Software Architecture Document

Fontys University of Applied Science | Rachelsmolen 1, 5612 MA Eindhoven

Driessen

Sanchez, Francisco; Thomas, Gilton; Pestana, Cristiano; WerneckRoale, Miguel; Le, Minh

2024

Table of Contents

[C4 Model Diagrams 2](#_Toc183375085)

[Level 1: System Context Diagram 2](#_Toc183375086)

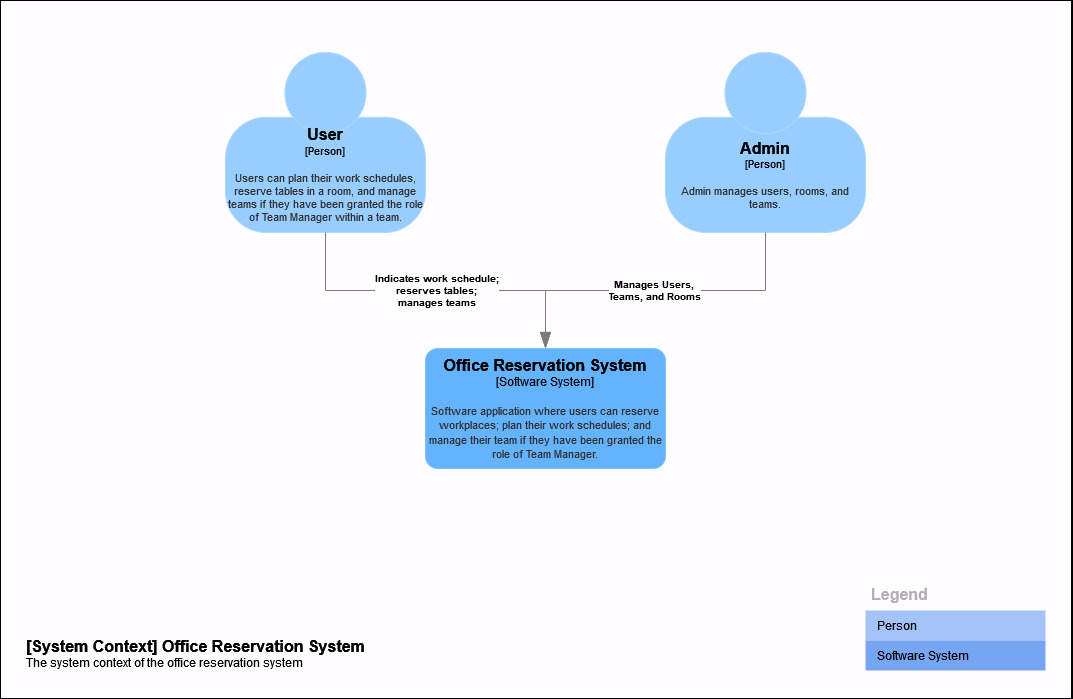
[Level 2: Container Diagram 3](#_Toc183375087)

[Level 3: Component Diagram 5](#_Toc183375088)

# C4 Model Diagrams

## Level 1: System Context Diagram

*A System Context diagram provides a starting point, showing how the software system in scope fits into the world around it.*



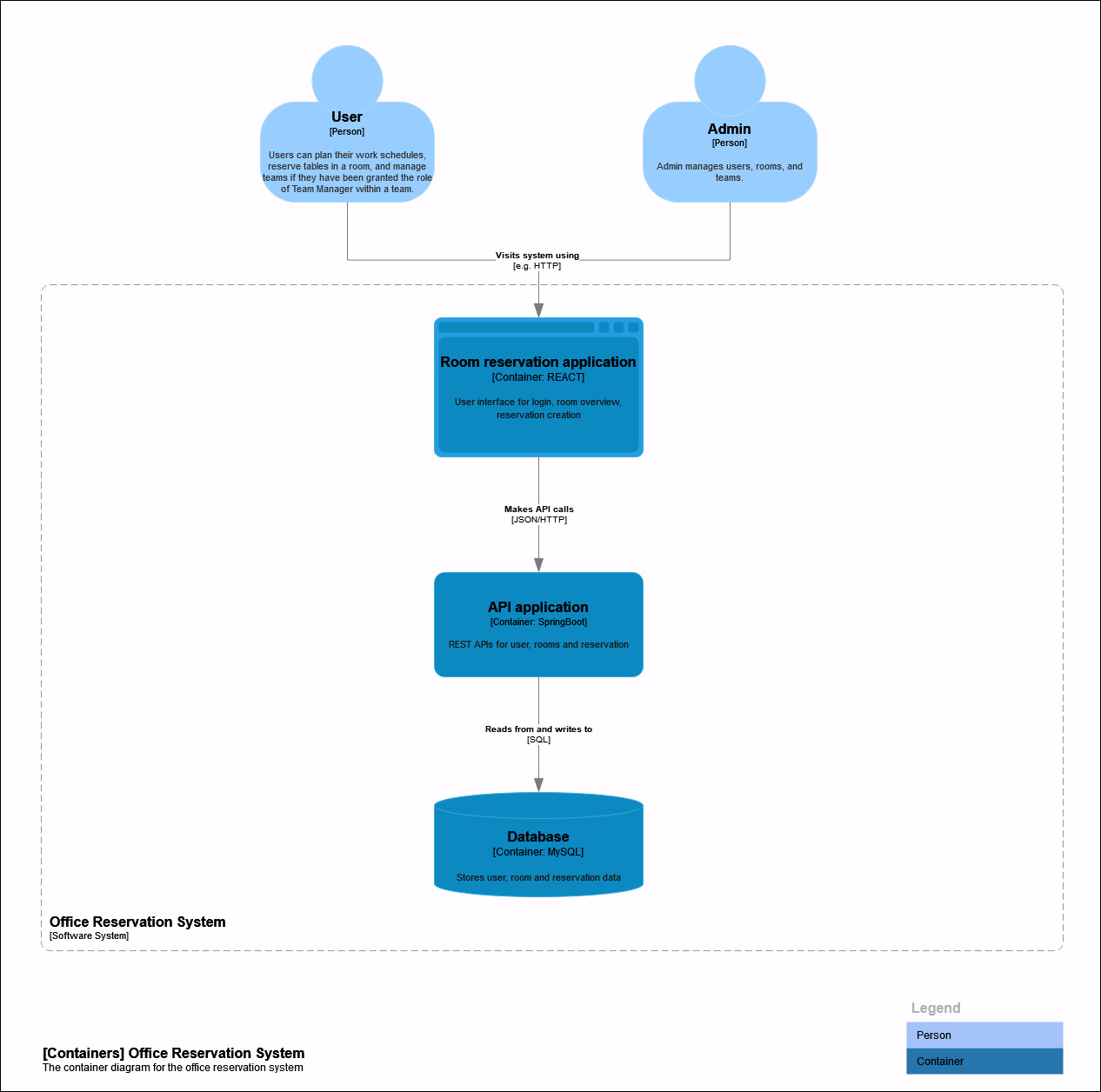
The system context diagram depicts the interactions between the users and the Office Reservation System. The system primarily interacts with two types of users:

1. Users: Regular users can plan their work schedules, reserve tables in rooms, and manage teams if granted the ‘Team Manager’ role within their team.
2. Admins: Admins are responsible for managing users, teams, and rooms.

The Office Reservation System is an application that facilitates these functionalities, serving as the main point of interaction for both user types.

## Level 2: Container Diagram

*A Container diagram zooms into the software system in scope, showing the high-level technical building blocks.*



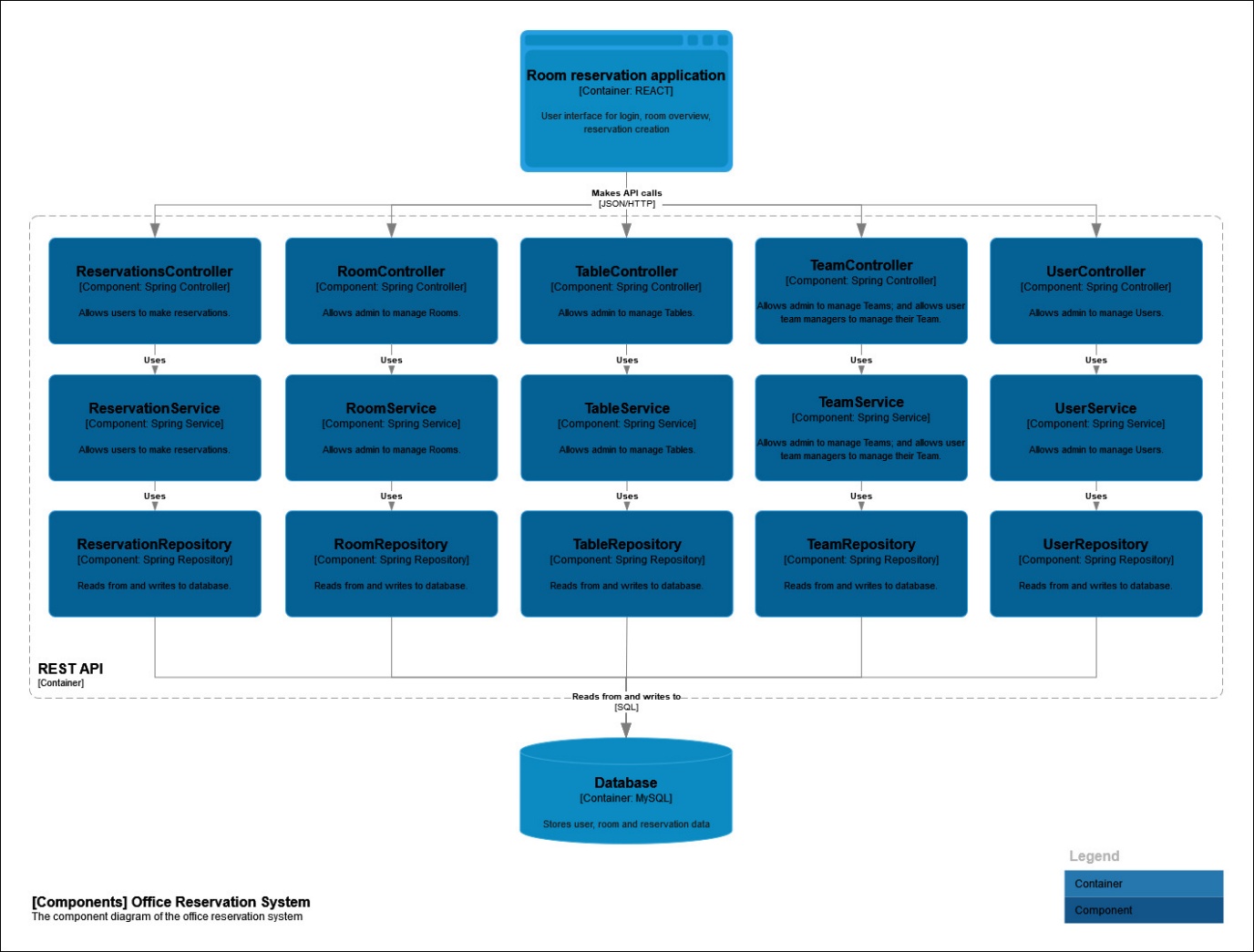
The container diagram illustrates the internal structure of the Office Reservation System, showing how various components interact. Key containers include:

1. Room Reservation Application (React): This is the user interface where users log in, view room availability, and make reservations.
2. API Application (Spring Boot): Serves as the backend, providing RESTful APIs for user, room, and reservation management. It handles requests from the Room Reservation Application.
3. Database (MySQL): Stores critical data, including user profiles, team information, room details, and reservation records.

The flow of data begins with users accessing the system through the React-based interface, which communicates with the API for all operations, with the database storing and retrieving persistent data.

## Level 3: Component Diagram

*A Component diagram zooms into an individual container, showing the components inside it.*



The component diagram breaks down the API application into smaller, focused components, showing their responsibilities and interactions:

1. Controllers: These handle incoming HTTP requests:
   1. ReservationsController: Manages reservations.
   2. RoomController: Manages rooms.
   3. TableController: Manages tables.
   4. TeamController: Manages teams.
   5. UserController: Manages users.
2. Services: These components implement business logic:
   1. ReservationService: Processes reservation-related operations.
   2. RoomService: Manages room-related tasks.
   3. TableService: Handles table-related logic.
   4. TeamService: Manages teams and roles.
   5. UserService: Administers user operations.
3. Repositories: Provide data access to and from the database:
   1. ReservationRepository, RoomRepository, TableRepository, TeamRepository, UserRepository.

The database remains the core data store, connected to repositories for seamless data persistence and retrieval. The REST API serves as the backend, implementing business logic and database interactions to provide services for the Room Reservation Application (frontend).