**Pakistan Railways Management System**



**Session 2023 - 2027**

**Submitted by:**

Abdul Rehman 2023-CS-73

**Supervised by:**

Dr. Muhammad Awais Hassan

**Course:**

CSC-102 Programming Fundamentals

Department of Computer Science

**University of Engineering and Technology, Lahore Pakistan**

**Table of contents**

**Contents**

[**1.** **Project description** 3](#_Toc183134137)

[**2.** **Users of Application** 3](#_Toc183134138)

[**3.** **Functional Requirements** 3](#_Toc183134139)

[3.1. Admin functional requirements 3](#_Toc183134140)

[3.2. Employees functional requirements 4](#_Toc183134141)

[3.3. Passengers’ functional requirements 5](#_Toc183134142)

[**4.** **Wireframes** 5](#_Toc183134143)

[**5.** **Data Structures (Parallel Arrays)** 9](#_Toc183134144)

[5.1. Passengers Data 9](#_Toc183134155)

[5.2. Passenger Ticket Data 10](#_Toc183134156)

[5.3. Employee Tick Data 10](#_Toc183134157)

[5.4. Train Data 10](#_Toc183134158)

[5.5. Variables used for all users 11](#_Toc183134159)

[**6.** **Function Prototypes** 11](#_Toc183134160)

[**7.** **Functions Working Flow** 14](#_Toc183134161)

[**8.** **Weakness in the Business Application** 15](#_Toc183134162)

[**9.** **Future Directions** 15](#_Toc183134163)

# **Project description**

* The main purpose of this project is to develop a railways management system which can be used to manage the train routes and ticketing system. This system will allow the administrators to add or remove data of trains and they can easily calculate and gather data related to the trains and booked tickets.
* In the field of Computer Science, this application will showcase a handy use of file handling for building console-based applications in C++ for solving real-world problems and their professional use.

# **Users of Application**

There are three types of users in the application based upon their role. The users include:

* + **Admin:** Admin can specifically manage employee’s data and can see the stats of the trains (No. of booked tickets, their revenue, total revenue). He has the access to all the data and functionalities in the application.
  + **Employees:** Employees can access passenger’s data and can make changes to it. They can add or remove passenger and can also add or delete train data.
  + **Passenger:** Passenger can book or cancel a ticket. He can view the timetable of the trains and can also view the details or the ticket booked.

# **Functional Requirements**

Functional requirements for each user are:

## Admin functional requirements

Functional requirements of the user as an admin are:

|  |  |  |
| --- | --- | --- |
| Admin | Manage  Employees  Data | Add an employee’s data in the application so that he can login to application and can work for admin. |
| Delete an employee’s data if he wants to remove any employee, can delete it. |
| Update an employee’s data. In case he wants to change any credential of the employee can do it easily. But userID cannot be change. |
| View all employee’s data so that employee’s data is displayed in tabular form |
| Search an employee’s data. If data exists, he can view it. |
| Manage  Passengers  Data | Add a passenger’s data in the application so that he can login and access the services of railways. |
| Delete a passenger’s data if he wants to remove any passenger, can delete it. |
| Update a passenger’s data. In case he wants to change any credential of the passenger can do it easily. But userID cannot be change. |
| View all passenger’s data so that passenger’s data is displayed in a tabular form. |
| Search a passenger’s data. If data exists, he can view it. |
| Manage  Train  Routes | Add a train route so that user can book ticket and can avail the services of railways. |
| Delete a train route. In case there occurs, some unexpected incident admin has the authority to delete a train’s route |
| Can view all the train routes available in a tabular form so that can easily manage further trains. |
| Manage  Ticketing  System | Book a ticket for a passenger if ticket is not already booked. |
| Cancel an already booked ticket of a passenger. |
| View all the booked tickets in a table form. |
| Tickets  Details | Views total number of tickets booked of each train and also revenue collected from each train and total revenue and booked tickets. |

## Employees functional requirements

Functional requirements of the user as an employee are:

|  |  |  |
| --- | --- | --- |
| Employee | Manage  Passengers  Data | Add a passenger’s data in the application so that he can login and access the services of railways. |
| Delete a passenger’s data if he wants to remove any passenger, can delete it. |
| Update a passenger’s data. In case he wants to change any credential of the passenger can do it easily. But userID cannot be change. |
| View all passenger’s data so that passenger’s data is displayed in a tabular form. |
| Search a passenger’s data. If data exists, he can view it. |
| Manage  Train  Routes | Add a train route so that passenger can book ticket and can avail the services of railways. |
| Delete a train route. In case there occurs, some unexpected incident admin has the authority to delete a train’s route |
| Can view all the train routes available in a tabular form so that can easily manage further trains. |
| Manage  Ticketing  System | Book a ticket for a passenger if ticket is not already booked. |
| Cancel an already booked ticket of a passenger. |
| View all the booked tickets in a table form. |
| Change Password | An employee can change its password by entering current password then is allowed to enter new password. |
| View  Data | An employee can view its own data. |

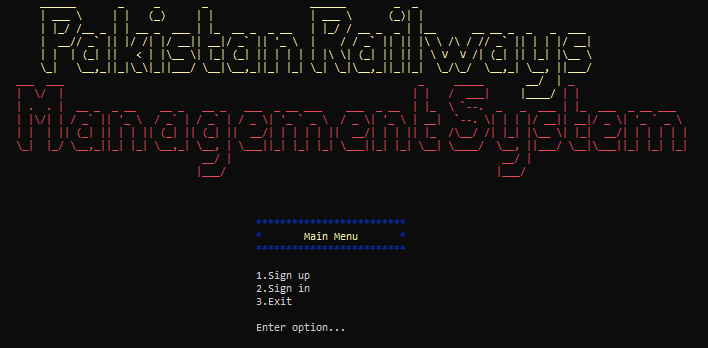
## Passengers’ functional requirements

Functional requirements of the user as a passenger are:

|  |  |  |
| --- | --- | --- |
| Passenger | Book  Ticket | Can book a ticket in train available. If ticket already booked, he cannot book a ticket again! |
| Cancel  Ticket | A passenger can change its password by entering current password then is allowed to enter new password. |
| View  Trains  Available | A passenger can view the data of all the trains available. |
| View  Data | A passenger can view its own data. |
| View  Ticket  Details | A passenger can view the details of the ticket booked. |
| Change Password | A passenger can change its password by entering current password then is allowed to enter new password. |

# **Wireframes**

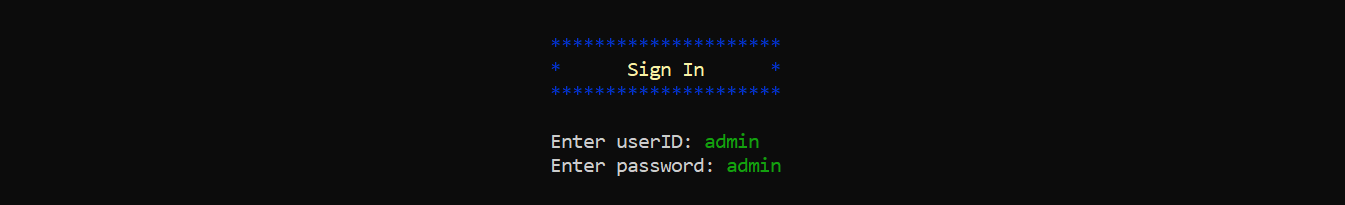
Wireframes of the railways management system are:

****

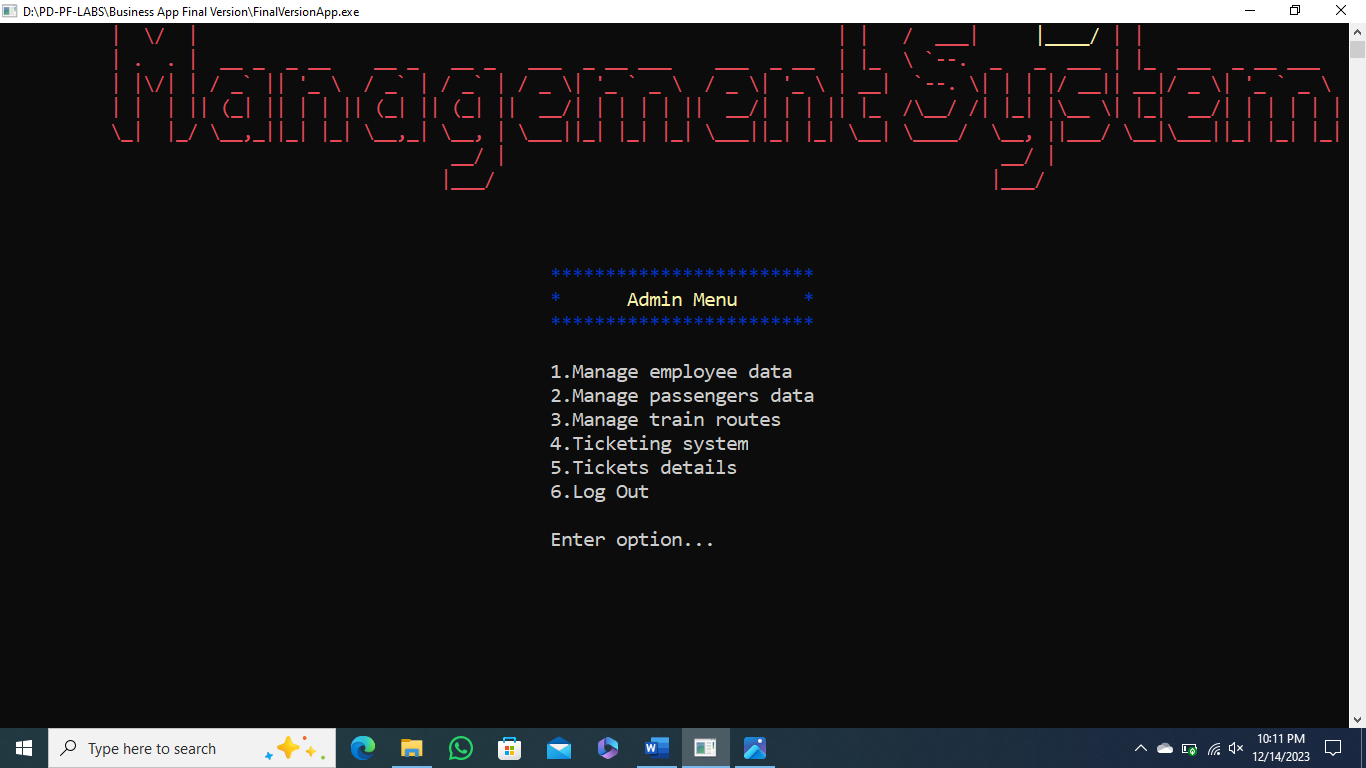
**Figure 1: Start Menu**

****

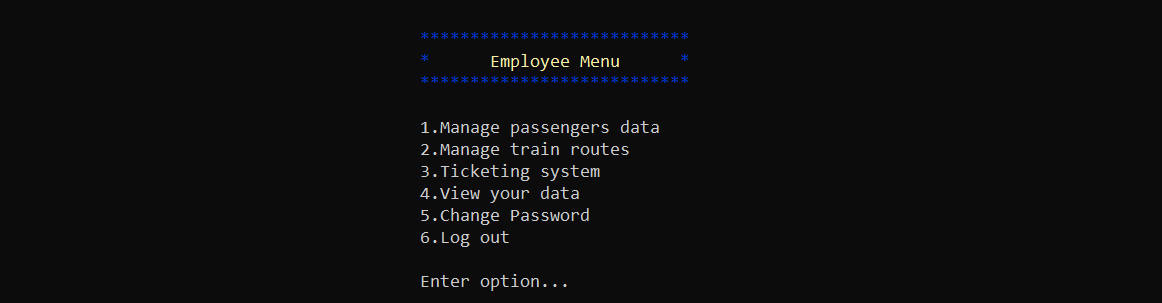
**Figure 2: User SignUp**

****

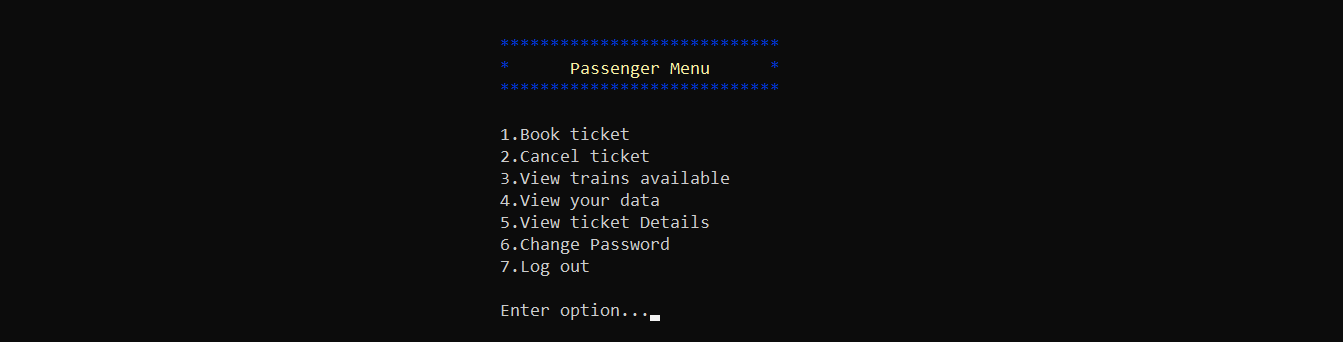
**Figure 3: User SignIn**

****

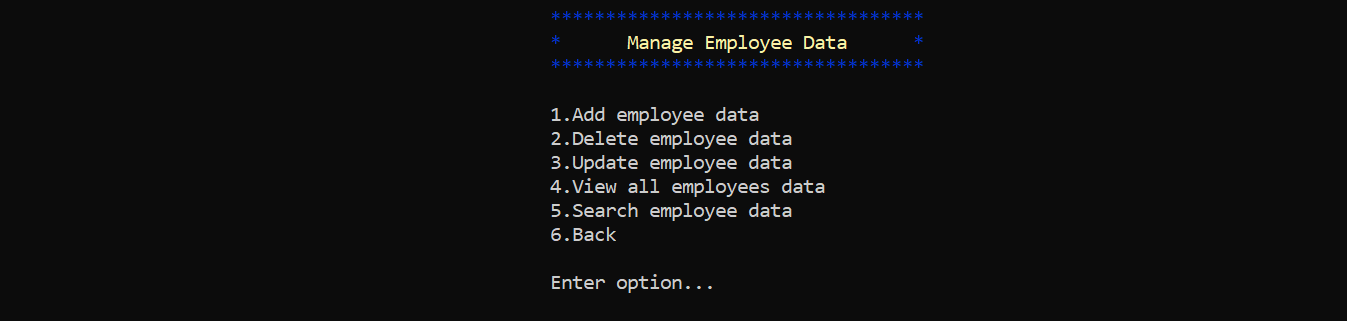
**Figure 4: Admin Menu**

****

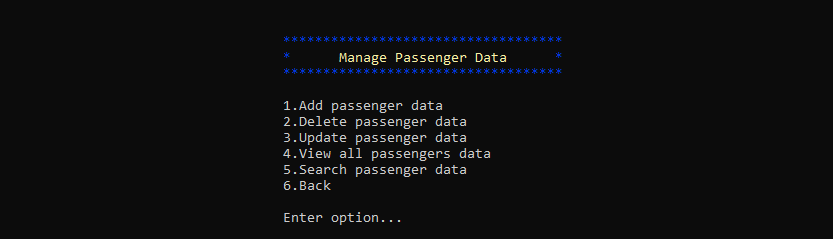
**Figure 5: Employee Menu**

****

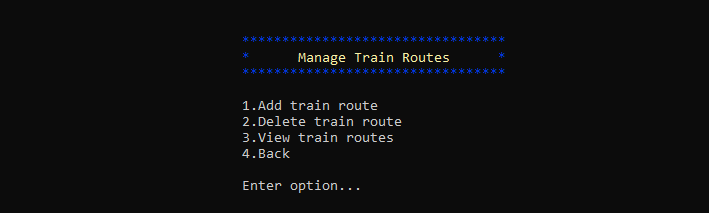
**Figure 6: Passenger Menu**

****

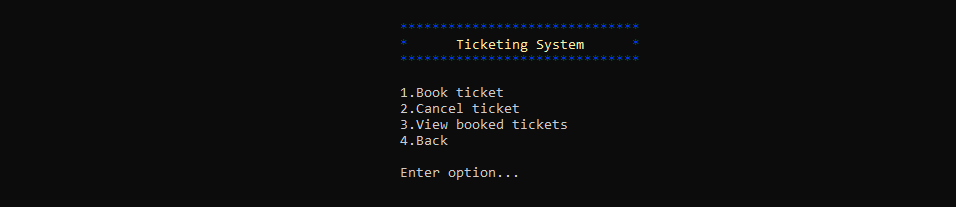
**Figure 7: Manage Employee Menu**

****

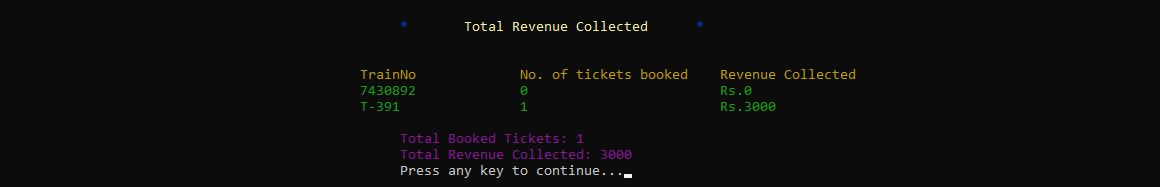
**Figure 8: Manage Passenger Menu**

****

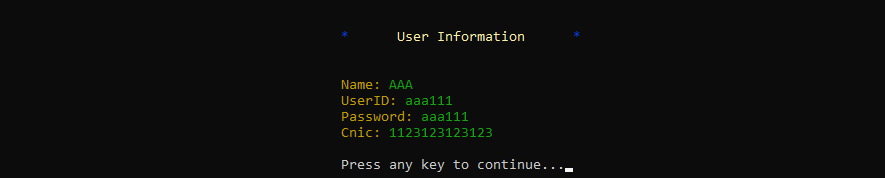
**Figure 9: Manage Train Routes Menu**

****

**Figure 10: Manage Ticketing System Menu**

****

**Figure 11: Total Revenue**

****

**Figure 12: View User Information**

# **Data Structures (Parallel Arrays)**

The parallel arrays along with counter variables are:



## Passengers Data

string passengerName[1000];

string passengerID[1000];

string passengerIDPassword[1000];

string passengerCnic[1000];

int passengerCountIdx = 0;

string passengerNameSU;

bool passengerNameCheck;

string userIDSU;

bool userIDCheckSU;

string userPasswordSU;

bool validation;

string userCnicSU;

bool cnicCheck;

int pasgIdx;

string pasgPasswordSI;

## Passenger Ticket Data

string passengerTicketStatus[1000];

string passengerTrainNo[1000];

string passengerTicketRoute[1000];

string passengerArrivalCity[1000];

string passengerDepartureCity[1000];

int passengerTicketPrice[1000];

string routeNo;

int indexTrain;

## Employee Tick Data

string employeeName[100];

string employeeID[100];

string employeeIDPassword[100];

string employeeCnic[100];

int employeeCountIdx = 0;

string empNamein;

bool empNameCheck;

string empIdin;

bool empIDCheck;

string empPasswordin;

bool passValidation;

string empCnicin;

bool cnicCheck;

int empIdx;

string empPasswordSI;

## Train Data

string trainNo[100];

string trainArrivalCity[100];

string trainDepartureCity[100];

string trainRoute[100];

int trainTicketPrice[100];

int trainCountIdx = 0;

string trainId;

string departureCity;

string arrivalCity;

string price;

string trainCode;

int totalRevenue = 0;

int totalTicketsSold = 0;

int numOfBookedTickets[trainCountIdx];

int revenueOfEachTrain[trainCountIdx];

## Variables used for all users

string ID;

string newPassword;

string userIDSI;

string role;

# **Function Prototypes**

All the functions used in the application are:

void header();

void startHeader();

void startMenuHeader();

void signUpHeader();

void signInHeader();

void adminHeader();

void employeeHeader();

void passengerHeader();

void manageEmployeeHeader();

void managePassengerHeader();

void manageTrainsHeader();

void manageTicketetingHeader();

void printSubHeader(string);

void noteUserName();

void noteuserIDpassenger();

void noteSUpassword();

void noteSUcnic();

void noteAddEmployee();

void noteAddTrain();

void noteDepartureCity(string[]);

void noteArrivalCity(string, string[]);

void noteRoutesavail(string[], string[], int);

void eraseInstruction();

string printMenu(string[], int);

string userNameSignUp(int);

bool userNameValidationCheck(string);

string userIDSignup(string[], int);

bool userIDCheckSignup(string, string[], int);

string userPasswordSignup(int);

bool passwordValidationCheckSignup(string);

string userCnicSignup(string[], string[], int, int, int);

bool userCnicValidationSignup(string, string[], string[], int, int);

void saveSUInformation(string, string[], string, string[], string, string[], string, string[], int &);

string userIDSignIn(string[], string[], int, int);

bool userCheckSignIn(string, string[], string[], int, int);

string adminPasswordCheck();

bool userCheck(string, string[], int);

string roleCheck(string);

int indexCheck(string, string[], int);

string userPasswordSignIn(string);

void addEmployeeData(string[], string[], string[], string[], string[], int &, int);

string empUserIDInput(string[], int);

bool employeeIDCheck(string, string[], int);

void addPassengerData(string[], string[], string[], string[], string[], string[], int &, int);

void deleteUserData(string[], string[], string[], string[], string, int &);

void deleteData(string[], string[], string[], string[], int &, int);

string YesNoChoice(string);

void deleteTicketData(string[], string[], string[], string[], string[], int[], int, int);

void updateUserData(string[], string[], string[], string[], string[], string, int, int);

string updateData(string[], string[], string[], string[], int, int, int, string);

string updateDataChoice();

string changePassword(string[], int);

void viewUserDataList(string[], string[], string[], string[], string, int);

void viewUserData(string[], string[], string[], string[], int, int, int);

void searchUserData(string[], string[], string[], string[], string, int);

void ticketStatusPassenger(string[], string[], string[], string[], string[], int[], int);

void addTrainRoute(string[], string[], string[], string[], int[], int &, string[]);

string trainNoInput(string[], int);

bool trainNoValidation(string, string[], int);

bool trainCheck(string, string[], int);

string trainArrivalCityInput(string, string[]);

string trainDepartureCityInput(string[]);

bool cityNameValidation(string, string[]);

int trainTicketPriceIn();

bool ticketPriceValidation(string);

int stringToIntConversion(string);

string deleteTrainRoute(string[], string[], string[], string[], int[], int &);

void deleteTrainTicketDetails(string[], string[], string[], string[], string[], int[], int,string);

void deleteData(string[], string[], string[], string[], int[], int, int &);

void viewTrainsAvailable(string[], string[], string[], string[], int[], int);

void bookTickets(string[], string[], string[], string[], string[], string[], string[], string[], string[], int[], int[], int, int);

void saveTicketData(string[], string[], string[], string[], string[], string[], string[], string[], string[], int[], int[], int, int);

void cancelTicket(string[], string[], string[], string[], string[], int[], int);

void viewTicketDetails(string[], string[], string[], string[], int[], int);

void viewBookedTickets(string[],string[],string[],string[],string[],string[],string[],int[], int);

void ticketsDetails(string[], string[], int[], int[], int, int);

void employeesNewDataFile(string[], string[], string[], string[], int);

void employeesDataUpdateFile(string[], string[], string[], string[], int);

void employeeDataLoad(string[], string[], string[], string[], int &);

void passengersNewDataFile(string[], string[], string[], string[], string[], string[], string[], string[], string[], int[], int);

void passengersDataUpdateFile(string[], string[], string[], string[], string[], string[], string[], string[], string[], int[], int);

void passengersDataLoad(string[], string[], string[], string[], string[], string[], string[], string[], string[], int[], int &);

void trainsNewDataFile(string[], string[], string[], string[], int[], int);

void trainsDataUpdateFile(string[], string[], string[], string[], int[], int);

void trainsDataLoad(string[], string[], string[], string[], int[], int &);

string loadUserAttribute(string, int &);

string userIDInput(int i, int y);

string inputs();

void pressAnyKey(int x, int y);

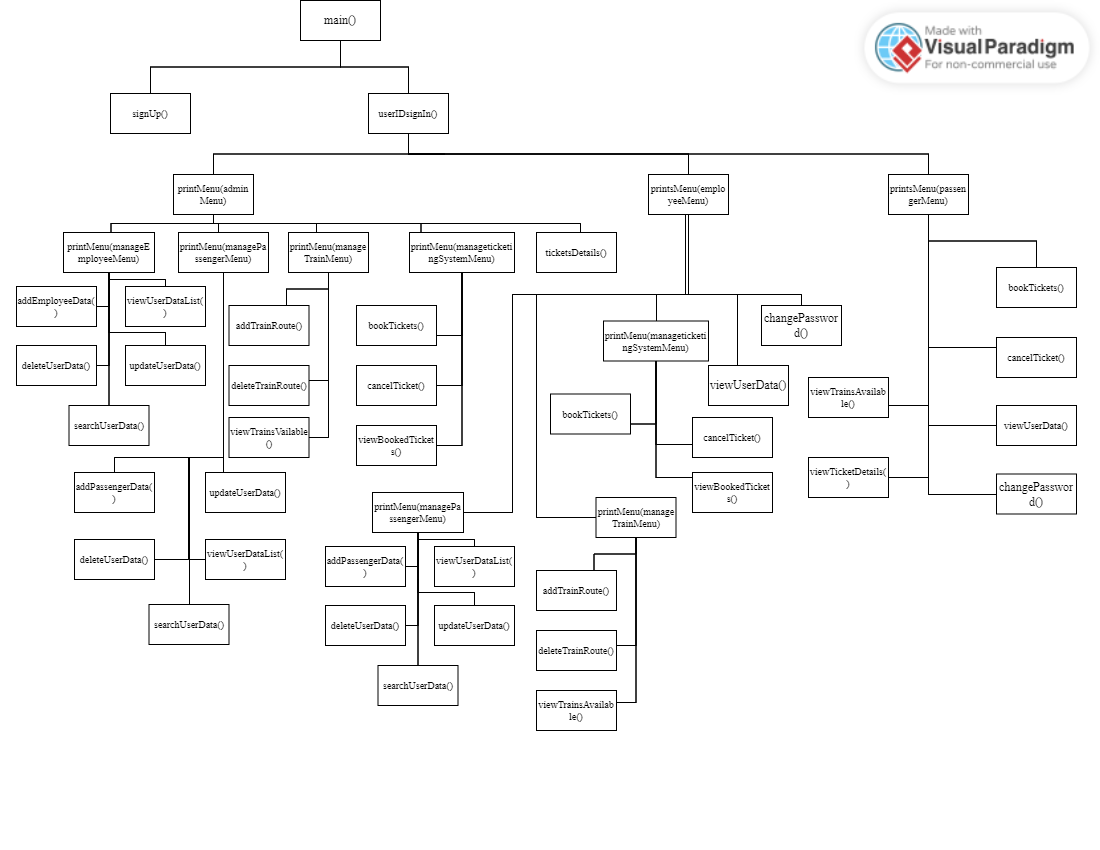
void space(int x, int y);

void headerCls();

void printStatement(string, int, int, string);

# **Functions Working Flow**

The flow of functions among the app is controlled as following:



# **Weakness in the Business Application**

The weaknesses in the business application are:

* A passenger can book a ticket for himself only.
* A user cannot book more than 1 tickets.
* Train data cannot be updated but deleted.
* Tickets details cannot be updated but deleted.
* There is no defined ticket type for train.
* There is no seats limit for the trains.
* A user cannot contact admin or employee to submit complaint.
* If password is forgotten you cannot recover it.

# **Future Directions**

I want to extend this application and add some more features i.e., booking tickets for other users, password recovery, limiting trains seats, message box etc. so that it can be more efficient in its working.