

## **SOFTWARE ENGINEERING AND PROJECT MANAGEMENT**



### **LOCAL TRAIN TICKETING SYSTEM**

**Submitted by: -**

BHARATHWAJ M                      RA2011026020065

P AKASH PRABHU                      RA2011026020072

NIKHIL RAVICHANDRAN                      RA2011026020102

<b>EX NO: 1</b>	<b>Identify the software project, create</b>
<b>DATE:</b>	<b>business case, arrive at a problem statement</b>

## **Introduction: -**

A local train ticketing system project for local trains that allows users to book local train tickets and get ticket receipt online. This local train project provides login rights for normal users and admin. A normal user may login and get a ticket online, print it and travel by train. The ticketing process consists of a ticket booking form. The form allows the user to choose his source and destination. The source is the station from where the user will be boarding the train. Destination is the station he needs to get down at. The project database is already filled with stations on Mumbai western central and harbor line. It can be modified for any other city as needed. The system also consists of an option to select weather ticket is a single journey or a return ticket and the journey will be commenced on first class or a second class. It also consists of an admin account. The admin may recharge user account balance and check for various journey tickets processed in the system.

## **1.1 Problem Statement: -**

- Online railway reservation is an efficient way to reserve tickets not by standing in the railway station queue.
- Now all railways has their own website for online reservation provide better customer service.
- The manual filling of reservation form cannot be changed once the details had been entered.
- The goal of online railway reservation is easing the tedious task of railway activity.
- Initially the customer has to create an ID in the appropriate website, so that the user can log into the system for doing further activities.
- An online manager will maintain a database

## **Objective:**

- All the manual work should be converted into computerized so that the load of employees should decrease.
- The database should be stored in computer rather than in register/manually.

### **Scope of the product:**

- All the manual work should be converted into computerized so that the load of employees should decrease.
- The database should be stored in computer rather than in register/manually.
- The train ticketing software is an easy-to-use self-service system which enables the customer buys train ticket online.
- After process buys train ticket is successfully, the customer can get the train ticket by print out the train ticket.

### **1.1. ONE PAGE BUSINESS CASE TEMPLATE:**

#### **The Project: -**

- Since, the number of customers using trains is gradually increasing every year, manual process becomes very much complicated.
- The development of this "Local Train Ticketing system" makes the entire process automated keeping in mind the view of database integration approach.
- Authentication is provided where only the registered users can access.

#### **The History: -**

- The manual system does not provide secured registration and profile management of the Applicants properly.
- In manual system, all details are taken in the form of documents, which can be misplaced anytime.
- Every individual had to stand in queue, which is more time consuming and involves more manpower.

**Constraints: -**

- The Applicants require a computer to submit their Information
- Prior knowledge of computers and English language should be known
- Although the security is given high importance, there is always a chance of intrusion in the web world which requires constant monitoring.
- The Applicant must be careful while submitting the information.

**Approach: -**

- We approach this solution from the technical point of view. So, we are keen in maintaining an easy way to execute interface for the applicant as well as the back-end maintenance.
- The development of this project will require front end and back-end programming knowledge. Thus, it is important to equip with the same.

<b>EX NO: 2</b>	<b>Stakeholder and user description,</b>
<b>DATE:</b>	<b>identify the appropriate process models, comparative study with agile model</b>

## **2.1 Identifying Stakeholders: -**

### **1.USER:**

- The persons who will use the project's service are known as users.
- In this project, the end user will be the Applicant of the Passport and the authority is the integrated user.

### **2.SPONSOR:**

Sponsor is the person or the organization that provides financial support for the project.

As of now, this project is self-sponsored.

Once we upgrade, we may require financial interference of a management.

### **3.PROGRAM MANAGER:**

A Program Manager articulates a program's strategy and objectives and assesses how it will impact.

### **4.PROJECT MANAGEMENT OFFICE (PMO):**

A project management office (PMO) is a team or department that sets and maintains standards for project management throughout an organization.

### **5.PROJECT MANAGER:**

Project Managers (PMs) are responsible for planning, organizing, and directing the completion of specific projects for an organization while ensuring these projects are on time, on budget, and within scope.

### **6.PROJECT TEAM:**

A project team is comprised of the project manager, project management team and the other members who carry out work

## **Benefits: -**

### **A. Economic Feasibility**

This is a very important aspect to be considered while developing a project.

We decided the technology based on minimum possible cost factor. All hardware and software cost has to be borne by the organization.

Overall, we have estimated that the benefits the organization is going to receive from the proposed system will surely overcome the initial costs and the later on running cost for system.

### **B. Technical Feasibility**

This included the study of function, performance and constraints that may affect the ability to achieve an acceptable system. For this feasibility study, we studied complete functionality to be provided in the system, as described in the System Requirement Specification (SRS) and checked if everything was possible using different type of frontend and backend platforms.

### **C. Operational Feasibility**

No doubt the proposed system is fully GUI based that is very user friendly and all inputs to be taken all self-explanatory even to a layman. Besides, a proper training has been conducted to let know the essence of the system to the users so that they feel comfortable with new system. As far our study is concerned the clients are comfortable and happy as the system has cut down their loads and doing.

## **2.2 IDENTIFY THE APPROPRIATE PROCESS MODELS: -**

### **LOGIN**

In this module, the passenger can enter the user name and password. If the user name and password is correct then it can be entered into the specific web page. Otherwise, re-enter the user name and password after the particular time only.

### **FILL THE DETAILS:**

The passenger is asked to fill in some details such as boarding point and destination.

### **VERIFICATION OF THE FORM:**

The responsible team verifies the given information and the validates the request.

## **2.2 COMPARISON BETWEEN WATERFALL AND AGILE MODEL: -**

### **WHY AGILE MODEL IS BETTER THAN WATERFALL MODEL?**

- The Agile Model is based on iterative development and hence it divides the entire project into smaller parts which reduces the risk factor which is not the case in waterfall model.
- The Waterfall model cannot accept the changes in requirements but in agile model it is easy to change the system requirements.
- In agile model, the entire project is divided into smaller parts which helps to minimize the project risk and to reduce the overall project delivery time requirements.
- In waterfall model since risk factor is high, it is not suitable for complex projects.
- In waterfall model the testing is done in later stage it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare, which is not the case in agile model.
- In waterfall model, it follows a sequential approach whereas in agile model it explains the process in order of incremental approach.
- In agile it performs the testing concurrently with software development whereas in waterfall model the testing comes after the build phase only.
- In agile model the distance between the customer and developer is in short whereas in waterfall model it is long.
- In agile there can be any change in the project but in waterfall model there is no change throughout the project work.

<b>EX NO: 3</b>	<b>Identify the Requirements, System</b>
<b>DATE:</b>	<b>Requirements, Functional Requirements, Non-functional Requirements</b>

### **3.1. REQUIREMENTS:**

Requirements are defined during the early stages of the system development as a specification of what should be implemented. A collection of requirements is a requirements document. They may be user level facility description, detailed specification of system behaviour, general system property, a specific constraint on the system or information on how to carry on computation. The three types of requirements are explained below.

#### **3.1.1. SYSTEM REQUIREMENTS:**

System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems. The former may prevent a device or application from getting installed, whereas the latter may cause a product to malfunction or perform below expectation or even to hang or crash.

##### **1. Hardware requirements: -**

- Processor: Intel Pentium 4.0
- Ram: 2GB
- Hard disk: 500GB

##### **2. Software requirements: -**

- Operating system: Windows XP.
- Front-End: HTML, PHP
- Database: MYSQL
- Model Design: Rational Rose

#### **3.1.2. FUNCTIONAL REQUIREMENTS: -**

- Subject Information Management



- Student Information Management
- Student's Subject Information Management
- Marks Information Management
- Mark Sheet Generation
- Report Generation
- User Account Management

### **3.1.4 NON-FUNCTIONAL REQUIREMENTS:**

- Readability
- Availability
- Maintainability
- Security
- User Friendly
- Performance
- Efficiency
- Privacy

<b>EX NO: 4</b>	<b>Prepare project plan based on scope, find</b>
<b>DATE:</b>	<b>job roles and responsibilities, calculate project effort based on resources</b>

#### **4.1. PROJECT PLAN: -**

##### **➤ PROJECT NAME:**

LOCAL TRAIN TICKETING SYSTEM

##### **➤ PROJECT MEMBERS:**

Our project consists of three members:

- BHARATHWAJ M RA2011026020065
- P AKASH PRABHU RA2011026020072
- NIKHIL RAVICHANDRAN RZ2011026020102

##### **➤ MODULES:**

- Login
- Enter the travel details
- Validation
- Payment
- Issue of ticket

##### **➤ SCHEDULING:**

<b>Task</b>	<b>Start date</b>	<b>End date</b>
Business case development	10 <sup>th</sup> April 2002	12 <sup>th</sup> April 2022
Identify Stakeholders, Process Models and Required Modules	20 <sup>th</sup> April 2022	22 <sup>nd</sup> April 2022
Identify requirements	2 <sup>nd</sup> May 2022	6 <sup>th</sup> May 2022
Setting cost estimates and budget	11 <sup>th</sup> May 2022	15 <sup>th</sup> May 2022
Coding	20 <sup>th</sup> May 2022	30 <sup>th</sup> May 2022
UML Diagrams	2 <sup>nd</sup> June 2022	5 <sup>th</sup> June 2022
Final Revisions	7 <sup>th</sup> June 2022	10 <sup>th</sup> June 2022

#### 4.2. JOB ROLES AND RESPONSIBILITIES: -

MEMBERS	ROLE AND RESPONSIBILITIES
<b>NIKHIL RAVICHANDRAN RA2011026020102</b>	<ul style="list-style-type: none"><li>• <b>TEAM LEADER:</b> Responsibility of coordination of the team, checking for errors, updating for the current status of the project, guiding the team.</li><li>• <b>DEVELOPER:</b> Responsibility of coding, compiling and debugging of the modules.</li><li>• <b>SOFTWARE ARCHITECT:</b> Design of UML Diagrams and other blueprint.</li></ul>
<b>BHARATHWAJ M RA2011026020065</b>	<ul style="list-style-type: none"><li>• <b>WEB DEVELOPER:</b> Responsibility of designing the website and interfacing with the server.</li><li>• <b>DEVELOPER:</b> Responsibility of coding compiling and debugging of the modules.</li><li>• <b>MANUAL TESTER:</b> Responsibility of testing the project at a smaller level and reporting the errors.</li></ul>
<b>P AKASH PRABHU RA2011206020072</b>	<ul style="list-style-type: none"><li>• <b>WEB DEVELOPER:</b> Responsibility of designing the website and interfacing with the website.</li><li>• <b>TESTER:</b> Responsibility of testing the project at a vast level and variety of ways.</li><li>• <b>DESIGNER:</b> Identifying areas for modification in existing programs .</li></ul>

### 4.3. PROJECT EFFORT BASED ON RESOURCES: -

- COCOMO2 (Constructive Cost Model 2) is an algorithmic cost estimation technique proposed by Boehm, which works in a bottom-up manner.
- It is designed to provide some mathematical equations to estimate software projects.
- These mathematical equations are based on historical data and use project size in the form of KLOC.
- The COCOMO model uses a multivariable size estimation model for effort estimation.

$$\text{OBJECT POINT} = \sum_{i=1}^3 \sum_{j=1}^3 c_{ij} \cdot w_{ij}$$

	Simple	Medium	Complex
Screens	1	3	2
Report	1	3	1
3GL	0	0	1

$$(1 \cdot 1 + 3 \cdot 2 + 2 \cdot 3) + (1 \cdot 2 + 3 \cdot 5 + 1 \cdot 8) + (1 \cdot 10) = 13 + 25 + 10$$
$$= 48$$

$$\text{NOP} = \text{Object Point} \cdot (1 - \% \text{ reuse} / 100)$$

$$= 48 \cdot (1 - 0)$$

$$= 48$$

$$\text{EFFORTS} = \text{NOP} / \text{PROD}$$

$$= 48 / 13$$

$$= 3.7$$

NOP = New Object Point PROD = Productivity

We have assumed nominal developer experience.

<b>EX NO: 5</b>	<b>Prepare the work breakdown structure based</b>
<b>DATE:</b>	<b>on timelines, Risk identification plan</b>

### 5.1. WORK BREAKDOWN STRUCTURE: -

A Work Breakdown structure is a deliverable-oriented hierarchical decomposition of the work to be executed by the project team to accomplish the project objectives and create the required deliverables. A WBS is the cornerstone of effective project planning, execution, controlling, monitoring, and reporting. All the work contained within WBS is to be identified, estimated, scheduled, and budgeted.

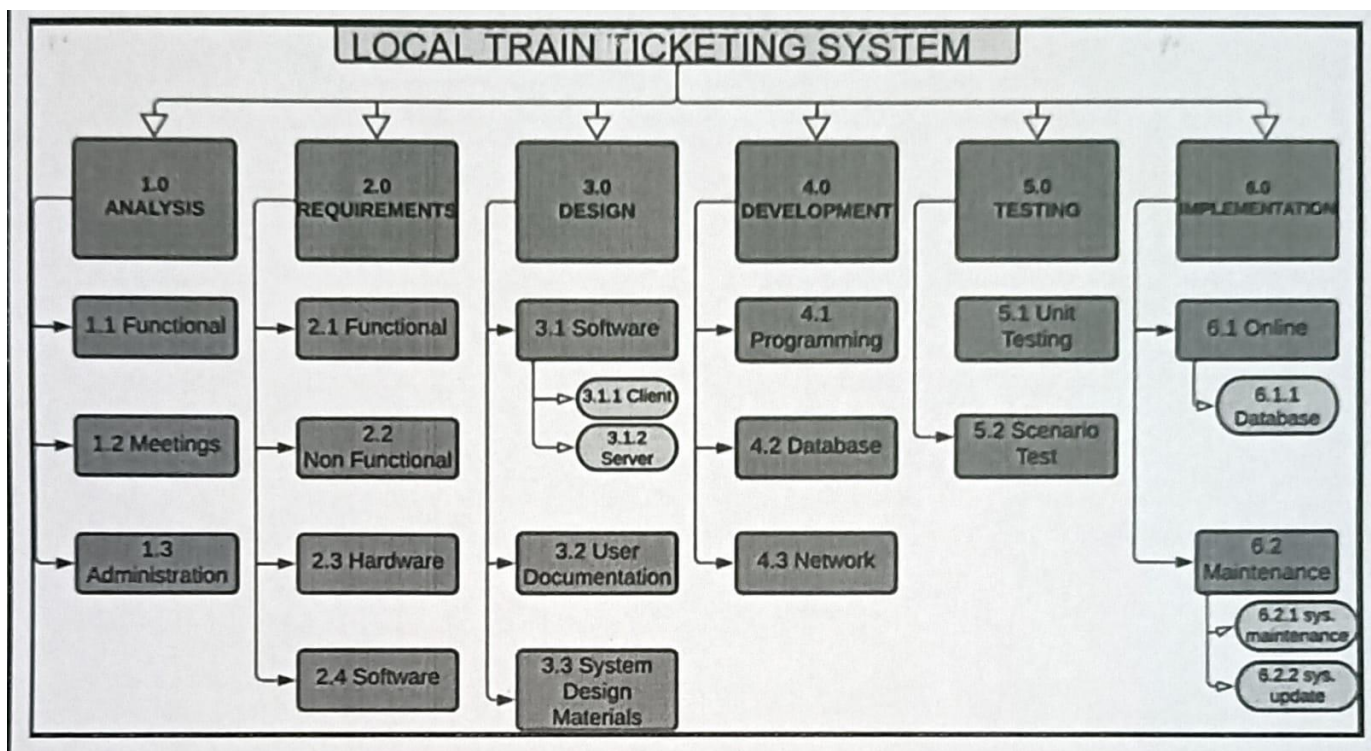


Figure 5.1 work breakdown structure

## **5.2. RISK MANAGEMENT: -**

### **DESCRIPTION:**

In the modern world, risk management refers to the practice of identifying potential risks in advance by analysing them and taking precautionary steps to curb the risk.

- Risk management is the identification, evaluation, and prioritization of risks, controlling the probability or impact of unfortunate events.
- When all risks have been identified, they will then be evaluated to determine their probability of occurrence.
- Plans will be made to avoid each risk, to track each risk to determine if it is more or less likely to occur, and to plan for those risks should they occur.
- The quicker the risks can be identified and avoided, the smaller the chances of having to face those particular risks consequence.

### **RISKS TO BE HANDLED:**

- Hacker's intent to get user data.
- Low website speed.
- Improper internet connection.
- Maintaining Database.

### **MANAGING RISKS:**

- Performing periodic maintenance of the server.
- Using of Captcha and other security protection things to protect from bot attack.
- The bugs must be removed, and the code must pass as many test cases as possible.

<b>EX NO: 6</b>	<b>Design a System Architecture, Use Case Diagram, ER Diagram, DFD Diagram, Class Diagram, Collaboration Diagram</b>
<b>DATE:</b>	

### 6.1. SYSTEM ARCHITECTURE: -

Here we have used the basic software front end design model in order to represent the system architecture of our software model.

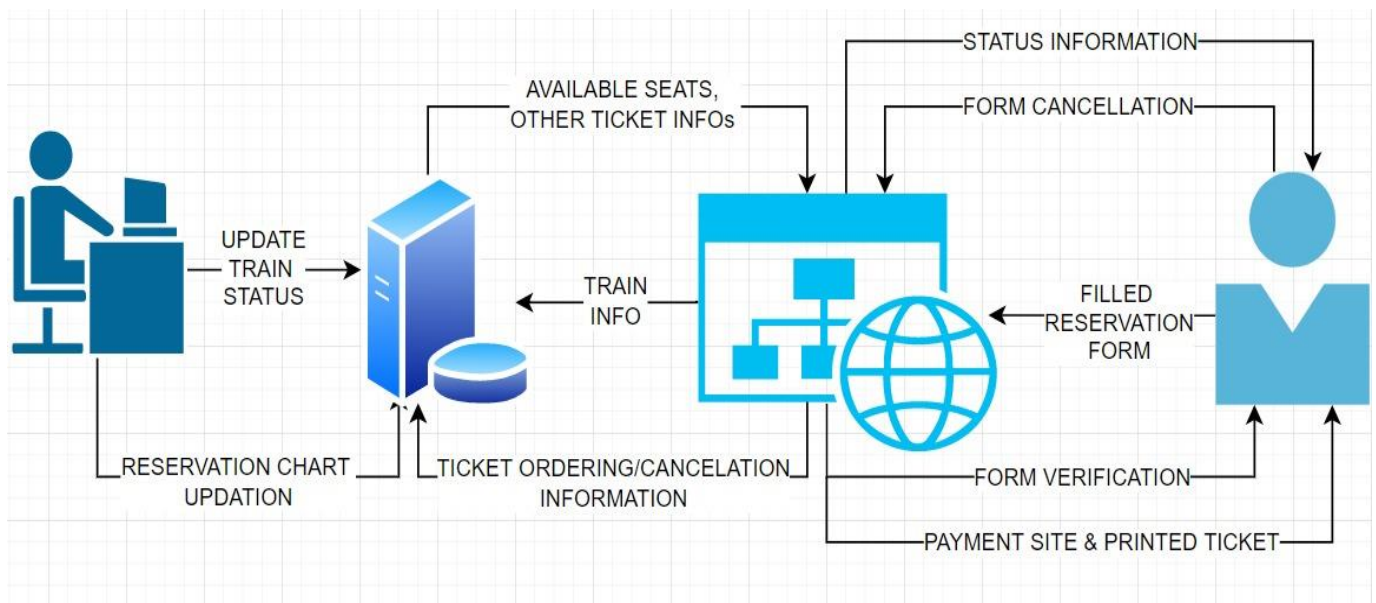


Figure 6.1 system architecture

The above is a simple form of system design diagram which uses front end design. This shows a loop of functions that need to be executed when this project is implemented. This is a chain of operations through which this project is implemented.

## **6.2MODELING USECASE DIAGRAM AND SCENARIOS: -**

### **6.2.1 USE CASE DIAGRAM DESCRIPTION:**

#### **USE CASE DIAGRAM:**

The purpose of a use case diagram in UML is to demonstrate the different ways that a user might interact with a system.

#### **USE CASE SYMBOLS AND NOTATION:**

The notation for a use case diagram is pretty straightforward and doesn't involve as many types of symbols as other UML diagrams.

##### **1] SYSTEM:**

A specific sequence of actions and interactions between actors and the system. A system may also be referred to as a scenario.

##### **2] USE CASES:**

Horizontally shaped ovals that represents an action which accomplishes some sort of task within the system.

##### **3] ACTORS:**

Stick figures that represent the people actually employing the use cases. It should be placed outside the system.

There are two types of Actors namely:

- **PRIMARY ACTOR:** Initiates the use of the system. It should be placed on the left side of the system.
- **SECONDARY ACTOR:** It is more reactionary and should be placed on the right side of the system.

##### **4] RELATIONSHIPS:**

- **INCLUDE:** This shows the dependency between base and included use case (it happens every time).
- **EXTENT:** This happens only when certain criteria are met.

##### **5] ASSOCIATION:**

A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.



### 6.1.1. USE CASE DIAGRAM:

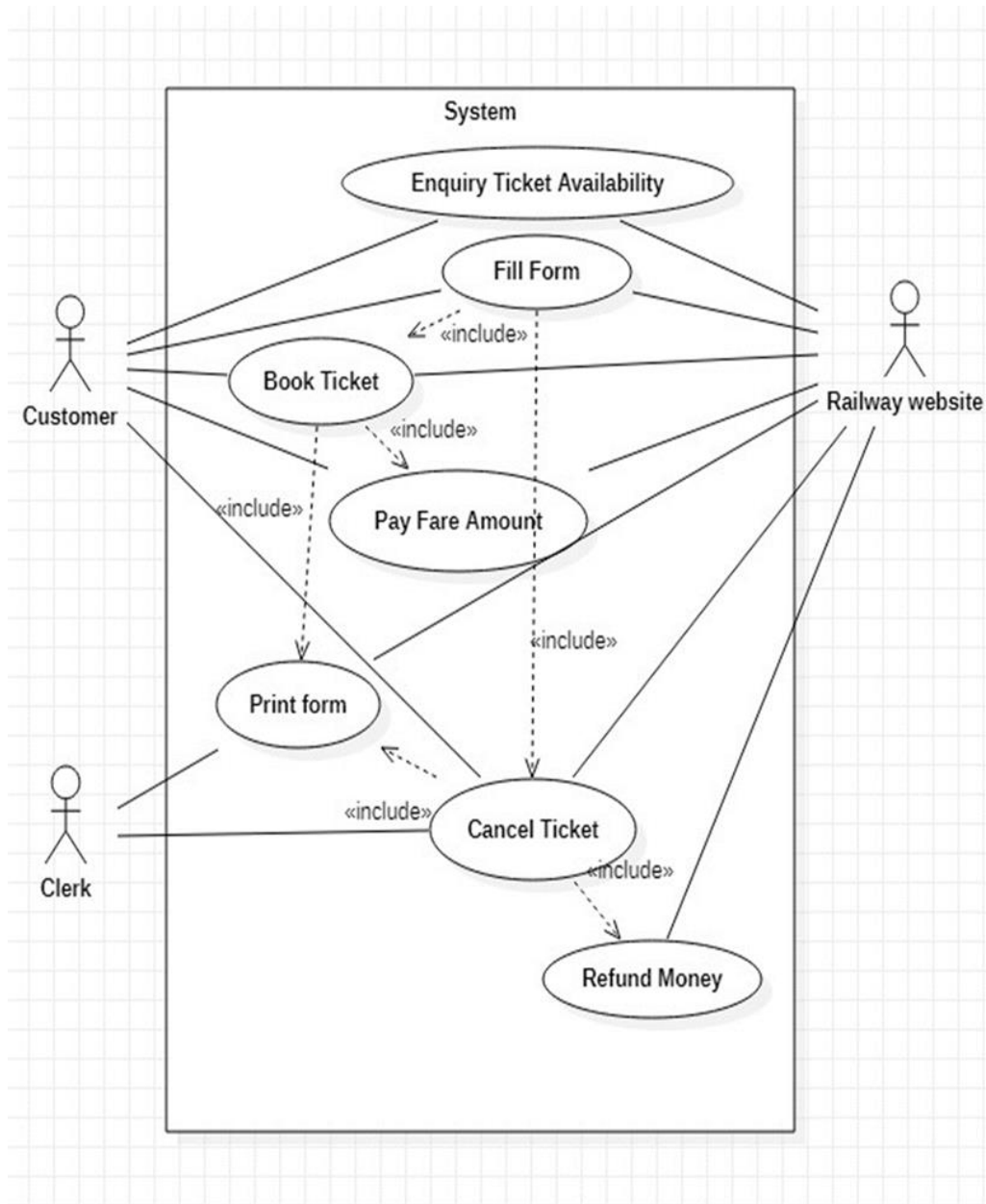


Figure 6.2 use case diagram

## 6.3 MODELING OF ER DIAGRAM: -

### 6.3.1 ER DIAGRAM DESCRIPTION:

- An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system.
- ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education, and research.

### USES OF ER DIAGRAM:

- Database design
- Database troubleshooting
- Business information systems
- Business process re-engineering (BPR)
- Education
- Research

### COMPONENTS OF ER DIAGRAM:

ER Diagrams are composed of entities, relationships (Cardinality) and attributes. They also depict cardinality, which defines relationships in terms of numbers.

**1] ENTITY:** A definable thing—such as a person, object, concept or event—that can have data stored in it.

**2] ATTRIBUTES:** A property or characteristic of an entity.

#### **3] KEYS:**

**PRIMARY KEY(PK):** It is unique, cannot be repeated and never null. **FOREIGN KEY(FK):** It is not unique and can be repeated.

**4] CARDINALITY:** Defines the numerical attributes of the relationship between two entities.

- One to one
- Many to one
- *One and only*
- Zero to one
- One or many
- Zero or many

### 6.3.2 ER DIAGRAM:

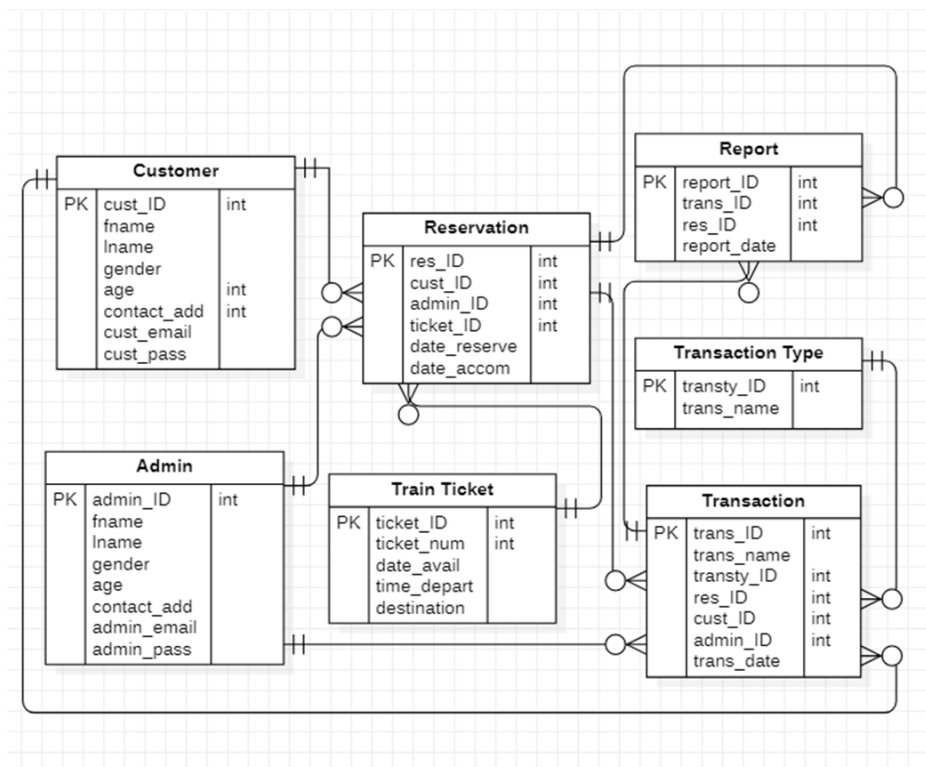


Figure 6.3 ER diagram

## 6.4 MODELING OF DATA FLOW DIAGRAM: -

### 6.4.2 DATA FLOW DIAGRAM DESCRIPTION:

Result Management System Data flow diagram is often used as a preliminary step to create an overview of the Result Management without going into great detail, which can later be elaborated. It normally consists of overall application dataflow and processes of the Result Management process. It contains all of the user flow and their entities such as all the flow of Student, Exam, Class, Subject, Result, Teacher, Semester. All of the below diagrams have been used for the visualization of data processing and structured design of the Result Management process and working flow.

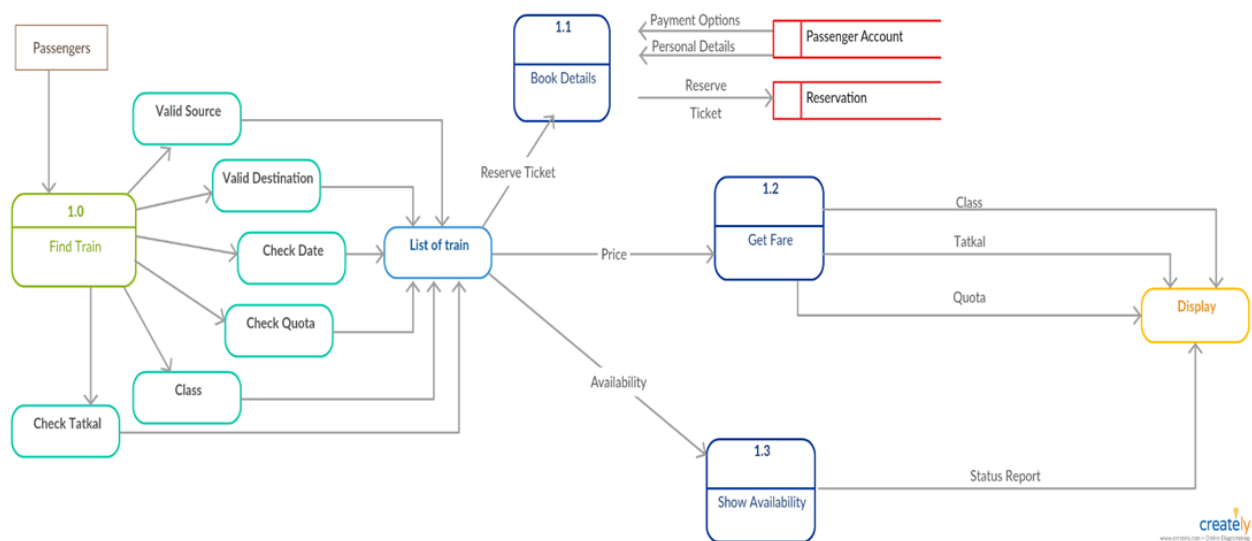


Figure 6.4 data flow diagram

## 6.5 MODELING OF CLASS DIAGRAM: -

### 6.5.2 CLASS DIAGRAM DESCRIPTION:

Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modelling its classes, attributes, operations, and relationships b/w objects.

#### COMPONENTS OF CLASS DIAGRAM:

The standard class diagram is composed of three sections:

- **UPPER SECTION:** Contains the name of the class. This section is always required, whether you are talking about the classifier or an object.
- **MIDDLE SECTION:** Contains the attributes of the class. Use this section to describe the qualities of the class. This is only required when describing a specific instance of a class.
- **BOTTOM SECTION:** Includes class operations (methods). Displayed in list format, each operation takes up its own line. The operations describe how a class interacts with data.

### 6.5.3 CLASS DIAGRAM:

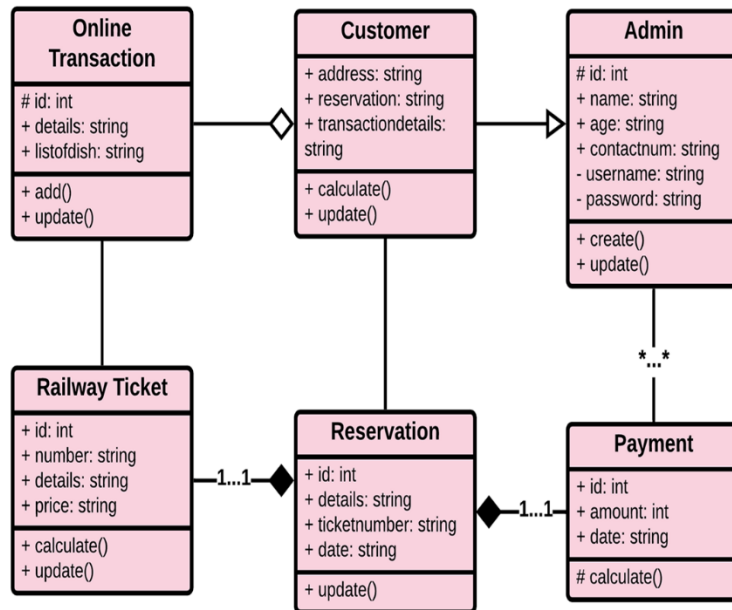


Figure 6.5 class diagram

## 6.6 MODELING OF COLLABORATION DIAGRAM: -

### 6.6.1. COLLABORATION DIAGRAM DESCRIPTION:

Communication diagrams, formerly known as collaboration diagrams, are almost identical to sequence diagrams in UML, but they focus more on the relationships of objects—how they associate and connect through messages in a sequence rather than interactions.

### COMPONENTS OF COMMUNICATION DIAGRAM:

#### 1) OBJECTS:

Objects can be classed as either a supplier or a client. Suppliers call the function that supplies the message. Clients send the message to the supplier, who receives it. It is represented by rounded rectangle.

## 2]ACTORS:

Stick figure represents the actor. It is the instances that invokes the interaction.Each actor has a specific name and a role.

## 3]LINKS:

A straight line connecting two objects indicates a relationship between them.The two objects are able to send messages to each other.

## 4]MESSAGES:

Typically, messages will have a number and description next to them. Thenumber determines the order in which messages should be read.

### 6.6.2. COLLABORATION DIAGRAM:

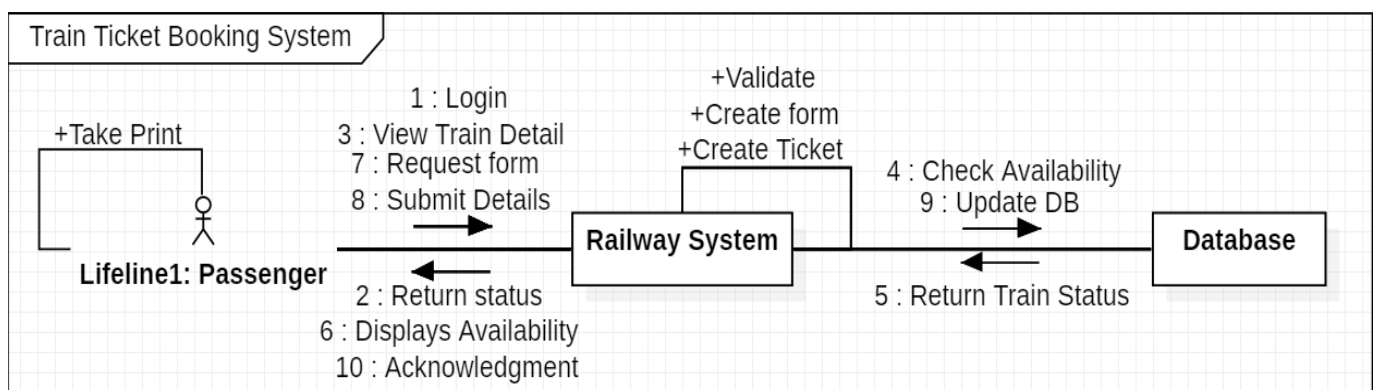


Figure 6.6 collaboration diagram

<b>EX NO: 7</b>	<b>State and Sequence Diagram,</b>
<b>DATE:</b>	<b>Deployment Diagram, Sample Frontend Design</b>

## **7.1. STATECHART DIAGRAM: -**

### **7.1.1. STATECHART DIAGRAM DESCRIPTION:**

State diagram describes the behaviour of a single object in response to a series of events in a system. This UML diagram models the dynamic flow of control from state to state of a particular object within a system.

### **COMPONENTS ARE:**

- **Initial State:**

A filled circle followed by an arrow represents the student's login (object's) initial state.

- **States**

States in state chart diagram represent situations during the life of an Object. You can easily illustrate a state in Smart Draw by using a rectangle with rounded corners.

- **Transition**

A solid arrow represents the path between different states of an object of Exam result management system.

- **Final State**

An arrow pointing to a filled circle nested inside another circle represents the (object's) result.

## 7.1.2. STATECHART DIAGRAM:

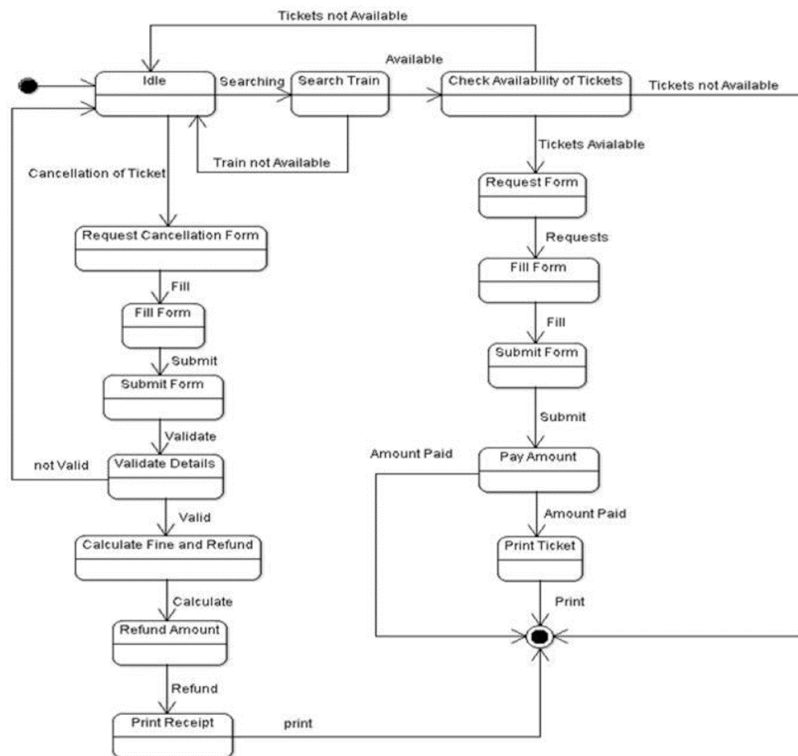


Figure 7.1 state chart diagram

## 7.2SEQUENCE DIAGRAM: -

### 7.2.1 SEQUENCE DIAGRAM DESCRIPTION:

Sequence diagram are a popular dynamic modelling solution in UML because they specifically focus on lifelines, or the processes and objects that live simultaneously, and the messages exchanged between them to perform a function before the lifeline ends.

### COMPONENTS IN SEQUENCE DIAGRAM:

#### 1] ACTOR:

Stick figure represents the actor. Shows entities that interact the external objects of the system.

#### 2] OBJECTS:

Rectangular boxes represent the object, demonstrates how an object will behave in the context of the system.



### **3] ACTIVATION BOXES:**

Represents the time needed for an object to complete a task. The longer the task will take, the longer the activation box becomes.

### **4] MESSAGE SYMBOLS:**

We use the following arrows and message symbols to show how information is transmitted between objects. These symbols may reflect the start and execution of an operation or the sending and reception of a signal.

- **SYNCHRONOUS MESSAGE:** Represented by a solid line with a solid arrowhead. This symbol is used when a sender must wait for a response to a message before it continues. The diagram should show both the call and the reply.
- **ASYNCHRONOUS MESSAGE:** Represented by a solid line with a lined arrowhead. Asynchronous messages don't require a response before the sender continues. Only the call should be included in the diagram.
- **REPLY MESSAGE:** Represented by a dashed line with a lined arrowhead, these messages are replies to calls.
- **DELETE MESSAGE:** Represented by a solid line with a solid arrowhead, followed by an X. This message destroys an object.

### 7.2.1 SEQUENCE DIAGRAM:

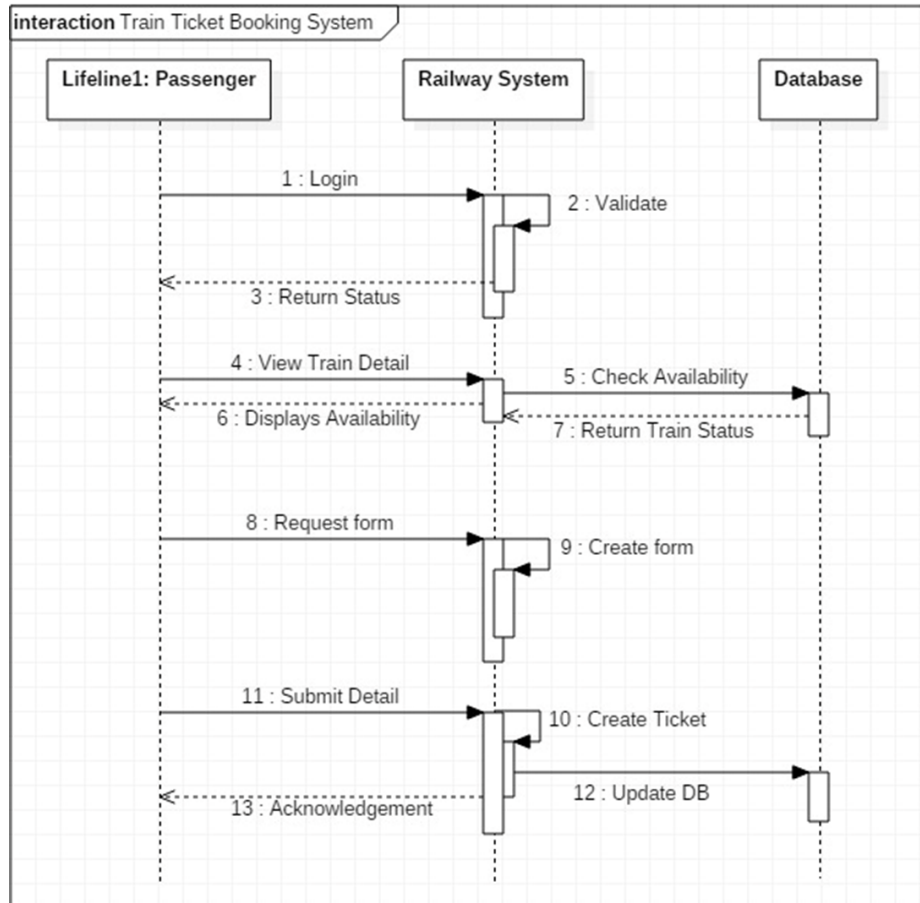


Figure 7.2 sequence diagram

## 7.3 DEPLOYMENT DIAGRAM: -

### 7.3.1. DEPLOYMENT DIAGRAM DESCRIPTION:

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware.

Deployment diagrams help model the hardware topology of a system compared to other UML diagram types which mostly outline the logical components of a system.

### 7.3.2. DEPLOYMENT DIAGRAM:

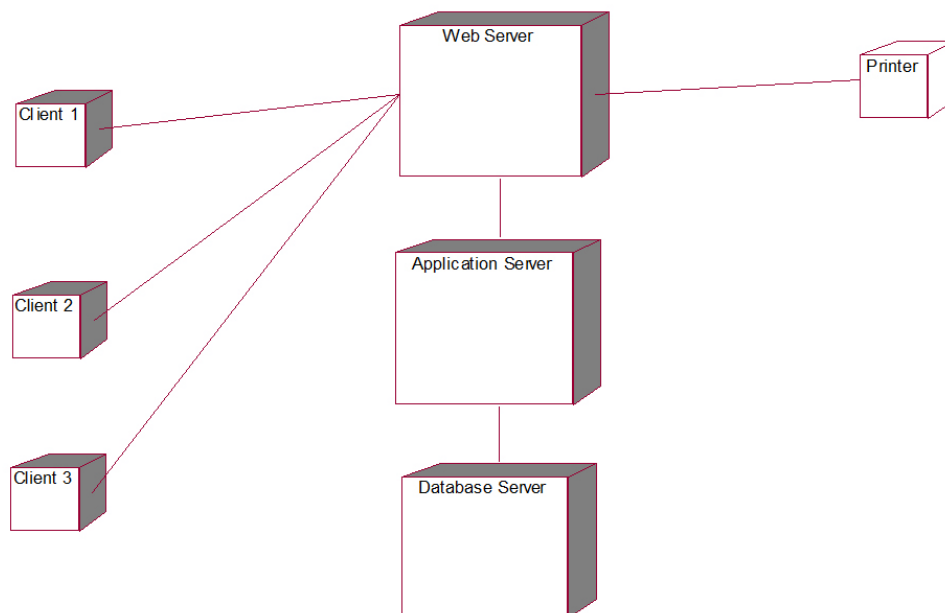



Figure 7.4 deployment diagram

#### 7.4. SAMPLE FRONTEND DESIGN: -

The figures 7.5 and 7.6 shows the home page of the ticketing system application after login in.

Railway e-ticketing

Sign InJoin Now



## Get Your e-Ticket Now

HomeContact Us

### Book Train Tickets

From

Select...

To

Select...

Train

Select...

Class

Select...

Time

Select...

No of Tickets

qty

HomeContact Us

### Book Train Tickets

From

Bandarawela

To

Colombo Fort

Train

Udarata Menike

Class

1st class

Time

05.00 am

No of Tickets

1

Date

2019-06-01

Amount	420.00 LKR
Discount	0.00 LKR
Total	420.00 LKR

Make Reservation

Figure 7.5 & 7.6 home page of the application

<b>EX NO: 8</b>	<b>Module      Description,      Module</b>
<b>DATE:</b>	<b>Implementation (phase 1) Using Agile</b>

## 8.1. MODULE DESCRIPTION: -

These are the main modules of the project.

- Ticket Module: We can create, read, update and delete Ticket from this module
- Train Route Module: All the operations related to Train Route, is managed by this module
- Train Module: Train Module is used to manage the Train
- Train Schedule Module: It has been developed for managing the Train Schedule
- Customer Module: It manages the Customer
- Payment Module: Payment operations will be managed by Payment module

### Features of Student Result Management System:

- User can search details of the Train Route, Ticket, Customer, Payment
- Railway Reservation System is an online application, from which user can easily manage Ticket details, Train details, Train Schedule details
- Admin can track all the information of Ticket, Train Route, Train ect
- Admin can edit, add, delete and update the records of Train Schedule, Customer, Payment
- Manage the information about of Train, Customer, Ticket

## 8.2. MODULE IMPLEMENTATION USING AGILE: -

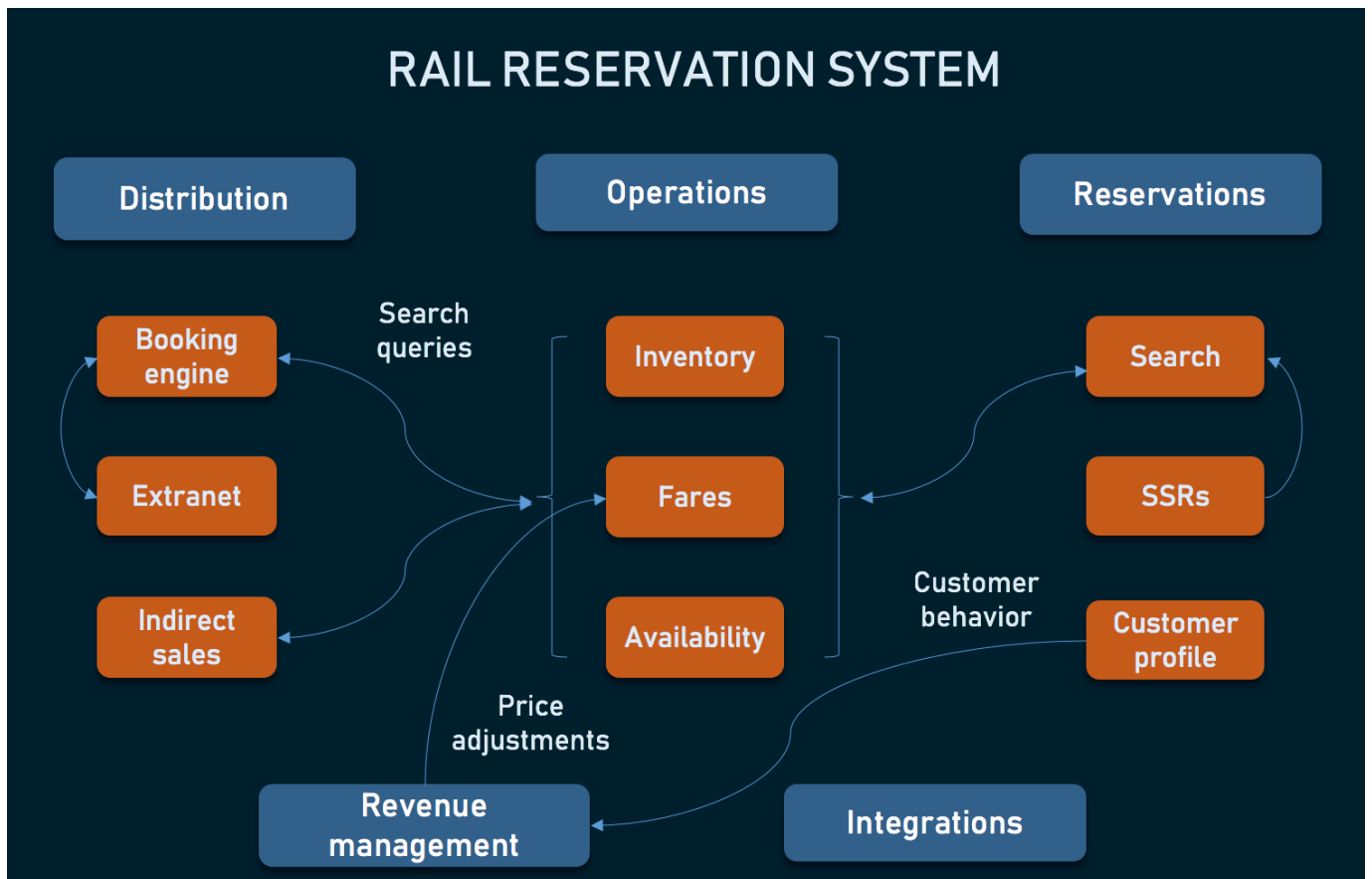


Figure 8.1 module implementation

<b>EX NO: 9</b>	<b>Module Implementation, Scrum Master to Induce New requirements in AgileDevelopment</b>
<b>DATE:</b>	

The fig.9.1 show the home page where you can login into your account where you can book the reservation for your upcoming journey.

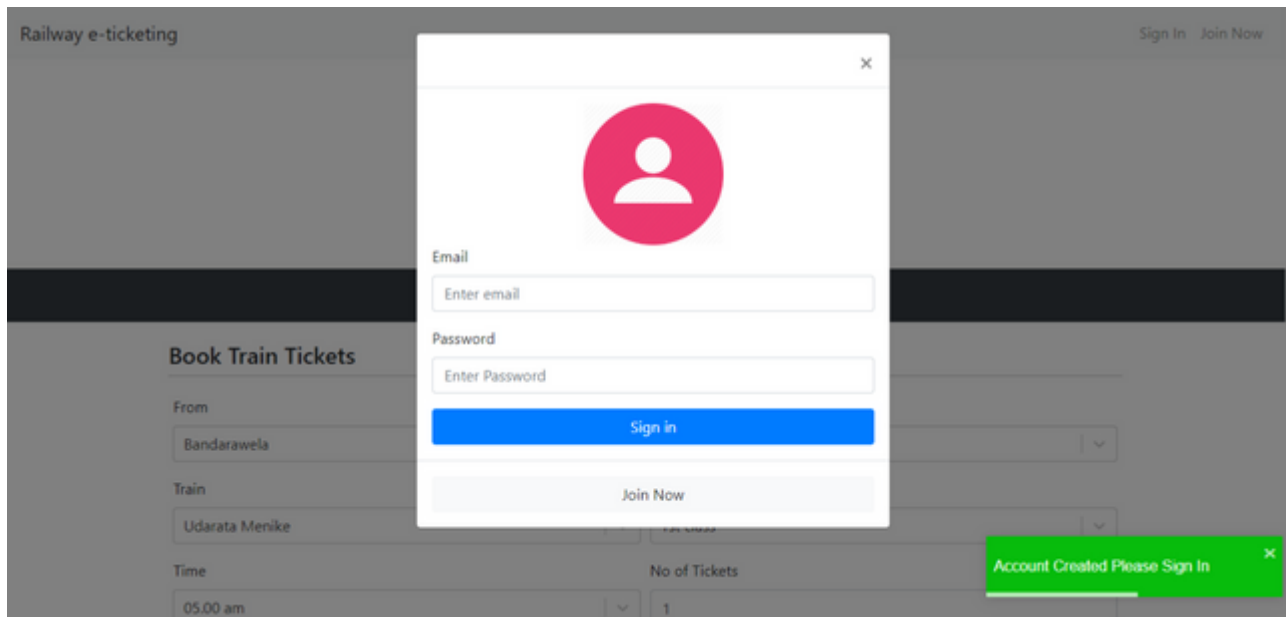


Figure 9.1 login page

<b>EX NO: 10</b>	<b>Module Implementation (Phase 2), Scrum</b>
<b>DATE:</b>	<b>Master to Induce New Issues in AgileDevelopment</b>

Figure 10.1 is the login page where you can login into your account

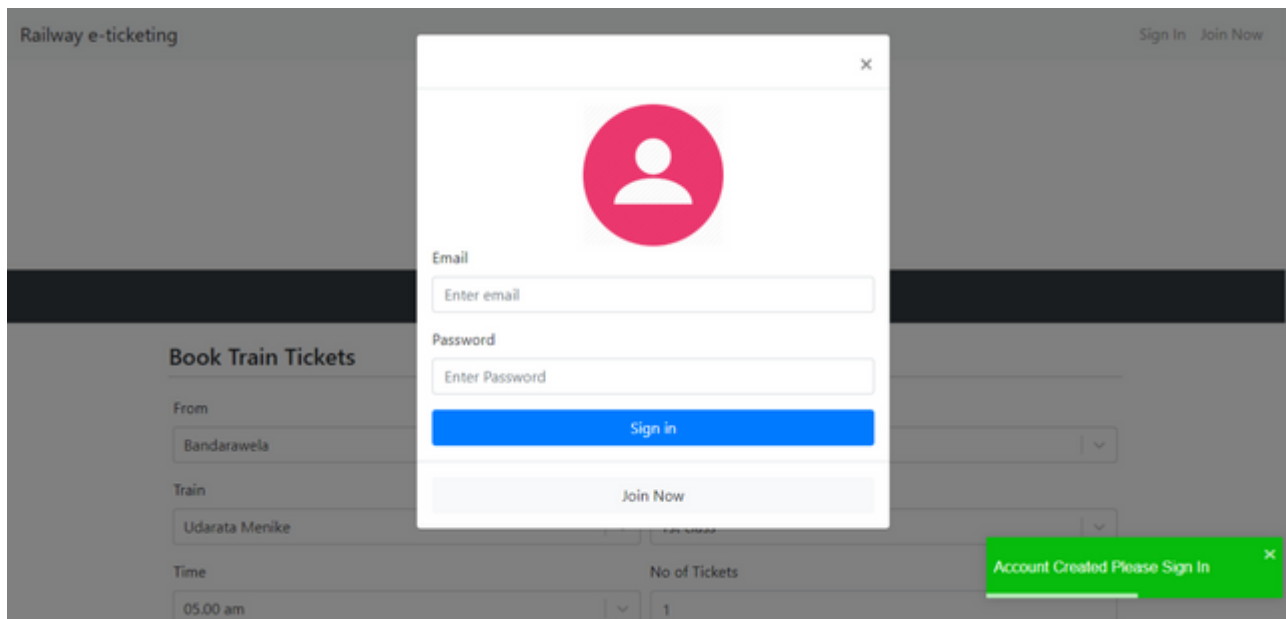


Figure 9.1 login page

This will be the page after logging in to your account



Figure 9.2 home window



Then you can fill in your journey details for booking the tickets in the following window

The screenshot shows a web form titled "Book Train Tickets" with a dark header bar containing "Home" and "Contact Us". The form fields are as follows:

- From:** Dropdown menu with "Bandarawela" selected.
- To:** Dropdown menu with "Colombo Fort" selected.
- Train:** Dropdown menu with "Udarata Menike" selected.
- Class:** Dropdown menu with "1st class" selected.
- Time:** Dropdown menu with "05.00 am" selected.
- No of Tickets:** Input field with "1" entered.
- Date:** Input field with "2019-06-01" entered.

Below the form is a summary table:

Amount	420.00 LKR
Discount	0.00 LKR
Total	420.00 LKR

A blue "Make Reservation" button is located at the bottom left of the form.

Figure 9.3 booking page

After filling all the details properly then the user is asked to enter the personal details for safety information.

The screenshot shows the "Railway e-ticketing" page with a modal window for personal information. The modal contains the following fields:

- First name:** Input field with placeholder "Enter first name".
- Last name:** Input field with placeholder "Enter last name".
- Phone:** Input field with placeholder "Enter Phone Number".
- NIC:** Input field with placeholder "Enter NIC (Optional)".
- Address:** Text area with placeholder "Enter Address".
- Email:** Input field with placeholder "Enter email".
- Password:** Input field with placeholder "Enter Password".

At the bottom of the modal are two buttons: "Create account" (blue) and "Sign in" (grey). The background shows the "Book Train Tickets" form from Figure 9.3, which is dimmed.

Figure 9.4 personal information page

<b>EX NO: 11</b>	<b>Module Implementation (Phase 3) Scrum Master to Induce New requirements in Agile Development, Scrum Master to Induce New</b>
<b>DATE:</b>	<b>Issues in Agile Development, Code development</b>

### ***11.1. MODULE IMPLEMENTATION (PHASE 3): -***

In result component, the user is finally redirected to the payment window where the payment for the ticket can be done .

**BOOK train tickets**

---

From  To

Train  Class

Time  No of Tickets

Date

Amount	420.00 LKR
Discount	42.00 LKR
Total	378.00 LKR

[Make Reservation](#)

Fig.11.1 payment window

## 11.2 CODE DEVELOPMENT: -

The local train ticketing system is developed using HTML, CSS, JAVASCRIPT. Functional decomposition of the system and its key modules are provided to explain the major functionalities proffered by the system. Also, use case diagram is presented to show the different categories of the system users and the various functionalities associated the different system user

```
1  import React, { Component } from 'react'
2  import { Modal, Button, Form, Image, Row } from 'react-bootstrap'
3  import { login } from '../Services'
4  import { getHash } from './commons/Functions'
5  class Login extends Component {
6    constructor(props, context) {
7      super(props, context)
8      this.state = {
9        // validated: false,
10       modalShowErr: false,
11       modalErrMsg: "Incorrect username or password!!!",
12       username: "",
13       password: ""
14     }
15     this.baseState = this.state
16   }
17   componentWillUnmount() {
18     this.setState(this.baseState)
19   }
20   handleChange = type => event => {
21     let value = event;
22     if (event.target) {
23       value = event.target.value;
24     }
25     this.setState({ [type]: value })
26   }
27   handleSubmit = event => {
28     this.setState({ modalShowErr: false })
29     const form = event.currentTarget
30     if (form.checkValidity() === true) {
31       login({ username: this.state.username, password:
32 getHash(this.state.password) })
33       .then(res => {
34         localStorage.setItem('user', JSON.stringify(res))
35         this.props.handleClose()
36       })
37     }
38   }
39 }
```

Figure 11.1 sample code

```

37  ✓ .catch(err => {
38    console.log(err)
39    this.setState({ modalShowErr: true })
40  })
41  }
42  event.preventDefault()
43  event.stopPropagation()
44  }
45  ✓ joinClick = () => {
46    this.props.handleClose()
47    this.props.handleRegisterShow()
48  }
49  ✓ render() {
50  ✓   return (
51  ✓     <Modal show={this.props.showLogin} onHide={this.props.handleClose}>
52  ✓     <Form onSubmit={e => this.handleSubmit(e)}>
53       <Modal.Header closeButton>
54     </Modal.Header>
55     <Modal.Body>
56     <Row style={{ alignItems: 'center', justifyContent: 'center'
57   }}>
58     <Image src={require("../images/login.png")} width='30%'
59   />
60   </Row>
61   <Form.Group controlId="formBasicEmail">
62     <Form.Label>Email</Form.Label>
63     <Form.Control required type="username" placeholder="Enter
64   email" onChange={this.handleChange('username')} />
65   </Form.Group>

```

```

66     <Form.Group controlId="formBasicPassword">
67       <Form.Label>Password</Form.Label>
68       <Form.Control required type="password" placeholder="Enter
69   Password" onChange={this.handleChange('password')} />
70     </Form.Group>
71     {this.state.modalShowErr && <p style={{ color: 'red'
72   }}>{this.state.modalErrMsg}</p>}
73     <Button variant="primary" type="submit" block>
74       Sign in
75     </Button>
76   </Modal.Body>
77   <Modal.Footer>
78     <Button variant="light" block onClick={this.joinClick}>
79       Join Now
80     </Button>
81   </Modal.Footer>
82   </Form>
83   </Modal>
84   );
85   }
86 }
87 export default Login;

```

Figure 11.2 & 11.3 sample code

<b>EX NO: 12</b>	<b>Master Test Plan, Test Case Design(Phase 1)</b>
<b>DATE:</b>	

### ***12.1. MASTER TEST PLAN: -***

<b>TESTING OBJECTIVE</b>	<b>FOCUSING ON PERFORMANCE ISSUE</b>
Test Items	Login system, Registration system, travel information, Payment System
Features to be tested	Login verification, Registration feature, travel information , Payment feature
Features not to be tested	Database Connectivity, Payment verifier
Approach	Method – Manual Testing
Required Hardware/Software	A PC with 8 GB RAM, Internet Connectivity
Risks	Instability of the product
Testers & Schedule	Tester: SELIN RIONA V Scheduling Information: 25 <sup>th</sup> April 2021, 3:00 PM
Estimate	Rs800/- (Excluding Tax and other charges)

### ***12.2. TEST CASE DESIGN: -***

#### **➤ Testing:**

- The process of executing a system with the intent of finding an error.
- Testing is defined as the process in which defects are identified, isolated, subjected for rectification and ensured that product is defect free in order to produce the quality product and hence customer satisfaction. Quality is defined as justification of the requirements

- Defect is nothing but deviation from the requirements.
- Testing --- The presence of bugs
- Testing can demonstrate the presence of bugs, but not their absence
- Debugging and Testing are not the same thing!
- Testing is a systematic attempt to break a program or the AUT
- Debugging is the art or method of uncovering why the script /program did not execute properly.

➤ **Testing Methodologies:**

- **Black box Testing:** is the testing process in which tester can perform testing on an application without having any internal structural knowledge of application. Usually Test Engineers are involved in the black box testing.
- **White box Testing:** is the testing process in which tester can perform testing on an application with having internal structural knowledge. Usually, The Developers are involved in white box testing.
- **Gray Box Testing:** is the process in which the combination of black box and white box techniques are used.

➤ **Positive Test Case:**

- The positive flow of the functionality must be considered
- Valid inputs must be used for testing.
- Must have the positive perception to verify whether the requirements are justified.

➤ **Negative Test Case: -**

- Must have negative perception.
- Invalid inputs must be used for test.

<b>EX NO: 13</b>	<b>Manual Testing</b>
<b>DATE:</b>	

### 13.1. MANULE TESTING: -

TEST AREA	INPUT	TEST DESCRIPTION	OUTPUT/RESULT
Login Module	Username and Password	Permits the user to enter into the application	Tested
Information Module	Fill the form	Allows the user to enter the journey details	Tested
Payment Module	Click validate and pay option	Checks whether the payment feature is functioning and secure	Tested

<b>EX NO: 14</b>	<b>User Manual, Analysis of Costing, Effort and Resources</b>
<b>DATE:</b>	

## ***14.1. USER MANUAL: -***

### ***14.1.1 Introduction:***

A local train ticketing system project for local trains that allows users to book local train tickets and get ticket receipt online. This local train project provides login rights for normal users and admin

### ***14.1.2. Getting Started:***

Go to our website and login for more details.

#### ***14.1.2 a. Quick Start:***

**Step 1:** Go to the website

**Step 2:** Users need to Log In to use the website by providing their username and password.

**Step 3:** Once you are logged in, your profile will be opened.

**Step 4:** Now, you can see the booking window open where you can fill in all the details

**Step 5:** After proceeding the user will be redirected to payment method after completing the payment the ticket will be issued.

#### ***14.1.2 b. System Requirements:***

- Smartphone with Android versions 5.0 and above.
- Internet connection for Application to function.
- Windows support up to latest version.

### ***14.1.3. Troubleshooting:***

Missing or Incorrect Password or E-Mail. A message will be displayed in the event Try again with proper credentials to access



## **14.2. ANALYSIS OF COSTING, EFFORT AND RESOURCES: -**

### **➤ DEVELOPMENT OF PROJECT:**

<b>RESOURCE REQUIREMENT</b>	<b>COST</b>
Computer with core i7 8 <sup>th</sup> gen processor, at least 8GB of RAM, running on windows 10.	Rs. 75000/-
Code	Open Source
Printing	Rs. 750/-

### **➤ SERVER-END:**

<b>RESOURCE REQUIREMENT</b>	<b>COST</b>
My SQL	Enterprise Edition Rs. 12000/-
http web services	Std edition Rs. 6000/-
UPS	Rs. 3000/-

### **➤ OTHER COSTS:**

Employee salary	-
Maintenance cost	Rs. 1500/- per month

