LAPORAN TUGAS UAS

REKAYASA PERANGKAT LUNAK

(Dosen : *Iwan Lesmana, S.Kom., M.Kom.*)

Nama : Muhammad Rizal Nurfirdaus
NIM : 20230810088
Kelas : TINFC-2023-04

TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS KUNINGAN

Ringkasan Sistem

Web ini adalah **Sistem Informasi Reservasi Layanan (Service Reservation System)** berbasis web. Fungsi utamanya adalah mendigitalkan proses pemesanan jasa, mulai dari pemilihan layanan oleh pelanggan, pengelolaan operasional oleh admin, hingga pemantauan kinerja bisnis oleh pemilik usaha.

Pengguna & Fitur Utama

Sistem ini membagi akses menjadi tiga peran (*role*) dengan hak akses yang berbeda:

1. **Customer (Pelanggan)**
    - **Fokus:** Transaksi dan Penggunaan Jasa.
    - **Fitur:** Dapat masuk menggunakan **Google OAuth** (tanpa perlu repot register manual), melihat katalog layanan, melakukan reservasi dengan formulir detail (termasuk pemilihan lokasi hingga tingkat desa), serta memantau status atau membatalkan pesanan.
2. **Admin (Operator)**
    - **Fokus:** Manajemen Operasional Harian.
    - **Fitur:** Mengelola data master (menambah/mengedit kategori dan item layanan beserta gambarnya), memverifikasi pembayaran masuk, dan memperbarui status pengerjaan reservasi (dari *Pending* hingga *Completed*).
3. **Owner (Pemilik Bisnis)**
    - **Fokus:** Strategi dan Analisis Data.
    - **Fitur:** Memiliki dashboard eksekutif untuk melihat grafik pendapatan (*Revenue Chart*), tren reservasi bulanan, dan statistik layanan terlaris untuk pengambilan keputusan bisnis.

Arsitektur & Teknologi (Tech Stack)

Berdasarkan *Sequence Diagram*, aplikasi ini dibangun dengan arsitektur modern (*Fullstack*):

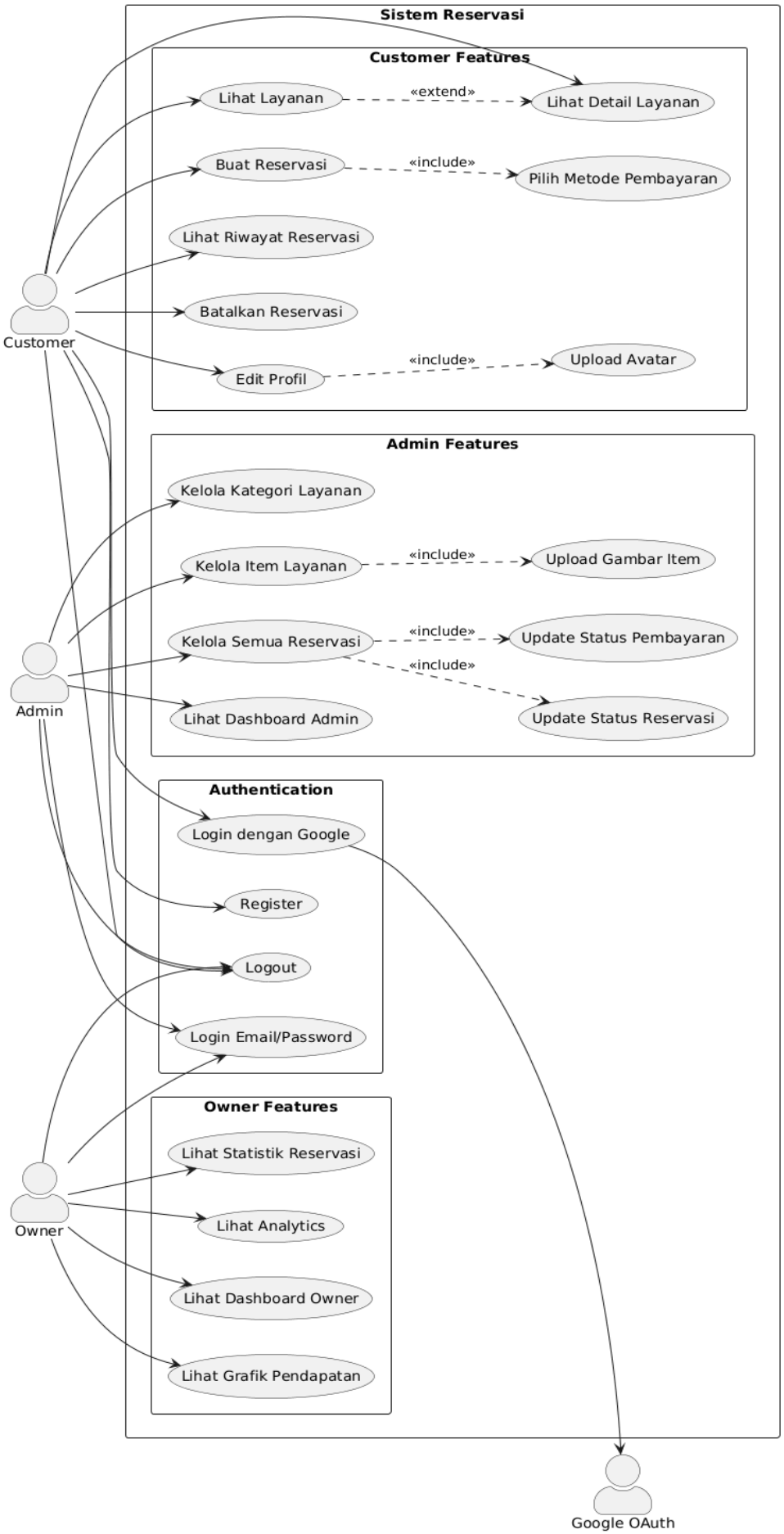
- **Frontend:** Menggunakan **React** untuk antarmuka pengguna yang interaktif, termasuk penggunaan library **Recharts** untuk visualisasi data grafik di dashboard Owner.
- **Backend:** Menggunakan **Express.js** (Node.js) sebagai server API yang menangani logika bisnis.
- **Autentikasi:** Menggunakan **Passport.js** untuk menangani keamanan login, khususnya integrasi dengan **Google OAuth**.
- **Database:** Menggunakan **PostgreSQL** sebagai tempat penyimpanan data relasional (User, Services, Reservations).

Alur Bisnis Singkat

1. Pelanggan login (via Google/Email) dan memilih layanan.

2. Pelanggan mengisi formulir reservasi dan metode pembayaran.
3. Sistem menyimpan status awal sebagai `PENDING`.
4. Admin memverifikasi pembayaran dan mengubah status menjadi `CONFIRMED`.
5. Setelah layanan selesai, Admin mengubah status menjadi `COMPLETED`.
6. Data transaksi tersebut otomatis terekap di Dashboard Owner sebagai laporan pendapatan.

1. Use Case Diagram



Use Case Diagram merepresentasikan interaksi fungsional antara aktor (pengguna) dengan sistem "Sistem Reservasi". Diagram ini memetakan batasan sistem (*system boundary*) dan mendefinisikan hak akses berdasarkan empat aktor utama: **Customer**, **Admin**, **Owner**, dan **Google OAuth** (sistem eksternal).

- **Aktor Customer:** Customer merupakan pengguna akhir yang menjadi target layanan. Fungsionalitas utama mereka berpusat pada transaksi dan manajemen profil. Customer dapat melakukan *Use Case* **"Lihat Layanan"** yang memiliki relasi `<<extend>>` ke **"Lihat Detail Layanan"**, artinya Customer bisa melihat daftar umum namun opsi melihat detail bersifat opsional/lanjutan. Dalam melakukan transaksi, terdapat *Use Case* **"Buat Reservasi"** yang memiliki relasi `<<include>>` dengan **"Pilih Metode Pembayaran"**. Ini menegaskan bahwa proses reservasi tidak akan dianggap valid atau selesai tanpa adanya pemilihan metode pembayaran yang jelas. Selain itu, Customer juga memiliki hak untuk **"Lihat Riwayat Reservasi"**, **"Batalkan Reservasi"**, serta **"Edit Profil"** yang mencakup (`<<include>>`) fungsi **"Upload Avatar"** untuk personalisasi akun.

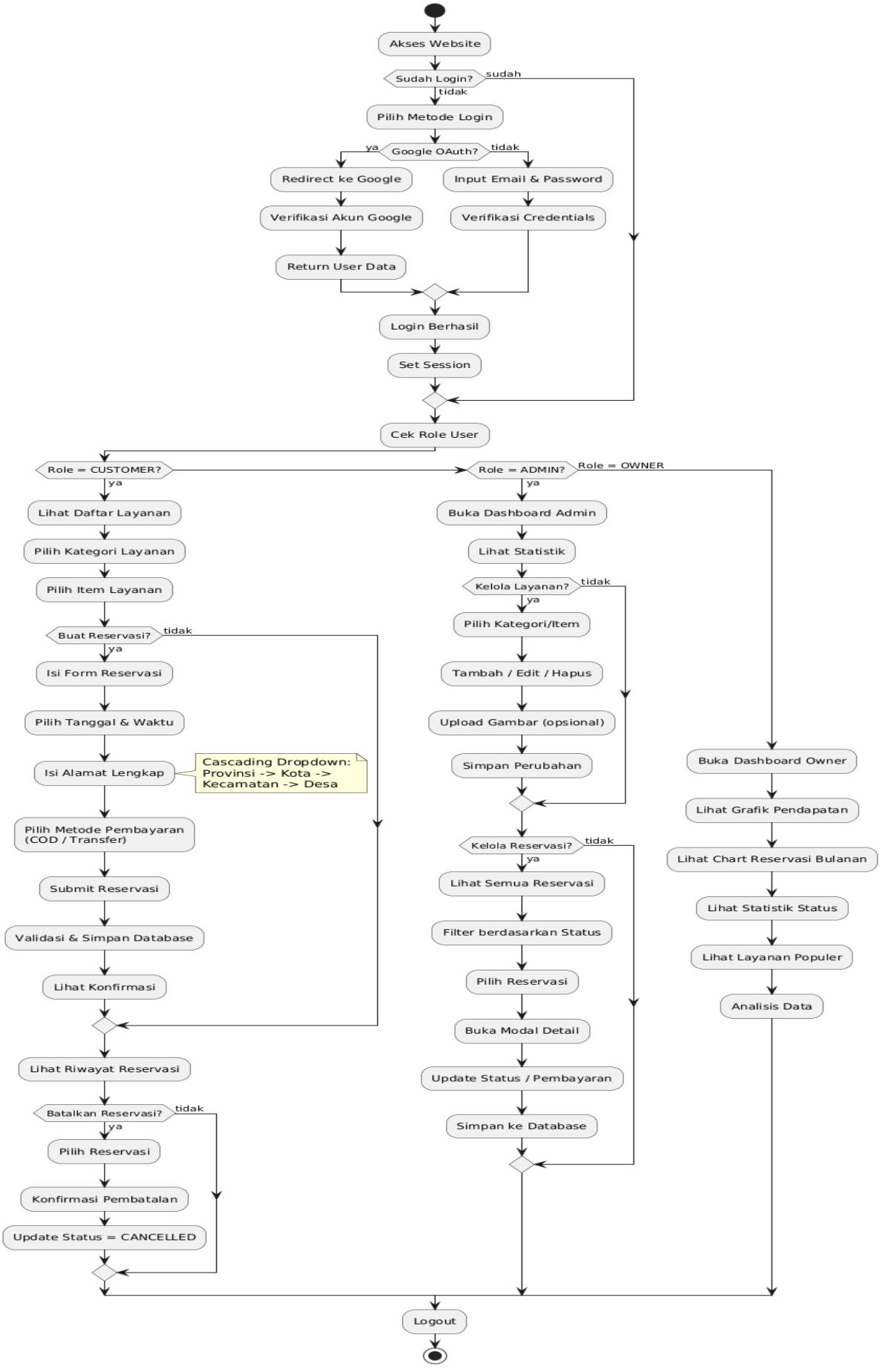
- **Aktor Admin:** Admin bertindak sebagai pengelola operasional sistem. Hak akses Admin dirancang untuk pemeliharaan data (*maintenance*). Admin bertanggung jawab atas **"Kelola Kategori Layanan"** dan **"Kelola Item Layanan"**. Pada pengelolaan item, terdapat relasi `<<include>>` ke **"Upload Gambar Item"**, yang berarti sistem mewajibkan adanya media gambar saat admin menambah atau mengedit layanan. Fungsi krusial lainnya adalah **"Kelola Semua Reservasi"**. *Use Case* ini mencakup dua fungsi wajib (`<<include>>`), yaitu **"Update Status Pembayaran"** (memverifikasi transfer masuk) dan **"Update Status Reservasi"** (mengubah status dari Pending ke Confirmed atau Completed). Admin juga memiliki akses ke **"Lihat Dashboard Admin"** untuk ringkasan operasional.

- **Aktor Owner:** Owner memiliki hak akses yang bersifat strategis dan analitik, terpisah dari operasional harian. Fitur utama Owner adalah **"Lihat Dashboard Owner"**, **"Lihat Statistik Reservasi"**, **"Lihat Analytics"**, dan **"Lihat Grafik Pendapatan"**. Semua *use case* ini dirancang untuk menampilkan data agregat yang membantu pemilik bisnis dalam pengambilan keputusan bisnis tanpa perlu terlibat dalam teknis pengelolaan pesanan satu per satu.

- **Modul Authentication (Autentikasi):** Ini adalah paket fitur yang diwariskan atau dapat diakses oleh ketiga aktor manusia di atas. Terdapat opsi **"Login dengan Google"** yang berinteraksi langsung dengan aktor eksternal **Google OAuth** untuk validasi identitas yang aman. Selain itu, tersedia opsi konvensional **"Login Email/Password"**, fitur **"Register"**

untuk pengguna baru, dan **"Logout"** untuk mengakhiri sesi.

2. Activity Diagram



Activity Diagram menggambarkan alur kerja (*workflow*) sistem secara *end-to-end*, mulai dari pengguna mengakses situs hingga keluar dari sistem, dengan penekanan pada logika percabangan (*decision nodes*).

- **Fase Inisiasi dan Autentikasi:** Aktivitas dimulai saat pengguna mengakses *website*. Sistem melakukan pengecekan kondisi awal: "Sudah Login?". Jika **Tidak**, pengguna diarahkan ke halaman pemilihan metode login. Jika memilih **Google OAuth**, sistem mengarahkan ke Google untuk verifikasi akun eksternal. Jika memilih **Email/Password**, sistem memverifikasi kredensial di database internal. Setelah verifikasi sukses, sistem mengembalikan data pengguna, menetapkan sesi (*Set Session*), dan melakukan pengecekan peran pengguna (*Cek Role User*).

- **Alur Percabangan Berdasarkan Role:**

    1. **Jalur Customer:** Pengguna melihat daftar layanan, memilih kategori, dan memilih item spesifik. Sistem mengecek keputusan: "Buat Reservasi?". Jika **Ya**, Customer mengisi form reservasi yang memiliki logika *Cascading Dropdown* (pilihan Provinsi memfilter Kota, Kota memfilter Kecamatan, dst). Setelah memilih tanggal, waktu, dan metode pembayaran, Customer menekan "Submit". Sistem melakukan validasi, menyimpan ke database, dan menampilkan konfirmasi. Customer kemudian dapat melihat riwayat. Di sini ada opsi pembatalan; jika Customer memilih membatalkan, status reservasi di-update menjadi `CANCELLED` sebelum proses selesai.

    2. **Jalur Admin:** Admin diarahkan ke Dashboard Admin untuk melihat statistik. Admin dapat memilih untuk **"Kelola Layanan"** (CRUD item & Upload Gambar) atau **"Kelola Reservasi"**. Jika mengelola reservasi, Admin dapat memfilter pesanan berdasarkan status, melihat detail, dan melakukan aksi kritis: **Update Status** (pengerjaan) atau **Update Pembayaran**. Perubahan ini disimpan ke database.

    3. **Jalur Owner:** Alur Owner lebih linier dan berfokus pada pemantauan (monitoring). Owner membuka Dashboard Owner untuk melihat grafik pendapatan, chart reservasi bulanan, statistik status, dan analisis layanan populer. Tidak ada aksi manipulasi data operasional di jalur ini, hanya analisis data.

- **Fase Terminasi:** Seluruh aktivitas dari ketiga peran tersebut bermuara pada satu titik akhir, yaitu proses **Logout**, yang mengakhiri aktivitas dalam sistem.

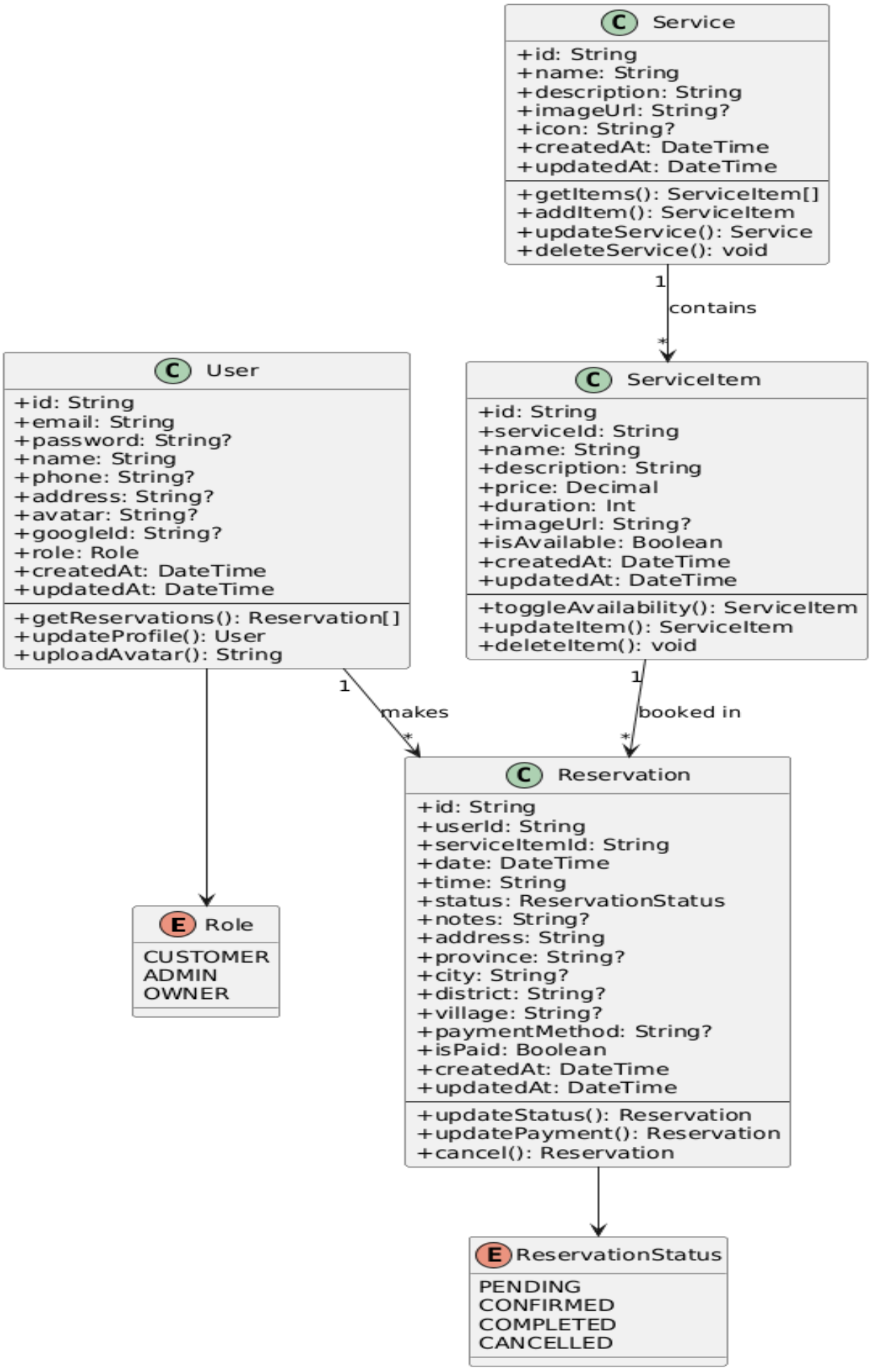
alamat pengiriman, kemudian mengirimkan pesanan.

Pada tahap pemrosesan transaksi, sistem membuat data transaksi dengan status PENDING, mengurangi stok motor secara otomatis, dan mengirimkan notifikasi pesanan baru kepada Cashier. Cashier kemudian memeriksa detail pesanan yang masuk. Apabila pesanan dinyatakan tidak valid, transaksi dibatalkan, stok motor dikembalikan, dan status transaksi berubah menjadi CANCELLED. Jika pesanan valid, cashier melanjutkan dengan memproses pembayaran dan memperbarui status transaksi menjadi PAID.

Setelah pembayaran selesai, proses masuk ke tahap pengiriman. Cashier melakukan penugasan Driver, kemudian sistem membuat data pengiriman dan mengubah status transaksi menjadi PROCESSING. Driver menerima tugas tersebut, mengambil unit motor dengan status PICKED UP, lalu mengantarkan motor ke alamat tujuan dengan status ON THE WAY.

Pada tahap akhir, driver menyerahkan motor kepada user dan memperbarui status pengiriman menjadi DELIVERED, sementara sistem mencatat waktu pengiriman. User selanjutnya melakukan konfirmasi penerimaan melalui aplikasi. Setelah konfirmasi dilakukan, sistem memperbarui status transaksi menjadi COMPLETED, yang menandakan bahwa seluruh proses pemesanan telah selesai.

3. Class Diagram



Class Diagram memvisualisasikan struktur statis sistem, menunjukkan kelas-kelas entitas, atribut yang dimilikinya, serta hubungan antar kelas (relasi) yang membentuk skema database.

- **Kelas User & Role:** Kelas `User` adalah entitas induk bagi semua pengguna. Atribut penting meliputi `googleId` (menyimpan token identitas OAuth), `email`, `password` (opsional jika menggunakan Google), dan `avatar`. Kelas ini memiliki relasi asosiasi dengan `Role`, yang merupakan sebuah *Enumeration* (Enum) berisi nilai konstan: `CUSTOMER`, `ADMIN`, `OWNER`. Ini menjamin integritas data bahwa pengguna hanya bisa memiliki salah satu dari tiga peran tersebut.

- **Kelas Service & ServiceItem (Manajemen Produk):** Terdapat relasi *Composition* (dilambangkan dengan *diamond* hitam) antara `Service` dan `ServiceItem`. Artinya, `ServiceItem` (produk spesifik) sangat bergantung pada `Service` (kategori). Jika kategori dihapus, item di dalamnya tidak relevan lagi.

    - `Service`: Berisi atribut umum seperti `name`, `description`, dan `icon`.

    - `ServiceItem`: Berisi detail spesifik seperti `price` (Decimal), `duration` (Int), `isAvailable` (Boolean), dan `imageUrl`. Kardinalitasnya adalah `1` Service memiliki `*` (banyak) ServiceItem.

- **Kelas Reservation (Inti Transaksi):** Ini adalah kelas transaksional utama. `Reservation` memiliki relasi asosiasi dengan `User` (User *makes* Reservation) dan `ServiceItem` (ServiceItem *booked in* Reservation).

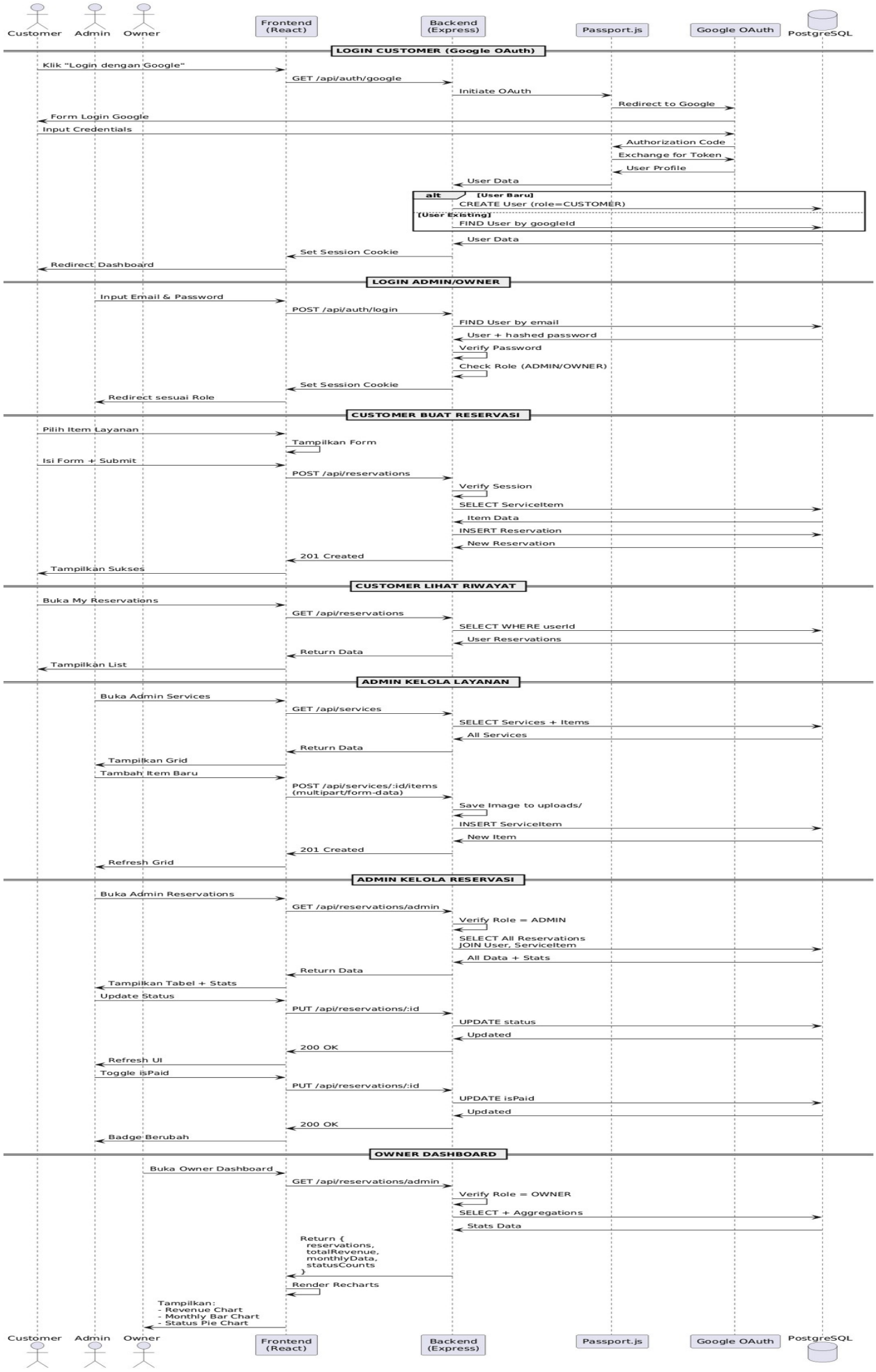
    - **Atribut:** Kelas ini menyimpan data logistik lengkap (`address`, `province`, `city`, `district`, `village`), data waktu (`date`, `time`), dan data keuangan (`paymentMethod`, `isPaid`).

    - **Status:** Status reservasi dikelola melalui relasi ke Enum `ReservationStatus`. Nilai yang diizinkan adalah `PENDING`, `CONFIRMED`, `COMPLETED`, dan `CANCELLED`. Penggunaan Enum mencegah kesalahan input status manual di luar alur bisnis yang valid.

- **Metode (Methods):** Setiap kelas mendefinisikan perilaku. Contohnya, `User` memiliki metode `getReservations()`, `ServiceItem` memiliki `toggleAvailability()` untuk mematikan stok sementara, dan `Reservation`

memiliki metode `updateStatus()` serta `cancel()` untuk mengubah *state* pesanan.

4. Sequence Diagram (Alur Interaksi Teknis)



Sequence Diagram merinci urutan pesan yang dipertukarkan antar komponen sistem (Frontend, Backend, Database, External API) berdasarkan waktu untuk skenario-skenario utama.

- **Skenario Login Customer (Google OAuth):** Proses dimulai ketika Customer mengklik "Login dengan Google". Frontend memanggil endpoint `GET /api/auth/google`. Backend (Express) menginisiasi OAuth ke Passport.js, yang kemudian me-redirect pengguna ke Google. Setelah Google memvalidasi kredensial pengguna, Google mengembalikan *Authorization Code*. Backend menukarkan kode ini dengan *Token* dan *User Profile*. Terdapat blok **Alternative (alt)** untuk logika bisnis: Jika `[User Baru]`, sistem menjalankan perintah `CREATE User` dengan role default CUSTOMER. Jika `[User Existing]`, sistem melakukan `FIND User by googleId`. Hasilnya adalah sesi pengguna terbentuk dan disimpan dalam Cookie browser.

- **Skenario Login Admin/Owner (Kredensial):** Menggunakan metode konvensional. Frontend mengirim `POST /api/auth/login` berisi email & password. Backend mencari email di database (`FIND User`). Jika ditemukan, backend membandingkan *hashed password*. Selanjutnya, sistem memverifikasi `Role`. Jika role sesuai (ADMIN/OWNER), sesi dibuat dan di-redirect ke dashboard masing-masing.

- **Skenario Customer Buat Reservasi:** Customer mengisi form dan Frontend mengirim `POST /api/reservations`. Backend memverifikasi sesi aktif, lalu melakukan `SELECT` ke tabel `ServiceItem` untuk memastikan harga dan ketersediaan valid. Setelah valid, Backend melakukan `INSERT Reservation` ke PostgreSQL. Database mengembalikan data reservasi baru, dan Backend mengirim respon `201 Created` ke Frontend untuk menampilkan pesan sukses.

- **Skenario Admin Kelola Layanan & Reservasi:**

    - **Tambah Item:** Admin mengirim data *multipart/form-data* ke `POST /api/services/:id/items`. Backend menyimpan file gambar ke folder *uploads*, lalu melakukan `INSERT ServiceItem` ke database.

    - **Update Status:** Admin mengirim `PUT /api/reservations/:id`. Backend memverifikasi bahwa `Role = ADMIN`, lalu menjalankan perintah `UPDATE status` di database.

    - **Toggle Paid:** Admin mengirim `PUT` untuk mengubah status pembayaran. Backend

mengupdate kolom `isPaid` di database. Setiap aksi ini diakhiri dengan respon `200 OK` yang memicu *refresh* UI (tabel/badge) di Frontend.

- **Skenario Owner Dashboard (Reporting):** Ketika Owner membuka dashboard, Frontend memanggil `GET /api/reservations/admin`. Backend memverifikasi `Role = OWNER`. Backend kemudian menjalankan kueri kompleks (`SELECT + Aggregations`) ke PostgreSQL untuk menghitung total pendapatan, jumlah reservasi per bulan, dan distribusi status. Data mentah dari database diolah menjadi format JSON (`totalRevenue`, `monthlyData`, `statusCounts`) dan dikirim ke Frontend. Frontend (React) menggunakan data ini untuk me-render komponen visual seperti *Bar Chart* dan *Pie Chart* menggunakan library Recharts.