BAB III

PELAKSANAAN KERJA PROFESI

3.1 Bidang Kerja

Penulis melangsungkan kegiatan KP (Kerja Profesi) pada PT. Bumi Rejeki Agung, penulis menempati divisi IT. Selama pelaksanaan KP (Kerja Profesi) pada bagian IT, praktikan ditugaskan sebagai *back-end* untuk membuat rancangan aplikasi booking berbasis mobile. Dalam proses pelaksanaannya, praktikan mempunyai mentor atau pembimbing yang merupakan Direktur Operasinal pada PT. Bumi Rejeki Agung. Komunikasinya menggunakan aplikasi online seperti WA dan zoom meet (*video conference*) atau langsung dikantor sesuai berdasarkan arahan yang telah disepakati sebelumnya.

Berikut ini adalah tahapan-tahapan yang praktikan yang dilakukan praktikan selama pelaksanaan KP (Kerja Profesi):

- Melakukan *system requirement* menggunakan elisitasi tuntuk menentukan masalah dan kebutuhan yang dihadapi perusahaan.
- Melakukan analisa hasil dari *system requirement* dan mencari solusi terhadap permasalahan yang dihadapi sebagai bahan untuk merancang sebuah sistem yang baru.
- Mengkaji dan Melakukan tahap perancangan sistem yang dibuat dengan memakai OOAD (Object-oriented analysis and design) lalu dibangun memakai UML (Unified Modelling *Language*) dan terdiri dari *use case diagram*, *use case description*, *activity diagram*, *sequence diagram* dan *class diagram*. yang nantinya akan dikembangkan menjadi sebuah aplikasi sebagai pemecahan masalah.

3.2 Pelaksanaan Kerja

Praktikan melaksanakan kegiatan KP (Kerja Profesi) selama tiga bulan yang dilaksanakan sejak 6 Juni 2022 sampai dengan 6 September 2022 (400 jam atau 90 hari). Selama pelaksanaan KP (Kerja Profesi), praktikan ditempatkan pada bagian atau divisi IT untuk membantu perusahaan dalam mengatasi masalah yang dihadapi. Pada divisi IT ini praktikan bergabung kedalam tim *mobile application* sebagai *backend* yang tugasnya menganalisa dan merancang sistem untuk membantu dalam proses pembuatan aplikasi tersebut. Selain itu praktikan harus mempunyai ilmu pengetahuan mengenai analisa dan perancangan suatu sistem agar nantinya dapat memahami proses bisnisnya pada aplikasi booking properti berbasis mobile yang ingin dibuat.

Aplikasi booking properti berbasis mobile ini adalah aplikasi yang digunakan untuk membantu aktifitas yang terdapat pada PT. Bumi Rejeki Agung. Berikut merupakan tujuan utama pembuatan aplikasi ini adalah:

- Mempermudah tata kelola proses booking yang dilakukan oleh manajer.
- Mempermudah proses verifikasi pembayaran yang dilakukan oleh finance.
- Membantu direktur untuk melihat dan mengunduh data booking secara digital.

Terdapat empat menu utama pada aplikasi booking properti, yaitu Menu Kelola Booking, Menu Siteplan, Menu Verifikasi Pembayaran dan Menu rangkuman penjualan. Masing-masing dari keempat menu tersebut dibagi berdasarkan peran akses yang berada dalam perusahaan tersebut. Berikut adalah gambaran lengkap tentang fitur dalam aplikasi *booking property* berbasis mobile:

1. Menu Kelola Booking

   Pada menu ini hanya manajer mendapatkan akses untuk masuk dan bertugas melakukan pengelolaan seperti menambah, mengubah, menghapus, dan melihat data booking yang tujuan untuk membantu memudahkan pengelolaan data.

2. Menu Siteplan

   Pada menu ini hanya manajer mendapatkan akses untuk masuk dan melihat siteplan yang berfungsi untuk membantu mencari rumah atau properti yang dipesan.

3. Menu Verifikasi Pembayaran

   Pada menu ini hanya finance yang mendapatkan akses untuk masuk dan melakukan verifikasi pembayaran. Finance bertugas untuk melakukan pengecekan terhadap pembayaran yang masuk lalu melakukan melakukan verifikasi pada aplikasi tersebut.

4. Menu rangkuman Penjualan

   Pada menu ini hanya direktur yang mendapatkan akes untuk masuk, menu ini dibuat untuk direktur agar dapat mengecek data penjualan yang masuk lalu dapat mengunduh data penjualan tersebut menjadi sebuah laporan secara otomatis.

Dalam pembuatan aplikasi booking properti berbasis mobile ini memiliki beberapa tahapan yang dilakukan seperti pengenalan lingkungan kerja, berdiskusi dengan mentor KP dan tim untuk menentukan user requirement, tahapan perancangan dan tahap pengembangan.

3.2.1 Analisa dan Perancangan Sistem

Perancangaan awal dilakukan dengan menentukan *user requirement* atau kebutuhan pengguna. Untuk menentukannya disini praktikan bekerja sama dengan tim dan mentor KP untuk melakukan diskusi mengenai kebutuhan pengguna. Setelah menentukan user requirement atau kebutuhan pengguna, praktikan melakukan perancangan menggunakan metode OOAD (*Object-oriented analysis and design*) dibangun dengan UML (*Unified Modelling Language*) lalu menghasilkan elisitasi, *use case diagram*, *use case description*, *activity diagram*, *sequence diagram* dan *class diagram*. Praktikan berdiskusi dengan tim dan mentor KP untuk menentukan kebutuhan pengguna terkait dengan aplikasi booking properti berbasis mobile yang akan dibuat.

3.2.2 Elisitasi

Elisitasi adalah langkah pertama untuk menentukan kebutuhan perangkat lunak, proses identifikasi Persyaratan perangkat lunak berasal dari tempat yang berbeda Sumber, yaitu wawancara, lokakarya, alur kerja, dll. Analisis Tugas, Analisis Dokumen, dan Mekanisme yang lain (Iswari, 2012). Dalam melakukan tahapan elisitasi, praktikan menggunakan tiga tahapan dalam melakukan perancangannya, yaitu:

- Elisitasi Tahap Satu

  Adalah suatu perancangan usulan yang telah dikasih oleh terkait kepada *developer* atau pengembang, pada tahap ini proses pembuatannya dapat dilakukan dengan menggunakan metode wawancara.

- Elisitasi Tahap Dua

  Adalah hasil klasifikasi pada tahap satu dengan menggunakan teknik MDI. Teknik MDI ini mempunyai tujuan untuk memisah antara rancangan utama dan wajib ada didalam sistem terbaru melalui pembuatan desain yang

disepakati pengembang agar dilakukan eksekusi. Berikut merupakan gambaran dari MDI:

- M : *Mandatory* (penting), artinya prasyaratan ini harus ada dan tidak dapat dikecualikan begitu membangun sistem yang terbaru.
- D : *Disireable*, artinya persyaratan ini tidak penting dan dapat diabaikan. Namun, jika persyaratan ini diterapkan saat membangun sistem baru, sistem akan menjadi lebih sempurna.
- I : *Inessential,* artinya persyaratan ini berada di luar sistem yang dijelaskan, bukan bagian darinya.

- Elisitasi Tahap Tiga

adalah output pengurangan elisitasi tahap II yaitu dengan menghapus semua persyaratan dengan pilihan I (*Inessential)* dengan teknik MDI. lalu, semua persyaratan yang tersisa dikelaskan lagi dengan prosedur TOE, yaitu:

- *Technical* (T) : dengan apa metode yang digunakan mengenai sistem baru yang diusulkan?
- *Operational* (O) : bagaimana menggunakan persyaratan yang akan dibuat?
- *Economic* (E) : Berapa biaya untuk membangun sistem baru?

Prosedur TOE diabagi menjadi beberapa opsi atau pilihan, yaitu:

- *High* (H) : sulit untuk dilakukan.
- *Middle* (M) : mampu dilakukan.
- *Low* (L) : sederhana atau mudah dikerjakan.

- Elisitasi Tahap Akhir

  Adalah tahap akhir atau hasil yang sudah melalui tiga tahap sebelumnya, lalu hasil dari tahapan ini akan dikembangkan untuk ketahap selanjutnya.

Setelah melakukan diskusi dan usulan bersama tim mengenai sistem baru yang akan di buat, Praktikan menerima output elisitasi tahap satu sampai akhir **Tabel 3.1** sampai **Tabel 3.4**

Tabel 3 1 Elisitasi Tahap 1 (satu)

Functional	
No	Analisa Kebutuhan
	Sistem yang ingin didapat:
1	Menyediakan fitur login
2	Menampilkan halaman kelola *booking*
3	Menyediakan fitur CRUD data *booking*
4	Menyediakan fitur search pada data *booking*
5	Menyediakan fitur sidebar menu
6	Menyediakan fitur refresh data
7	Menyediakan halaman konfirmasi pembayaran
8	Menyediakan fitur konfirmasi pembayaran
9	Menyediakan halaman site map
10	Menyediakan halaman rangkuman penjualan
11	Menyediakan data rangkuman penjualan
12	Menyediakan fitur unduh data rangkuman penjualan
13	Menyediakan fitur logout
Non Functional	
No	analisis kebutuhan
	saya ingin sistem dapat:

1	Memberikan tampilan user friendly
2	Memberikan tampilan responsive pada mobile
3	Memberikan tampilan yang indah

Sumber: Hasil Dokumentasi Praktikan

Tabel 3 2 Elisitasi Tahap 2 (dua)

Functional				
No	Analisa Kebutuhan			
	Saya ingin sistem dapat melakukan:	**M**	**D**	**I**
1	Menyediakan fitur login	✓		
2	Menampilkan halaman kelola booking	✓		
3	Menyediakan fitur CRUD data booking	✓		
4	Menyediakan fitur search pada data booking	✓		
5	Menyediakan fitur sidebar menu	✓		
6	Menyediakan fitur refresh data		✓	
7	Menyediakan halaman konfirmasi pembayaran	✓		
8	Menyediakan fitur konfirmasi pembayaran	✓		
9	Menyediakan halaman site map	✓		
10	Menyediakan halaman direktur	✓		
11	Menyediakan data rangkuman penjualan	✓		
12	Menyediakan fitur unduh data rangkuman penjualan		✓	
13	Menyediakan fitur logout	✓		
Non Functional				
	Analisa Kebutuhan			

No	Saya ingin sistem dapat melakukan:	M	D	I
1	Memberikan tampilan user friendly	✓		
2	Memberikan tampilan responsive pada mobile	✓		
3	Memberikan tampilan yang indah	✓		

Sumber: Hasil Dokumentasi Praktikan

Tabel 3 3 Elisitasi Tahap 3 (tiga)

Functional										
Analisa Kebutuhan										
Saya ingin sistem dapat melakukan:										
No	Feasibility	**Technical**			**Operational**			**Economy**		
	Risk	**H**	**M**	**L**	**H**	**M**	**L**	**H**	**M**	**L**
1	Menyediakan fitur login		✓			✓			✓	
2	Menampilkan halaman kelola booking		✓			✓			✓	
3	Menyediakan fitur CRUD data booking	✓			✓			✓	✓	
4	Menyediakan fitur search pada data booking			✓			✓			✓
5	Menyediakan fitur sidebar menu			✓			✓			✓
6	Menyediakan fitur refresh data			✓			✓			✓
7	Menyediakan halaman konfirmasi pembayaran		✓			✓			✓	
8	Menyediakan fitur konfirmasi pembayaran		✓			✓			✓	

9	Menyediakan halaman site map			✓			✓			✓
10	Menyediakan halaman direktur		✓			✓			✓	
11	Menyediakan data rangkuman penjualan		✓			✓			✓	
12	Menyediakan fitur unduh data rangkuman penjualan		✓			✓			✓	
13	Menyediakan fitur logout			✓		✓			✓	
Non Functional										
Analisa Kebutuhan										
saya ingin sistem dapat:										
No	Feasibility	**Technical**			**Operational**			**Economy**		
	Risk	**H**	**M**	**L**	**H**	**M**	**L**	**H**	**M**	**L**
1	Memberikan tampilan user friendly		✓			✓			✓	
2	Memberikan tampilan responsive pada mobile		✓			✓			✓	
3	Memberikan tampilan yang indah		✓			✓			✓	

Sumber: Hasil Dokumentasi Praktikan

Tabel 3 4 Elisitasi Tahap Akhir

Functional	
No	Analisa Kebutuhan
	Sistem yang ingin didapat:
1	Menyediakan fitur login

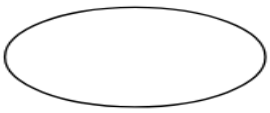
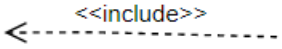
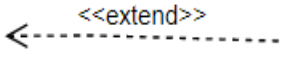
2	Menampilkan halaman kelola booking
3	Menyediakan fitur CRUD data booking
4	Menyediakan fitur search pada data booking
5	Menyediakan fitur sidebar menu
6	Menyediakan fitur refresh data
7	Menyediakan halaman konfirmasi pembayaran
8	Menyediakan fitur konfirmasi pembayaran
9	Menyediakan halaman site map
10	Menyediakan halaman direktur
11	Menyediakan data rangkuman penjualan
12	Menyediakan fitur unduh data rangkuman penjualan
13	Menyediakan fitur logout
Non Functional	
No	analisis kebutuhan
	saya ingin sistem dapat:
1	Memberikan tampilan user friendly
2	Memberikan tampilan responsive pada mobile
3	Memberikan tampilan yang indah

Sumber: Hasil Dokumentasi Praktikan

3.2.3 *Use Case* Diagram

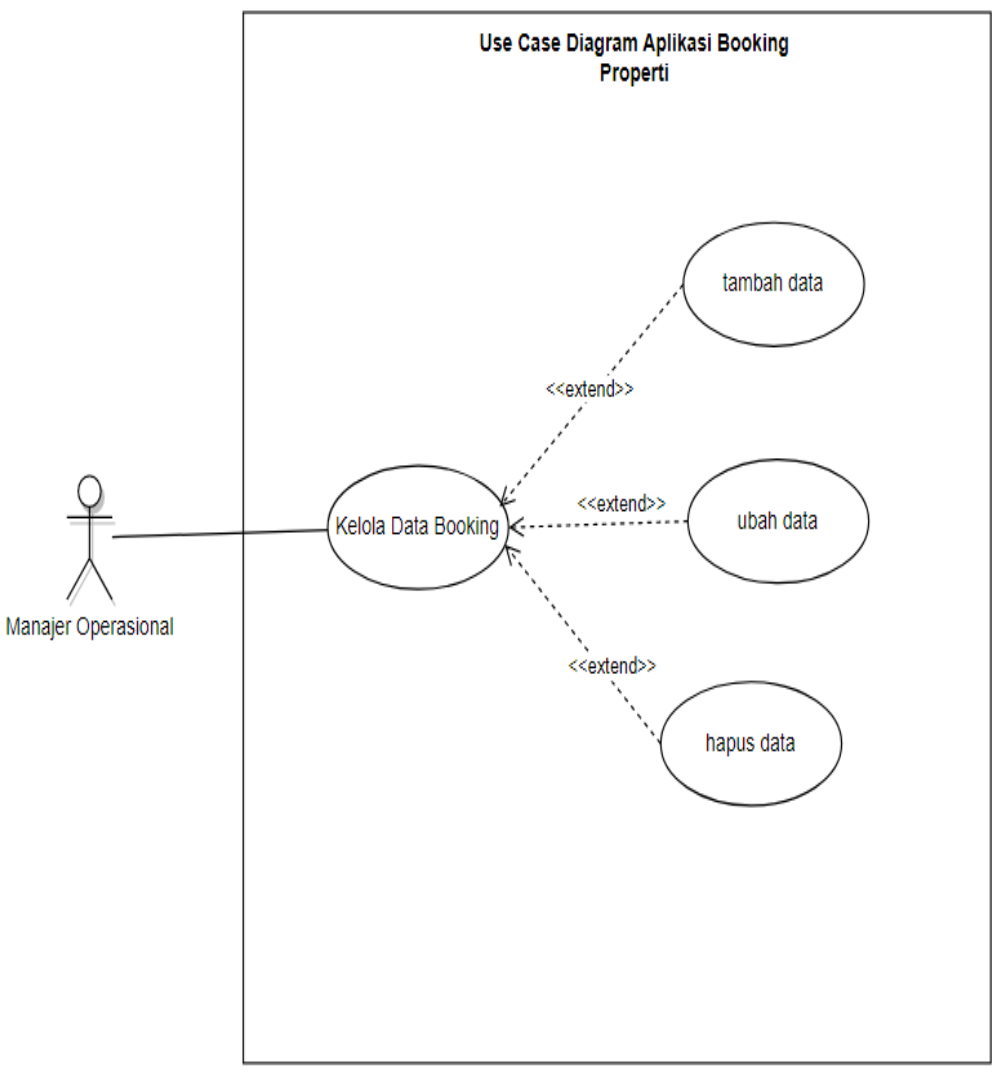
Use case diagram ialah sebuah metode umum untuk dipakai dalam *development* suatu *software* yang tujuannya agar mengetahui keperluan kebutuhan dari suatu sistem. Pengertian dari use case diagram sendiri adalah sebuah proses desain yang menggambarkan hubungan antara pengguna dengan sistem yang dirancang (Robith Adani, 2021). Berikut ini merupakan hasil *use case* diagram yang praktikan buat berdasarkan hasil akhir elisitasi yang telah dibuat. Berikut merupakan penjelasan mengenai symbol-simbol dari *use case* diagram disajikan pada **Tabel 3.5**

Tabel 3 5 Simbol Pada Use Case Diagram

Simbol	Nama	Penjelasan
	*Actor*	Merupakan sebuah ikon untuk orang yang berinteraksi bersama use case didalam sistem.
	*Use Case*	Merupakan suatu fitur dari bentuk dalam suatu sistem yang nantinya akan di implementasi
	Association	Mengabstraksi hubungan antara Aktor dengan kasus penggunaan
	*Include*	Menunjukkan relasi use case yang dimana memerlukan use case tersebut untuk dapat menjalankannya
	*Extend*	Menunjukkan relasi use yang dimana merupakan tambahan dan tidak perlu use case tambahan untuk menjalankannya

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 3.1** adalah *use case* diagram aplikasi *booking* berbasis *mobile* yang mempunyai peran sesuai dari gambar dibawah:



Gambar 3 1 use case diagram booking properti

Sumber: Hasil Dokumentasi Praktika

3.2.4 *Use Case Description*

Use case description ialah suatu teks yang dapat digunakan dalam menjelaskan langkah-langkah terperinci dari interaksi pengguna-sistem. Deskripsi dari *use case* ditulis dari sudut pandang aktor dan menghindari aspek internal. *Use case* ini memiliki dua kunci: *use case* dan skenario. Skenario adalah bagaimana seorang aktor menggunakan solusi untuk mencapai tujuan. *Use case Description* digunakan antara pengembang sistem dan tim pengujian ketika mereka ingin mengomunikasikan aktivitas di mana sistem digunakan. Hal ini agar tujuannya dapat dengan mudah dijelaskan dan publik dapat melihat apakah harapannya terpenuhi atau tidak (Richard Iwansaputra, 2022). Berikut merupakan use case description yang dibuat oleh praktikan pada **Tabel 3.6** sampai **Tabel 3.15**

Tabel 3 6 Spesifikasi use case Kelola booking

Use Case Name	Kelola Data Booking	
Actor	Manajer	
Trigger	Melihat data booking	
Pre-Conditions	Sudah melakukan login	
Post Conditions	Melihat data booking	
Success Scenario	Aktor	*System*
	1. Aktor memasuki halaman kelola data *booking*	
		2. Menampilkan data *booking*
Alternative Flows	-	

Sumber: Hasil Dokumentasi Praktikan

Tabel 3 7 Spesifikasi use case Kelola booking (tambah)

Use Case Name	Kelola Data Booking (Tambah)	
Actor	Manajer	
Trigger	Melakukan tambah data booking	
Pre-Conditions	Sudah melakukan login	
Post Conditions	Terjadi penambahan data booking	
Success Scenario	Aktor	*System*
	1. Aktor memasuki halaman kelola booking	
		2. Menampilkan data

		booking
	3. Klik tombol tambah	
		4. Menampilkan form tambah data
	5. Menginput pada form tambah data	
	6. Klik tombol submit	
		7. Menampilkan notifikasi “data berhasil ditambah”
		8. Menampilkan halaman data booking
Alternative Flows	-	

Sumber: Hasil Dokumentasi Praktikan

Tabel 3 8 Spesifikasi use case Kelola booking (ubah)

Use Case Name	Kelola Data Booking (Ubah)	
Actor	Manajer	
Trigger	Melakukan ubah data booking	
Pre-Conditions	Sudah melakukan login	
Post Conditions	Terjadi Pengubahan data booking	
Success Scenario	Aktor	*System*
	1. Aktor memasuki halaman kelola booking	
		2. Menampilkan data booking
	3. Klik tombol edit	
		4. Menampilkan form data yang dapat diubah
	5. Mengubah data booking	
	6. Klik tombol submit	
		7. Menampilkan notifikasi “data berhasil diubah”
		8. Menampilkan halaman data booking
Alternative Flows	-	

Sumber: Hasil Dokumentasi Praktikan

Tabel 3 9 Spesifikasi use case Kelola booking (hapus)

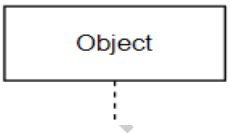
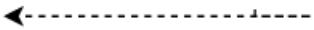
Use Case Name	Kelola Data Booking (Hapus)	
Actor	Manajer	
Trigger	Melakukan Hapus data booking	
Pre-Conditions	Sudah melakukan login	
Post Conditions	Terjadi penghapusan data booking	
Success Scenario	Aktor	*System*
	1. Aktor memasuki halaman kelola booking	
		2. Menampilkan data booking
	3. Klik tombol hapus	
		4. Notifikasi “konfirmasi penghapusan”
	5. Klik tombol yes	
		6. Menampilkan notifikasi “data berhasil dihapus
		7. Menampilkan halaman kelola data booking
Alternative Flows	-	

Sumber: Hasil Dokumentasi Praktikan

3.2.5 Sequence Diagram

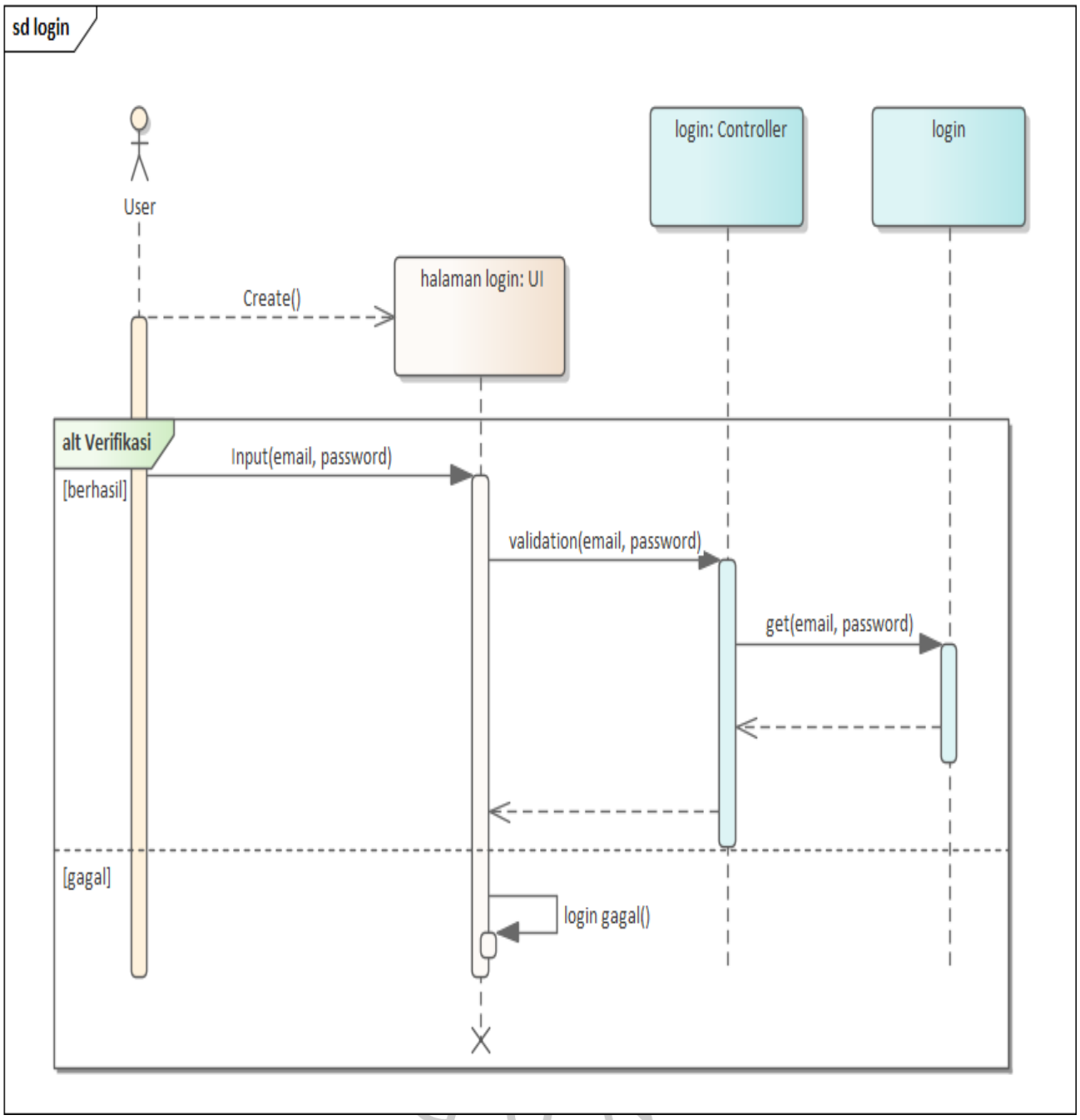
Sequence Diagram merupakan diagram yang diapakai dalam merinci lalu menunjukkan interaksi antara obyek yang berada pada suatu sistem. sequence diagram dapat menggambarkan suatu pesan atau perintah yang dikirim dan akan dieksekusi. Objek untuk pelaksanaan proses operasional biasanya diurutkan dari kiri ke kanan (Rony Setiawan, 2021). Berikut ini merupakan Berikut adalah sequence diagram yang dibuat oleh praktisi berdasarkan *activity* diagram yang dilakukan oleh *frontend developer*. Berikut merupakan penjelasan dari simbol-simbol pada **Tabel 3.13** di bawah ini.

Tabel 3 10 Simbol pada sequence diagram

Simbol	Nama	Penjelasan
	*Actor*	Menggambarkan pengguna yang melakukan interaksi secara langsung di sistem
	*Lifeline*	Menggambarkan sebuah class atau object entity
	Message	Menggambarkan pesan dari interaksi objek-objek
	*Message to life*	Mendeskripsikan umpan balik atau reaksi dengan objek sebelumnya

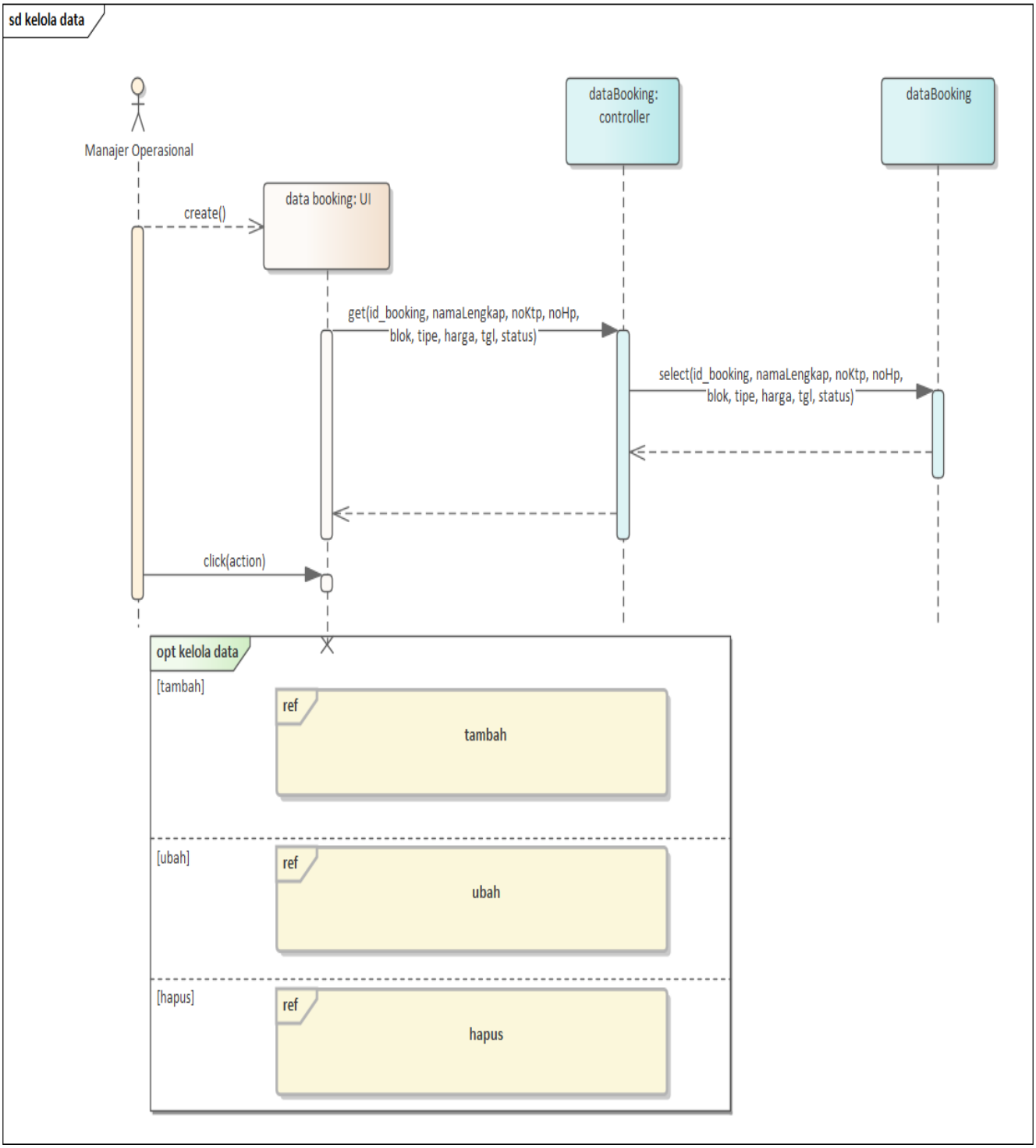
Sumber: Hasil Dokumentasi Praktikan

Berikut merupakan hasil dari sequence diagram yang telah dibuat oleh praktikan dari **Gambar 3.2** sampai **Gambar 3.6.** dibawah ini.



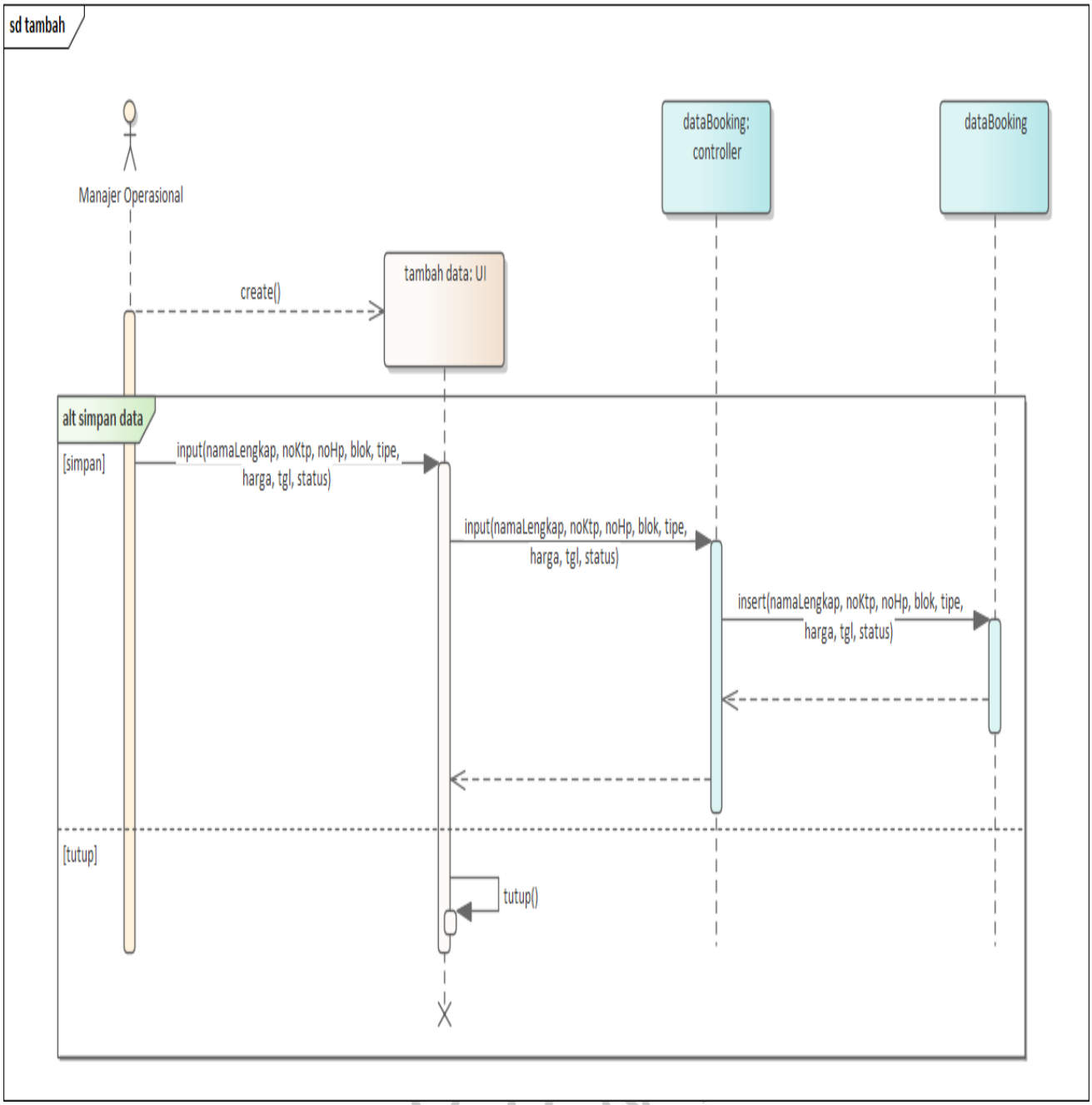
Gambar 3 2 sequence diagram login user

Sumber: Hasil Dokumentasi Praktikan



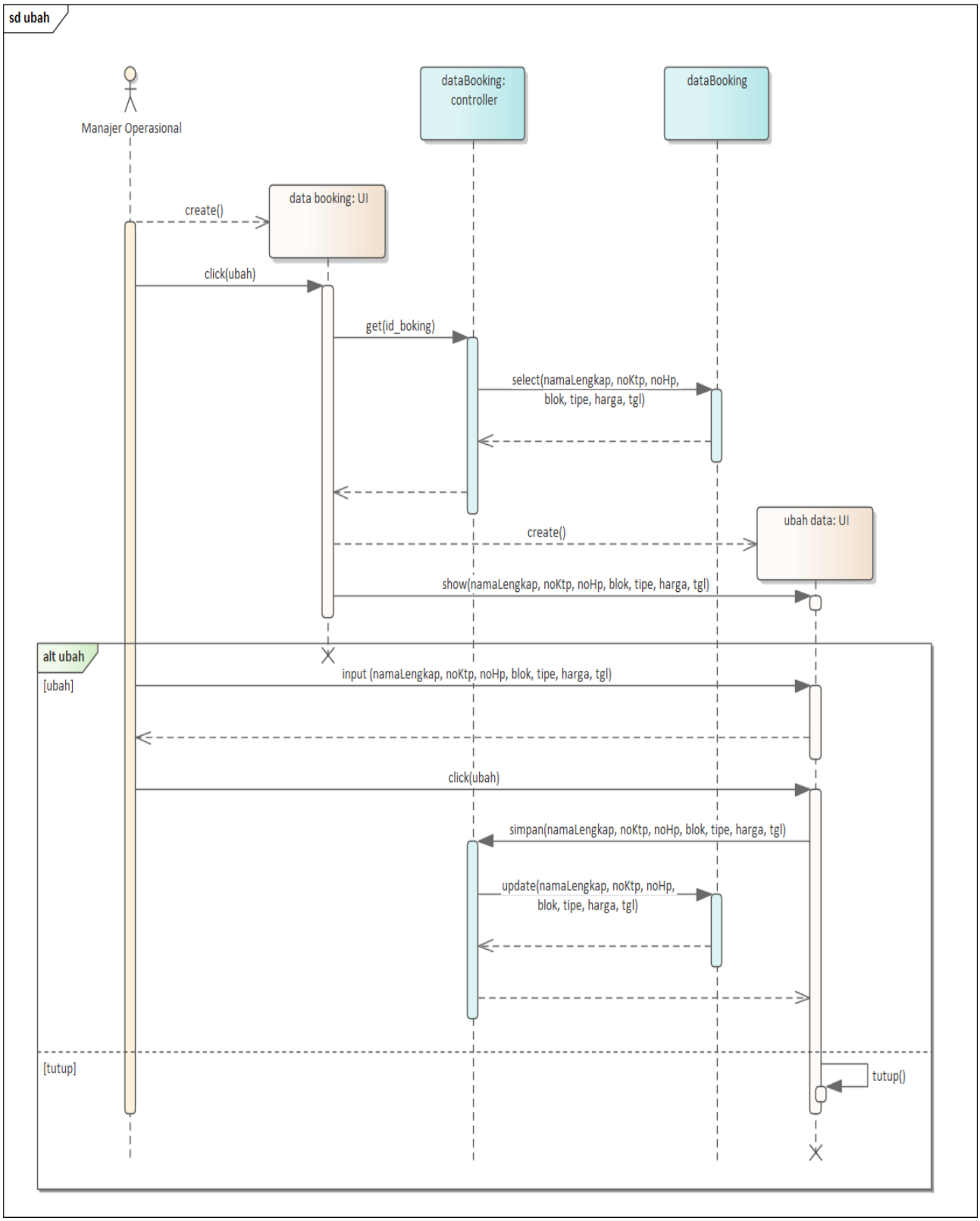
Gambar 3 3 sequence diagram kelola data booking

Sumber: Hasil Dokumentasi Praktikan



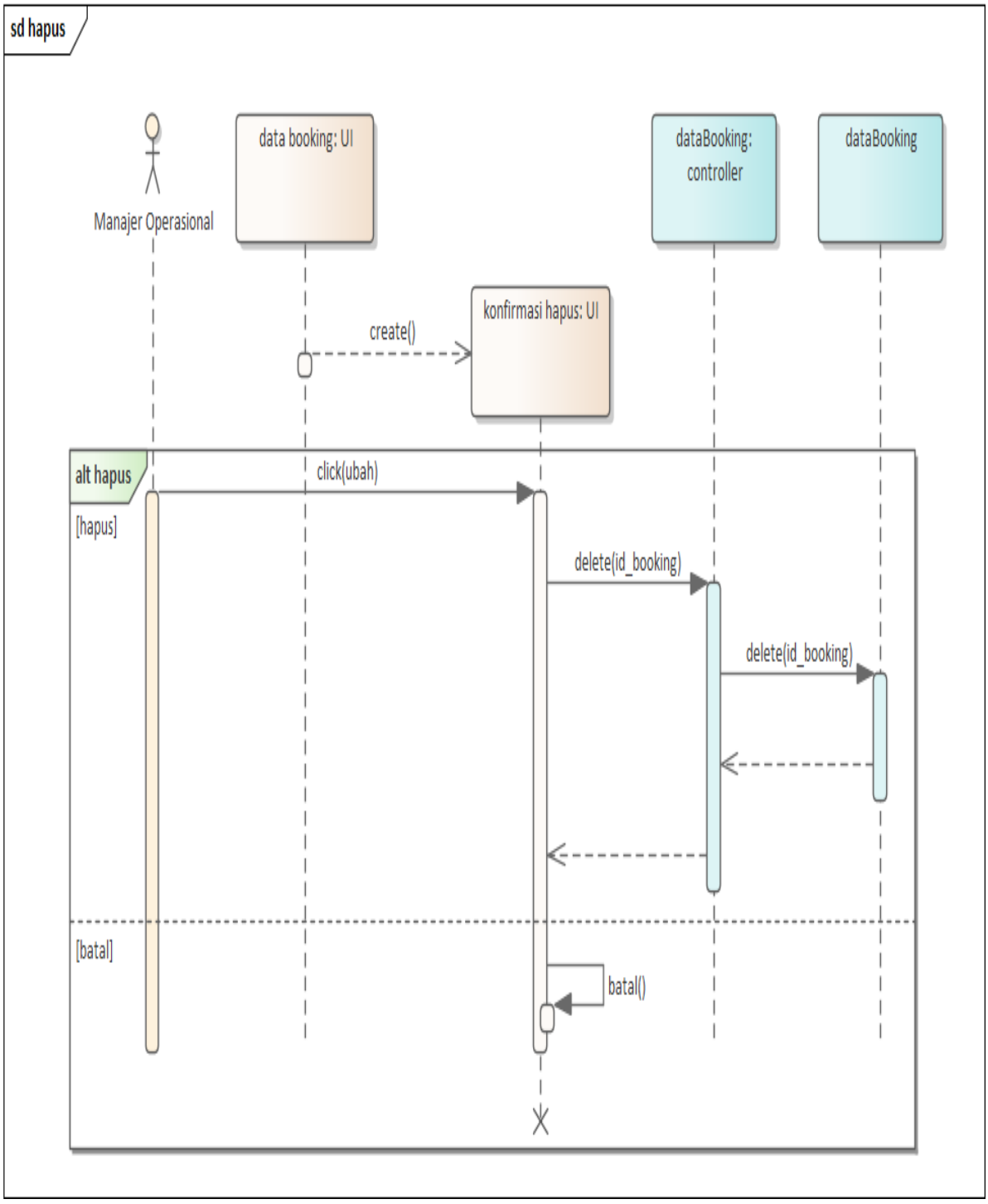
Gambar 3 4 sequence diagram tambah data booking

Sumber: Hasil Dokumentasi Praktikan



Gambar 3 5 sequence diagram ubah data booking

Sumber: Hasil Dokumentasi Praktikan

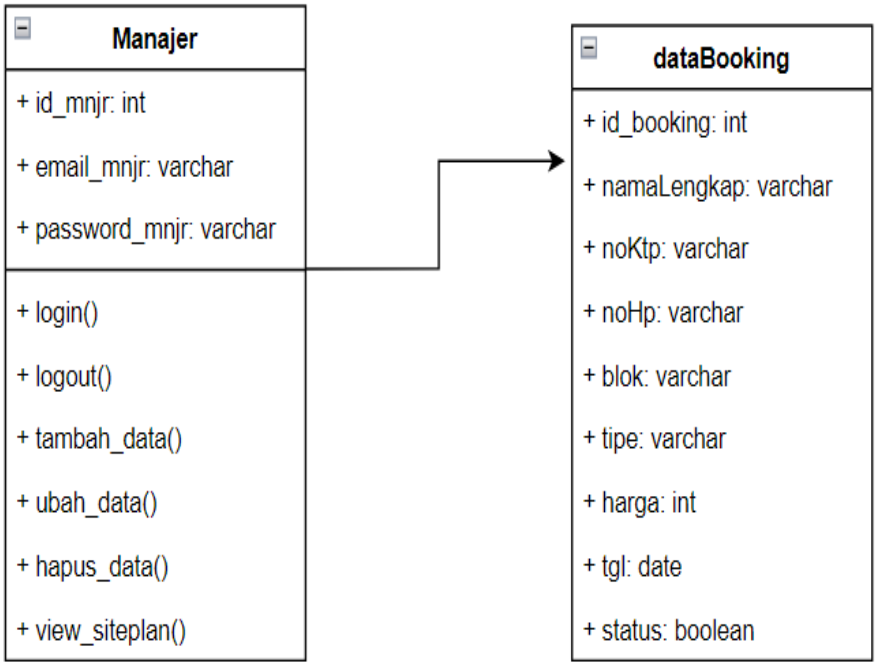


Gambar 3 6 sequence diagram hapus data booking manajer

Sumber: Hasil Dokumentasi Praktikan

3.2.6 *Class Diagram*

class diagram ialah sebuah diagram bangun UML yang dengan jelas menunjukkan struktur dan metode, atribut, hubungan antar objek dan deskripsi setiap kelasnya, dan. Diagram kelas memiliki sifat yang statis karena menggambarkan hubungan apa yang ada, bukan apa yang terjadi (Rony Setiawan, 2021). Berikut merupakan hasil class diagram yang praktikan buat pada **Gambar 3.7**.



Gambar 3 7 Class diagram aplikasi booking

Sumber: Hasil Dokumentasi Praktikan

3.2.7 Implementasi

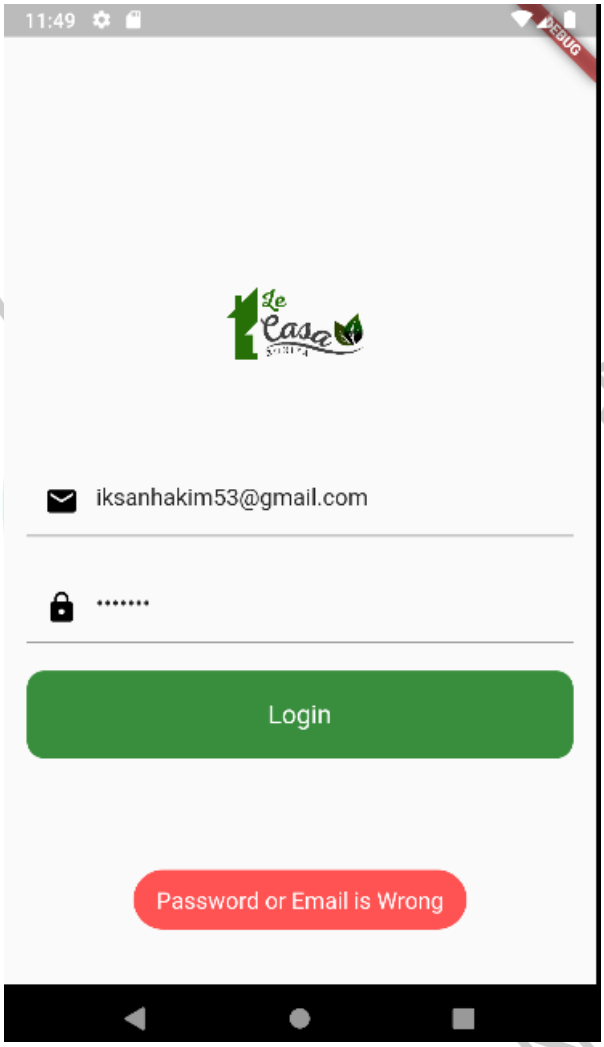
Setelah tahapan *system requirement* sampai dengan tampilan front-end dibuat, praktikan melakukan implementasi dengan menggunakan Firebase. Firebase adalah layanan Google yang didirikan James Trampling dan Andrew Lee untuk memberikan kemudahan dan bahkan memudahkan pengembang perangkat lunak untuk mengembangkan aplikasi. Firebase merupakan BaaS (*Backend as a Service*) merupakan suatu pemecahan masalah yang telah disediakan oleh Google yang dapat memudahkan kerja para *developer*. Dengan menggunakan Firebase pengembang dapat fokus pada pengembangan aplikasi tanpa menghabiskan banyak energi untuk hal-hal *backend*. layanannya dapat digunakan secara cuma-cuma (gratis) dengan batasan tertentu. Firebase mempunyai fitur yang dapat dimanfaatkan, salah satunya adalah *authentication* dan *cloud firestore*. Fitur authentication ini bermanfaat untuk melakukan autentikasi pengguna yang akan masuk kedalam aplikasi tersebut, sedangkan cloud firestore bermanfaat untuk menampung data secara *real time* dan menghubungkannya dengan aplikasi yang dibuat.(Aprilia, 2020).

Gambar 3 8 logo firebase

Sumber: Wikipedia

Berikut merupakan hasil implementasi yang sudah selesai dilakukan oleh praktikan:

1. Login Sistem



Gambar 3 9 tampilan gagal login

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 3.9**, diperlihatkan tampilan jika user gagal dalam melakukan login, sehingga mengharuskan pengguna untuk memasukkan *email* dan *password* yang benar agar dapat akses untuk mengakses kedalam aplikasi booking properti. Berikut merupakan source code untuk sistem login pada dibawah ini.

```dart
class _HomePageState extends State<HomePage> {
//Initialize Firebase App
  Future<FirebaseApp> _initializeFirebase() async {
    FirebaseApp firebaseApp = await Firebase.initializeApp();
    return firebaseApp;
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: FutureBuilder(
        future: _initializeFirebase(),
        builder: (context, snapshot) {
          if (snapshot.connectionState == ConnectionState.done) {
            return LoginScreen();
          }
          return const Center(
            child: CircularProgressIndicator(),
          ); // Center
        },
      ), // FutureBuilder
    ); // Scaffold
  }
}
```

Gambar 3 10 source code inisialisasi firebase

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 3.10** diatas diperlihatkan *source code* untuk melakukan inisialisasi Firebase agar bisa berjalan pada aplikasi booking properti yang dibuat.

```dart
static Future<User?> loginUsingEmailPassword(
    {required String email,
    required String password,
    required BuildContext context}) async {
  FirebaseAuth auth = FirebaseAuth.instance;
  User? user;
  try {
    UserCredential userCredential = await auth.signInWithEmailAndPassword(
        email: email, password: password);
    user = userCredential.user;
  } on FirebaseAuthException catch (e) {
    Fluttertoast.showToast(
        msg: "Password or Email is Wrong",
        backgroundColor: Colors.redAccent,
        textColor: Colors.white,
        fontSize: 16.0,
        gravity: ToastGravity.BOTTOM);
  }
  return user;
}
```

Gambar 3 11 source code Function login

Sumber: Hasil Dokumentasi Praktikan

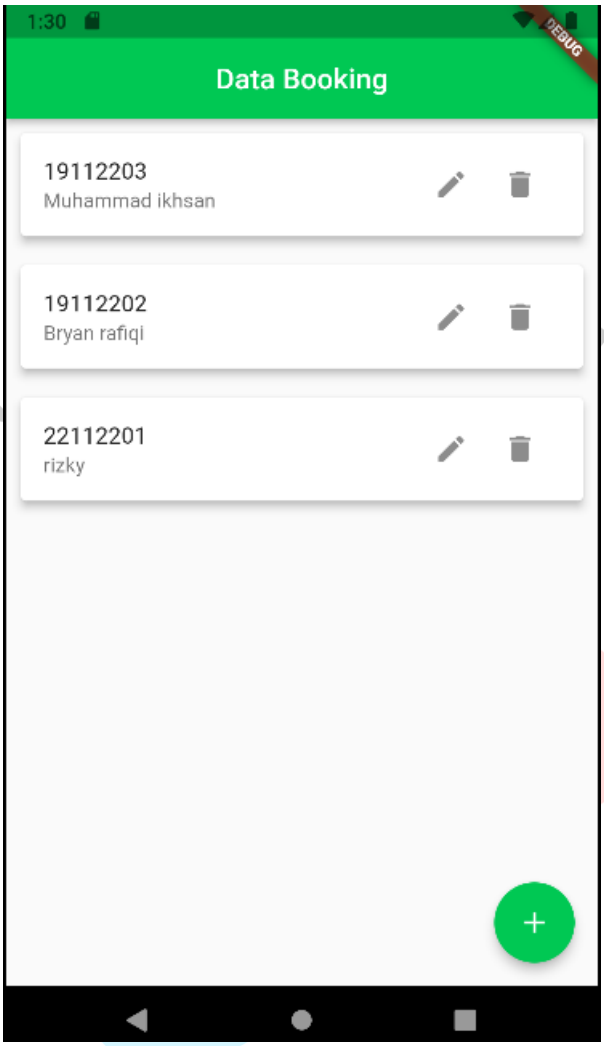
Pada **Gambar 3.11** diatas diperlihatkan *source code* untuk melakukan verifikasi login, jika salah akan muncul toast dan akan meminta kita untuk menginput dengan benar. Jika sudah benar nantinya akan dilakukan perintah untuk dapat masuk kehalaman data *booking* seperti **Gambar 3.12**.

```dart
onPressed: () async {
  //test the app
  User? user = await loginUsingEmailPassword(
      email: _emailController.text,
      password: _passwordController.text,
      context: context);
  print(user);
  if (user != null) {
    Navigator.of(context).pushReplacement(
        MaterialPageRoute(builder: (context) => BookingPage()));
  }
```

Gambar 3 12 login action

Sumber: Hasil Dokumentasi Praktikan

2. Kelola Data Booking

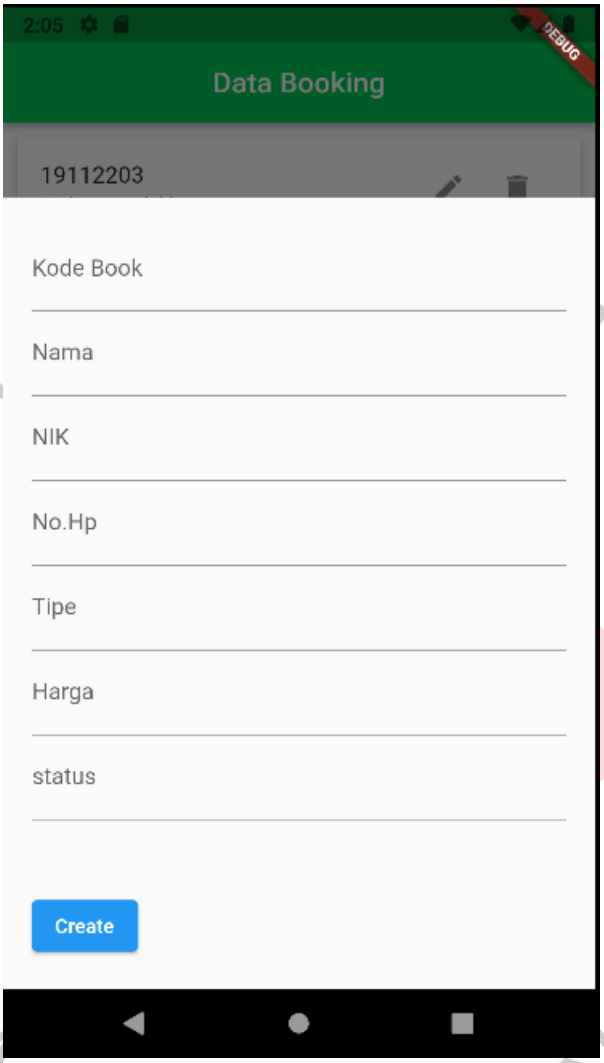


Gambar 3 13 Data kelola data booking

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 3.13** diatas diperlihatkan data Kelola *booking* yang sudah terintegritas menggunakan firebase. Di data Kelola *booking* ini memiliki fitur untuk melakukan tambah, ubah, dan hapus.

- Tambah Data



Gambar 3 14 form tambah data

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 3.14** diatas diperlihatkan form untuk melakukan tambah data pada data booking. Untuk melakukan tambah data hanya perlu menginput data dan setelah itu menekan tombol ubah untuk menyimpan data tersebut. Berikut merupakan *source code* untuk melakukan untuk melakukan tambah data pada **Gambar 3.15** dan **Gambar 3.16**.

```
// text fields' controllers
  final TextEditingController _kdbookController = TextEditingController();
  final TextEditingController _nameController = TextEditingController();
  final TextEditingController _nikController = TextEditingController();
  final TextEditingController _nohpController = TextEditingController();
  final TextEditingController _tipeController = TextEditingController();
  final TextEditingController _hargaController = TextEditingController();
  final TextEditingController _statusController = TextEditingController();

  final CollectionReference _bookingdata =
      FirebaseFirestore.instance.collection('bookingdata');
```

Gambar 3 15 source code controller tambah

Sumber: Hasil Dokumentasi Praktikan

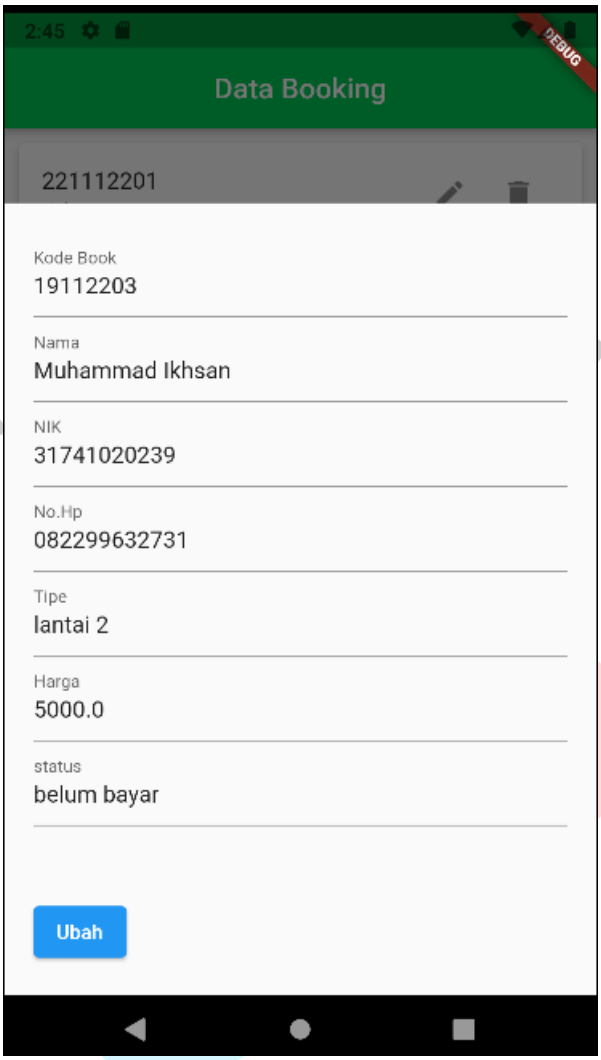
Pada **Gambar 3.15** diatas diperlihatkan sebuah *source code* untuk mengambil data pada tabel "*bookingdata*"yang sudah dibuat dalam firebase. Setelah itu dibuatlah function tambah data pada **Gambar 3.16**

```
child: const Text('Tambah'),
onPressed: () async {
  final String kdbook = _kdbookController.text;
  final String name = _nameController.text;
  final String nik = _nikController.text;
  final String nohp = _nohpController.text;
  final String tipe = _tipeController.text;
  final double? harga =
      double.tryParse(_hargaController.text);
  final String status = _statusController.text;
  if (kdbook != null) {
    await _bookingdata.add({
      "kdbook": kdbook,
      "name": name,
      "nik": nik,
      "nohp": nohp,
      "tipe": tipe,
      "harga": harga,
      "status": status
    });
    _kdbookController.text = '';
    _nameController.text = '';
    _nikController.text = '';
    _nohpController.text = '';
    _tipeController.text = '';
    _hargaController.text = '';
    _statusController.text = '';
    Navigator.of(context).pop();
```

Gambar 3 16 Source code tambah data

Sumber: Hasil Dokumentasi Praktikan

- Ubah Data



Gambar 3 17 form edit data

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 3.17** diatas diperlihatkan sebuah form untuk melakukan ubah data pada data booking yang dipilih. untuk melakukan ubah data hanya perlu memilih datanya, selanjutnya tinggal mengubah data tersebut dan terakhir menekan tombol ubah untuk menyimpan perubahan tersebut. Berikut merupakan source code untuk melakukan perubahan data pada **Gambar 3.18** dan **Gambar 3.19**

```
Future<void> _update([DocumentSnapshot? documentSnapshot]) async {
  if (documentSnapshot != null) {
    _kdbookController.text = documentSnapshot['kdbook'];
    _nameController.text = documentSnapshot['name'];
    _nikController.text = documentSnapshot['nik'];
    _nohpController.text = documentSnapshot['nohp'];
    _tipeController.text = documentSnapshot['tipe'];
    _hargaController.text = documentSnapshot['harga'].toString();
    _statusController.text = documentSnapshot['status'];
  }
```

Gambar 3 18 source code controller update

Sumber: Hasil Dokumentasi Praktikan

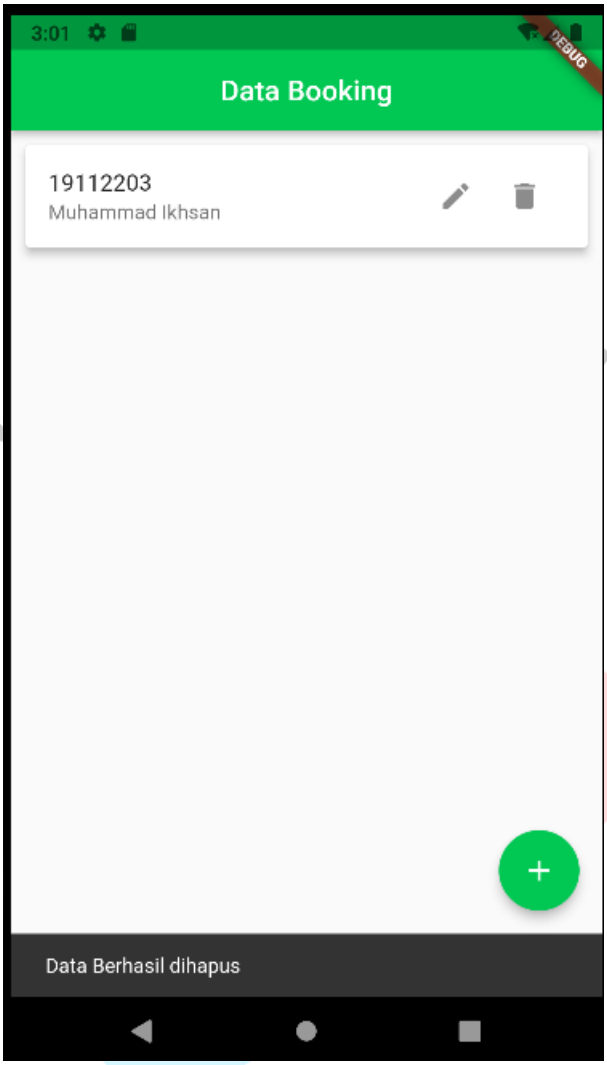
Pada **Gambar 3.18** diatas diperlihatkan source code untuk mengambil data dan menampilkannya pada form ubah. Lalu setelah dilakukan perubahan akan dijalankan function ubah yang akan disimpan kedalam database pada **Gambar 3.19** dibawah.

```
onPressed: () async {
  final String kdbook = _kdbookController.text;
  final String name = _nameController.text;
  final String nik = _nikController.text;
  final String nohp = _nohpController.text;
  final String tipe = _tipeController.text;
  final double? harga =
      double.tryParse(_hargaController.text);
  final String status = _statusController.text;
  if (harga != null) {
    await _bookingdata.doc(documentSnapshot!.id).update({
      "kdbook": kdbook,
      "name": name,
      "nik": nik,
      "nohp": nohp,
      "tipe": tipe,
      "harga": harga,
      "status": status
    });
    _kdbookController.text = '';
    _nameController.text = '';
    _nikController.text = '';
    _nohpController.text = '';
    _tipeController.text = '';
    _hargaController.text = '';
    _statusController.text = '';
    Navigator.of(context).pop();
```

Gambar 3 19 source code ubah data

Sumber: Hasil Dokumentasi Praktikan

- Hapus Data



Gambar 3 20 tampilan hapus data

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 2.20** diatas diperlihatkan tampilan saat data berhasil dihapus. Untuk menghapus data hanya memilih yang ingin dihapus lalu data tersebut akan terhapus. Berikut merupakan *source code* untuk menghapus data pada **Gambar 3.21** dibawah.

```
Future<void> _delete(String bookingId) async {
  await _bookingdata.doc(bookingId).delete();

  ScaffoldMessenger.of(context)
      .showSnackBar(const SnackBar(content: Text('Data Berhasil dihapus')));
}
```

Gambar 3 21 source code hapus data

Sumber: Hasil Dokumentasi Praktikan

Pada **Gambar 3.21** diatas diperlihatkan sebuah fungsi untuk melakukan hapus dengan mengambil id data tersebut, selanjutnya akan menampilkan *snackbar* sebagai notifikasi bahwa data berhasil dihapus dari *database.*

3.3 Kendala Yang Dihadapi

Pada berlangsungnya Kerja Profesi (KP) dalam melakukan tugasnya sebagai back-end untuk merancang aplikasi booking berbasis mobile. terdapat hambatan yang dialami sehingga waktu pengerjaannya pun menjadi terhambat. Berikut merupakan kendala-kendala yang dihadapi oleh praktikan selama mengerjakan tugasnya pada PT. Bumi Rejeki Agung:

- Pelaksanaan Kerja Profesi (KP) yang dilakukan pada masa covid-19 dan lokasi kantor yang jauh mengharuskan praktikan untuk melakukan pekerjaannya secara *work from home* (WFH) yang membuat komunikasi dengan tim dan pembimbing kurang maksimal.
- Terbatasnya akses data mengenai informasi yang dibutuhkan dan verifikasi hasil pekerjaan yang susah membuat waktu pengerjaan terhambat.

3.4 Cara Mengatasi Kendala

Dari berbagai kendala yang ada saat penulis melangsungkan KP (Kerja Profesi), berikut merupakan solusi yang ditemukan untuk mengatasi kendala atau permasalahan yang dihadapi praktikan:

- Melakukan komunikasi dengan mentor dan tim dengan menggunakan *video conference* google *meet*.
- Menanyakan secara langsung kepada pembimbing KP mengenai akses data kebutuhan informasi dan verifikasi hasil pekerjaan agar pekerjaan dapat berjalan dengan lancar.

3.5 Pembelajaran yang diperoleh

Pelajaran yang telah didapatkan oleh praktikan selama melakukan KP (kerja profesi) antara lain sebagai berikut:

- Praktikan mempelajari hal baru seperti pembuatan aplikasi berbasis mobile menggunakan Firebase sebagai database.
- Praktikan mempunyai rasa tanggung jawab terhadap tugas yang diberikannya.
- Praktikan mengimplementasikan teori yang telah dipelajari dari perkuliahan.
- Praktikan mendapatkan pengalaman terjun langsung terhadap pembuatan aplikasi berdasarkan masalah yang dihadapi.
- Praktikan mendapatkan pelajaran bahwa komunikasi atau koordinasi merupakan hal yang penting