

SISTEM INFORMASI BOOKING LAPANGAN - BookMyField

Mata Kuliah: Pemrograman Web Lanjut



Anggota Kelompok:

Qhaulan Syaqhila (F1D022152)

Izzat Nazhiefa (F1D02310114)

M. Wahyu Hilal Abroor (F1D02310123)

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MATARAM

2025

BAB I – DESKRIPSI PROJECT

1. Latar Belakang

Olahraga adalah kegiatan yang dilakukan dengan tujuan tertentu, seperti melatih tubuh agar tetap sehat secara jasmani maupun rohani. Dengan rutin berolahraga, seseorang dapat memperoleh manfaat untuk menjaga kesehatan tubuh. Masyarakat bisa melakukan olahraga di mana saja, tetapi beberapa jenis olahraga memerlukan tempat atau fasilitas khusus yang sesuai dengan kebutuhannya. Contoh olahraga yang membutuhkan sarana khusus antara lain basket, voli, futsal, dan badminton. Beberapa jenis olahraga membutuhkan tempat khusus karena memerlukan fasilitas pendukung, seperti ring, net, gawang, dan lainnya. Untuk memenuhi kebutuhan tersebut, banyak orang memilih menyewa lapangan. Berdasarkan survei yang dilakukan melalui kuesioner terhadap 94 responden, terdiri dari 53,8% mahasiswa, 20,4% siswa SMA, 17,2% karyawan, dan 1,1% lainnya, diketahui bahwa responden pernah menyewa lapangan olahraga. Lapangan yang disewa antara lain futsal, voli, basket, dan badminton. Para responden juga menginginkan adanya inovasi berupa sistem penyewaan lapangan olahraga secara online agar lebih mudah mendapatkan informasi mengenai lapangan yang tersedia dan dapat mempermudah serta mengefisienkan proses penyewaan[1].

Oleh karena itu, responden perlu menyewa lapangan untuk melakukan kegiatan olahraga. Saat ini, sistem pemesanan lapangan sebagian besar masih dilakukan secara manual, yaitu pengguna harus datang langsung ke tempat penyewaan atau menghubungi lewat telepon untuk mengecek ketersediaan lapangan. Namun, pemesanan melalui telepon sering menghadapi kendala terkait validasi yang akurat, sehingga pihak penyedia lapangan tidak selalu menerima pemesanan dengan cara ini. Sistem manual ini cukup merepotkan bagi pengguna karena kurang efisien dari segi waktu, tenaga, dan biaya, karena mereka harus mengecek jadwal dan memesan lapangan satu per satu.

Berdasarkan kondisi tersebut, diperlukan otomatisasi sistem dengan merancang dan membuat aplikasi pemesanan lapangan olahraga berbasis web. Sistem ini diharapkan dapat memberikan validasi yang akurat terkait penjadwalan dan pemesanan lapangan, mengefisienkan waktu, serta mengurangi biaya transportasi. Selain itu, pemesanan berbasis web akan memudahkan pengguna karena dapat diakses melalui berbagai perangkat.

2. Tujuan *Project*

Adapun tujuan dari proyek ini adalah sebagai berikut:

- a. Proyek ini bertujuan untuk menggantikan metode pemesanan lapangan yang masih dilakukan secara manual dengan sistem berbasis web, sehingga seluruh proses menjadi lebih terstruktur, dan mudah dikelola.
- b. Sistem ini diharapkan dapat memberikan kemudahan bagi pengguna untuk memperoleh informasi terkait ketersediaan lapangan secara cepat dan *real-time*, sehingga pengguna tidak perlu lagi melakukan pengecekan secara langsung atau melalui telepon.
- c. Dengan adanya sistem berbasis web, proses pemesanan lapangan dapat dilakukan dengan lebih efisien, baik dari segi waktu, tenaga, maupun biaya, sehingga pengguna dan pengelola lapangan dapat memanfaatkan sumber daya secara optimal.
- d. Sistem ini dirancang untuk memberikan validasi yang tepat terhadap penjadwalan dan pemesanan lapangan, sehingga meminimalkan risiko kesalahan atau bentrok jadwal, serta meningkatkan kepercayaan pengguna terhadap layanan.
- e. Pengguna dapat melakukan pemesanan kapan saja dan dari mana saja melalui berbagai perangkat, seperti komputer, tablet, atau smartphone, sehingga sistem ini mampu mendukung kebutuhan fleksibilitas dan kenyamanan bagi seluruh pengguna.

3. Teknologi yang Digunakan (*Backend, Frontend, Database*)

a. *Backend*:

1) Node.js dan Express.js

Node.js digunakan untuk *runtime* JavaScript di sisi server dan Express.js digunakan sebagai *framework* untuk membangun REST API dengan arsitektur MVC.

2) JWT (JSON Web Token)

JWT digunakan untuk autentikasi dan otorisasi pengguna agar akses API lebih aman.

3) bcrypt

bcrypt digunakan untuk mengenkripsi (hash) kata sandi pengguna sebelum disimpan ke dalam database, sehingga meningkatkan keamanan data pengguna.

4) `dotenv`

`dotenv` digunakan untuk mengatur variabel lingkungan, seperti konfigurasi database dan secret key JWT, sehingga tidak perlu dituliskan secara langsung di dalam kode program.

b. *Frontend*:

1) `Vue.js 3`

`Vue 3` digunakan untuk membuat aplikasi Single Page Application (SPA) yang interaktif.

2) `Vue Router`

`Vue Router` digunakan untuk mengatur navigasi antar halaman, seperti halaman login, dashboard pengguna, dan halaman *booking* lapangan.

3) `Axios`

`Axios` digunakan sebagai library HTTP *client* yang berfungsi menghubungkan *frontend* dengan *backend* melalui API, sehingga proses pengiriman dan penerimaan data dapat dilakukan secara *real-time*.

4) `Vite`

`Vite` digunakan sebagai alat *build* sekaligus *development server* yang mendukung proses pengembangan yang cepat dan efisien, sehingga mempermudah kegiatan pengujian serta pengembangan antarmuka pengguna.

c. *Database*:

1) `MySQL`

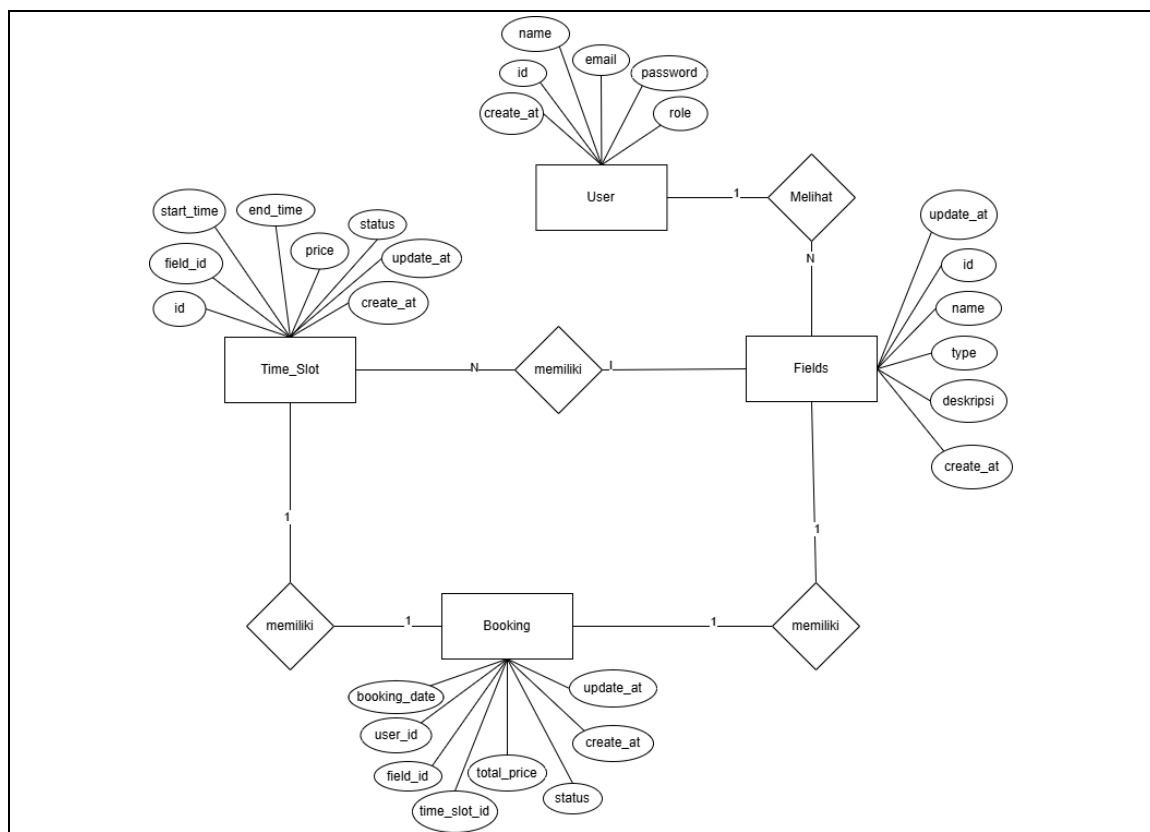
`MySQL` digunakan sebagai database relasional yang berfungsi menampung dan mengelola berbagai data penting dalam sistem, seperti data pengguna, informasi lapangan, dan data booking. Dengan penerapan struktur tabel yang sistematis serta dukungan perintah SQL, `MySQL` membantu menjaga keakuratan dan konsistensi data serta meningkatkan efisiensi dalam proses pengolahan dan penyimpanan data aplikasi.

BAB II – PERANCANGAN SISTEM

1. Deskripsi Singkat Sistem

Sistem Manajemen Booking Lapangan berbasis web ini merupakan aplikasi yang memungkinkan pengguna untuk melakukan pemesanan lapangan olahraga secara online dengan mudah dan cepat. Sistem ini menyediakan dua peran utama, yaitu Admin yang dapat mengelola data lapangan dan memantau seluruh booking, serta User yang dapat melakukan booking dan melihat riwayat pemesanan.

2. Desain Database (ERD dan penjelasan singkat tabel)



Penjelasan Tabel:

1. Users

Tabel ini digunakan untuk menyimpan data pengguna, baik itu user maupun admin. Tabel ini berisi kolom id, name, email, password, dan role (admin/user). Role disini digunakan untuk menentukan hak akses di sistem (misalnya admin bisa menambah dan menghapus lapangan).

2. Fields

Tabel ini digunakan untuk menyimpan data lapangan yang tersedia untuk booking. Tabel ini berisi kolom id, name, type, deskripsi, price, dan lokasi. Tabel ini juga digunakan untuk menampilkan informasi lapangan pada frontend.

3. Time_Slot

Tabel ini digunakan untuk menyimpan data jam atau slot waktu yang tersedia di lapangan. Tabel ini berisi kolom id, start_time, dan end_time. Tabel ini juga digunakan untuk menentukan kapan lapangan bisa di pesan.

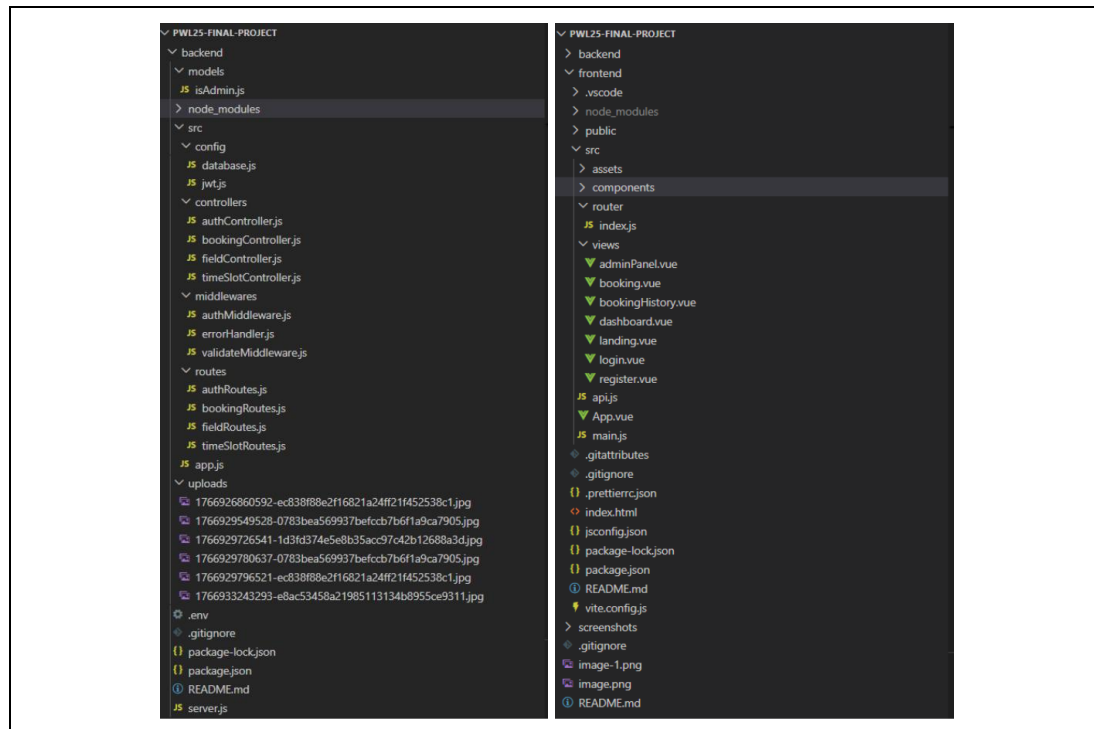
4. Bookings

Tabel ini digunakan untuk menyimpan data pemesanan yang dilakukan pengguna. Tabel ini berisi kolom id, user_id, field_id, time_slot_id, booking_date, dan status. Status bisa berupa aktif atau selesai untuk menandai proses booking.

BAB III – IMPLEMENTASI

1. Implementasi *Backend* (struktur folder, API, middleware)

a. Struktur folder



Dari gambar di atas dapat dilihat struktur folder *backend* dan *frontend* pada sistem informasi *booking* lapangan, pada struktur folder backend disesuaikan dengan struktur MVC begitupun *frontend* juga memiliki struktur folder yang mirip dengan MVC.

b. API

Adapun daftar API *endpoint* adalah sebagai berikut:

1) Auth (/api/auth)

Method	Endpoint	Keterangan
POST	/register	Register user baru
POST	/login	Login user dan dapat token

2) Fields (/api/fields)

Method	Endpoint	Keterangan
GET	/	Ambil semua lapangan
POST	/	Tambah lapangan (Admin)
PUT	/:id	Update lapangan (Admin)
DELETE	/:id	Hapus lapangan (Admin)

3) Bookings (/api/bookings)

Method	Endpoint	Keterangan
GET	/	Ambil semua booking
POST	/	Booking baru (user)
PUT	/:id/status	Update status booking (Admin)
GET	/fields	Ambil daftar lapangan untuk booking

GET	/time-slots	Ambil daftar time slot tersedia
GET	/user	Riwayat booking user
PUT	/:id/complete	Tandai booking selesai & buka slot (Admin)

4) Time Slots (/api/time-slots)

Method	Endpoint	Keterangan
GET	/	Ambil semua time slot
POST	/	Tambah time slot (Admin)
PUT	/:id	Update time slot (Admin)
DELETE	/:id	Hapus time slot (Admin)

c. Middleware

Untuk meningkatkan keamanan API, digunakan *middleware* autentikasi berbasis JSON Web Token (JWT). Middleware ini bertugas memvalidasi token yang dikirim oleh klien melalui header *Authorization* sebelum memberikan izin akses ke *endpoint* yang bersifat terbatas.

1) authMiddleware.js

- verifyToken digunakan untuk memeriksa apakah request membawa JWT yang valid
- isAdmin digunakan untuk memeriksa apakah user memiliki role admin

2) validateMiddleware.js

Validasi input body sesuai field yang wajib diisi, berikut implementasinya:

```
router.post('/', verifyToken, isAdmin, validate(['name', 'type']), addField)
```

Dari potongan kode tersebut dapat disimpulkan bahwa akses ke *route* dibatasi hanya untuk pengguna yang telah login dengan peran admin. Selain itu, request yang dikirimkan wajib menyertakan data *name* dan *type* pada body. Jika salah satu syarat tersebut tidak terpenuhi, maka akses ke *route* tidak akan diizinkan.

2. Implementasi *Frontend* (halaman, *routing*, konsumsi API)

Frontend dibangun menggunakan Vue 3 (Composition API) dan berfungsi sebagai antarmuka pengguna untuk melakukan *booking* lapangan, melihat riwayat *booking*, serta panel admin untuk pengelolaan data.

a. Halaman

- Halaman dashboard publik
- Halaman login dan registrasi
- Halaman dashboard awal setelah login

- 4) Halaman *booking* lapangan
- 5) Halaman *view history booking*
- 6) Halaman dashboard admin yang terdiri dari CRUD Lapangan, *Time slot* dan *booking*

b. *Routing*

Routing menggunakan Vue Router untuk navigasi antar halaman. Berikut ini adalah kode penggunaan *routing*.

```
const routes = [  
  { path: '/', component: Landing }, // Landing bebas akses  
  { path: '/login', component: Login },  
  { path: '/register', component: Register },  
  { path: '/dashboard', component: Dashboard, meta: {  
    requiresAuth: true } },  
  { path: '/booking', component: Booking, meta: { requiresAuth:  
    true } },  
  { path: '/admin', component: AdminPanel, meta: { requiresAuth:  
    true, requiresAdmin: true } },  
  { path: '/booking-history', component: BookingHistory, meta: {  
    requiresAuth: true } }  
]
```

Kode routes tersebut adalah konfigurasi *routing* Vue Router yang menentukan halaman apa (*component*) yang ditampilkan berdasarkan URL (path). Halaman seperti /, /login, dan /register bisa diakses bebas tanpa login, sedangkan /dashboard, /booking, dan /booking-history hanya bisa diakses oleh *user* yang sudah login karena memiliki “meta.requiresAuth”. Khusus halaman /admin, selain harus login juga membutuhkan hak akses admin (meta.requiresAdmin), sehingga user biasa tidak dapat membukanya. Properti meta digunakan sebagai penanda tambahan yang kemudian dicek pada *navigation guard* untuk membatasi akses halaman sesuai status login dan *role* pengguna.

c. Konsumsi API

Komunikasi data antara frontend dan backend dilakukan menggunakan library Axios. Berikut contoh implemtnasi API untuk *request update* status *booking*.

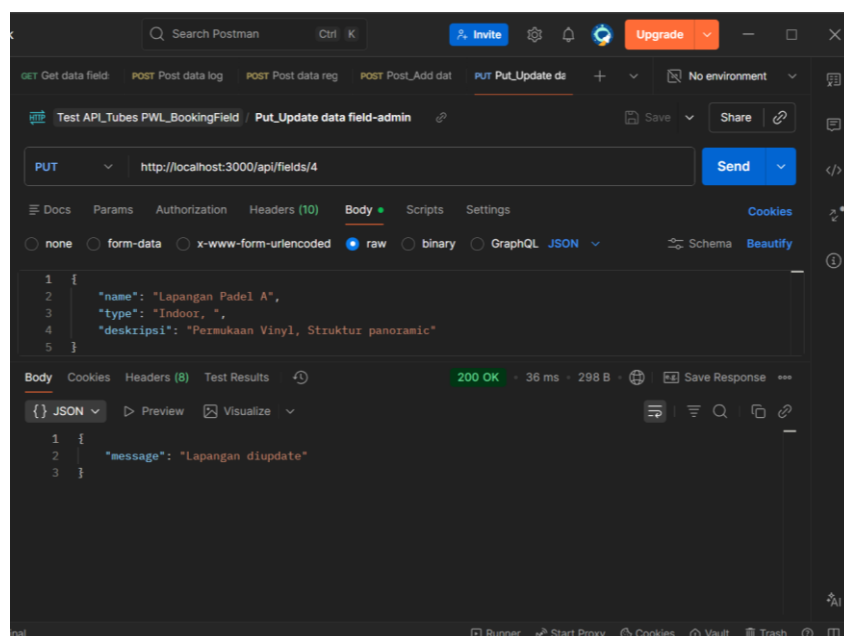
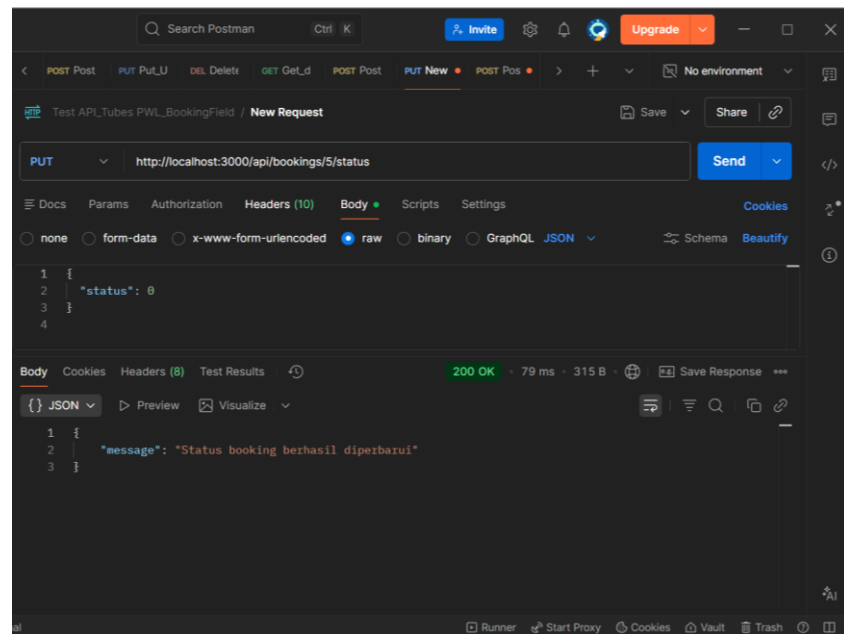
```
async function completeBooking(booking) {  
  if (!confirm('Tandai booking ini sebagai selesai?')) return  
  const res = await  
  fetch(`http://localhost:3000/api/bookings/${booking.id}/status`, {  
    method: 'PUT',  
    headers: {  
      'Content-Type': 'application/json',  
      'Authorization': 'Bearer ' + token  
    },  
    body: JSON.stringify({ status: 1 }) // 1 = complete  
  })  
  const data = await res.json()  
  alert(data.message)
```

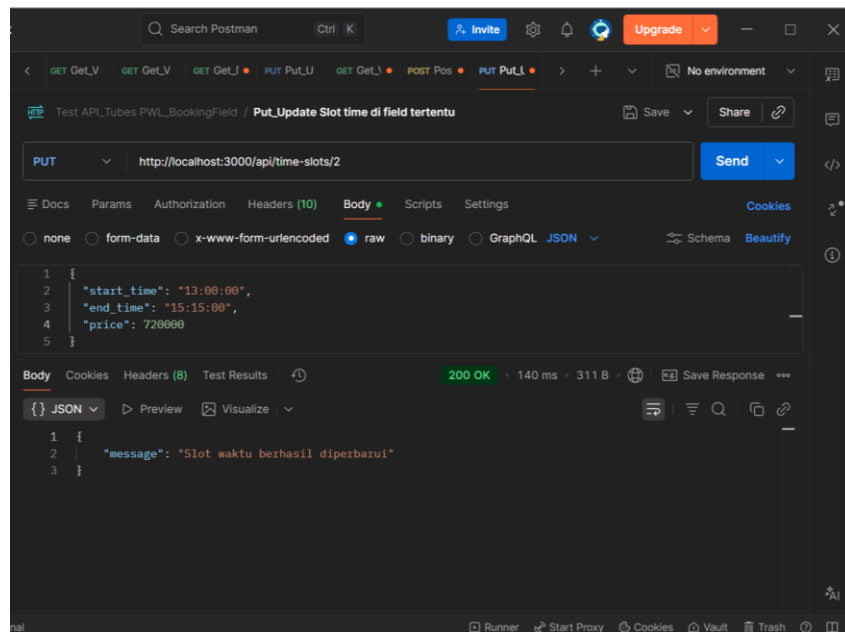
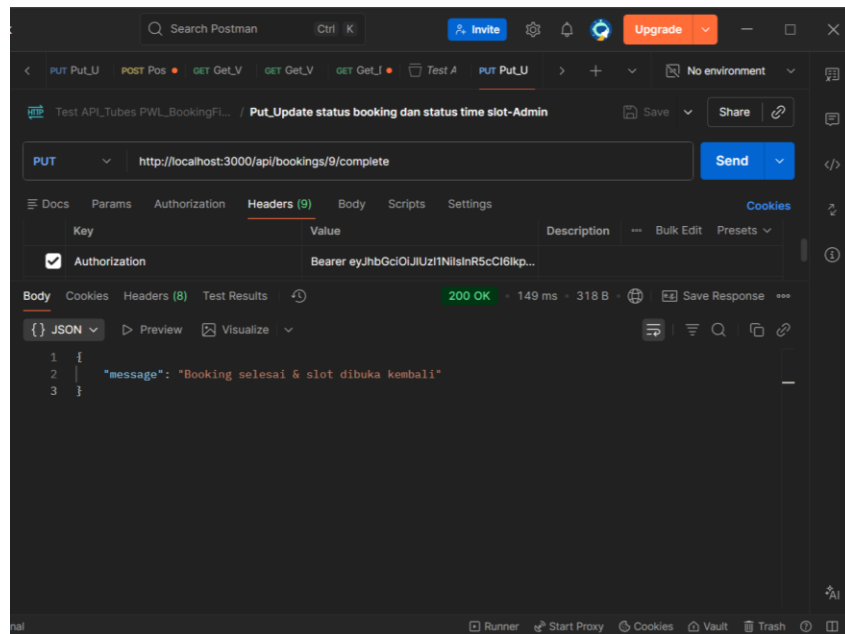
```
loadBookings()  
loadTimeSlots()  
}
```

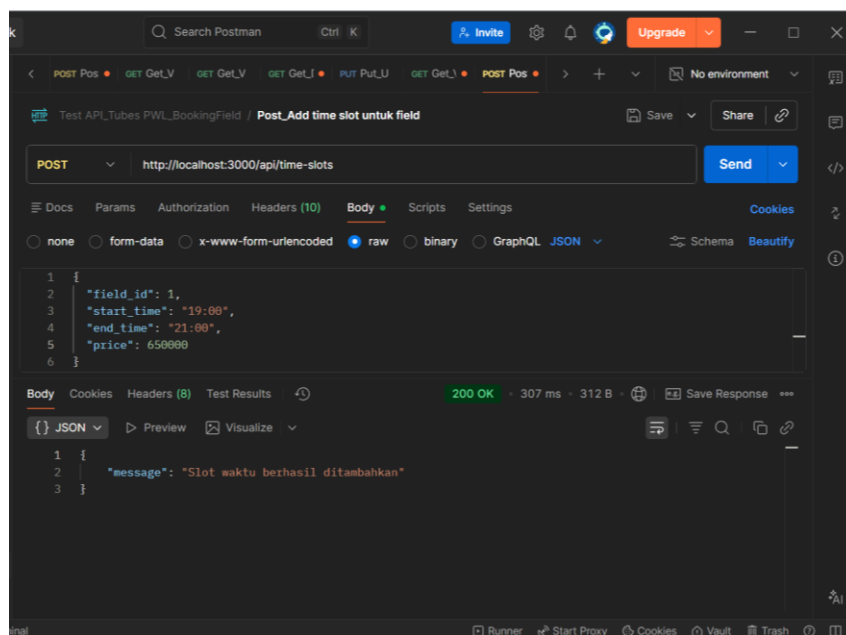
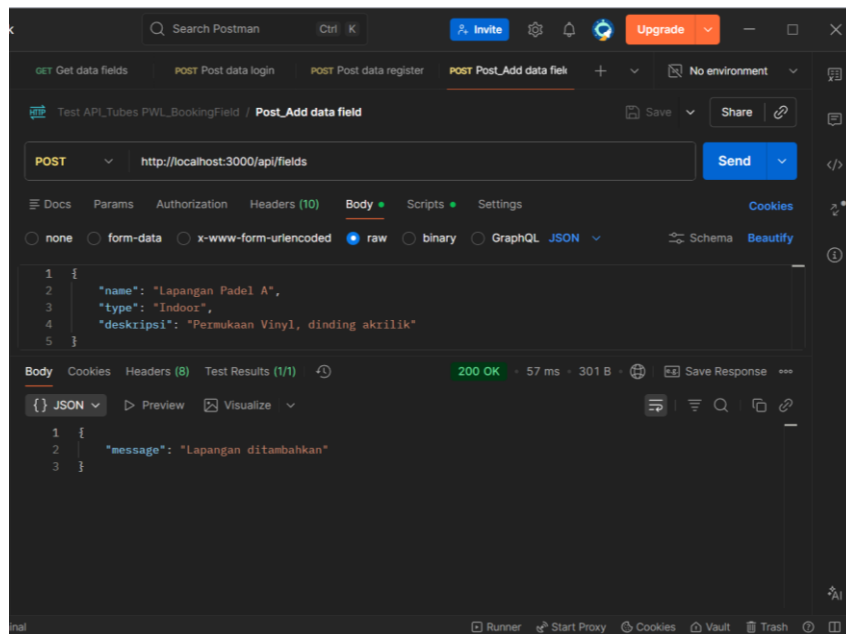
Fungsi `completeBooking` ini digunakan untuk mengubah status *booking* melalui *backend*. Ketika admin menekan tombol, sistem terlebih dahulu menampilkan konfirmasi agar tidak terjadi perubahan tidak sengaja. Jika disetujui, *frontend* mengirim request PUT ke *endpoint backend* dengan menyertakan “`booking.id`” pada URL dan token autentikasi pada *header* untuk memastikan hanya pengguna yang berwenang yang dapat mengaksesnya. Data status: 1 dikirim dalam body sebagai penanda bahwa booking telah selesai. Setelah *backend* memproses permintaan dan mengembalikan respons, *frontend* menampilkan pesan hasil proses dan memuat ulang data booking serta time slot agar tampilan langsung mencerminkan perubahan status terbaru.

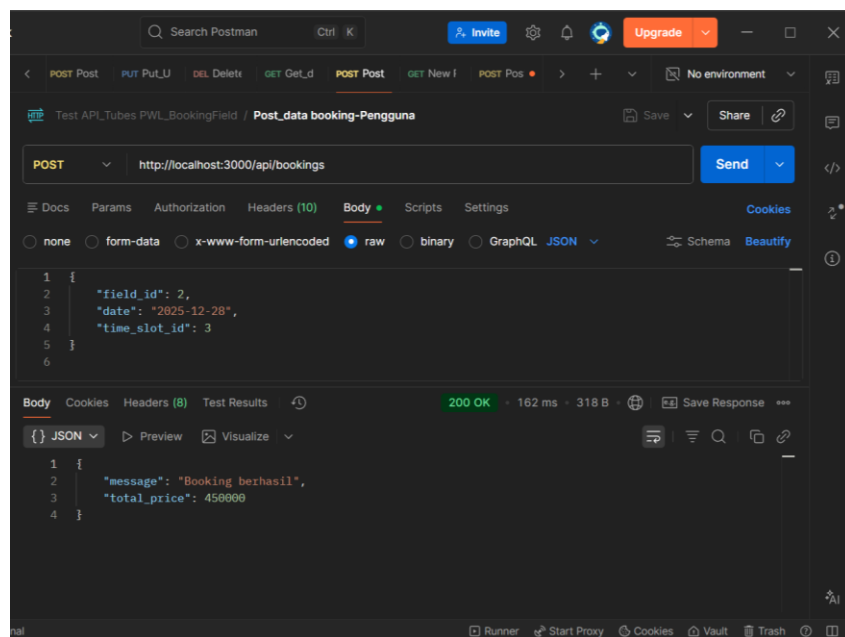
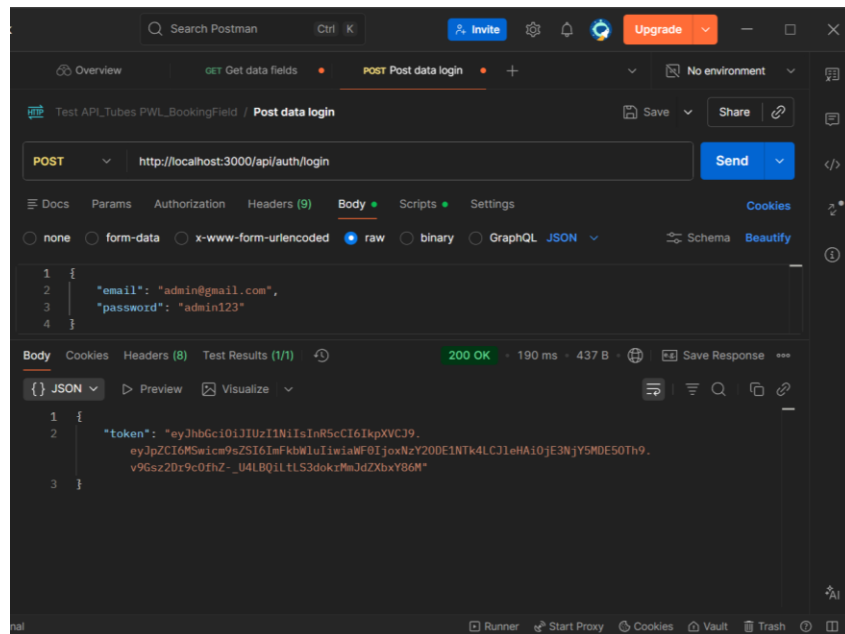
BAB IV – PENGUJIAN

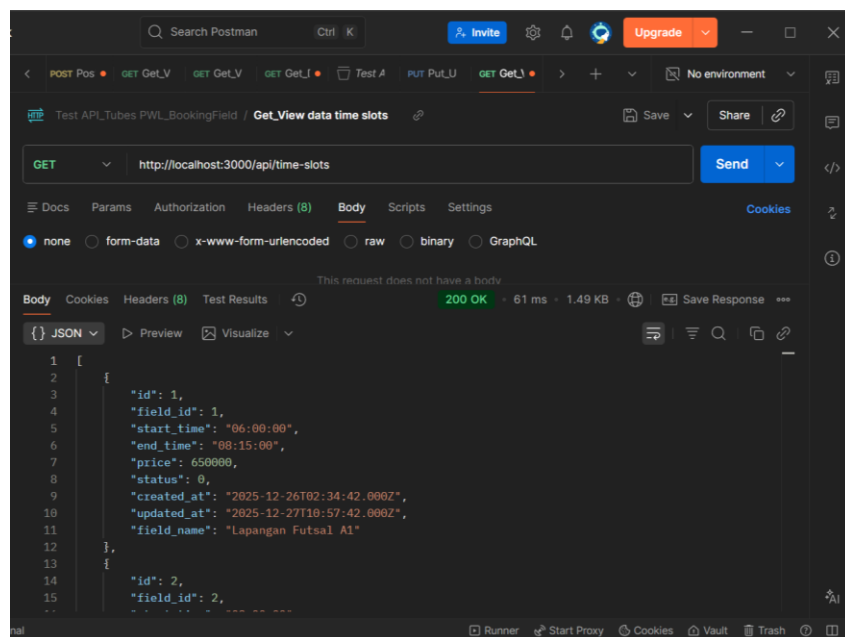
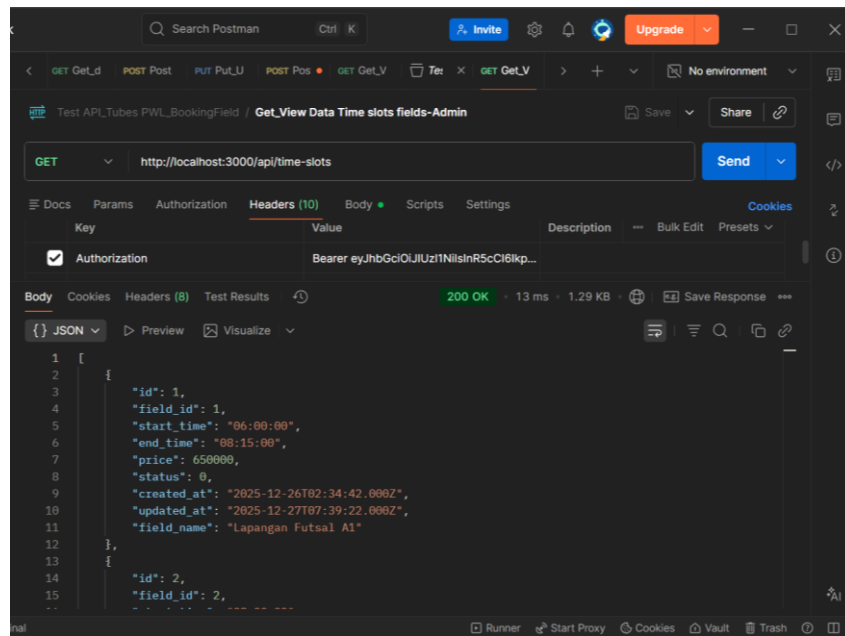
1. Pengujian API (Postman/Thunder Client)

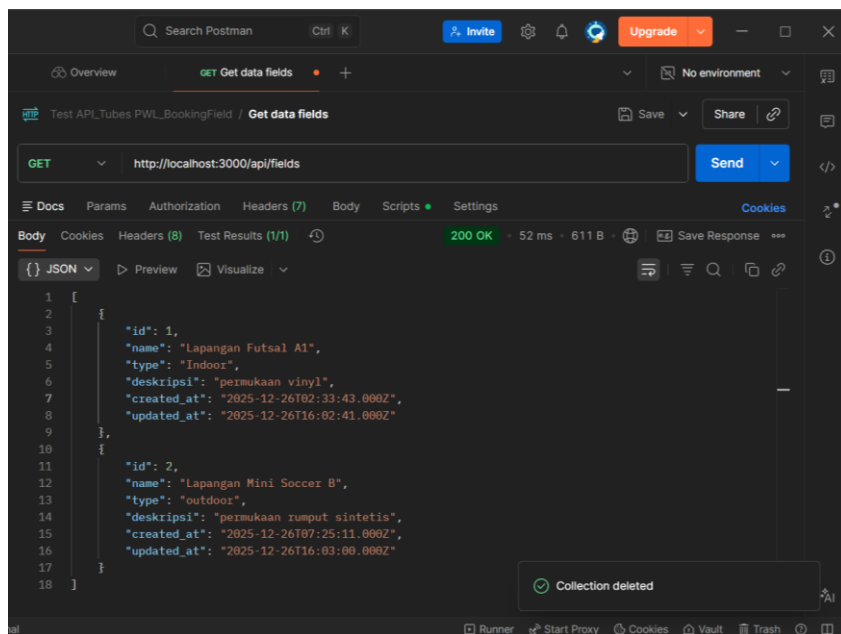
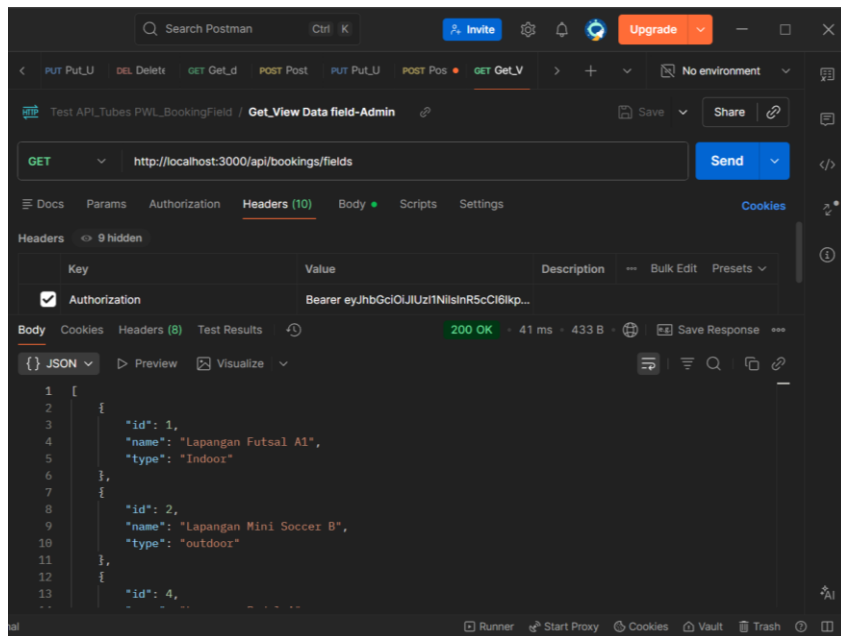


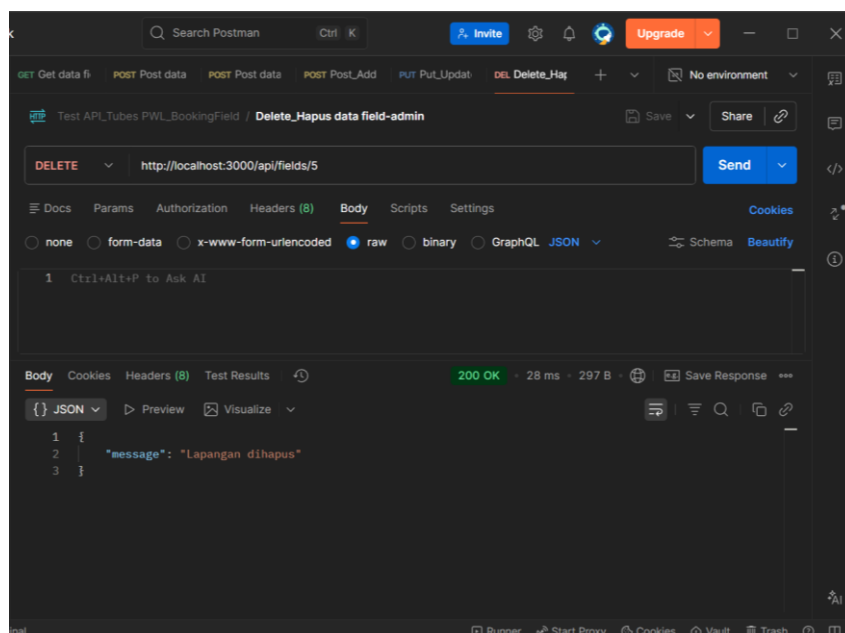
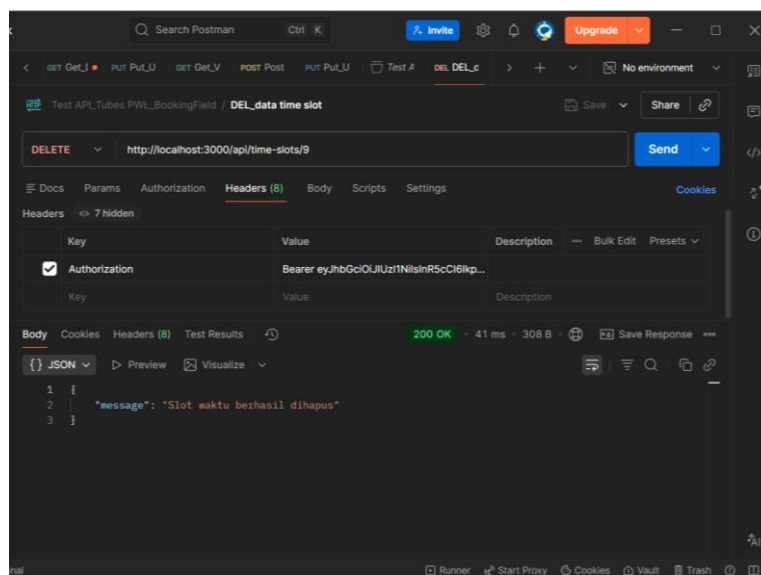
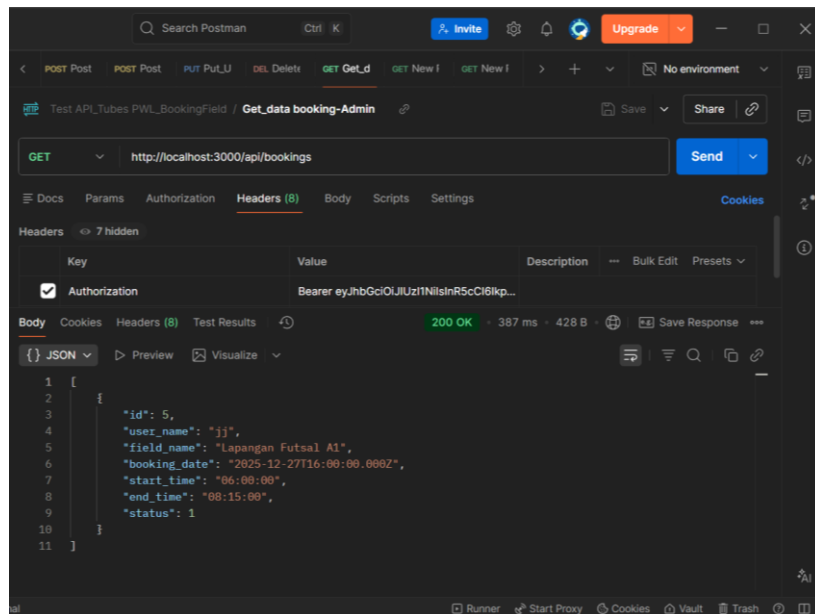






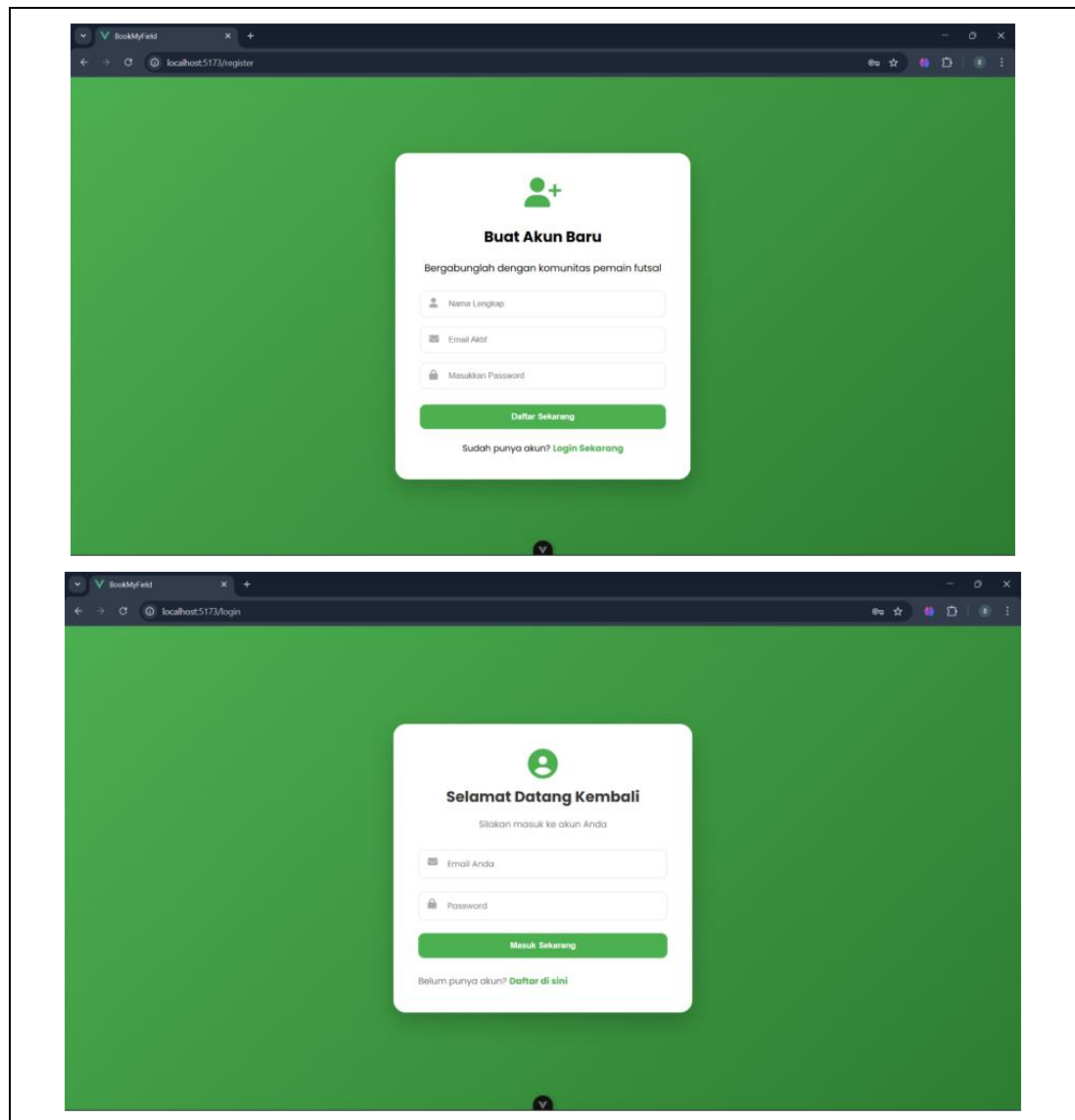






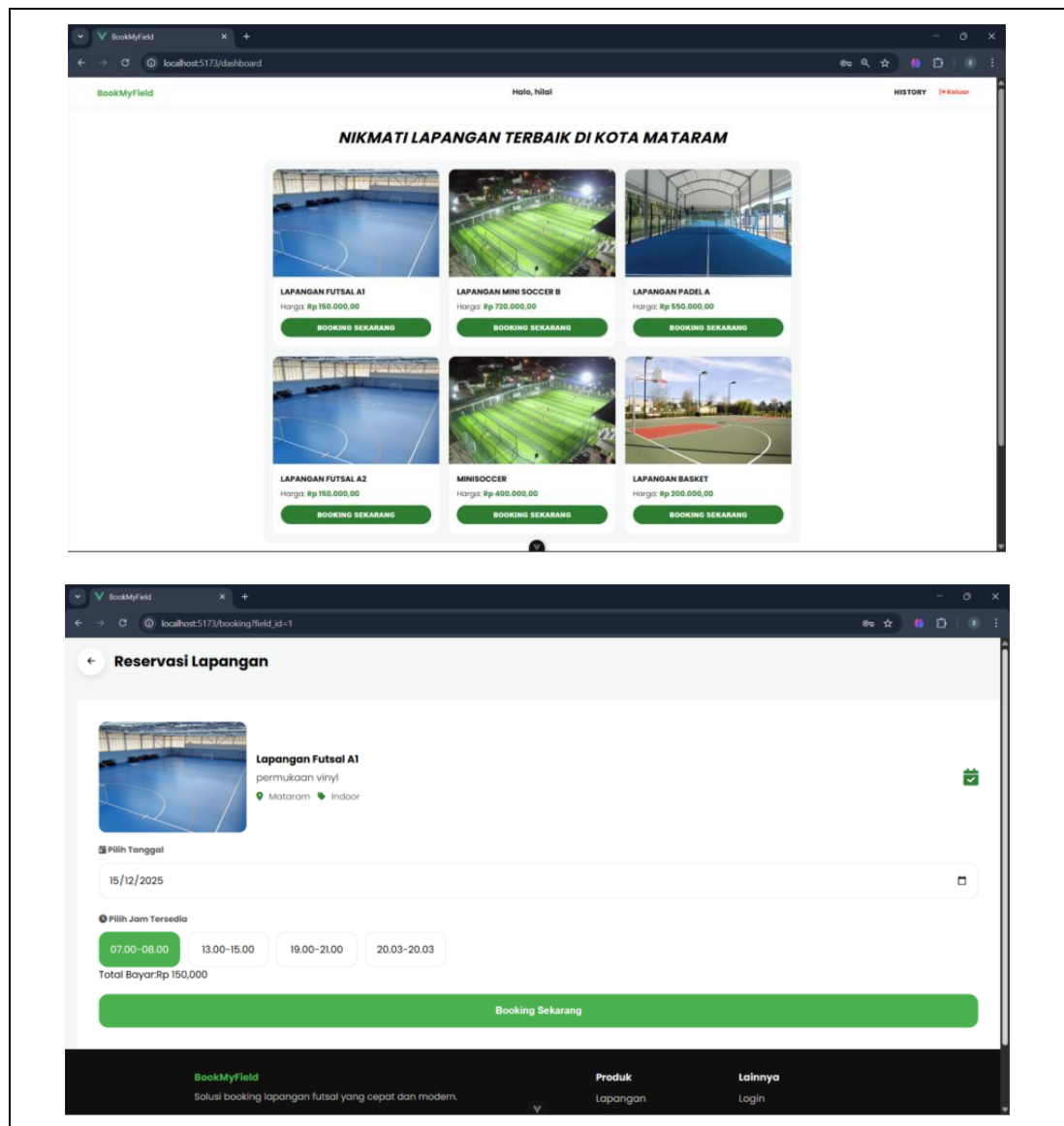
2. Pengujian Aplikasi (screenshot fitur utama)

a. Halaman Login dan Registrasi



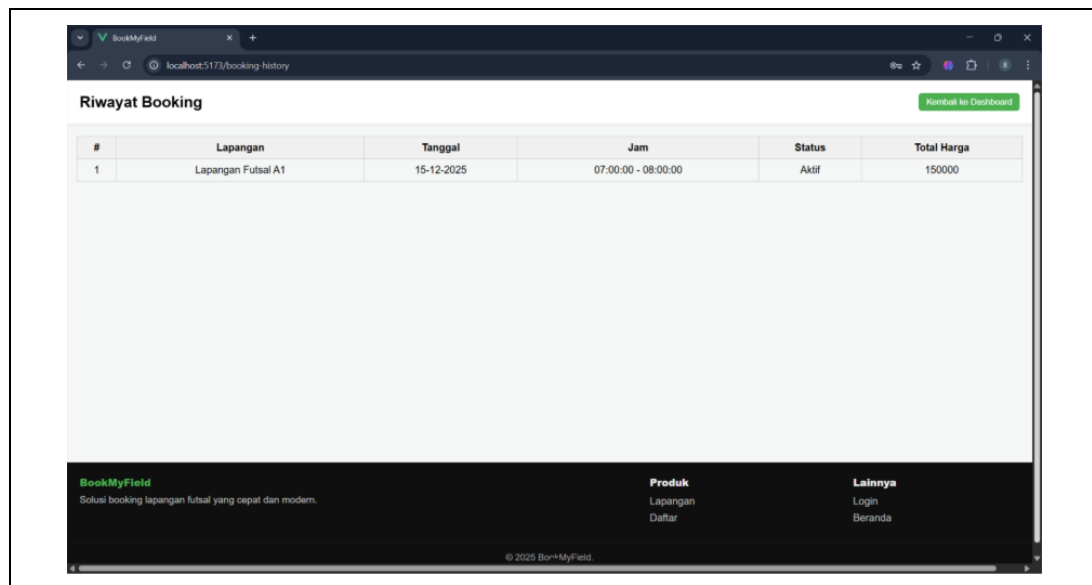
Dari gambar di atas dapat diketahui bahwa sebelum masuk ke dalam website, pengguna perlu melakukan registrasi jika belum memiliki akun, setelah melakukan registrasi pengguna perlu melakukan login untuk masuk ke dalam website

b. Booking Lapangan



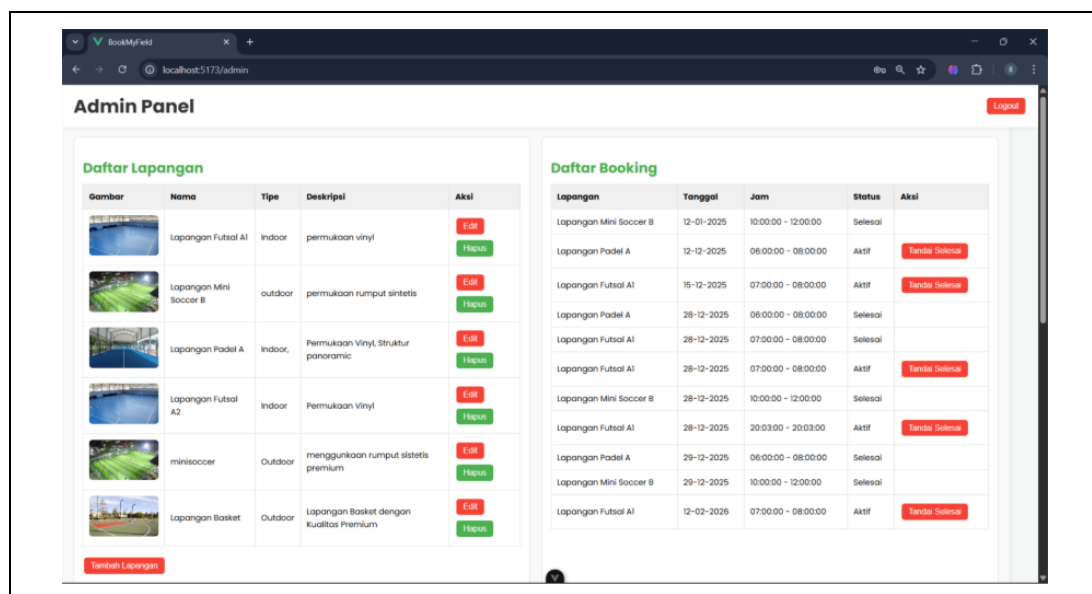
Dari gambar di atas dapat diketahui bahwa setelah berhasil login, pengguna masuk ke dalam halaman awal, di halaman tersebut pengguna dapat melihat lapangan yang tersedia, pengguna dapat melakukan booking dengan klik button booking, setelah klik button tersebut pengguna akan masuk ke halaman booking yang berisi detail booking yang dapat diisi oleh pengguna.

c. History Booking



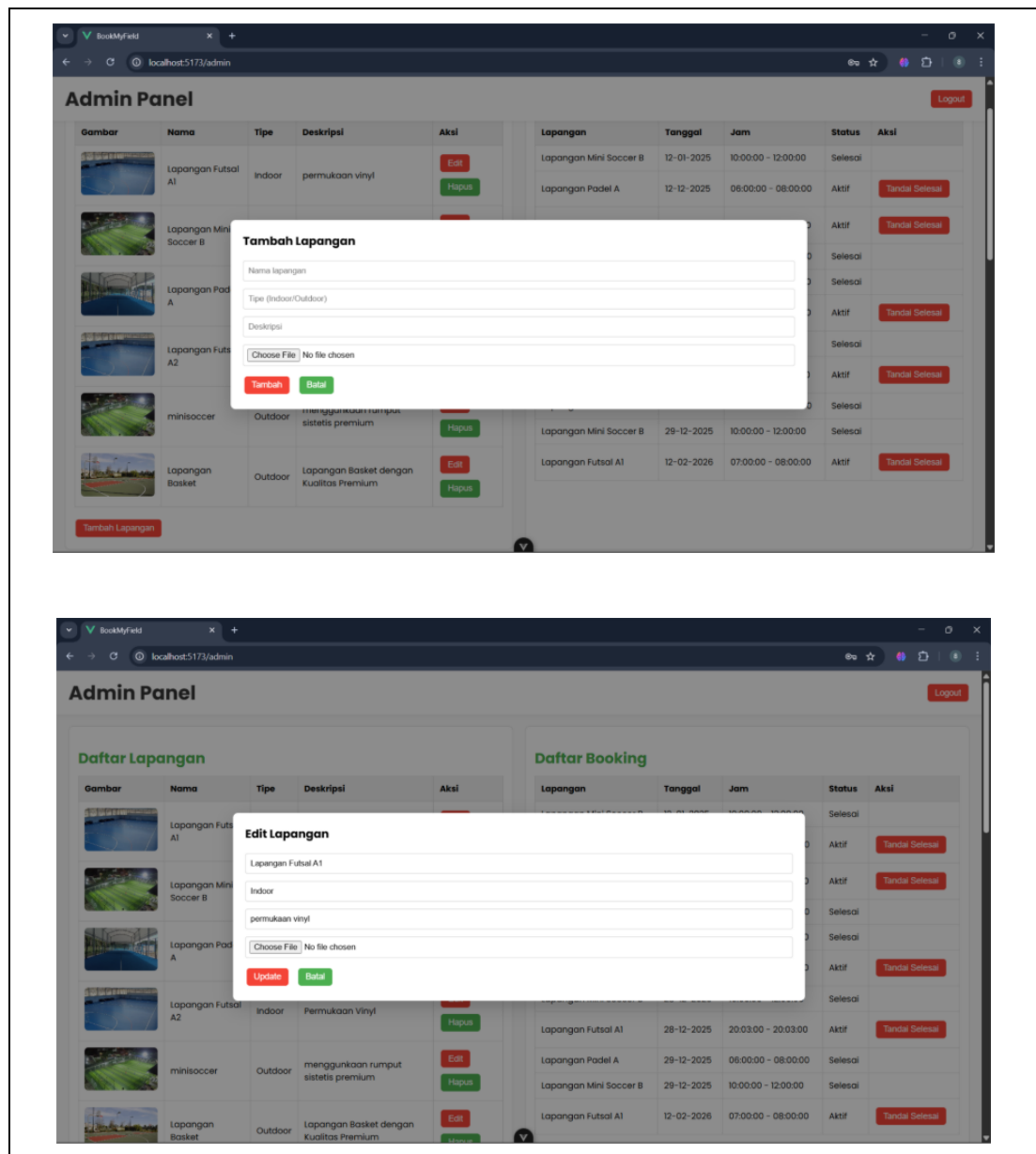
Berdasarkan gambar, sistem menyediakan fitur history booking yang memungkinkan pengguna melihat riwayat pemesanan sebelumnya, termasuk detail lapangan, tanggal, waktu, dan status booking. Fitur ini memudahkan pemantauan aktivitas dan pengecekan ulang pemesanan.

d. Halaman Admin Panel

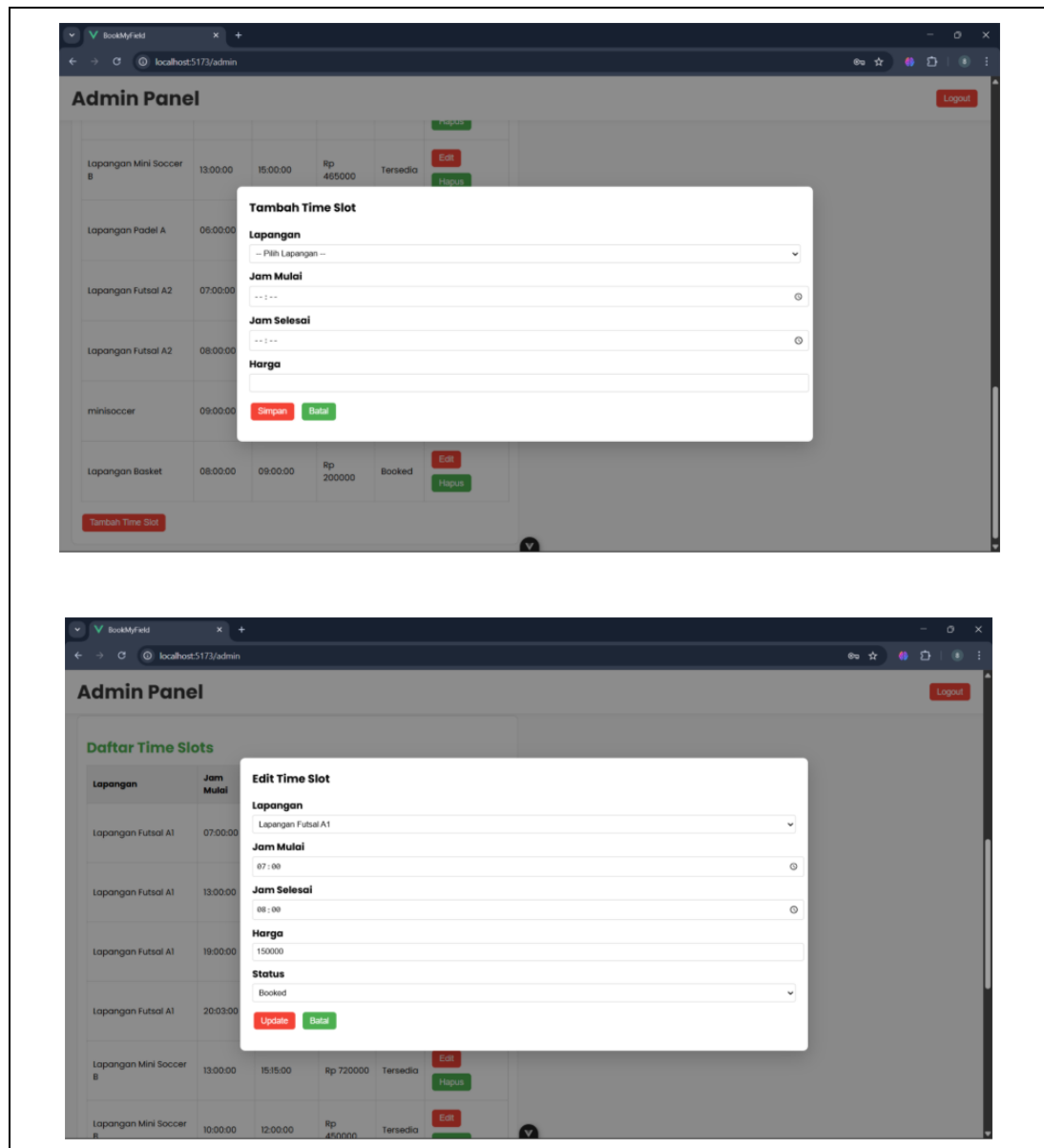


Dari gambar di atas dapat diketahui bahwa setelah admin login, terlihat admin panel seperti pada gambar tersebut yang didalamnya berisi detail informasi dari website.

e. Admin CRUD Lapangan

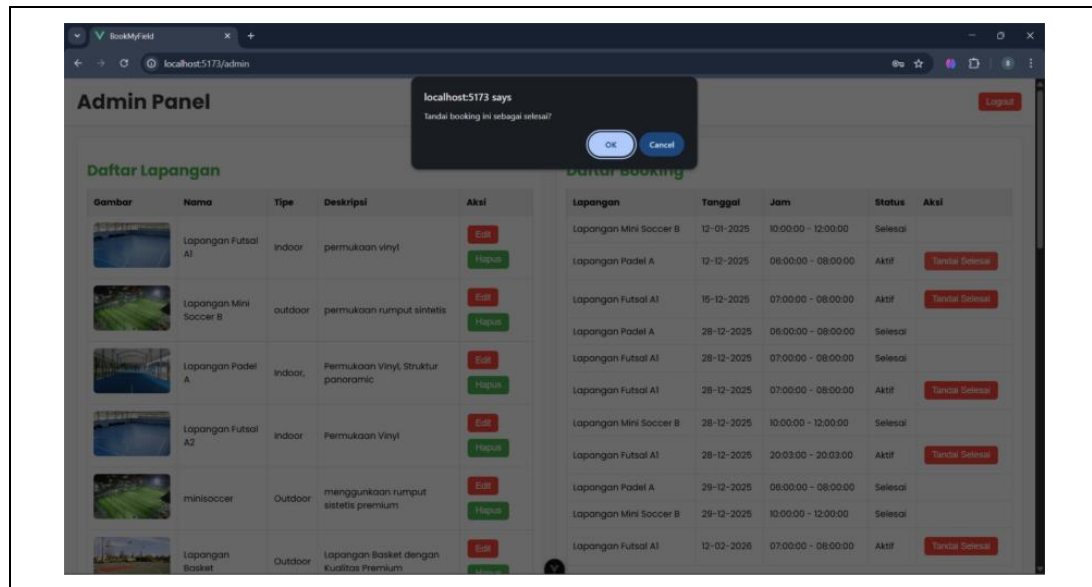


Berdasarkan gambar tersebut, dapat disimpulkan bahwa pengguna dengan peran admin memiliki hak akses penuh untuk mengelola data lapangan. Admin dapat melakukan seluruh operasi CRUD, yaitu membuat data lapangan baru, melihat dan menampilkan data yang ada, memperbarui informasi lapangan, serta menghapus data lapangan sesuai kebutuhan sistem. Admin CRUD Time Slot



Berdasarkan sistem yang ditunjukkan, admin memiliki kemampuan untuk mengelola data timeslot secara penuh melalui operasi CRUD. Admin dapat membuat timeslot baru sesuai jadwal yang tersedia, membaca atau menampilkan daftar timeslot yang ada, memperbarui informasi timeslot jika terjadi perubahan waktu atau status, serta menghapus timeslot yang tidak lagi digunakan. Dengan demikian, admin dapat memastikan pengelolaan timeslot berjalan tertib dan sesuai kebutuhan operasional.

f. Admin Edit Status Booking



Berdasarkan sistem yang ditunjukkan, admin memiliki kemampuan untuk mengubah **status booking** yang telah dibuat oleh pengguna. Hal ini memungkinkan admin untuk memperbarui status pemesanan dari “aktif” menjadi “selesai”, sehingga memudahkan pengelolaan dan pemantauan transaksi. Fitur ini berhubungan dengan lapangan, Ketika booking selesai time slot pada lapangan tersebut akan berubah menjadi tersedia yang mana sebelumnya masih berstatus booked.

LAMPIRAN

Link GitHub Repository: <https://github.com/hilalabroor/pwl25-final-project.git>

Link Deployment (jika ada):

Screenshot Tambahan:

- [1] H. Rasikhah and A. R. Adriansyah, “Jurnal Informatika Terpadu PERANCANGAN DAN IMPLEMENTASI BOOKING SYSTEM LAPANGAN MENGGUNAKAN FRAMEWORK MVC BERBASIS WEB,” *Jurnal Informatika Terpadu*, vol. 8, no. 1, pp. 8–12, [Online]. Available: <https://journal.nurulfikri.ac.id/index.php/JIT>