System Architecture
for
LivingPlus

Prepared by

Muhammad Sibtain – 22i-0887

Abdullah Shakir – 22i-1138

Aqib Mehmood– 22p-9430

Identifying Subsystems

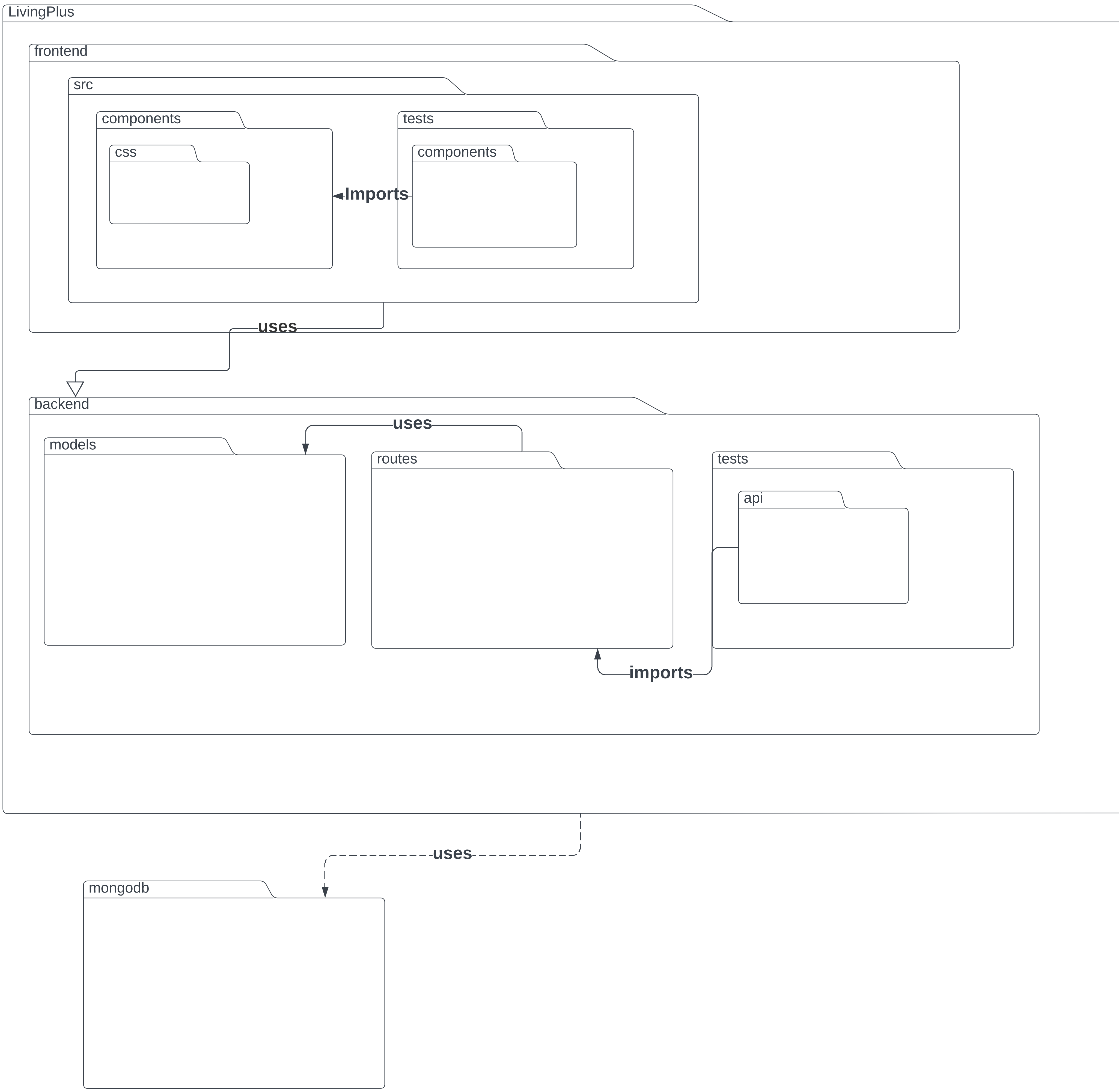
LivingPlus (Top-Level Package): This is the main package, representing the entire application.

frontend: This package encapsulates all the client-side code, built with React.js.

- **src:** The source directory, a standard convention in React projects, containing the main application code.
- **components:** This sub-package holds the reusable UI components of the application (e.g., login form, sidebar, dashboard).
- **CSS:** Contains the stylesheets for styling the application.
- **tests:** Includes tests for the frontend components.
- **Uses (Dependency Arrow to components):** Indicates that the src package uses or depends on the backend, to access the database.
- **Imports (Dependency Arrow from components):** the components package in test imports main components so that they can be tested.

backend: This package contains the server-side code, built with Node.js and Express.js.

- **models:** Defines the data structures for the application, using Mongoose to interact with MongoDB. (e.g., schemas for rentals, Tenant, owner, Payment).
- **routes:** Handles the API endpoints that define how the server responds to requests (e.g., /rentals, /tenants).
- **tests:** Includes tests for the backend logic.
- **uses (Dependency Arrow to models and routes):** Indicates that the frontend package uses both the models (for data access) and routes (to define the API).
- **imports (Dependency Arrow from routes):** Suggests that the routes package uses models.

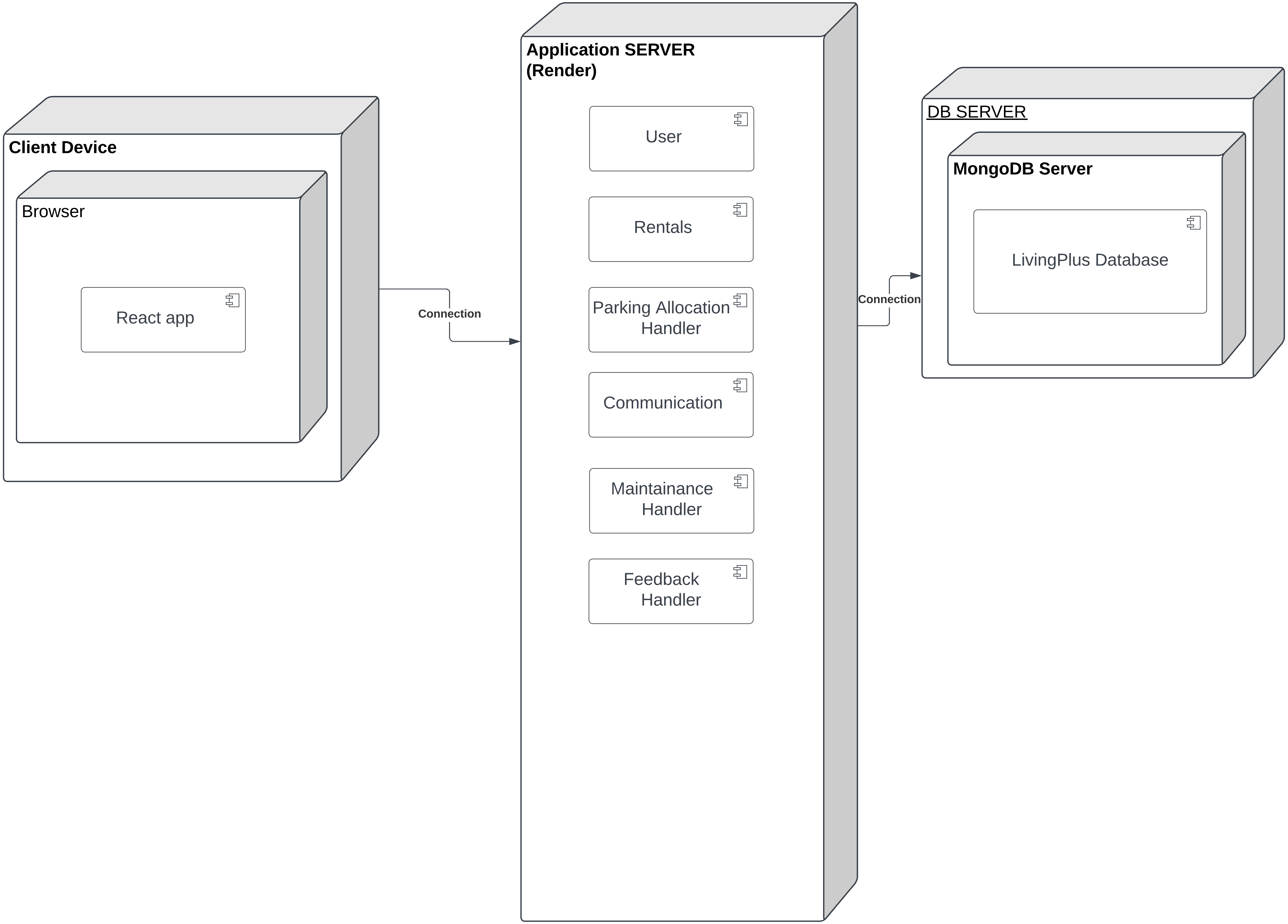

LivingPlus
frontend
src
components
css
tests
components
Imports
uses
backend
uses
models
routes
tests
api
imports
uses
mongodb

Architecture Styles

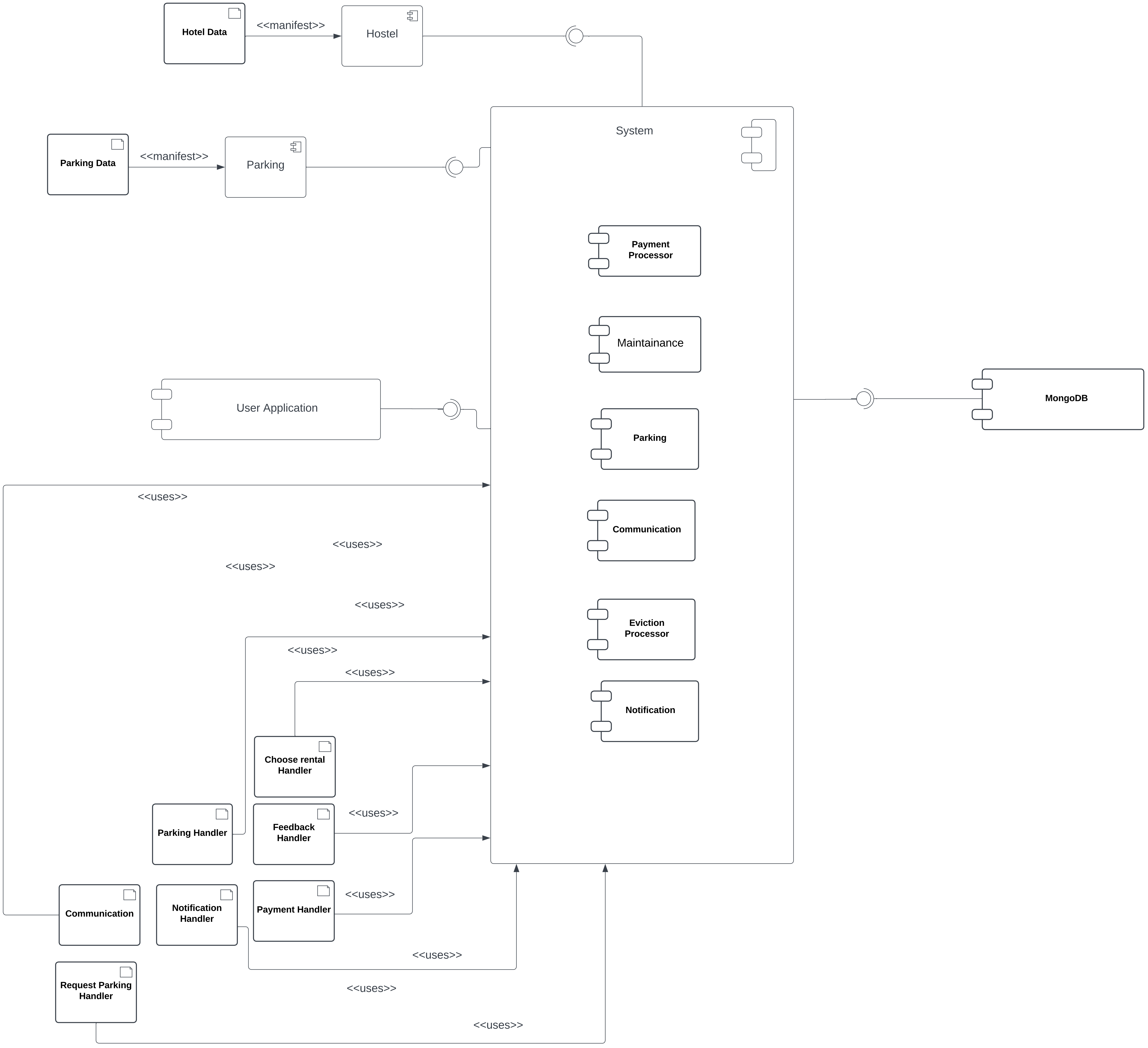
The LivingPlus web-based application is built using the MERN stack (MongoDB, Express.js, React.js, Node.js) and follows the **Model-View-Controller (MVC)** architectural style.

- **Model**: The data layer is handled using **MongoDB**, where all rental management-related data (e.g., properties, tenants, payments) are stored. The models define the structure of the database collections and are used to interact with the database.
- **View**: The frontend is developed using **React.js**, which acts as the View in the MVC pattern. React is responsible for presenting data to users and providing a dynamic, responsive user interface. It communicates with the backend via APIs.
- **Controller**: The controllers are implemented in **Express.js** within the Node.js backend. Controllers handle the business logic, processing incoming HTTP requests, interacting with the models, and sending appropriate responses back to the client.

By using MVC, LivingPlus separates concerns clearly, making the system more organized, easier to maintain, scalable, and testable. Each part (Model, View, Controller) is independently developed and maintained, improving collaboration and future extension of the application.


Client Device
Browser
React app
Connection
Application SERVER
(Render)
User
Rentals
Parking Allocation Handler
Communication
Maintainance Handler
Feedback Handler
Connection
DB SERVER
MongoDB Server
LivingPlus Database

8. Component Diagram


Hotel Data
<<manifest>>
Hostel
Parking Data
<<manifest>>
Parking
System
Payment Processor
Maintainance
User Application
MongoDB
Parking
Communication
Eviction Processor
Notification
<<uses>>
<<uses>>
<<uses>>
<<uses>>
<<uses>>
<<uses>>
Choose rental Handler
Parking Handler
Feedback Handler
<<uses>>
Communication
Notification Handler
Payment Handler
<<uses>>
<<uses>>
Request Parking Handler
<<uses>>
<<uses>>