Hotel Reservation System - Project Design Document

Name: Pooja Nadagouda

The main aim of hotel reservation system is to allow customers to reserve rooms in the hotel through online booking. The functional and non-functional requirements are given below:

FUNCTIONAL REQUIREMENTS:

This hotel reservation system provides following services:

1. Provide room availability details for the given dates
2. Reserve rooms for required dates
3. Send booking confirmation emails to customers
4. View booking details by using Booking ID
5. View the types of rooms and facilities offered for the room by the hotel
6. View the contact information of the hotel along with google map location.

This hotel reservation system can be used by the hotel administration to take online bookings from the customers. This eliminates the manual task of taking bookings, keeping track of available rooms, explaining the facility details offered for different room types.

NON- FUNCTIONAL REQUIREMENTS:

1. Scalability – as the number of requests increase or decrease the hotel reservation service should be scaled up or down.
2. Availability – hotel reservation service should be highly available (e.g.: P99.9)
3. Consistency – the requests to the hotel reservation service should be highly consistent.

Note: Functionalities like payment completion, admin actions like add/remove room are not done as part of this project. These functions can be provided as part of add-ons depending on the time constraints.

ARCHITECTURE:

The hotel reservation system consists of 2 main components – web service and web application. The web service is a REST based. The web application consists of a client and a user interface. The user can use a browser on his device to access the hotel reservation system. The requests made from the browser are handled by the web application and the client within the web application calls the web service. For cases which requires interaction with AWS services (AWS SES) or fetching information from database (e.g.: Booking a room), web application client makes appropriate requests to the web service. The web service processes the request and sends the processed data to the client. Now the web application sends this data to the user interface which renders HTML content to the user as response. For situations which require to show static data (e.g.: view hotel contact information), web application client sends appropriate response to the user interface. Below *Figure 1* shows the overall high-level architecture diagram for hotel reservation system.

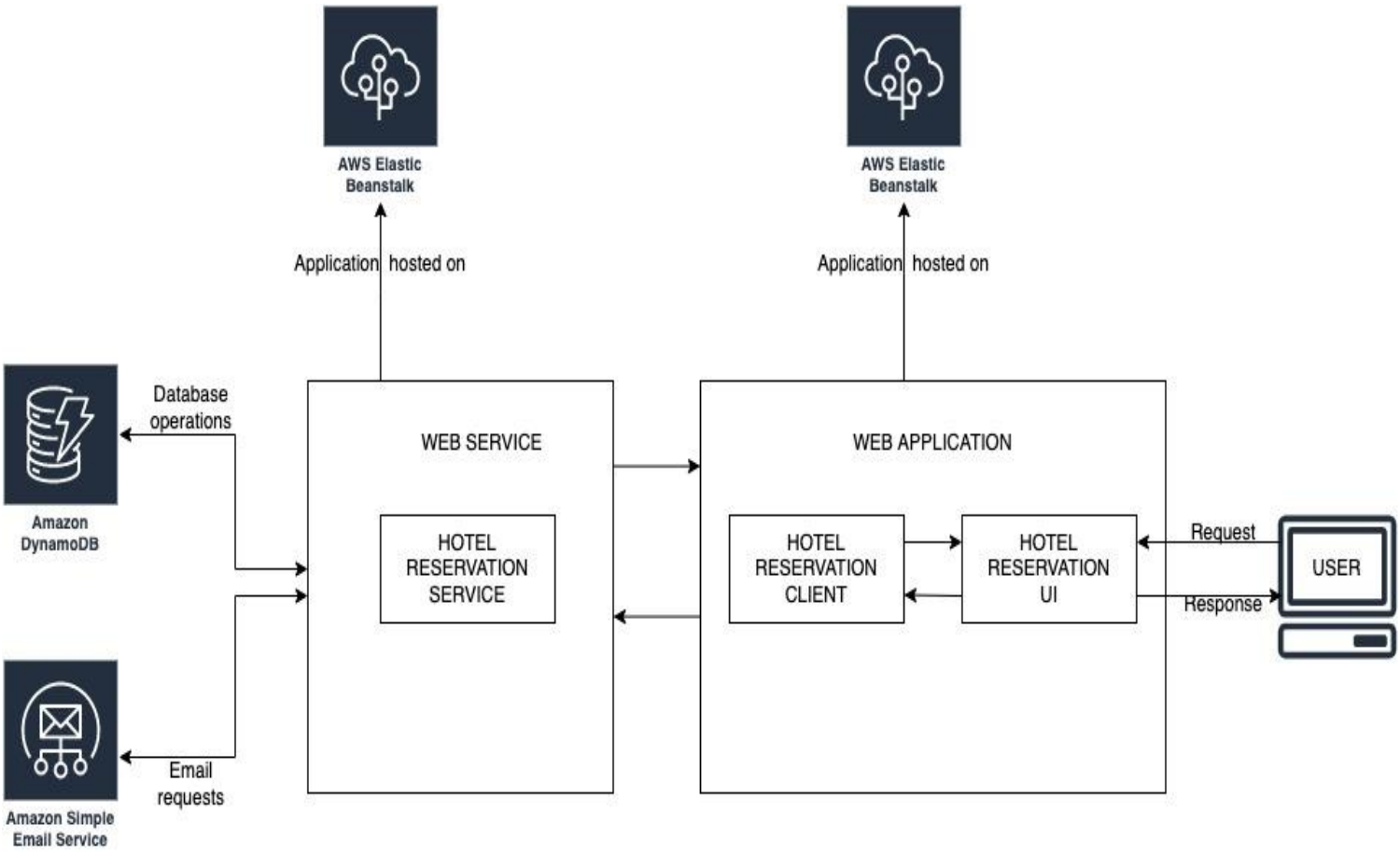

Figure 1: Architecture diagram for Hotel Reservation System

The web service and web application are hosted on AWS Elastic Beanstalk. The web application handles every event performed on the UI. The UI holds all the JSP pages required for hotel reservation system. The web service provides all the services which require processing or interaction with database and AWS services.

HOTEL RESERVATION UI:

This is visible for the customers on their browser i.e., it is responsible for the user interface of the hotel reservation system.

The hotel reservation system has the following tabs:

1. **HOME TAB:** In this page user can enter check-in and check-out dates and check room availability.
   Subpages in the hotel reservation page: These pages are displayed when user is booking the room:
   In the home page, user enters check-in and check-out dates, on clicking the check availability button RoomSelection Page is displayed with available room types. The user can click BOOK NOW button - on the type of room he wishes to book. This displays BookingDetails Page, where user should enter all the details to complete the booking.

2. **ROOMS TAB**: This page shows details of all types of rooms and its facilities offered by the hotel.
3. **BOOKINGS TAB**: In this page the user can enter Booking ID to view his booking details
4. **CONTACT TAB**: This page shows contact details of the hotel along with the google map location.

One controller is created for each tab. Following controllers are created for this project - HomeTabController, RoomsTabController, BookingsTabController, ContactTabController.

HOTEL RESERVATION CLIENT:

The controllers use hotel reservation client to make requests to hotel reservation services. Also, the client handles both positive and negative responses from the service and gives appropriate message to display on the UI.

HOTEL RESERVATION WEB SERVICE:

This is responsible for processing the requests which require interaction with the database.
The hotel reservation system provides the following services:

RoomService:

Provides services related to room details like adding a new room type, fetching details of a room type.

Interfaces: /Room/

- /Room/POST – adds new room-type to the database
- /Room/GET – gets details of a room-type from the database

Here the negative scenario can be entering wrong room-type, appropriate error messages are shown to the user.

RoomAvailabilityService:

Provides service related to room availability like adding availability for a date, updating an availability based on bookings added, fetching availability for given date.

Interfaces: /RoomAvailability/Availability/

- /RoomAvailability/Availability/POST – adds room availability details for a particular date to the database
- /RoomAvailability/Availability/GET – gets the room availability details for a given date from the database
- /RoomAvailability/Availability/PUT – updates the room availability details for a given date to the database

Here negative scenario can be checking rooms availability for past dates.

RoomBookingService:

Provides services related to room booking like adding new booking, fetching details of a booking.

Interfaces: /RoomBooking/Bookings

- /RoomBooking/Bookings/POST – adds the new booking details record to the database
- /RoomBooking/Bookings/GET – gets the booking details from the database for given booking ID

Here negative scenarios can be user entering wrong booking ID, invalid appropriate error messages are shown to the user. For adding new booking, negative scenario can be providing invalid booking details.

DATABASE SCHEMA:

For this project using the AWS DynamoDB for storing the data.
Following 3 tables are required for this project:

1. ROOM_DETAILS_TABLE
2. BOOKING_DETAILS_TABLE
3. ROOM_AVAILABILITY_TABLE

ROOM_DETAILS_TABLE Schema:

- RoomType (String) – Partition key – refers to type of the room e.g.: single, deluxe, suite etc.
- Facilities (String) – Facilities offered for the room
- Capacity (Integer) – Max number of people room can accommodate
- Price (Double) – Price of the room
- Total_rooms (Integer): Total number of rooms in the hotel

BOOKING_DETAILS_TABLE Schema:

- BookingID (String) – Partition key – ID assigned for each booking
- First_Name (String) – First name of the person doing the booking
- Last_Name (String) – Last name of the person doing the booking
- Address (String) - Address of the person doing the booking
- City (String) - City of the person doing the booking
- State (String) - State of the person doing the booking
- Country (String) - Country of the person doing the booking
- Pincode (String) - Pincode of the person doing the booking
- Phone (String) - Phone of the person doing the booking
- Email (String) - Email of the person doing the booking
- Check_in (String) – check in date for the booking
- Check_out (String) - check out date for the booking
- Rooms_booked (Integer) – count of rooms booked
- RoomType (String) – type of room booked
- Adults_count (Integer) – number of adults to be hosted
- Price (Double) – total price for the booking

ROOM_AVAILABILITY_TABLE Schema:

- Availability_Date (String) – Partition key – calendar date (includes dates up till next 6 months) (Here at the end of current day, a new availability record is added automatically to the table. A task is scheduled to add new availability record for a new day - for a date 6 months after the current).
- Availability_Details (JSON) –JSON string with room-type as key and count of available rooms as value.
  E.g.: 04/13/2022 {“single_room”:20, “deluxe_room”:15, “suite_room”:10}

END-TO-END DATA FLOW FOR A SERVICE:

EXPLANATION FOR DATA FLOW:

Consider user wants to check room availability function. End-to-end data flow is shown below in *Figure 2.* Using the browser user visits the home page of hotel reservation system. User enters the check-in / check-out dates, and clicks check availability button. The home tab view sends this request to the client - HomeTabController.

The controller calls the service: *RoomAvailability/Availability/GET*

The web service interacts with the database table ROOM_AVAILABILITY_TABLE to fetch the room availability. The service returns a JSON response of availability count for each room type. The controller processes the JSON response to a map of room-type and availability count. This is sent to the appropriate JSP page i.e., RoomSelection page. The UI displays the available room details for the user on the browser.

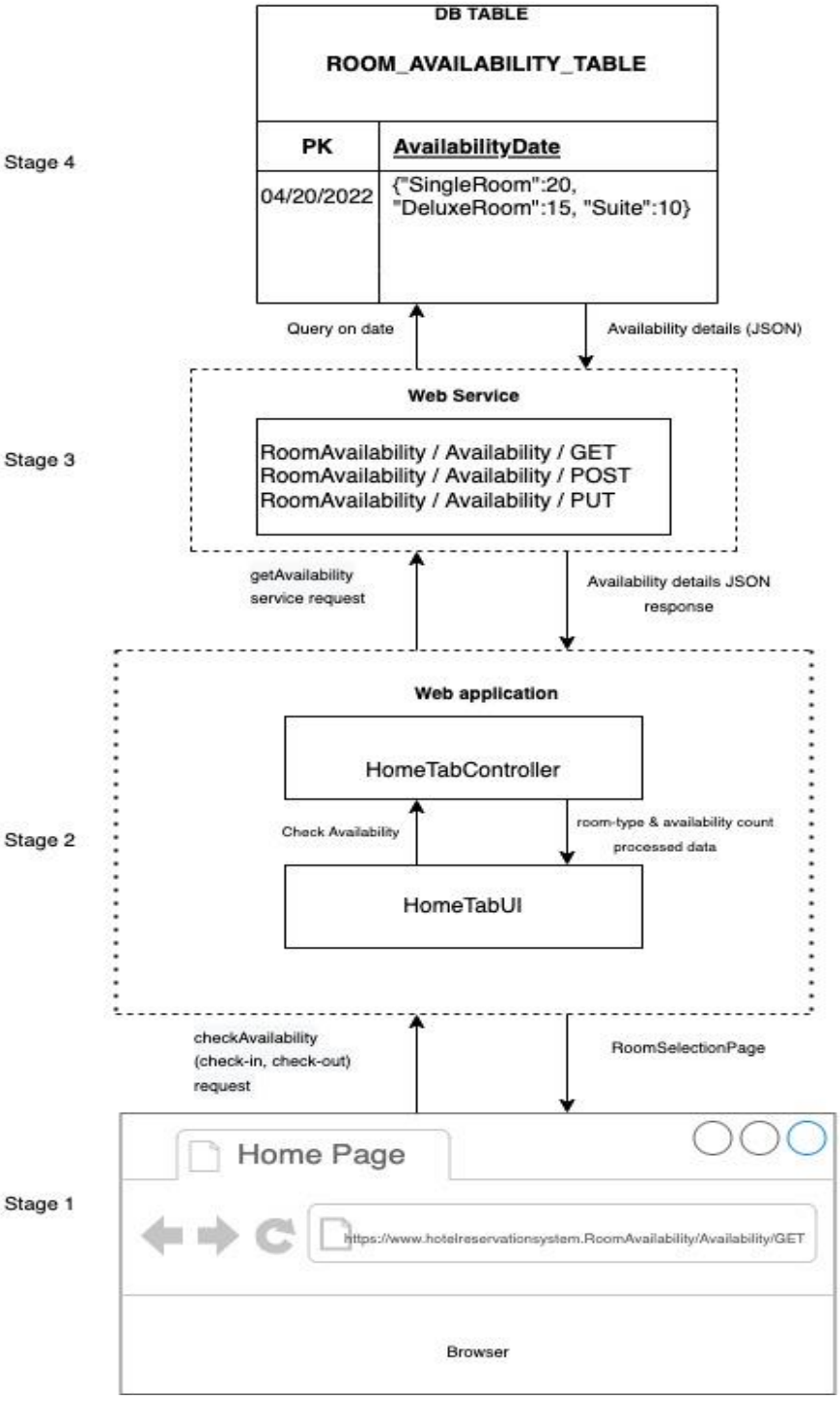

Figure 2: Check Room Availability function end-to-end data flow

TECHNOLOGIES, LANGUAGES, FRAMEWORK USED FOR EACH TIER:

- **Technologies:**
  - AWS Dynamo DB for database
  - AWS Simple Email Service (SES) for sending email notifications to users
  - AWS Cloud watch to monitor the metrics
  - AWS Elastic beanstalk to host the web application
- **Languages:**
  - Java for backend i.e., for web service and web application
  - HTML, CSS, JavaScript, JSP for user interface
- **Framework:**
  - Spring boot
- **Testing:**
  - JUNIT for module testing