



Adventist University of Central Africa

P.O. Box 2461 Kigali, Rwanda | www.auca.ac.rw | info@auca.ac.rw

WEBTECH FINAL PROJECT

HOTEL RESERVATION SYSTEM

NAME: Manzi Danny

ID: 24470

Table of Content

1. Project Requirement(Functional and Non-Functional requirements)
2. Project Plan
3. Source Code
4. Database Schema
5. User Documentation
6. Technical Documentation

1. Project Requirement(Functional and Non-Functional requirements)

1.1 Functional Requirement of the system

1. User Authentication and Account Creation:
 - 1.1. Allow all users to log in and create an account within the system.
2. User Exploration and Reservation:
 - 2.1. Enable users or clients to log in and explore system functionalities, including viewing available rooms and services.
 - 2.2. Provide the capability to reserve rooms at a specified time and monitor the reservation progress.
3. Room Reservation Process:
 - 3.1. Allow clients to browse available rooms, select their desired room, and make a reservation.
4. Reservation Reporting:
 - 4.1. Implement a system feature to generate reports or lists for users, detailing the reservations they have made.
5. User Logout Functionality:
 - 5.1. Provide users with the option to log out of the system at their discretion.
6. Authentication and Authorization:
 - 6.1. Implement robust authentication and authorization using Spring Security for all users seeking access to privileged system resources.

1.2 Non-Functional Requirement of the System

1. User Data Privacy and Security:
 - 1.1. Ensure user data's utmost privacy and security, restricting access to only authenticated, authorized individuals with the appropriate privileges.
2. System Performance and Availability:

- 2.1. Maintain high system performance and availability to ensure seamless operation at all times.
3. User-Friendly Interface Design:
 - 3.1. Design an intuitive, visually appealing, and easily navigable interface for the system to enhance user experience.
4. Compatibility Across Devices and Browsers:
 - 4.1. Guarantee compatibility with various devices and browsers, ensuring the Hotel Reservation System functions correctly and consistently across different platforms.

2. Project Plan

1. Project Scope:

- Develop a basic Hotel Reservation System for individual users within a two-week timeframe.

2. Project Timeline:

Week 1: Planning and Development

- Days 1-2: Project Setup
 - Define project scope and requirements.
 - Set up the development environment.
- Days 3-5: Design and Architecture
 - Design the system architecture.
 - Plan the user interface.
- Days 6-7: Initial Development
 - Implement user authentication and account creation.

Week 2: Implementation and Testing

- Days 8-10: Core Functionality
 - Develop room exploration, reservation, and reporting features.
- Days 11-12: Testing and Refinement

- Conduct unit testing and resolve bugs.
- Optimize system performance.
- Day 13-14: Deployment and Documentation
 - Deploy the system for testing.
 - Document code and provide user instructions.

3. Resources:

- Programming Languages:
 - Java (for backend development)
 - HTML, CSS, JavaScript (for frontend development)
- Frameworks:
 - Spring Boot (for backend development)
 - Thymeleaf (for frontend templates)
- Database:
 - MySQL (for data storage)
- Development Tools:
 - IntelliJ IDEA (IDE for Java development)
 - Git (version control)
- Documentation:
 - Markdown for documentation.

3. Source Code

When you clone a Spring Boot project from GitHub, you must ensure that you have the necessary configuration files and scripts.

1. `application.properties`:

Adjust the properties in this file according to your actual database setup.

```
# src/main/resources/application.properties
```

```
spring.datasource.url=jdbc:mysql://localhost:3306/hotel_reservation
```

spring.datasource.username=root

spring.datasource.password=password

spring.jpa.hibernate.ddl-auto=update

2. pom.xml:

Ensure your `pom.xml` file contains the necessary dependencies for Spring Boot and Thymeleaf.

3. Running the Application:

When you clone the repository, navigate to the project's root directory and run the following command to build and run the Spring Boot application:

```
./mvnw spring-boot: run
```

This assumes you are using the Maven Wrapper (`mvnw`). If not, you can use `mvn` instead.

Visit `http://localhost:8080` in your web browser to access the application.

4. Database Schema

The hotel reservation system consists of 3 tables described below:

-Users: Contains all the information of the user of the application.

-Reservation: Contains all information about rooms, services, and orders to be requested by the clients.

-Role: Specify roles for each user, implementing role-based authentication.

Relationships:

Users and Roles (Many-to-Many):

- A User can have multiple Roles.
- A Role can be associated with multiple Users.

Join Table (users_roles):

- user_id (Foreign Key referencing Users Table)
- role_id (Foreign Key referencing Roles Table)

Users and Reservations (One-to-Many):

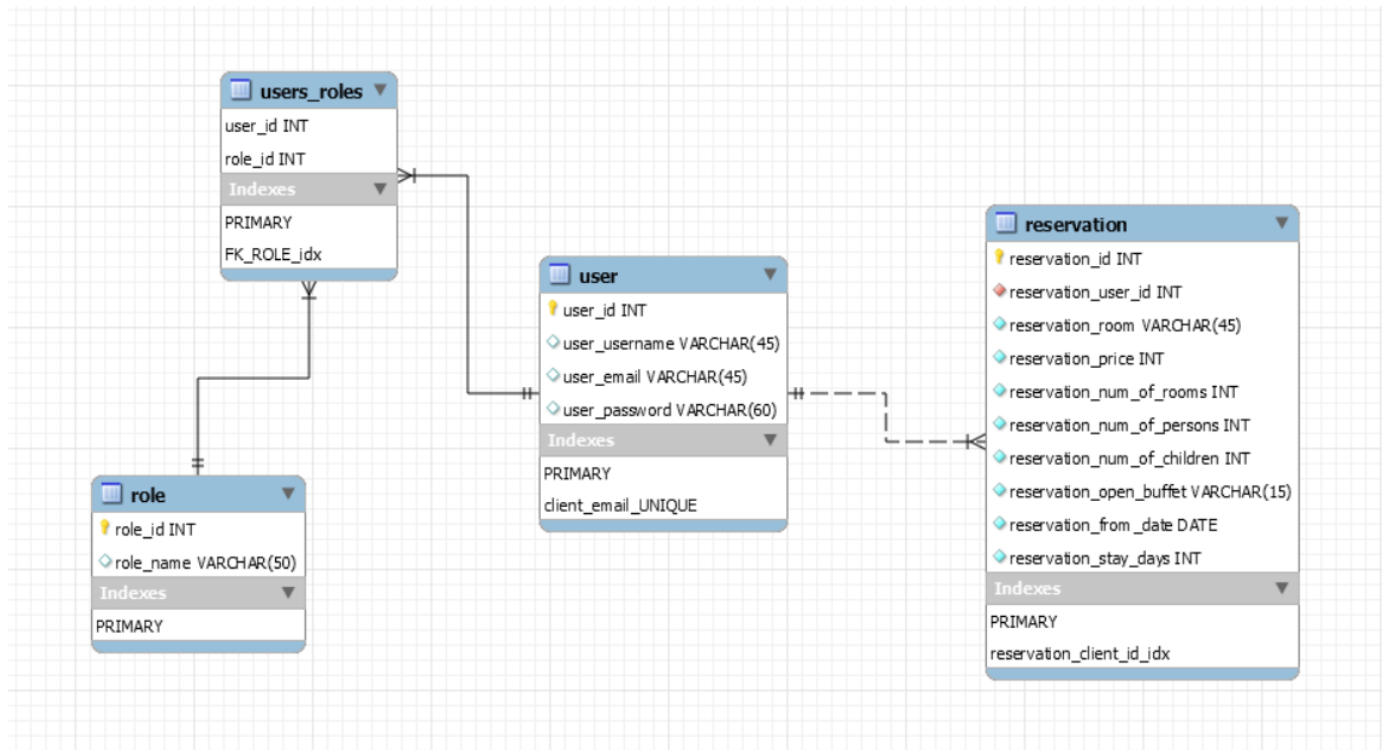
- A User can have multiple Reservations.

- A Reservation is associated with only one User.

Foreign Key in Reservations Table (reservation_user_id):

- reservation_user_id (Foreign Key referencing Users Table)

ERD DIAGRAM



5. User Documentation

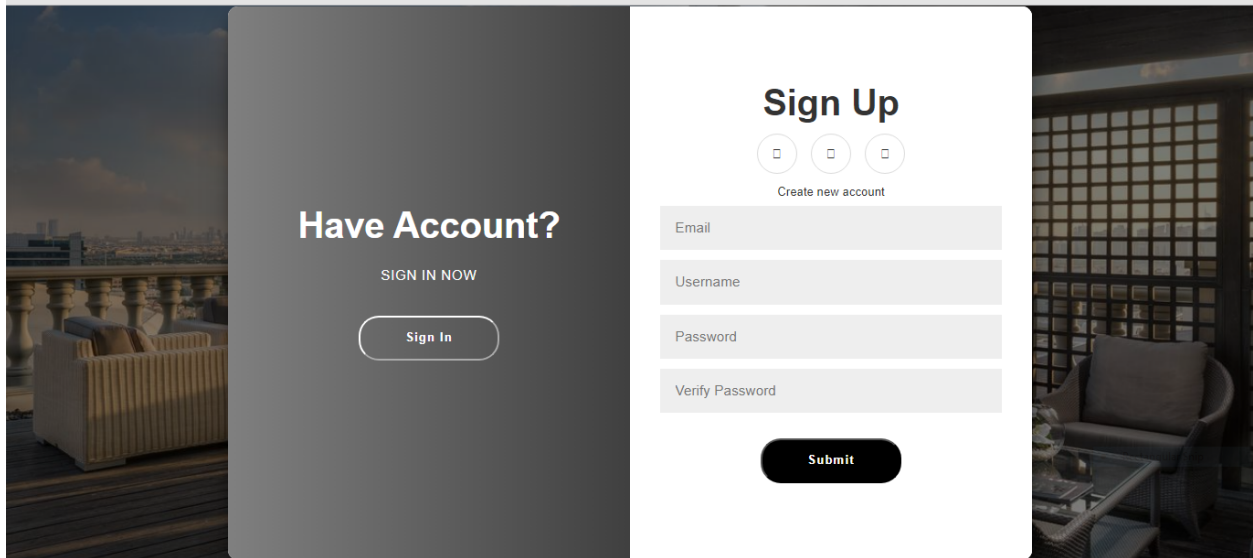
1. Accessing the System:

- Run your Spring Boot application.
- Open your web browser and navigate to the localhost:8080.
- You will be greeted by a login page.

2. Login and Sign Up:

- If you don't have an account create one.
- Use the provided credentials to log in:
 - Email: [Your email]

- Password: [Your password]



The image shows a 'Sign Up' form overlay on a background image of a hotel balcony with a city skyline. The form is white with a dark border. It has a title 'Sign Up' and three social media icons (Facebook, Twitter, Google+) with the text 'Create new account' below them. The form fields are: Email, Username, Password, and Verify Password. A 'Submit' button is at the bottom right. To the left of the form, there is a dark grey panel with the text 'Have Account?' and 'SIGN IN NOW' with a 'Sign In' button.

Sign Up

Create new account

Email

Username

Password

Verify Password

Submit

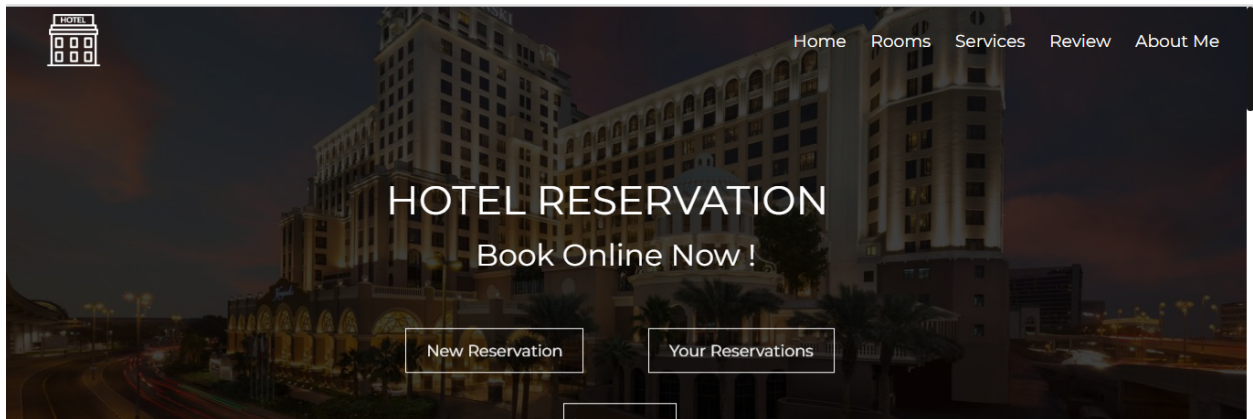
Have Account?

SIGN IN NOW

Sign In

3. HomePage:

- Upon successful login, you will land on the system HomePage.
- The HomePage provides an overview of reservation details, services, and your account information.



Hotel Rooms and Suites

Hotel Rooms and Suites

Single Room:

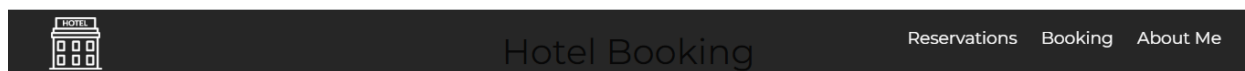


Double Room:



4. Making a Reservation:

- Navigate to the " New Reservations" button.
- Explore available rooms and services.
- Select your desired room, specify additional services, and confirm the reservation.



Booking Form

Room/SuiteType
Single Deluxe

Period of Stay
0

Number of room/suite
0

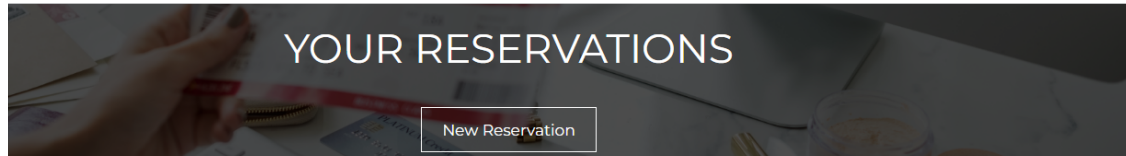
Number of persons

5. Viewing Reservations:

- Head to the "My Reservations" section to view details of your existing reservations.
- Check the status, room details, and any additional services associated with each reservation.

6. Managing Your Reservations:

- You can update and delete your reservation order based on your preference.



User Reservations

Reservation List

Res ID	Room	Rooms	Persons	Children	Open Buffet	Stay Period	Arrival Date	Total Price	Action
2	Single Standard	1	1	0	YES	3	2023-12-30 00:00:00.0	1100	<div><div>Update</div><div>Delete</div></div>

7. Role-Based Authentication:

- The system employs role-based authentication for enhanced security and personalized experiences.
- Your role determines the features and privileges available to you within the system.

8. Logging Out:

- To log out, click on the "Log Out" button.

6. Technical Documentation

1. Architecture Overview:

- The Hotel Reservation System is designed as a web application with a client-server architecture.
- The backend is implemented using the Spring Boot framework, and the frontend utilizes Thymeleaf templates.

2. Backend Implementation:

- Spring Boot Framework:
 - The backend is developed using Spring Boot, providing a robust and scalable foundation.
 - Spring Boot enables rapid development with its convention-over-configuration approach.
- Spring Data JPA:
 - JPA (Java Persistence API) is used for data access.
 - Spring Data JPA simplifies database operations, allowing seamless integration with relational databases.
- Database:
 - The system uses a relational database (e.g., MySQL) to store user, reservation, and role information.
 - Hibernate, a JPA implementation, is used for ORM (Object-Relational Mapping).
- Spring Security:
 - Spring Security is implemented for authentication and authorization.
 - Role-based access control ensures secure access to different features based on user roles.

3. Frontend Implementation:

- Thymeleaf Templates:
 - Thymeleaf is employed for server-side templating.
 - Dynamic content is seamlessly integrated into HTML templates.
- HTML, CSS, and JavaScript:
 - Standard web technologies are used for crafting the user interface.
 - HTML structures the content, CSS styles it and JavaScript enhances user interactions.

4. Database Schema:

- The relational database schema includes tables for users, reservations, and roles.
- Relationships are established through foreign key constraints.

5. Build and Dependency Management:

- Maven:
 - Maven is used for project management and build automation.
 - Dependencies and plugins are defined in the `pom.xml` file.

6. Version Control:

- The source code is managed using a version control system (e.g., Git) with a repository hosted on GitHub.

7. Security Measures:

- Passwords are securely stored using hashing algorithms.
- HTTPS is enforced for secure communication.

8. Future Enhancements:

- Suggestions for future improvements include implementing additional features such as email notifications, more robust reservation management, and further optimization for scalability.

This documentation provides an insight into the architecture, technologies, and implementation details of the Hotel Reservation System. Further details and clarification can be found in the source code and associated documentation files.

THANKS!

