ST. ANTONY'S PUBLIC SCHOOL

CBSE AFFILIATION NO: 930037, SCHOOL NO: 75036

CLASS XII

RECORD OF PROJECT WORK IN COMPUTER SCIENCE

Name : Karthik Sambhu R

CBSE Roll No :

ST. ANTONY'S PUBLIC SCHOOL

CBSE AFFILIATION NO: 930037, SCHOOL NO: 75036

Project Report submitted in fulfillment of Class XII

Syllabus Requirement By

Karthik Sambhu R

CERTIFICATE

This is to certify that this Project titled RAILWAY MANAGEMENT SYSTEM is the record of bonafide project work carried out by Karthik Sambhu R Roll . No. _____ of class XII, Section D, during the academic year 2023 – 2024

Teacher in Charge Principal Examiner

ACKNOWLEDGEMENT

This project is the fruit of the diligent labor that has been invested in it by our group. Apart from our personal hard work, this project has received great support from the school, without which its presentation would have been impossible.

We take this opportunity to thank our manager Rev. Fr. Sabu Panachikkal, our Principal Fr. Antony Thokkanattu, Vice Principal Fr. Shiju Varghese and our teachers who have helped us in our endeavor. A special mention is to make about our computer teachers, Mr. Cherian K Abraham, and Mrs. Thanuja Mathew in this regard. We would also like to thank our friends for their co-operation and suggestions. Finally, we thank the Almighty for all his blessings.

ABSTRACT

Railway Management System is a system which maintains the information about the trains operating in the railway network, their routes, tickets, and administration. This is very difficult to organize manually. Maintenance of all these information manually is a very complex task. Owing to the advancement of technology, organization of a Digital Railway Management System is a need of the hour. The Digital Railway Management has been designed to computerize and automate the operations performed over the information about the trains, passengers, and all other operations. This computerization of the railway helps in many instances of its maintenance. It reduces the workload of the management as most of the manual work done is reduced and automate through the Digital system.

CONTENTS

1. INTRODUCTION

1.1 PROJECT AIMS AND OBJECTIVES

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

- Railway Management System
- Admin/Staff login menu
- Facility to add trains.
- Facility to display all available trains.
- An Admin operation where admin can modify train details, and add new trains.
- An User operation where users can purchase tickets.

1.2 BACKGROUND OF THE PROJECT

The development of a Railway Management System stems from the growing complexity and demands of modern railways. Railways serve as vital hubs for transportation, connecting people and goods across vast distances. Managing the multitude of operations within a railway network requires a sophisticated system that can streamline processes, enhance efficiency, ensure safety, and improve the overall passenger experience.

2.SYSTEM ANALYSIS

In this chapter, we will discuss and analyze the developing process of Railway Management System, including software requirement specification (SRS) and comparison between existing and proposed systems. The functional and non-functional requirements are included in the SRS part to provide a complete description and overview of the system requirements before the developing process is carried out. Besides that, existing vs proposed provides a view of how the proposed system will be more efficient than the existing one.

2.1 SOFTWARE REQUIREMENT SPECIFICATION

Railway Management System is a computerized system that helps the user / admin to manage the daily activities of a railway station in an electronic format. It reduces the risk of paperwork, such as loss of registers, damage to registers, and other time-consuming activities in the station. It can help the user manage transactions or records more effectivelyand, in a time-saving manner.

PROBLEM STATEMENT:

The problem occurred before having computerized system includes:

- Loss of registers

  When a computerized system is not implemented, records are always under the threat of mishandling. Sometimes, due to some human error, there may be a lossof records.

- Damaged records

  When a computerized system is not there, files are always lost due to accidentslike spilling of water by some member on records accidentally. Besides, some natural disasters like floods or fires may also damage the files.

- Difficult to search records

  When there is no computerized system there is always a difficulty in searching of

  records if the records are large in number.

- Space consuming

  As the number of records become more, the space for physical storage of records also increases if no computerized system is implemented.

- Cost consuming

  As there is no computerized system, every year records/ registers will be required, which will increase the cost for the management of the directory.

2.2 EXISTING VS PROPOSED

Existing System:

Early days Directories are managed manually. It required a lot of time to record or to retrieve the details. The employees who must record the details must perform their job very carefully. Even a small mistake would create a lot of problems. Security of information is very less. Reporting generations of all the information is a very tough task. Maintenance of the Directory catalogue and arrangement of the numbers in the catalogue is a very complex task. In addition to its maintenance of member details, manually is a complex task.

Proposed System:

To solve the problems mentioned in the existing system, a Railway Management System is proposed. The proposed system contains the following features.

- Add Train
- Get details about train and tickets
- Book tickets as the user
- Manage Bookings and cancelations.

2.3 SYSTEM SPECIFICATIONS

HARDWARE SPECIFICATIONS

The following is the hardware specification of the system on which the software has been developed:-

Operating System : Windows 7 / 8 / 10 / 11

Windows 11 is used as the operating system as it is stable and supports more features and is more user friendly.

Machine : Pentium Dual Core Processor 2.6 GHz or above,

2 GB RAM or above,

500 GB Hard Disk or above

We used Intel core i5-11th generation-based system, it is faster than other processors and provides reliable and stable performance and we can run our PC for a long time. By using this processor, we can keep on developing our project without any worries. 8 GB RAM is used as it will provide fast reading and writing capabilities and will support processing.

SOFTWARE SPECIFICATIONS

Front End Used : PYTHON 3.8.0 or above

Backend Used : MySQL 5.7.43

3.SYSTEM DESIGN

3.1 MYSQL TABLE STRUCTURE

```
mysql> use railway;
Database changed
mysql> show tables;
+--------------------+
| Tables_in_railway |
+--------------------+
| tickets            |
| trains             |
| users              |
+--------------------+
3 rows in set (0.00 sec)
```

```
mysql> desc trains;
+------------------+--------------+------+-----+---------+-------+
| Field            | Type         | Null | Key | Default | Extra |
+------------------+--------------+------+-----+---------+-------+
| id               | int          | NO   | PRI | NULL    |       |
| name             | varchar(255) | YES  |     | NULL    |       |
| capacity         | int          | YES  |     | NULL    |       |
| starting_station | varchar(255) | YES  |     | NULL    |       |
| ending_station   | varchar(255) | YES  |     | NULL    |       |
| price_1st_class  | int          | YES  |     | NULL    |       |
| price_2nd_class  | int          | YES  |     | NULL    |       |
| price_3rd_class  | int          | YES  |     | NULL    |       |
| price_general    | int          | YES  |     | NULL    |       |
+------------------+--------------+------+-----+---------+-------+
9 rows in set (0.00 sec)
```

```
mysql> desc users;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| username | varchar(100) | NO   | PRI | NULL    |       |
| PASSWORD | varchar(100) | YES  |     | NULL    |       |
| email    | varchar(255) | YES  |     | NULL    |       |
| NAME     | varchar(255) | YES  |     | NULL    |       |
| phoneno  | varchar(15)  | YES  |     | NULL    |       |
| address  | varchar(255) | YES  |     | NULL    |       |
| money    | int          | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

```
mysql> desc tickets;
+---------------------+--------------+------+-----+---------+-------+
| Field               | Type         | Null | Key | Default | Extra |
+---------------------+--------------+------+-----+---------+-------+
| ticket_id           | int          | NO   | PRI | NULL    |       |
| train_id            | int          | YES  |     | NULL    |       |
| username            | varchar(255) | YES  |     | NULL    |       |
| departure_station   | varchar(255) | YES  |     | NULL    |       |
| destination_station | varchar(255) | YES  |     | NULL    |       |
| price               | int          | YES  |     | NULL    |       |
+---------------------+--------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Sql Data

```
mysql> select * from trains;
+------+-----------------+----------+------------------+----------------+-----------------+-----------------+-----------------+---------------+
| id   | name            | capacity | starting_station | ending_station | price_1st_class | price_2nd_class | price_3rd_class | price_general |
+------+-----------------+----------+------------------+----------------+-----------------+-----------------+-----------------+---------------+
| 1523 | Hampi Express   |      100 | Mysuru           | Hubballi       |            2500 |            1500 |            1000 |           500 |
| 1524 | Chennai Express |       90 | Chennai          | Bangalore      |             450 |             350 |             250 |           150 |
| 1600 | Shatabdi Express|      120 | Chennai          | Bangalore      |            3000 |            2500 |            2000 |          1000 |
+------+-----------------+----------+------------------+----------------+-----------------+-----------------+-----------------+---------------+
3 rows in set (0.00 sec)
```

```
mysql> select * from users;
+----------+-----------+-------------------+---------------------+----------------+---------------+-------+
| username | PASSWORD  | email             | NAME                | phoneno        | address       | money |
+----------+-----------+-------------------+---------------------+----------------+---------------+-------+
| admin    | admin@123 | admin@gmail.com   | Administrator       | +91 1234567890 | SAPS, Anakkal | 10000 |
| harrin   | 12345     | harrin@gmail.com  | Harrin Joy Nelwin   | 12345366789    | Trissur       | 48000 |
| karthik  | 54321     | karthik@gmail.com | Karthik Sambhu R    | 1234567890     | Kottayam      | 69000 |
| rohan    | Abc123    | rohan@gmail.com   | Rohan Joseph Sam    | +91 1231234560 | Kanjirappally |  1400 |
| yaseen   | 12@12     | yaseen@gmail.com  | K S Muhammad Yaseen | 9495020761     | Anakkal       |  5700 |
+----------+-----------+-------------------+---------------------+----------------+---------------+-------+
5 rows in set (0.00 sec)
```

```
mysql> select * from tickets;
+-----------+----------+----------+--------------------+----------------------+-------+
| ticket_id | train_id | username | departure_station | destination_station | price |
+-----------+----------+----------+--------------------+----------------------+-------+
|         0 |        0 | 0        | 0                  | 0                    |     0 |
|         1 |     1600 | karthik  | Chennai            | Bangalore            |  3000 |
|         2 |     1523 | yaseen   | Mysuru             | Devanagari           |  1500 |
+-----------+----------+----------+--------------------+----------------------+-------+
3 rows in set (0.00 sec)
```

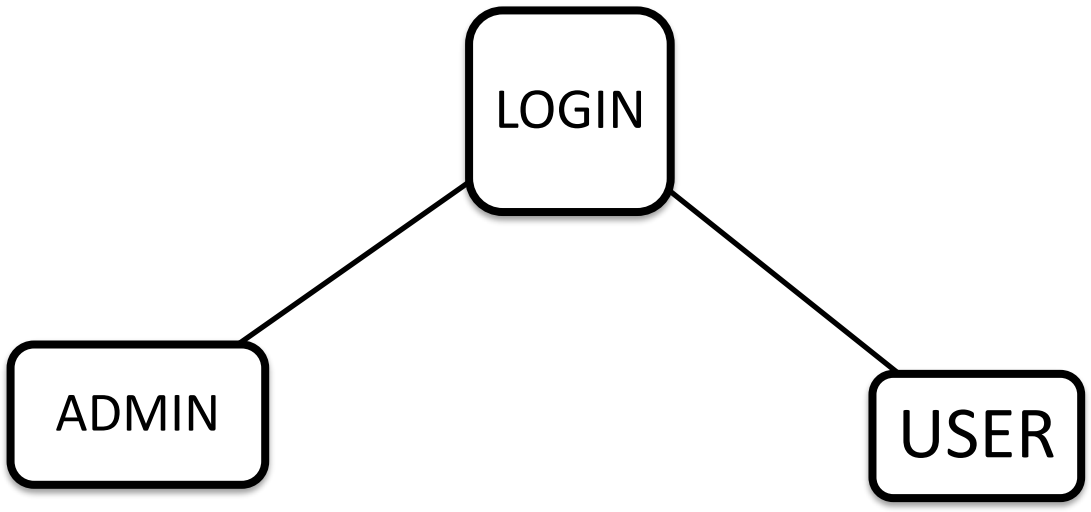
3.2 MENU STRUCTURE

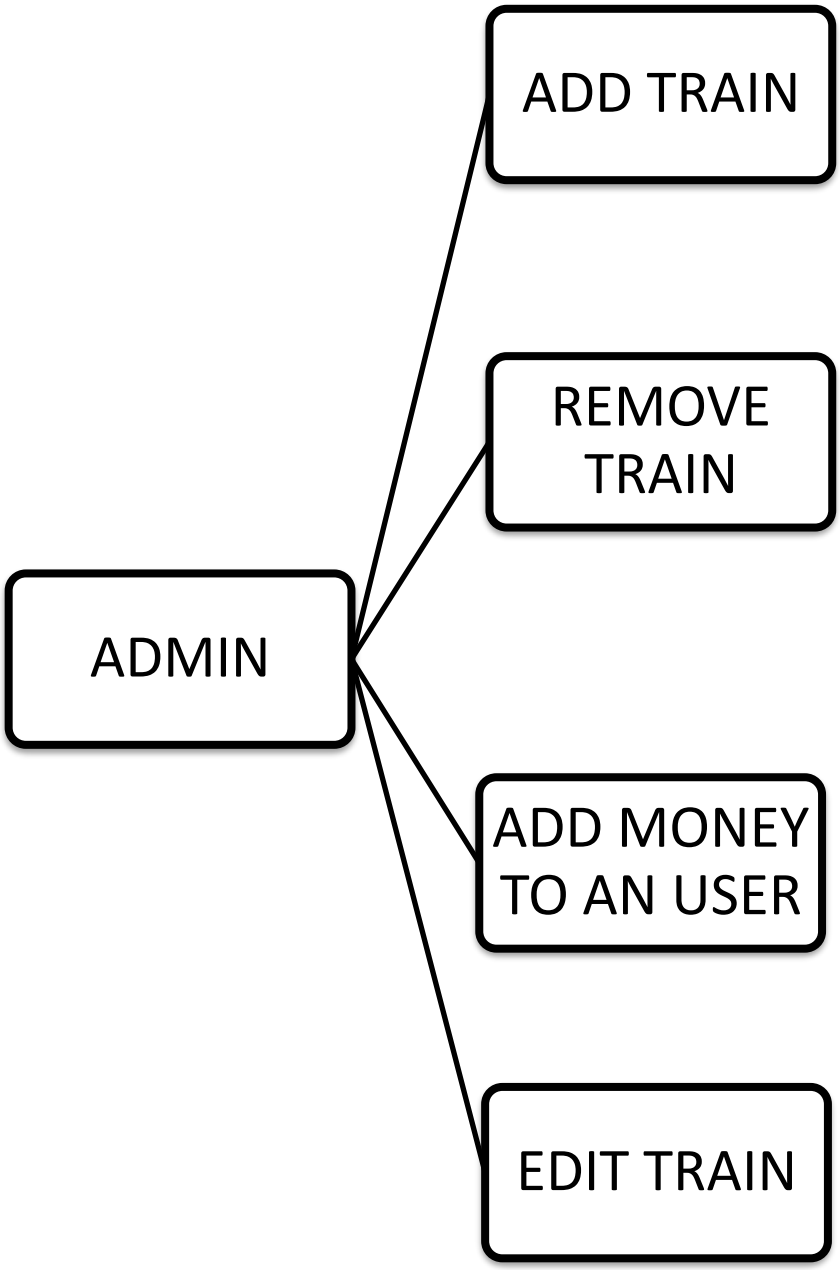
- MAIN MENU
  - LOGIN
  - SIGNUP
  - EXIT
- ADMIN MENU
  - VIEW TRAINS
  - VIEW TRAINS
  - ADD TRAIN
  - RMOVE TRAIN
  - EDIT TRAIN DETAILS
  - ADD BALANCE TO A USER
  - LOGOUT
- USER MENU
  - VIEW TRAINS
  - BUY TICKET
  - VIEW TICKETS
    - VIEW ALL TICKETS
    - SEE DETAILS OF A SPECIFIC TICKET
    - CANCEL TICKET

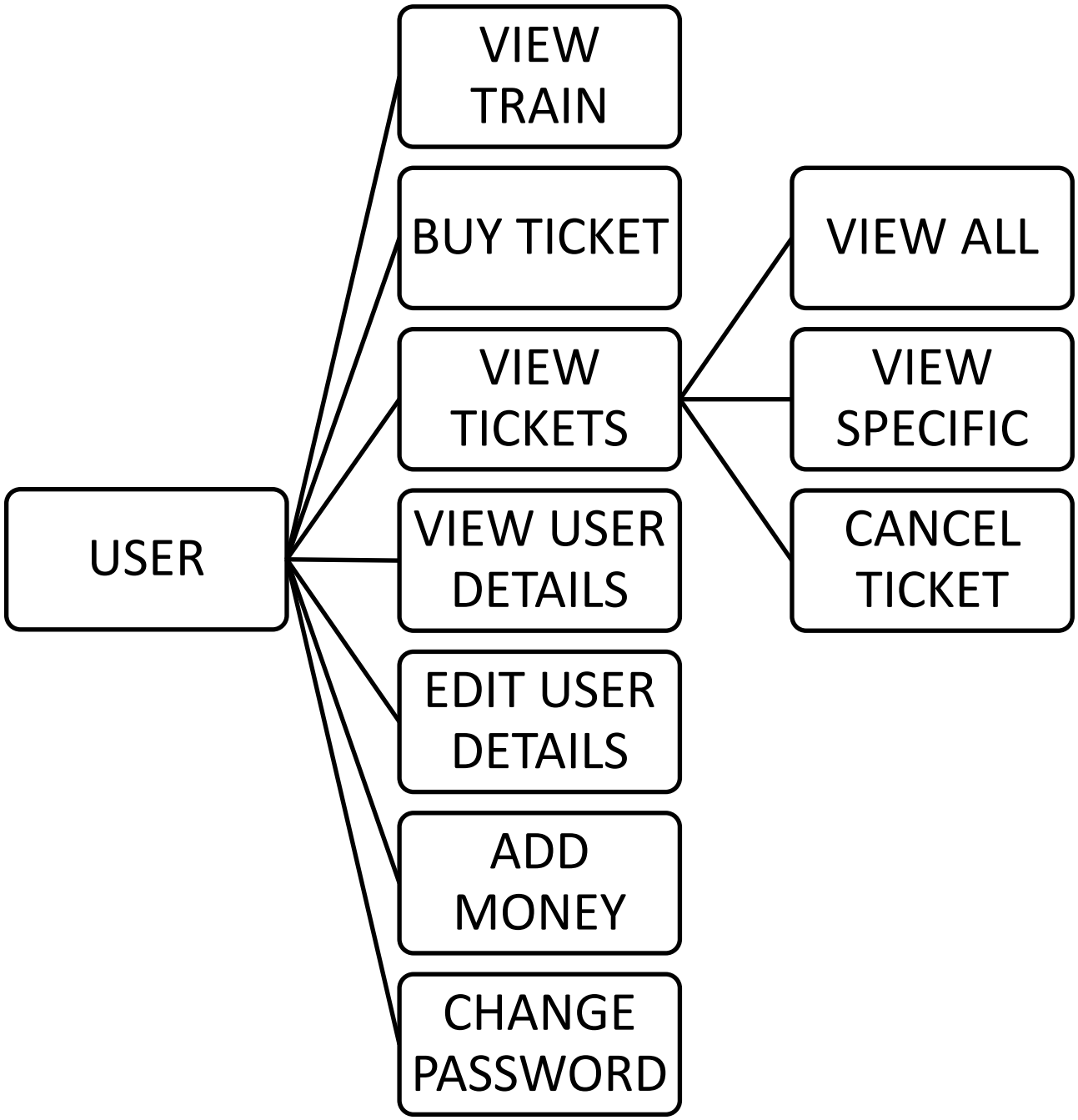
- GO BACK

- VIEW USER DETAILS
- EDIT USER DETAILS
- DEPOSIT MONEY
- CHANGE PASSWORD
- LOGOUT

3.3 DATA FLOW DIAGRAMS




ADMIN
ADD TRAIN
REMOVE TRAIN
ADD MONEY TO AN USER
EDIT TRAIN


USER
VIEW TRAIN
BUY TICKET
VIEW TICKETS
VIEW USER DETAILS
EDIT USER DETAILS
ADD MONEY
CHANGE PASSWORD
VIEW ALL
VIEW SPECIFIC
CANCEL TICKET

4. SYSTEM IMPLEMENTATION

4.1 SOURCE CODE AND MODULE DESCRIPTION

main.py

```
import mysql.connector

print(
   """
============================================================
      Welcome to the Railway Management System
============================================================"""
)


def signup():
   print()

   cursor.execute("SELECT username FROM users")
   users = cursor.fetchall()

   while True:
      username = input("Enter Username (0 To Cancel): ")

      if username == "0":
         return 0
      username = username.lower()

      if (username,) not in users:
         break
      else:
         print("\nUsername Already Exists Try Again!\n")

   password = input("Enter A Password: ")
   name = input("Enter Your Name: ")
   email = input("Enter Your Email: ")
   phoneno = input("Enter Your Phone Number: ")
   address = input("Enter Your Address: ")

   query = 'INSERT INTO users VALUES ("{}","{}","{}","{}","{}","{}",0)'.format(
      username, password, email, name, phoneno, address
   )
```

```
    cursor.execute(query)
    connection.commit()

    return username


def login():
    print()

    cursor.execute("SELECT username FROM users")
    users = cursor.fetchall()

    while True:
        username = input("Enter Your Username: ")

        if (username,) in users:
            password = input("Enter Your Password: ")
            cursor.execute(
                "SELECT password FROM users WHERE username = '{}'".format(username)
            )

            if password == cursor.fetchall()[0][0]:
                return username
            else:
                print("\nWrong Password!\n")

        elif username == "0":
            return 0
        else:
            print("\nWrong Username!\n")


def Ticket(id):
    query = """SELECT *
               FROM tickets
               WHERE ticket_id = {}""".format(
        id
    )
    cursor.execute(query)
    ticket = cursor.fetchone()

    ticket_id = ticket[0]
    train_id = ticket[1]
    username = ticket[2]
    departure_station = ticket[3]
    destination_station = ticket[4]
    price = ticket[5]
```

```
    query = """SELECT name,starting_station,ending_station
            FROM trains
            WHERE id = {}""".format(
        train_id
    )
    cursor.execute(query)
    train = cursor.fetchone()

    train_name = train[0]
    starting_station = train[1]
    ending_station = train[2]

    query = '''SELECT name
            FROM users
            WHERE username = "{}"'''.format(
        username
    )
    cursor.execute(query)

    name = cursor.fetchone()[0]

    print(
        """--------------------------------------------------------------------------------
{:^70}
Ticket ID: {}                          Ticket Price: {} Rupees
                        Name: {}
                        Train Starting Station: {}
                        Train Destination: {}
                        From: {}
                        To: {}
--------------------------------------------------------------------------------""".format(
            train_name,
            ticket_id,
            price,
            name,
            starting_station,
            ending_station,
            departure_station,
            destination_station,
        )
    )

    print()


def TicketMenu():
```

```
    while True:
        print(
            """
_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
            TICKET MENU
        1. View All Tickets
        2. Get Details Of A Specific Ticket
        3. Cancel Ticket
        4. Go Back"""
        )
        choice = input("\n       Enter A Choice >>> ")

        if choice == "1":
            query = """SELECT ticket_id,departure_station,destination_station,name,tickets.train_id
            FROM tickets,trains
            WHERE tickets.username = "{}" AND tickets.train_id = trains.id""".format(
                username
            )

            cursor.execute(query)
            tickets = cursor.fetchall()

            if tickets == []:
                print("No Tickets!")
            else:
                print(
                    "\n---------------------------------------------------------------------------------"
                )
                for ticket in tickets:
                    print(
                        "Ticket:({}) | Train:  {} (ID: {}) From {} To {}".format(
                            ticket[0], ticket[3], ticket[4], ticket[1], ticket[2]
                        )
                    )
                    print(
                        "---------------------------------------------------------------------------------"
                    )

        elif choice == "2":
            query = """SELECT ticket_id
                    FROM tickets"""
            cursor.execute(query)
            ids = cursor.fetchall()
            while True:
                id = int(input("Enter The ID Of The Ticket(0 To Go Back): "))

                if id == 0:
```

```
            continue

        elif (id,) in ids:
            Ticket(id)
            break

        else:
            print("\nEnter A Valid ID!\n")

    elif choice == "3":
        cursor.execute("SELECT ticket_id,username FROM tickets")
        ids = cursor.fetchall()

        while True:
            ticket_id = int(
                input("Enter ID Of The Ticket You Want To Cancel (0 To Go Back): ")
            )
            if ticket_id == 0:
                break

            elif (ticket_id, username) in ids:
                query = """SELECT departure_station,destination_station,price,name
                FROM tickets, trains
                WHERE tickets.train_id = trains.id AND tickets.ticket_id = {}""".format(
                    ticket_id
                )

                cursor.execute(query)
                details = cursor.fetchone()
                if (
                    input(
                        """Are You Sure You Want To Cancel You Ticket On
        {} From {} To {}
Enter Your Choice (Y/N): """.format(
                            details[3], details[0], details[1]
                        )
                    ).upper()
                    == "Y"
                ):
                    query = "DELETE FROM tickets WHERE ticket_id = {}".format(
                        ticket_id
                    )
                    cursor.execute(query)
                    connection.commit()

                    print("Your Ticket Has Been Cancelled!\n")
```

```
                query = "UPDATE users SET money = money + {} WHERE username = '{}'".format(
                    details[2], username
                )
                cursor.execute(query)
                connection.commit()

                print("You Have Been Refunded Rupees: {}!".format(details[2]))
            else:
                print("\nTicket Not Found!\n")

        elif choice == "4":
            break

        else:
            print("\nEnter A Valid Option!\n")


def AdminMenu():
    while True:
        choice = input(
            """\n============ ADMIN MENU ============
        1. View All Trains
        2. View Prices
        3. Add train
        4. Remove Train
        5. Edit Train Details
        6. Add Balance To A User
        7. Logout

        Enter a choice >>> """
        )
        if choice == "1":
            print(
                "\n-------------------------------------------------------------------------------------------------------------"
            )

            print("|", end="")

            for i in ("ID", "Name", "Capacity", "Starting Station", "Ending Station"):
                print("{:^20}|".format(i), end="")

            print(
                "\n-------------------------------------------------------------------------------------------------------------"
            )

            cursor.execute(
                "SELECT id,name,capacity,starting_station,ending_station FROM trains"
```

```
        )
        trains = cursor.fetchall()

        for train in trains:
            print("|", end="")

            for i in train:
                print("{:^20}|".format(i), end="")

            print(
                "\n-----------------------------------------------------------------------------------------------------------"
            )

        print()
    elif choice == "2":
        print()
        cursor.execute("SELECT id FROM trains")
        ids = cursor.fetchall()

        while True:
            id = int(input("Enter ID Of The Train To View Price (0 To Go Back): "))

            if id == 0:
                print()
                break

            elif (id,) in ids:
                query = "SELECT name,price_1st_class,price_2nd_class,price_3rd_class,price_general
FROM trains WHERE id = {}".format(
                    id
                )
                cursor.execute(query)
                train = cursor.fetchone()

                if train == ():
                    print("\n>>>>> Train Not Found!\n")
                    continue

                print(
                    """\n          ----------------------------------------
                Train ID: {}
                Train Name: {}
                First Class Price: {}
                Second Class Price: {}
                Third Class Price: {}
                General Price: {}
            ----------------------------------------\n""".format(
```

```
                id, train[0], train[1], train[2], train[3], train[4]
            )
        )

        break
    else:
        print("\n>>>>> Train Not Found!\n")
        continue
elif choice == "3":
    print()
    cursor.execute("SELECT id FROM trains")
    ids = cursor.fetchall()

    while True:
        id = int(input("Enter An ID For The New Train (0 To Go Back): "))

        if id == 0:
            break

        elif (id,) in ids:
            print("\n>>>>> Train Already Exists!\n")
            continue
        else:
            name = input("Enter A Name For The Train: ")
            starting_station = input("Enter The Starting Station: ")
            ending_station = input("Enter The Ending Station: ")
            capacity = int(input("Enter The Capacity: "))
            price_1st_class = int(input("Enter The Price For First Class: "))
            price_2nd_class = int(input("Enter The Price For Second Class: "))
            price_3rd_class = int(input("Enter The Price For Third Class: "))
            price_general = int(input("Enter The Price For General: "))

            query = "INSERT INTO trains VALUES ({},'{}',{},'{}','{}',{},{},{},{})".format(
                id,
                name,
                capacity,
                starting_station,
                ending_station,
                price_1st_class,
                price_2nd_class,
                price_3rd_class,
                price_general,
            )
            cursor.execute(query)
            connection.commit()

            print("Train Added!\n")
```

```
            break
    elif choice == "4":
        print()
        cursor.execute("SELECT id FROM trains")
        ids = cursor.fetchall()

        while True:
            id = int(input("Enter The ID Of The Train To Remove (0 To Go Back): "))
            if id == 0:
                break

            elif (id,) in ids:
                print("Train Removed!\n")

                query = "DELETE FROM trains WHERE id = {}".format(id)
                cursor.execute(query)
                connection.commit()

                break
            else:
                print("\n>>>>> Train Not Found!\n")
                continue
    elif choice == "5":
        print()
        cursor.execute("SELECT id FROM trains")
        ids = cursor.fetchall()

        while True:
            id = int(input("Enter ID Of The Train To Edit (0 To Go Back): "))
            if id == 0:
                break

            elif (id,) in ids:
                name = input("Enter New Name Of The Train: ")
                starting_station = input("Enter The Starting Station: ")
                ending_station = input("Enter The Ending Station: ")
                capacity = int(input("Enter The Capacity: "))
                price_1st_class = int(input("Enter The Price For First Class: "))
                price_2nd_class = int(input("Enter The Price For Second Class: "))
                price_3rd_class = int(input("Enter The Price For Third Class: "))
                price_general = int(input("Enter The Price For General: "))

                query = """UPDATE trains
                SET name = '{}',
                starting_station = '{}',
                ending_station = '{}',
                capacity = {},
```

```
            price_1st_class = {},
            price_2nd_class = {},
            price_3rd_class = {},
            price_general = {}
            WHERE id = {}""".format(
                name,
                starting_station,
                ending_station,
                capacity,
                price_1st_class,
                price_2nd_class,
                price_3rd_class,
                price_general,
                id,
            )
            cursor.execute(query)
            connection.commit()

            print("Train Edited!\n")
            break
        else:
            print("\n>>>>> Train Not Found!\n")
            continue
    elif choice == "6":
        print()
        cursor.execute("SELECT username FROM users")
        usernames = cursor.fetchall()

        while True:
            username = input("Enter Username Of User (0 To Go Back): ")
            if username == "0":
                print()
                break

            elif (username,) in usernames:
                balance = int(input("How Much Money Do You Want To Add: "))

                query = "UPDATE users SET money = money + {} WHERE username = '{}'".format(
                    balance, username
                )
                cursor.execute(query)
                connection.commit()

                break
            else:
                print("\n>>>>> Username Not Found!\n")
    elif choice == "7":
```

```
            print("\nYou have been logged out!\n")
            break
        else:
            print("\n>>>>> Enter a valid option!\n")


def UserMenu(username):
    while True:
        choice = input(
            """\n============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout

        Enter a choice >>> """
        )

        if choice == "1":
            print(
                "\n-----------------------------------------------------------------------------------------------------------"
            )

            print("|", end="")

            for i in ("ID", "Name", "Capacity", "Starting Station", "Ending Station"):
                print("{:^20}|".format(i), end="")

            print(
                "\n-----------------------------------------------------------------------------------------------------------"
            )

            cursor.execute(
                "SELECT id,name,capacity,starting_station,ending_station FROM trains"
            )
            trains = cursor.fetchall()

            for train in trains:
                print("|", end="")

                for i in train:
                    print("{:^20}|".format(i), end="")
```

```
            print(
                "\n-----------------------------------------------------------------------------------------------------------"
            )

        print()

    elif choice == "2":

        cursor.execute("SELECT MAX(ticket_id) FROM tickets")
        ticket_id = cursor.fetchone()[0] + 1
        cursor.execute("SELECT id FROM trains")
        ids = cursor.fetchall()

        while True:
            train_id = int(input("Enter Train ID: "))

            if (train_id,) in ids:
                departure_station = input("Enter Station Of Departure: ")
                destination_station = input("Enter Destination: ")

                while True:
                    choice = input(
                        """                 Select Class
                    1. 1st Class
                    2. 2nd Class
                    3. 3rd Class
                    4. General
Enter Your Choice >>> """
                    )

                    if choice == "1":
                        Class = "price_1st_class"
                        break
                    elif choice == "2":
                        Class = "price_2nd_class"
                        break
                    elif choice == "3":
                        Class = "price_3rd_class"
                        break
                    elif choice == "4":
                        Class = "price_general"
                        break
                    else:
                        print("\nEnter A Valid Option!\n")

                query = "SELECT {} FROM trains WHERE id = {}".format(
                    Class, train_id
```

```
            )
            cursor.execute(query)
            price = cursor.fetchone()[0]

            cursor.execute(
               'SELECT money FROM users WHERE username = "{}"'.format(username)
            )
            balance = cursor.fetchone()[0]

            if balance < price:
               print("\nNot Enough Money In Your Account!\n")
               break

            query = (
               'INSERT INTO tickets VALUES ({},{},"{}","{}","{}",{})'.format(
                  ticket_id,
                  train_id,
                  username,
                  departure_station,
                  destination_station,
                  price,
               )
            )
            cursor.execute(query)
            newbalance = balance - price

            cursor.execute(
               'UPDATE users SET money = {} WHERE username = "{}"'.format(
                  newbalance, username
               )
            )

            connection.commit()
            print("Ticket Purchased!")
            break
         else:
            print("\nEnter A Valid Train ID!\nPress 1 To View Trains.\n")
            break

   elif choice == "3":
      TicketMenu()

   elif choice == "4":
      query = 'SELECT email,name,phoneno,address,money FROM users WHERE username =
"{}"'.format(
         username
      )
```

```
    cursor.execute(query)
    details = cursor.fetchone()

    print(
        """
    Name: {}
    Username: {}
    Email: {}
    Phone No: {}
    Address: {}
    Balance: {}\n""".format(
            details[1], username, details[0], details[2], details[3], details[4]
        )
    )

elif choice == "5":
    print("\nEnter New Details")
    email = input("Enter A Email: ")
    name = input("Enter Name: ")
    phoneno = input("Enter Phone No: ")
    address = input("Enter Address: ")

    query = """UPDATE users SET
            email = '{}',
            name = '{}',
            phoneno = '{}',
            address = '{}'
            WHERE username = '{}'""".format(
        email, name, phoneno, address, username
    )

    cursor.execute(query)
    connection.commit()

    print("Details Edited!\n")
elif choice == "6":
    balance = int(input("\nHow Much Money Do You Want To Add: "))

    query = "UPDATE users SET money = money + {} WHERE username = '{}'".format(
        balance, username
    )
    cursor.execute(query)
    connection.commit()

    print("Money Deposited!\n")
elif choice == "7":
    newpassword = input("\nEnter New Password: ")
```

```
            query = "UPDATE users SET password = '{}' WHERE username = '{}'".format(
                newpassword, username
            )
            cursor.execute(query)
            connection.commit()

            print("Password Changed!\n")
        elif choice == "8":
            print("\nYou have been logged out!\n")
            break
        else:
            print("\nEnter A Valid Option!\n")


def sql():
    cursor.execute("CREATE DATABASE IF NOT EXISTS railway")

    cursor.execute("USE railway")

    cursor.execute(
        """CREATE TABLE IF NOT EXISTS users (
                username VARCHAR(100) PRIMARY KEY,
                PASSWORD VARCHAR(100),
                email VARCHAR(255),
                NAME VARCHAR(255),
                phoneno VARCHAR(15),
                address VARCHAR(255),
                money INT
                )"""
    )

    cursor.execute(
        """CREATE TABLE IF NOT EXISTS tickets (
                ticket_id INT PRIMARY KEY,
                train_id INT,
                username VARCHAR(255),
                departure_station VARCHAR(255),
                destination_station VARCHAR(255),
                price INT
                )"""
    )
    cursor.execute("SELECT ticket_id FROM tickets")

    if (0,) not in cursor.fetchall():
        cursor.execute('INSERT INTO tickets VALUES (0,0,"0","0","0",0)')
```

```
    connection.commit()

    cursor.execute(
        """CREATE TABLE IF NOT EXISTS trains (
                id INT PRIMARY KEY,
                name VARCHAR(255),
                capacity INT,
                starting_station VARCHAR(255),
                ending_station VARCHAR(255),
                price_1st_class INT,
                price_2nd_class INT,
                price_3rd_class INT,
                price_general INT
            )"""
    )
    cursor.execute("SELECT username FROM users")

    if ("admin",) not in cursor.fetchall():
        cursor.execute(
            'INSERT INTO users VALUES
("admin","admin@123","admin@gmail.com","Administrator","+91 1234567890","SAPS,
Anakkal",10000)'
        )
    connection.commit()
username = input("Enter username for the sql connection: ")
pword = input("Enter password for the sql connection: ")
try:
    connection = mysql.connector.connect(
        host="localhost", user=username, password=pword
    )

    if connection.is_connected():
        cursor = connection.cursor()
        sql()

        while True:
            choice = input(
                """\n
=============== Menu ===============
            1. Login
            2. Signup
            3. Exit

            Enter your choice >>> """
            )

            if choice == "1":
```

```
            username = login()
            if username == "admin":
                AdminMenu()
            elif username == 0:
                continue
            else:
                UserMenu(username)
        elif choice == "2":
            username = signup()
            if username == 0:
                continue
            elif username == "admin":
                AdminMenu()
            else:
                UserMenu(username)
        elif choice == "3":
            break
        else:
            print("\n>>>>> Enter a valid option!\n")

    connection.close()
except Exception as e:
    print(e)
    print("Couldn't Connect To Database")
```

SIGNUP()

This module is used by the user to create account so as to book train tickets.

LOGIN()

This module is used to manage the login action by the user/admin.

TICKET()

This module is used to print the ticket is a user friendly manner.

TICKETMENU()

This module provides a menu to:

- View All Tickets
- Get Details Of A Specifc Ticket
- Cancel Ticket

ADMINMENU()

This module provides a menu to:

- View All Trains
- View Prices
- Add Train
- Remove Train
- Edit Train Details
- Add Balance To A User

USERMENU()

This module provides a menu to:

- View All Trains
- Buy Tickets

- View Tickets
- View User Details
- Edit User Details
- Deposit Money
- Change Password

SQL()

This module is used to setup th sql database for use.

4.2 SCREEN SHOTS

```
============================================================
        Welcome to the Railway Management System
============================================================
Enter username for the sql connection: root
Enter password for the sql connection: tiger


===============  Menu ===============
            1. Login
            2. Signup
            3. Exit

            Enter your choice >>> 1

Enter Your Username: admin
Enter Your Password: admin@123

============ ADMIN MENU ============
        1. View All Trains
        2. View Prices
        3. Add train
        4. Remove Train
        5. Edit Train Details
        6. Add Balance To A User
        7. Logout
        Enter a choice >>> 1
```

ID	Name	Capacity	Starting Station	Ending Station
1523	Hampi Express	100	Mysuru	Hubballi
1524	Chennai Express	90	Chennai	Bangalore
1600	Shatabdi Express	120	Chennai	Bangalore

```
============ ADMIN MENU ============
        1. View All Trains
        2. View Prices
        3. Add train
        4. Remove Train
        5. Edit Train Details
        6. Add Balance To A User
        7. Logout
        Enter a choice >>> 2

Enter ID Of The Train To View Price (0 To Go Back): 1523

            ----------------------------------------
                    Train ID: 1523
                    Train Name: Hampi Express
                    First Class Price: 2500
                    Second Class Price: 1500
                    Third Class Price: 1000
                    General Price: 500
            ----------------------------------------

============ ADMIN MENU ============
        1. View All Trains
        2. View Prices
        3. Add train
        4. Remove Train
        5. Edit Train Details
        6. Add Balance To A User
        7. Logout
        Enter a choice >>> 3

Enter An ID For The New Train (0 To Go Back): 1234
Enter A Name For The Train: XYZ Express
Enter The Starting Station: Kottayam
Enter The Ending Station: Thiruvananthapuram
Enter The Capacity: 100
Enter The Price For First Class: 1500
Enter The Price For Second Class: 1000
Enter The Price For Third Class: 750
Enter The Price For General: 500
Train Added!
```

```
============ ADMIN MENU ============
        1. View All Trains
        2. View Prices
        3. Add train
        4. Remove Train
        5. Edit Train Details
        6. Add Balance To A User
        7. Logout
        Enter a choice >>> 4

Enter The ID Of The Train To Remove (0 To Go Back): 1234
Train Removed!

============ ADMIN MENU ============
        1. View All Trains
        2. View Prices
        3. Add train
        4. Remove Train
        5. Edit Train Details
        6. Add Balance To A User
        7. Logout
        Enter a choice >>> 5

Enter ID Of The Train To Edit (0 To Go Back): 1523
Enter New Name Of The Train: Hampi Express
Enter The Starting Station: Mysuru
Enter The Ending Station: New Delhi
Enter The Capacity: 100
Enter The Price For First Class: 3000
Enter The Price For Second Class: 2500
Enter The Price For Third Class: 1500
Enter The Price For General: 1000
Train Edited!
============ ADMIN MENU ============
        1. View All Trains
        2. View Prices
        3. Add train
        4. Remove Train
        5. Edit Train Details
        6. Add Balance To A User
        7. Logout
        Enter a choice >>> 6

Enter Username Of User (0 To Go Back): xyz
How Much Money Do You Want To Add: 10000
```

```
===============  Menu ===============
            1. Login
            2. Signup
            3. Exit

            Enter your choice >>> 2

Enter Username (0 To Cancel): abc
Enter A Password: 12345
Enter Your Name: Abc Xyz
Enter Your Email: abc@xyz.com
Enter Your Phone Number: +91 1234567890
Enter Your Address: Anakkal



============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout
        Enter a choice >>> 2
Enter Train ID: 1524
Enter Station Of Departure: Chennai
Enter Destination: Bangalore
                    Select Class
                        1. 1st Class
                        2. 2nd Class
                        3. 3rd Class
                        4. General
    Enter Your Choice >>> 1
Ticket Purchased!
```

```
============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout
        Enter a choice >>> 3

_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
            TICKET MENU
        1. View All Tickets
        2. Get Details Of A Specific Ticket
        3. Cancel Ticket
        4. Go Back

        Enter A Choice >>> 1

----------------------------------------------------------------------
Ticket:(3) | Train:  Chennai Express (ID: 1524) From Chennai To Bangalore
----------------------------------------------------------------------

_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
            TICKET MENU
        1. View All Tickets
        2. Get Details Of A Specific Ticket
        3. Cancel Ticket
        4. Go Back

        Enter A Choice >>> 2
Enter The ID Of The Ticket(0 To Go Back): 3

----------------------------------------------------------------------
                        Chennai Express
Ticket ID: 3                                         Ticket Price: 450 Rupees
                  Name: Abc Xyz
                  Train Starting Station: Chennai
                  Train Destination: Bangalore
                  From: Chennai
                  To: Bangalore
----------------------------------------------------------------------
```

```
_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
            TICKET MENU
        1. View All Tickets
        2. Get Details Of A Specific Ticket
        3. Cancel Ticket
        4. Go Back

        Enter A Choice >>> 3
Enter ID Of The Ticket You Want To Cancel (0 To Go Back): 3
Are You Sure You Want To Cancel You Ticket On
                Chennai Express From Chennai To Bangalore
Enter Your Choice (Y/N): y
Your Ticket Has Been Cancelled!

You Have Been Refunded Rupees: 450!
Enter ID Of The Ticket You Want To Cancel (0 To Go Back): 0

_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
            TICKET MENU
        1. View All Tickets
        2. Get Details Of A Specific Ticket
        3. Cancel Ticket
        4. Go Back

        Enter A Choice >>> 4
============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout
        Enter a choice >>> 4

            Name: Abc Xyz
            Username: abc
            Email: abc@xyz.com
            Phone No: +91 1234567890
            Address: Anakkal
            Balance: 10000
```

```
============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout
        Enter a choice >>> 5

Enter New Details
Enter A Email: xyz@abc.com
Enter Name: Xyz Abc
Enter Phone No: +91 9876543210
Enter Address: Kanjirapally
Details Edited!
============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout
        Enter a choice >>> 6

How Much Money Do You Want To Add: 1500
Money Deposited!


============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout
        Enter a choice >>> 7

Enter New Password: 112233
Password Changed!
```

```
============ MENU ============
        1. View Trains
        2. Buy Ticket
        3. View Tickets
        4. View User Details
        5. Edit User Details
        6. Deposit Money
        7. Change Password
        8. Logout
        Enter a choice >>> 8

You have been logged out!



===============  Menu ===============
            1. Login
            2. Signup
            3. Exit

            Enter your choice >>> 3
```

5. SYSTEM TESTING

Software Testing is an empirical investigation conducted to provide stakeholders with information about the quality of the product, with respect to the context in which it is intended to operate. Software Testing also provides an objective, independent view of the software to allow the management to appreciate and understand the risks during the implementation of the software.

The aim of the system testing process was to determine all defects in our project. The program was subjected to a series of trial operations with test inputs and various observations were made and based on these observations, changes were made and again tested for better results. Our Project went through two levels of testing

1. Unit testing
2. Integration testing

5.1 UNIT TESTING

Unit testing was undertaken when a module has been created and successfully reviewed. In order to test a single module, we need to provide a complete working environment.

5.2 INTEGRATION TESTING

After integrating the entire modules developed, we performed various checks by providing different set of test input. The primary objective is to test all the modules in order to ensure that no errors are occurring when one module invokes the other module.

6. CONCLUSION

In conclusion, the Railway Management System is a comprehensive software solution designed to streamline and optimize various aspects of railway operations. It plays a pivotal role in enhancing the overall efficiency, safety, and passenger satisfaction of modern railways.

It effectively manages train schedules, arrivals, and departures in real-time, automating coordination with stations and optimizing resource utilization to minimize delays. It facilitates smooth passenger movement and enhances overall satisfaction. It is also found to be efficient and user-friendly, eliminating the need for specialized programming knowledge. It automates data storage and retrieval procedures, reducing user burden.

Overall, it is a powerful tool that can transform the way railways operate, leading to improved efficiency, enhanced safety, and a more enjoyable travel experience for passengers.

.

7. REFERENCES

1) Computer Science with Pythonby Sumita Arora

2) Python The Complete Referenceby Martin C Brown

3) Programming & Problem Solving Through Pythonby Sathish Jain & Shashi Singh

4) Python for Beginners by Prof. Rahul E. Borate

5) Computer Science with Python Language MadeSimple
by Sathish Jain & Shashi Singh