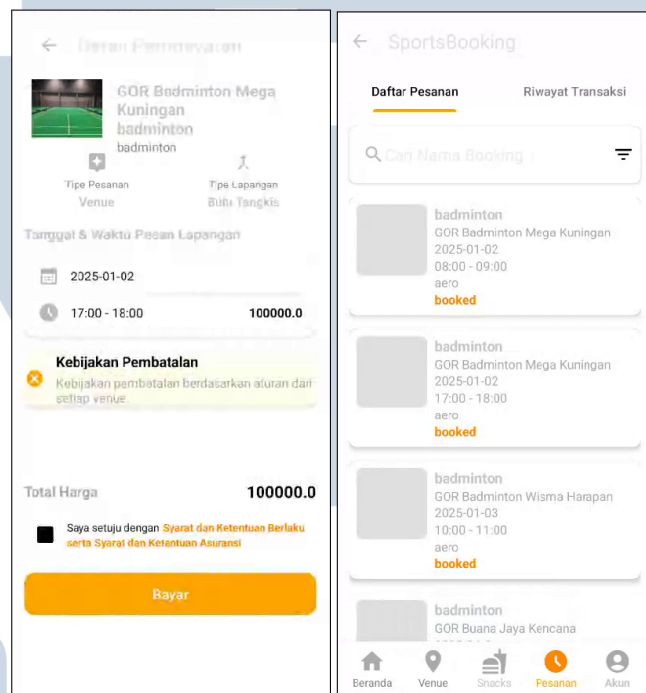
Untuk gambar 3.13 yang menunjukkan halaman *booking*, di halaman ini terdapat *venue* pilihan *user* di *venue page* yang akan otomatis tersimpan dan datanya akan dikirim ke *page-page* berikutnya untuk memastikan *booking user* akan tersimpan di jadwal *venue* tersebut. Menggunakan *Intent.action* untuk memastikan data akan berpindah ke *page* berikut. Untuk *page* berikutnya harus menerima data tersebut dengan melakukan *binding* data ke dalam variabel yang dideklarasikan lagi khusus untuk *page* tersebut.



Gambar 3.14. Tampilan Firebase Dari Aplikasi SportsBooking

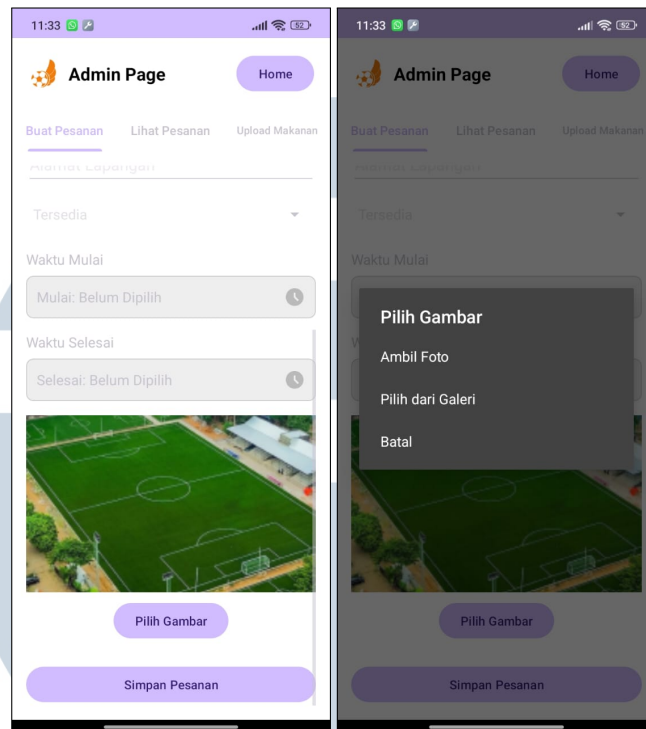
Pada gambar 3.14, struktur untuk sebuah *booking* dalam *venue* diinput

dengan format seperti : `/sports_center/badminton/courts/GOR Badminton Mega Kuningan/bookings/2024-12-31`. Hal ini memudahkan pendataan untuk setiap "booking" karena terdapat dalam setiap *venue* yang di-booking. Untuk *slot* jam sudah dibuat untuk setiap jamnya setiap hari. Jadi ketika *user* akan booking pada *slot* jam tertentu dan pada tanggal tertentu, dipastikan tidak akan terjadi *double booking* di mana pada jam tertentu dan tanggal tertentu terdapat dua booking yang sama.



Gambar 3.15. Detail Pembayaran dan Daftar Pesanan Dari Aplikasi *SportsBooking*

Pada Gambar 3.15, setelah *user* melakukan pemilihan tanggal pemesanan dan juga *slot* jam yang diinginkan. Maka *user* akan diarahkan ke detail pembayaran di mana sudah ditampilkan data pemesanan dan total harga. Lalu setelah melakukan pembayaran, data akan di-input ke *Firestore* dan pada Daftar Pesanan akan ditampilkan pesanan yang sudah dibuat oleh *user* berdasarkan *ID user* yang sudah ter-login di aplikasi saat itu.



Gambar 3.16. Halaman khusus *admin* untuk menambahkan *venue* olahraga baru

Gambar 3.16 menunjukkan halaman *admin* yang berfungsi untuk melakukan penambahan *venue* dari aplikasi yang nantinya akan di-input ke dalam *firebase* untuk penambahan *venue* yang praktis. Misal *admin* ingin mengambil gambar *venue* secara langsung, *admin* juga bisa mengaktifkan fitur *native* dari *Android* yaitu untuk membuka kamera dan melakukan pemotretan untuk *venue* langsung.

3.8 Kendala dan Solusi yang Ditemukan

3.8.1 Kendala

Beberapa kendala yang muncul selama proses pengerjaan proyek, termasuk kendala umum di divisi IT, adalah sebagai berikut:

1. Keterbatasan pada sisi *hardware*. Perangkat yang digunakan memiliki spesifikasi yang kurang memadai untuk menjalankan emulator *Android Device*, khususnya saat menggunakan *Android Studio* yang membutuhkan sumber daya tinggi.
2. Perubahan kebutuhan pengguna (*user requirement*) yang terjadi secara berulang. Perubahan ini seringkali muncul di tengah proses pengembangan,

sehingga memengaruhi alur kerja yang telah direncanakan sebelumnya.

3. Kurangnya komunikasi lintas tim. Koordinasi antara tim pengembang, manajemen, dan pengguna seringkali tidak sinkron, yang menyebabkan miskomunikasi terkait prioritas dan kebutuhan sistem.
4. Gangguan pada infrastruktur jaringan. Koneksi internet atau server internal yang tidak stabil dapat menghambat proses sinkronisasi, pengujian sistem, dan akses ke layanan berbasis cloud.

3.8.2 Solusi

Untuk mengatasi kendala-kendala tersebut, solusi yang diterapkan antara lain:

1. Memanfaatkan fitur *remote desktop*, yang memungkinkan akses dan kendali terhadap komputer lain dengan spesifikasi lebih tinggi dari jarak jauh untuk menjalankan emulator dan proses berat lainnya.
2. Meminta konfirmasi tertulis dan dokumentasi resmi atas kebutuhan akhir sebelum setiap fase proyek dimulai. Langkah ini bertujuan untuk meminimalkan perubahan yang tidak direncanakan serta memastikan seluruh pihak memiliki pemahaman yang selaras mengenai ruang lingkup dan tujuan proyek.
3. Mengadakan pertemuan rutin lintas tim untuk menyamakan pemahaman, menyusun prioritas, dan mempercepat pengambilan keputusan yang berkaitan dengan proyek.
4. Menyediakan cadangan koneksi internet dan memastikan infrastruktur jaringan memiliki tingkat ketersediaan (*uptime*) tinggi agar proses pengembangan dan pengujian tidak terganggu.