**Online Hotel Management System**

**POC**

**Low Level Design (LLD)**



Date: 07/07/2022

Current Document Version: [*1.0*]

DOCUMENT APPROVAL

**Approvers of this document**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Department** | **Role** | **Signature** | **Date** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Document Change History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Version #** | **Author** | **Date** | **Description** |
| 1.0 | Kavya Goudar | 07/07/2022 | Online Hotel Management System LLD |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1.0 Document Purpose 1](#_Toc109206821)

[2.0 Intended Audience 1](#_Toc109206822)

[3.0 Project Background, Objective(s) 1](#_Toc109206823)

[3.1 Project Background 1](#_Toc109206824)

[3.2 Project Objective 1](#_Toc109206825)

[4.0 Design Pattern 2](#_Toc109206826)

[5.0 Solution Diagram 2](#_Toc109206827)

[6.0 Solution Steps 4](#_Toc109206828)

[7.0 Classes/function 6](#_Toc109206829)

[8.0 Data Models 7](#_Toc109206830)

[9.0 API Canvas 8](#_Toc109206831)

[10.0 HTTP Status Code 9](#_Toc109206832)

# Document Purpose

This document describes the solution architecture for Online Hotel Management System.

# Intended Audience

This document is intended as a reference for the following roles and stakeholders who are interested in the Hotel Management System technical architecture.

|  |  |
| --- | --- |
| Role | Nature of Engagement in WB Classics Portal Technical Architecture |
| Product Owners/SME | Key stakeholder to ensure that the architecture is aligned with business goals. |
| Business Analysts | Business analysts are one of the stakeholders who are informed with the key architectural decisions. |
| Enterprise Architects | To enforce Hotel management Platform Architecture is aligned to business goals and architecture, architectural guidelines. |
| Solution Architects | To ensure solution design and architecture is aligned to business requirements, architectural guidelines. |
| Developers | Use Technical Architecture Document as the guiding document for detail design and implantation approach to align with Hotel Management Microservice |

# Project Background, Objective(s)

## Project Background

Hotel management system is developed to automate the major operations of the hotel where each end users can register themselves and perform various operations.

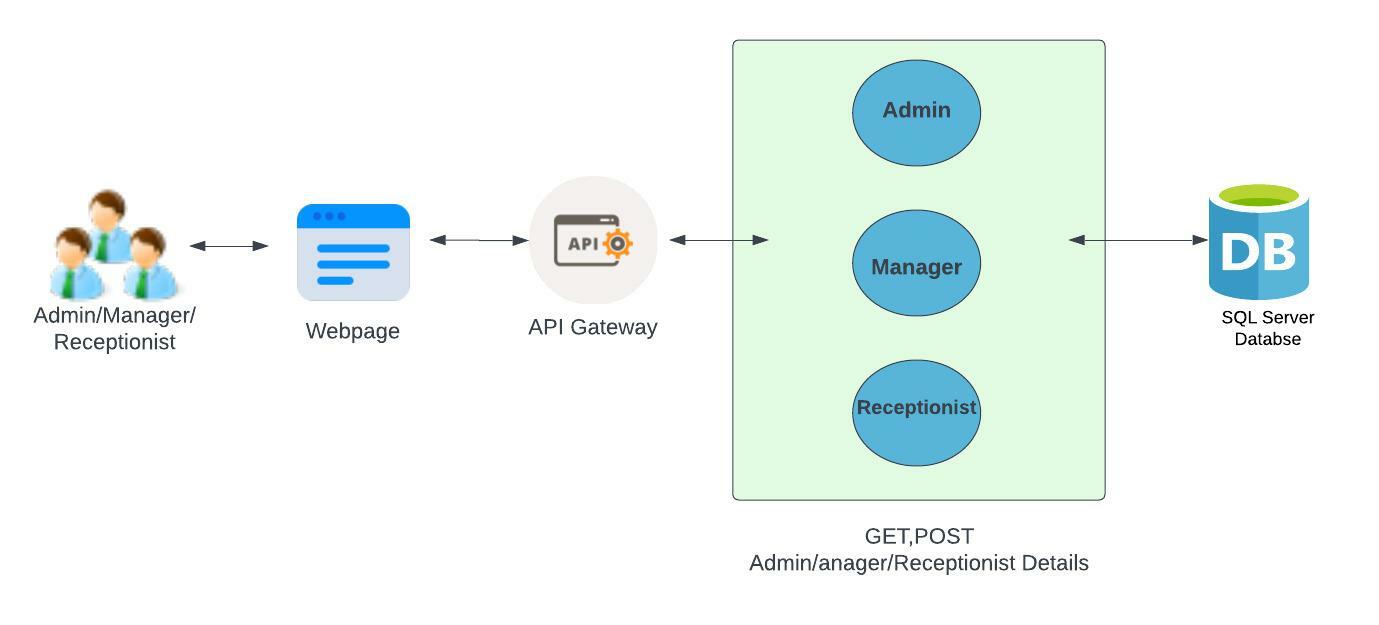
## Project Objective

The objectives of the automated Hotel Management System are to simplify the day-to-day processes of the hotel. The system will be able to handle many services to take care of all customers in a quick manner.

# Design Pattern

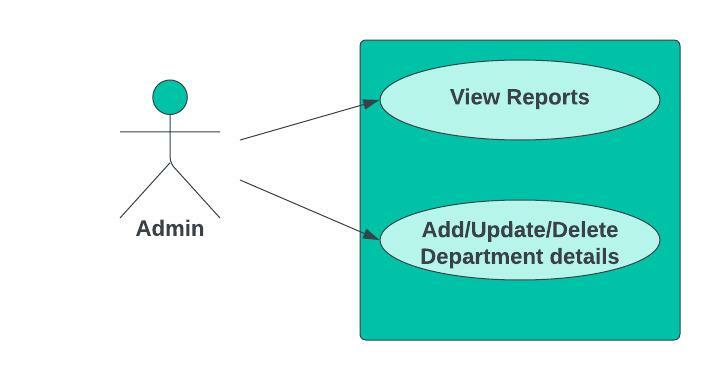
|  |  |  |
| --- | --- | --- |
| # | Name | Description |
| 1 | API | Using HTTP requests, we will use the respective action to trigger various operations |
| 2 | Web App | A web app is created using Angular and connected with web API to perform various operations. |
| 3 | Database | A database is created with various tables with constraints,keys |

# Solution Diagram

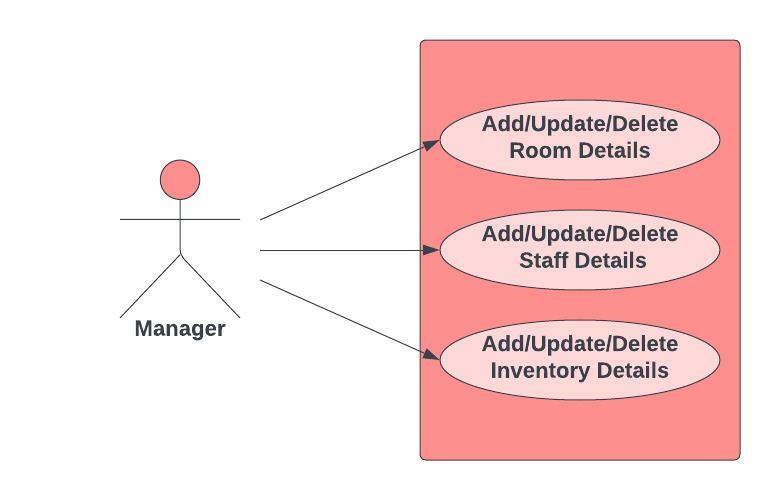


**Use Case Diagram**

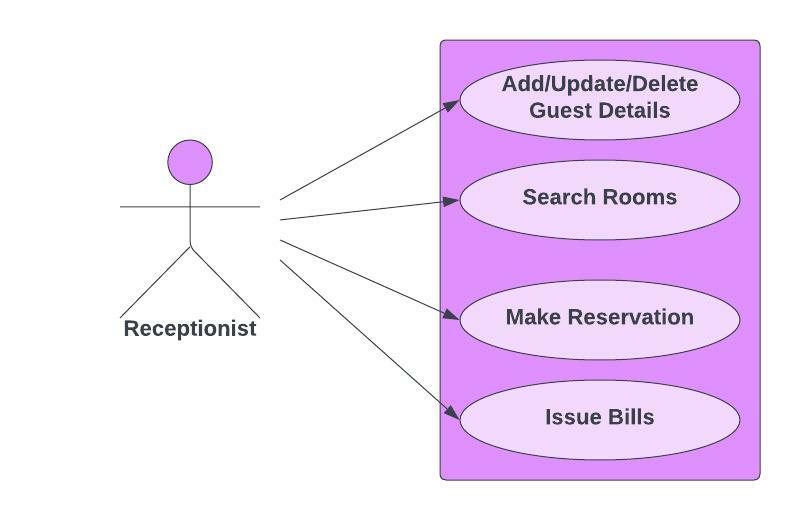
1. Admin Module



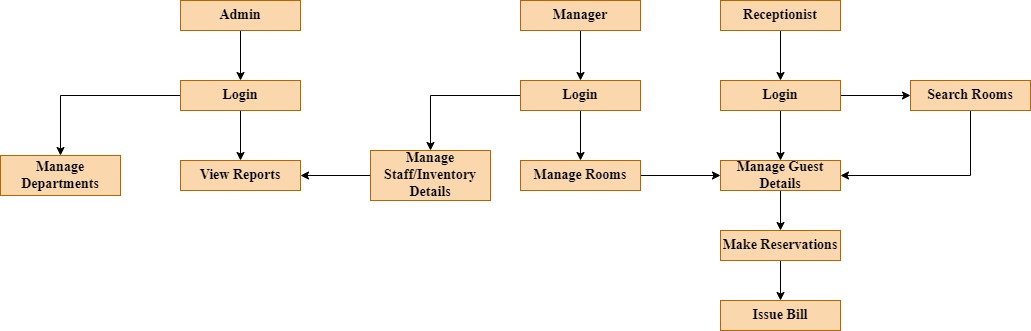
1. Manager Module



1. Receptionist Module



**Workflow**



# 6.0 Solution Steps

**Make Reservation**

1. Receptionist will enter the required customer details such as Code, Number of children, Number of adults, check-in date, check out date, status, Number of nights and click submit button after which the browser directs the request to make reservation API
2. This request reaches the API gateway
3. Routing is processed by API gateway and forwards the request to the API controller which handles all the requests of making reservations.
4. All the fields entered by the receptionist will be validated. In case of successful validation, the data will be recorded in the database. If validation fails then appropriate message will be displayed and browser redirects to the make reservation page.
5. When the database is updated, the receptionist will be notified that the reservation is successful.

**Guest Details**

1. Receptionist will enter the required guest details such as Member code, Phone number, Company, Name, E-mail, Gender, Address and click submit button after which the browser directs the request to add guest API
2. This request reaches the API gateway
3. Routing is processed by API gateway and forwards the request to the API controller which handles all the requests of adding guests.
4. All the fields entered by the receptionist will be validated. In case of successful validation, the data will be recorded in the database. If validation fails then appropriate message will be displayed and browser redirects to add guest page.
5. Along with adding the guests, the receptionist can also modify the guest details if required.
6. When there is a need to remove the guest details from database, delete option will be given to the receptionist.
7. When the database is updated, the receptionist will be notified that the guest is added/updated/deleted successfully.

**Search Rooms**

1. In order to search the rooms, the receptionist must enter details such as Period, Check-in, Check-out, Guests and click search button.
2. Based on the details entered, search operation will be performed in the database.
3. When a match is found, a message along with the available rooms is displayed to the receptionist.
4. Based on the search details provided, the receptionist can make reservation for the customers.

**Issue Bill**

1. In order to issue bill for the customer, the receptionist must enter the required bill details such as Billing no, Quantity, Price, Taxes, Date, Services, Unit and click generate bill button.
2. All the fields entered by the receptionist will be validated.
3. In case of successful validation, the receptionist can print the bill by clicking print button.
4. If validation fails then appropriate message will be displayed and browser redirects to issue bill page.

**Admin Dashboard**

1. Admin can view the reports such as staff payment and Income.
2. Admin can add department to the database.
3. When needed, the department details can also be modified by admin.
4. In case the department is no longer needed, admin can remove department from database.

**Hardware and Software Requirements**

Hardware Requirements

|  |  |
| --- | --- |
| Processor | Core i3 or above |
| Primary Memory | 8GB |
| Hard Disk | 60GB free space |

Software Requirements

|  |  |
| --- | --- |
| Operating System | Windows XP or above |
| Framework | Angular |
| Frontend | HTML5, CSS3, JavaScript, Bootstrap |
| Middleware | ASP .net core Web API |
| Backend | SQL Server |
| Tools used | Visual Studio, Vs code, SSMS |

# Classes/function

|  |  |
| --- | --- |
| Class | Description |
| Guest.cs | Model holds the Guest schema details |
| addGuest.cs | It contains the logic for the registration of new guest |
| getGuestDetails.cs | It contains the logic for getting the details of guest |
| updateGuest.cs | It contains the logic for updating the details of a guest |
| deleteGuest.cs | It contains the logic for deleting a guest |
| Staff.cs | Model holds the Staff schema details |
| addStaff.cs | It contains the logic for adding a new staff |
| getStaffDetails.cs | It contains the logic for getting the details of a staff |
| updateStaff.cs | It contains the logic for updating the details of a staff |
| deleteStaff.cs | It contains the logic for deleting a staff |
| Room.cs | Model holds the Room schema details |
| addRoom.cs | It contains the logic for adding a new room |
| getRoomDetails.cs | It contains the logic for getting the details of a room |
| updateRoom.cs | It contains the logic for updating the details of a room |
| deleteRoom.cs | It contains the logic for deleting a room |
| Inventory.cs | Model holds the Inventory schema details |
| addInventory.cs | It contains the logic for the registration of new inventory |

# 8.0 Data Models

**T\_Login**

|  |  |  |
| --- | --- | --- |
|  | Email | VARCHAR(200) |
|  | Password | VARCHAR(200) |

**T\_Reservation**

|  |  |  |
| --- | --- | --- |
| PK | Reservation Code | INT |
|  | No of children | INT |
|  | No of Adults | INT |
|  | Check-in | DATE |
|  | Check-out | DATE |
|  | Status | VARCHAR(200) |
|  | No. of nights | INT |
|  | No of rooms | INT |

**T\_Guests**

|  |  |  |
| --- | --- | --- |
| PK | Guest Code | INT |
|  | Ph no. | INT |
|  | Company | VARCHAR(255) |
|  | Name | VARCHAR(255) |
|  | Email | VARCHAR(255) |
|  | Gender | VARCHAR(255) |
|  | Address | VARCHAR(255) |

**T\_Staff**

|  |  |  |
| --- | --- | --- |
| PK | Staff ID | INT |
|  | Staff Name | VARCHAR(255) |
|  | Address | VARCHAR(255) |
|  | NIC | VARCHAR(250) |
|  | Salary | MONEY |
|  | Age | INT |
|  | Occupation | VARCHAR(255) |
|  | Email | VARCHAR(255) |

**T\_Rooms**

|  |  |  |
| --- | --- | --- |
| PK | Room ID | INT |
| FK | Check-in | DATE |
| FK | Check-out | DATE |
|  | Status | INT |

**T\_Rate**

|  |  |  |
| --- | --- | --- |
|  | First Night Price | MONEY |
|  | Extension | MONEY |
|  | Check-in | DATE |
|  | Check-out | DATE |
|  | No of days | INT |
| FK | Room ID | INT |
|  | Total Amount | MONEY |

**T\_Inventory**

|  |  |  |
| --- | --- | --- |
| PK | Inventory ID | INT |
|  | Inventory name | VARCHAR(250) |
|  | Quantity | INT |
|  | Price | MONEY |

**T\_Payment**

|  |  |  |
| --- | --- | --- |
| PK | Payment ID | INT |
| FK | Guest name | VARCHAR(250) |
|  | Card Holder Name | VARCHAR(250) |
|  | Card Number | NVARCHAR(16) |
| FK | Amount | MONEY |

# 9.0 API Canvas

**Admin Module**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Micro Service | Path | Verb | API Description | Role | Auth |
| hotel-mgmt-ms | /admin | GET | Display Admin Dashboard | Admin | True |
| hotel -mgmt-ms | /admin/view-reports | GET | To get generated reports | Admin | True |
| hotel -mgmt-ms | /admin/department | POST | To add or modify department details | Admin | True |
| hotel -mgmt-ms | /department/id | GET | To get details of a particular department | Admin | True |
| hotel -mgmt-ms | /customers/id | DELETE | To remove the department | Admin | True |

**Manager Module**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Micro Service | Path | Verb | API Description | Role | Auth |
| hotel-mgmt-ms | /manager | GET | Display manager Dashboard | No | True |
| hotel -mgmt-ms | /manager/view-rooms | GET | To get room details | No | True |
| hotel -mgmt-ms | /manager/view-inventory | GET | To get inventory details | No | True |
| hotel -mgmt-ms | /manager/view-staffs | GET | To get staff details | No | True |

**Receptionist Module**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Micro Service | Path | Verb | API Description | Role | Auth |
| hotel-mgmt-ms | / receptionist | GET | Display receptionist Dashboard | No | True |
| hotel -mgmt-ms | / receptionist /guest | POST | To add guest details. | No | True |
| hotel -mgmt-ms | /guest/id | GET | To get details of a particular room | No | True |
| hotel -mgmt-ms | /guest/id | DELETE | To remove the guest | No | True |
| hotel -mgmt-ms | /receptionist/make-reservation | POST | To add reservation | No | True |
| hotel -mgmt-ms | /receptionist/issue-bill | GET | To issue bill to the customer | No | True |

# 10.0 HTTP Status Code

200 - Request succeeded

400 – Inputs are invalid

404 – Customer Not found

502 – Bad gateway