Online Booking System

Arleth Martinez

CST-451 Capstone Project Final Architecture & Design

Grand Canyon University

Revision: 2.0

Date: August 3, 2024

**ABSTRACT**

The online hotel booking system aims to develop a user-friendly web application that enables users to select, view, and reserve hotel rooms online. It will feature a database of hotel rooms with room availability, pricing information, and details. Users will be able to create accounts and manage their bookings. The application seeks to streamline the hotel booking process enhancing convenience for users.

|  |
| --- |
| History and Signoff Sheet |

**Change Record**

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Revision Notes** |
| 11/04/2023 | Arleth Martinez | Initial draft for review/discussion |
| 08/03/2024 | Arleth Martinez | Finalized details |
|  |  |  |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |

**Integrated Instructor Feedback into Project Documentation**

Yes  No

**TABLE OF CONTENTS**

Design Overview 4

Detailed High-Level Solution Design 5

Detailed Technical Design 6

Appendix A – Technical Issue and Risk Log 7

Appendix B – References 8

Appendix C – External Resources 9

Design Introduction

1. Provide the high-level design of the proposed solution or business case with supporting narrative text. This design should include mock-up screenshots for the proposed user interface, pseudocode, or flowcharts that show the logic for the program, as well as the anticipated process flow. The purpose of the solution/business case design is to allow the stakeholder to approve the concepts before committing resources to the technical design.

The hotel room database will contain information about each hotel room such as price, availability, room number, etc. The user database will be used to store and authenticate user information. The web app will directly connect to the user database to store newly created users and validate user login information. The hotel room database will be accessible only to the web app via the Room API interface. Through this interface, the web app will be able to perform CRUD operations such as reading all the available rooms and updating the availability. Here is the high-level design:

**A diagram of a computer

Description automatically generated**

Once the User creates an account and is logged in, the application will take both data from the User database and Hotel Room API to display the hotel rooms. The user can then use the application to book a room via the UI shown below:

A screenshot of a hotel

Description automatically generated

The UI then updates the database via the hotel room API showing the as now booked by Username. Here is the overall process diagram:

**A diagram of a hotel room

Description automatically generated**

1. Use the template to list the project deliverables that are to be included external to this Design Specification (Data Dictionary, API Design, etc.).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Deliverable Acceptance Log | | | | | |
| ID | Deliverable Description | Comments | Evaluator (internal or external as applicable) | Status | Date of Decision |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |

***NOTE: If necessary, you may add subsections to those listed in order to match the requirements in the assignment description. Do not remove any top level sections (those that are listed in the Table of Contents).***

Detailed High-Level Solution Design

1. Provide a detailed overview of how the proposed design fits into the overall solution.
   1. Create diagrams to logically and physically depict the system. This can be illustrated using UML Component, UML Deployment, and UML Activity diagrams or simply block diagrams done in a drawing tool such as Visio.
   2. This section should also include any solution configuration changes that will be required to develop and implement the proposed solution.
   3. Describe the approach and resources required to assure non-functional requirements (such as security and performance) will be met with this solution.
   4. The purpose of the detailed solution architecture is to provide sufficient information for a developer to produce the system.
2. Provide a detailed inventory of hardware and software technologies that will be used in the solution:
   1. List any Frameworks or third party libraries that will be used.
   2. List any Proof of Concepts to be completed (POC) to ensure that the technologies and frameworks selected are the best fit for purpose, cost effective, and proper to solve the problem. This section should also be updated with the purpose/rationale for the POC and the results of the POC.

Use the templates below to list the Proof of Concepts, hardware, and technologies.

|  |  |  |
| --- | --- | --- |
| Proof of Concepts | |  |
| **Description** | **Rationale** | **Results** |
| 1. |  |  |
| 2 - |  |  |
| 3 - |  |  |
| 4 - |  |  |
| 5 - |  |  |

|  |
| --- |
| Hardware and Software Technologies |
| 1 – React JS |
| 2 – MySQL |
| 3 – Node.js (Express Framework) |
| 4 - Postman |
| 5 -Bootstrap Framework |

**Logical Solution Design:**

A diagram of a software system

Description automatically generated

**Physical Solution Design:**

A diagram of a server

Description automatically generated

Detailed Technical Design

**General Technical Approach:**

To develop a hotel booking system with ReactJS, Node.js, and MySQL, the technical approach involves creating a robust, secure, and user-friendly application. The system would employ ReactJS for the front end, featuring intuitive interfaces for browsing hotel rooms and managing bookings. The Node.js backend would handle hotel management, and booking operations via RESTful API endpoints, with MySQL as the relational database for data storage. We will also be using a MySQL database for user authentication. Maintenance and documentation would ensure long-term reliability and ease of use.

**Key Technical Design Decisions:**

Front-End Framework: ReactJS will be used to create a responsive and interactive user interface for the booking system.

Database Management: A MYSQL hotel and user database will be used to store information about hotels, rooms, reservations, user data, and other relevant details.

Back-End Framework: The Express framework (Node.js) will be used to create and handle API endpoints. Any update to the Hotel database will be handled by the API. Postman will also be required to test the endpoints.

**Database ER Diagram:**

A diagram of a computer

Description automatically generated

**Flow Charts/Process Flows:**

**A diagram of a hotel room

Description automatically generated**

**Sitemap Diagram:**

A diagram of a hotel room

Description automatically generated

**User Interface Diagrams:**

Homepage

A screenshot of a hotel registration form

Description automatically generated

Booking Page

A screenshot of a hotel

Description automatically generated

Reservations Page

A white background with black text

Description automatically generated

Login / SignUp page

A screenshot of a hotel registration form

Description automatically generatedA screenshot of a hotel registration form

Description automatically generated

**UML Diagrams:**

**Express API:**

**A screenshot of a computer

Description automatically generated**

**ReactJS App**

A screenshot of a computer program

Description automatically generated

**Service API Design:**

This section should fully document any third party Service Interface APIs being consumed or application specific Service API’s being published, how to access the service, what parameters are required by the API, and the detailed JSON data format specification that could be used by a third party developer to integrate with the service and API.

**NFR’s (Security Design, etc.):**

To ensure non-functional requirements are met, logs will be in place to detect and address issues promptly. The user experience should remain central, ensuring responsive design, efficient navigation, and fast page load times. Extensive testing such as usability testing, is vital to validate the system's capabilities such as cross-browser compatibility and authentication.

**Other Documentation:**

You should insert any additional drawings, storyboards, white board pictures, project schedules, tasks lists, etc. that support your approach, design, and project. If you have no supporting documentation, explain the rationale why you are able to leave this section as N/A.

Appendix A – Technical Issue and Risk Log

1. Use the template to identify and monitor project issues and risks.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Issues and Risk Log | | | | | | | | |
| **Issue or Risk** | **Description** | **Project Impact** | **Action Plan/Resolution** | **Owner** | **Importance** | **Date Entered** | **Date to Review** | **Date Resolved** |
| I/R | What is the issue or risk? | How will this impact scope, schedule, and cost? | How do you intend to deal with this issue? | Who manages this issue? |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

Appendix B – References

*List all Project Documentation References*

*List all references using APA style*

Appendix C – External Resources

|  |  |
| --- | --- |
| **GIT URL:** | *The GIT URL (if applicable).* |
| **Hosting URL:** | *The Hosting URL (if applicable).* |