**RAILWAY RESERVATION MICROSERVICE**

**POC**

**Low Level Design (LLD)**



***PRESENTED BY:***

*Nandini Tripathy*

*Deepsubho Gayen*

*Shubham Raj*

**Date: 12/09/2022**

**Current Document Version: [2*.0*]**

**DOCUMENT APPROVAL**

**Approvers of this document**

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

**DOCUMENT CHANGE HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Version** | **Author** | **Date** | **Description** |
| 1.0 | Nandini Tripathy,  Deepsubho Gayen,  Shubham Raj | 07/09/2022 | Railway Reservation Microservice LLD |
| 2.0 | Nandini Tripathy,  Deepsubho Gayen,  Shubham Raj | 12/09/2022 | Incorporated Dataflow Diagram, ER Diagram and Made changes in the database |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S. No** | **TOPIC** | **PAGE NO.** |
| 1 | Document Purpose | 4 |
| 2 | Intended Audience | 4 |
| 3 | Project Background and Objective | 4 |
| 4 | Design Pattern | 5 |
| 5 | Solution Diagram | 6 |
| 6 | Solution Steps | 9 |
| 7 | Classes/Functions | 11 |
| 8 | Data Models/Tables | 14 |
| 9 | API Canvas | 16 |
| 10 | Azure Role | 16 |
| 11 | HTTP Status Code | 16 |

1. **DOCUMENT PURPOSE**

The purpose of the document is to provide a detailed description of the requirements for the Railway Reservation System and to describe it, which provides train timetable and fare details, reservation of berths, online payment and cancellation. This document will allow the end user to develop the required software and be used for the development of future project stages.

**2.0 INTENDED AUDIENCE**

The intended audience of this document comprises the **administrator** and the **customer.**

|  |  |
| --- | --- |
| **Role** | **Nature of Engagement in Technical Architecture** |
| Administrator | The administrator has the prerogative of monitoring and updating the information for the efficient and effective management of the Railway Reservation Microservice |
| Customer | The Customer forms the section of the audience for serving whom this microservice has been created |

**3.0 PROJECT BACKGROUND AND OBJECTIVE**

***3.1 Project Background***

The traditional way of booking a ticket is very time-consuming and cancellation is also a long process. The existing system has Latency while we search for train details. The existing system also gets crashed many times while booking a ticket. So, we will design a system which will enable the booking in any class, and a maximum of six berths/seats at a time, for a journey between any two stations served by a train.

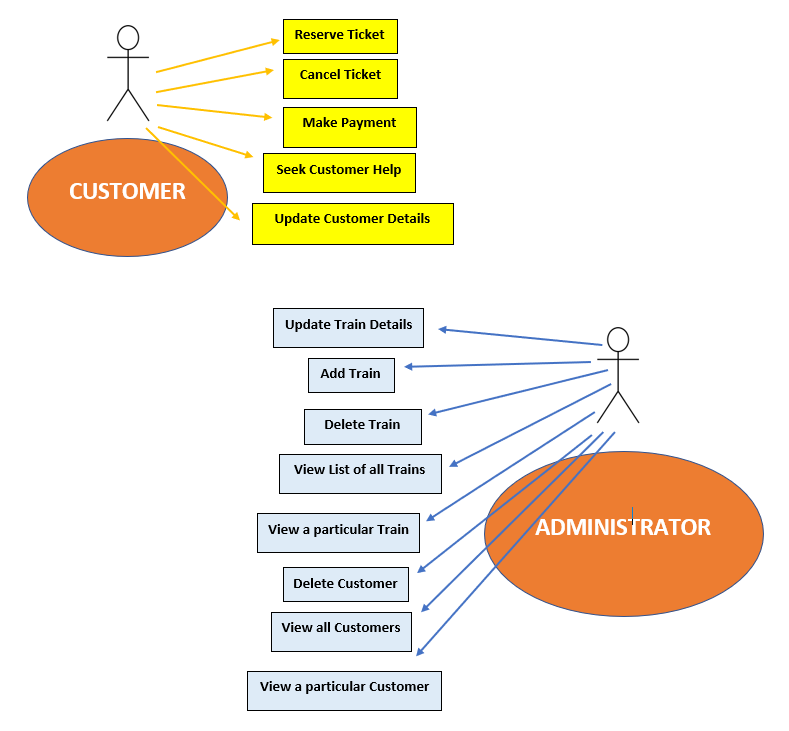
***3.2 Project Objective***

The project’s main objective is to develop the railway reservation system which will provide the most important customer interface for the business. The objective is also to give structural design to the railway system. The project provides functionality and flexibility to the railway system such that one can operate that system easily and efficiently. Using this system, a customer will be able to perform various operations like booking a ticket, canceling a ticket, viewing ticket fare, and viewing train details. Users should first register themselves and then they can perform all the operations.

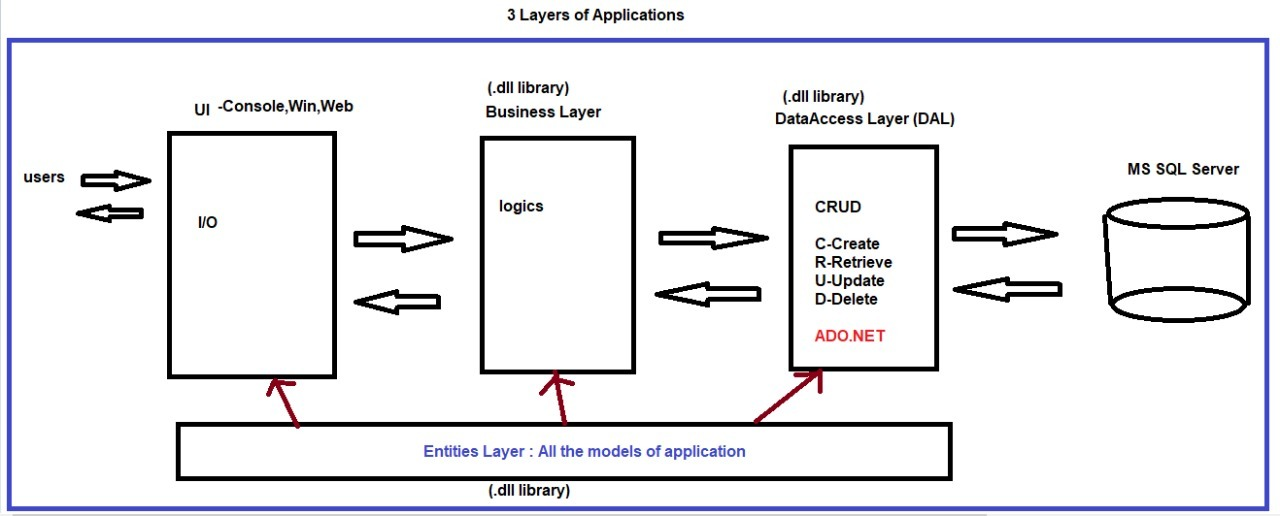
**4.0 DESIGN PATTERN**

|  |  |  |
| --- | --- | --- |
| **#** | **Name** | **Description** |
| 1 | Angular | For the Frontend. |
| 2 | API | Using HTTP requests, we will use the respective actions to trigger various operations. |
| 3 | SQL Server | For Data Storing and connection to the application frontend. |

***5.1 Use Case Diagram***



***5.2 Architecture***

****



The above architecture shows that the architecture of our Microservice can be broadly divided into the following five components:

1. UI (User Interface)
2. BL (Business Layer)
3. DAL (Data Access Layer)
4. EL (Entity Layer)
5. DB (Database)

The relationships among the above-stated five components is as follows:

The Users interact with the Microservice through the UI which can access the BL which can access the DAL which in turn can access the DB. All of the UI, BL and DAL can access the EL. Both EL and database cannot access anything and only have the privilege of being accessed. On the other hand, UI only enjoys the privilege of accessing and cannot be accessed. The remaining two components, BL and DAL share both of the privileges of accessing as well as being accessed.

**6.0 SOLUTION STEPS**

There are ***two*** types of users of this Microservice and accordingly, their experiences will be as follows-

**1. Customer-**

1. The new Customer will have to register by providing his/her details- name, gender, age, email id and password (minimum 6 characters).
2. After registration, he/she will have to login using the customer id (auto-generated by the system) and password.
3. Thereafter, the Customer will be redirected to a screen wherein the following operations can be performed-
4. ***Reserve Ticket:***

The customer can reserve ticket by filling a reservation form present in the website. The reservation form will contain the source and destination places of the customer along with the selected date. The trains available for the particular destinationsource and date combination will be shown listwise and a PNR number (unique for every reservation) will also be issued to the customer during the reservation process. A Customer can view a train’s details like its class (sleeper/AC), ticket fare and the number of berths available by clicking on it is the list provided. In the final step of reservation, the customer will have to make payment by entering his/her card details, bank account name and CVV in the redirected window for payment where he/she will **make the payment**.

1. ***Cancel Ticket:***

The customer can also perform cancellation of the ticket which he/she had reserved earlier by entering the PNR no. This PNR no will be checked with the PNR no in the database. If it exists then the booking will be cancelled. After the cancellation process, the confirmation message will be sent by the server.

1. ***Make Payment:***

The customer will make payment for the train he/she wishes to reserve as discussed in point A.

1. ***Customer Help:***

A Customer can seek help by seeing his/her doubts/problems in the FAQs provided in this section.

1. ***Update Details:***

A Customer can update his/her details.

**2. Administrator-**

After logging in with the credentials (hardcoded in the system), the Administrator will be redirected to a screen wherein the following operations can be performed-

1. ***Update Train Details:***

The Administrator can update the train details of a particular train in the train database.

1. ***Add Train:***

The Administrator can add the details of a new train to the train database.

1. ***Delete Train:***

The Administrator can delete the details of a train from the train database.

1. ***View List of All Trains:***

The Administrator can view the list of all trains in the train database.

1. ***View a Particular Train***

The Administrator can view the details of a particular train in the Train database.

1. ***Delete Customer:***

The Administrator can delete a Customer of a Customer from the Customer database.

1. ***View List of All Customers:***

The Administrator can view the list of all Customers present in the Customer database.

1. ***View A Particular Customer:***

The Administrator can view the details of a particular Customer in the Customer database using his/her Customer Id.

**7.0 CLASSES /FUNCTIONS**

***7.1 Classes***

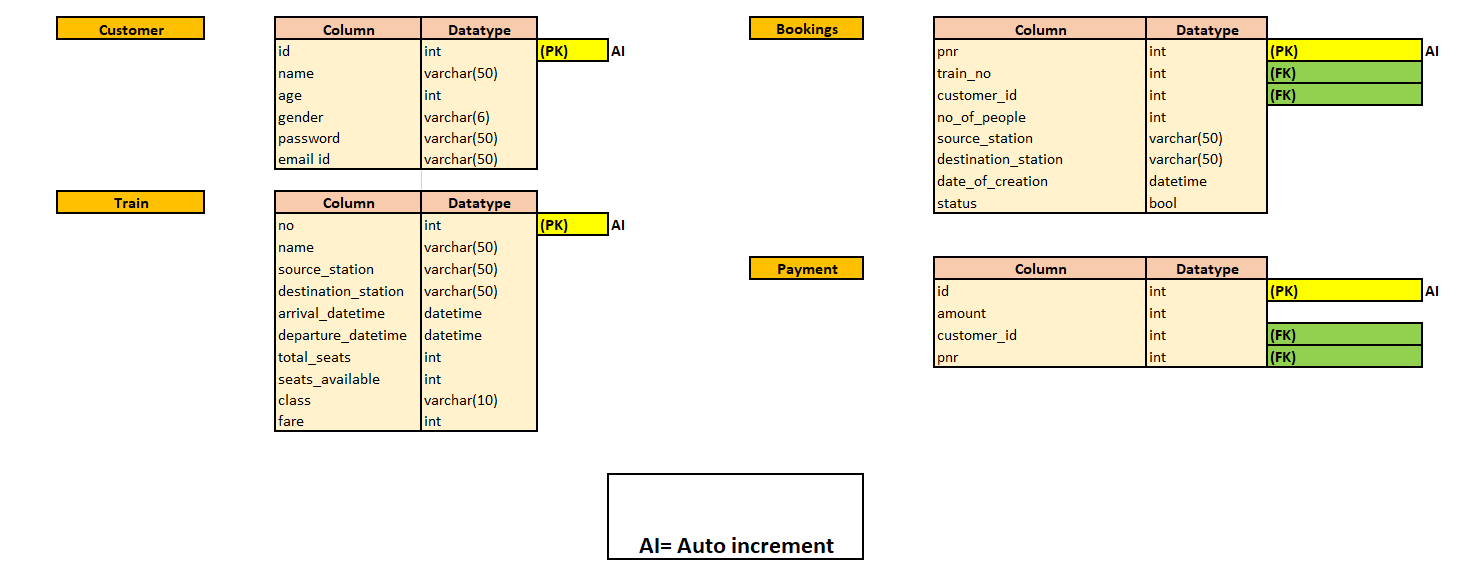
|  |  |  |
| --- | --- | --- |
| **#** | **Class** | **Description** |
| 1 | customer() | This class lays down the model for the customer entity |
| 2 | train() | This class lays down the model for the train entity |
| 3 | bookings() | This class lays down the model for the booking entity |
| 4 | payment() | This class lays down the model for the payment entity |

***7.2 Functions***

|  |  |  |
| --- | --- | --- |
| **#** | **Function** | **Description** |
| 1 | addCustomer() | This function adds a new customer to the customer table |
| 2 | deleteCustomer() | This function deletes an existing customer from the customer table |
| 3 | updateCustomer() | This function updates the details of an existing customer in the customer table |
| 4 | viewAllCustomers() | This function views a list of all the customers in the increasing order of their Customer Ids |
| 5 | viewSpecificCustomer() | This function views the details of a particular Customer (through his/her ID) |
| 6 | addTrain() | This function adds a new train to the train table |
| 7 | deleteTrain () | This function deletes an existing train from the train table |
| 8 | updateTrain () | This function updates the details of an existing train in the train table |
| 9 | viewAllTrains() | This function views a list of all the trains in the increasing order of their train nos |
| 10 | viewSpecificTrain () | This function views the details of a particular train (through its train no) |
| 11 | bookTrain() | This function books a train |
| 12 | cancelBookedTrain() | This function cancels a booked train |
| 13 | makePayment() | This function allows the user to make the payment during the process of booking a train |
| 14 | customerHelp() | This function shows the FAQs through which a Customer can check his/her doubts |

1. **DATA MODEL/TABLES**

***8.1 Database Design:***



***8.2 ER Diagram***

**No**

Name

Source

Arrival

Destination

Age

Name3

Departure

Gender3

**ID**

Total seats

Available seats

Password

Email

Class

Fare

**CUSTOMER**

**TRAIN**

Train No

NoNo

**PNR**

***BOOKS***

**ID**

Customer ID

**BOOKING**

Customer ID

**PAYMENT**

No of people

***HAS***

Status

Amount

Source

PNR

Date of Creation

Destination

**9.0 API Canvas**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Microservice** | **Path** | **Verb** | **API Description** | **Role** | **Auth** |
| railway-rsrvn-ms | /customer | POST | Customer registration | No | True |
| railway-rsrvn-ms | /customer | GET | To get customers’ list | Admin | True |
| railway-rsrvn-ms | /train | GET | To get trains’ list | Admin | True |
| railway-rsrvn-ms | /customer | PUT | To update customer details | No | True |
| railway-rsrvn-ms | /customer/id | GET | To get details of a particular customer | Admin | True |
| railway-rsrvn-ms | /customer/id | DELETE | To remove the details of a particular customer | Admin | True |
| railway-rsrvn-ms | /train/id | DELETE | To remove the details of a particular train | Admin | True |
| railway-rsrvn-ms | /train | POST | To get the details of a new train | Admin | True |
| railway-rsrvn-ms | /bookings | POST | To reserve a train | No | True |

**11.0 HTTP STATUS CODES**

201 – Customer Registered

200 - Request succeeded

400 – Inputs are invalid

404 – Customer Not found

502 – Bad gateway