

**Project Report**

Submitted By:

Mohammad Hasan Hael Al-Salehi (SP22-BSE-053)

- Haris Ali(SP22-BSE-016)

Mian Zaid Zujaj (SP22 - BSE - 026)

**Submitted to: Mr.**

Rizwan Rashid

**Date of Submission: 10 / 01 / 2023**

Dated: 08 January 2023

**In Driver System**

**Introduction:**

In Driver system consists of Three main functionalities further which are given below.

1. Login Page
2. Admin Page
3. Customer Page
4. Menu-Based Program

In Driver system allows you to register into the system by using signup option from login page where you are required to input your username and strong password. After successfully registering into your in-Driver account then you can also be able to travel, add amount and exit from the account. Also, from login page when you are again wanting to sign in into your account so there you can also be able to change your password as well. Now if a customer successfully registered into the account so, now they can be able to add amount in their account for using the system with the help of visa card so for this purpose first visa card number will be verified now if the amount is added to the account so then he/she can be able to travel. After they pressed travel menu based system will be shown to them where they can chose their ride type, current location, desired location, and then another menu will be shown in which they can be able to choose the type of drivers according to their prices and so after they chose they can be able to travel and the amount of ride will be deducted from their account. In this program two text files will be made at backend and the data will store there for example username, password, and balance etc. Also, arrays are used in this program to save data of the locations. In the end if the user press exit the program will terminate. Also, the admin can be able to check the records and delete the data of the customer from the system. For admin different command is made.

**Task Distribution:**

* Mian Zaid Zujaj (SP22 - BSE - 026):

1. Checkers (Password Strength, phone number,showRideHistory,getBalance)

* Haris Ali(SP22-BSE-016):

1. Field validation
2. Testing
3. Exceptions handling

* Mohammad Hasan Hael Al-Salehi (SP22-BSE-053):

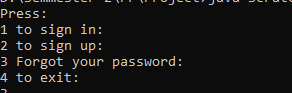
1. File handing
2. Inner Logic

**Features of project:**

* Privacy protected project that you can only use after correcting credentials (i.e., password and id’s)
* Use of Modules to make the main program shorter and easier to understand
* Data stored in files to store permanently unlike the arrays.
* Properly indented code
* Easy logic designing to make code much easier for any new reader or coder.
* Uses of methods for the implementation of different tasks.

1. **Login Page:**

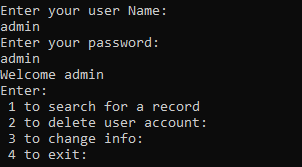
First, when the program executed the menu will show like this.



Here a user can be able to sign in if he/she is already in the system otherwise he/she can just signup and create their account by putting their username and their password which must be secured, their age and phone number. After successfully logged in the system the user can be also able to change their old password using the third option which is forgot password if they forgot their password, they can access their account with the help of their username and phone number if they have forgot their password.

1. **Admin Page:**

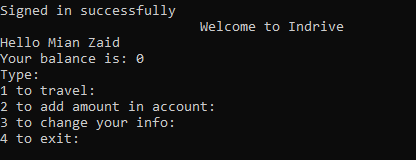
The menu of the admin page will show like this



Admin page can be accessed from login page and will like this if the admin entered their user name and password as admin so admin page can be accessed. Now the functionality of admin page is that here the admin can be able to check about all the rides which has been done and the full information of that ride. Also from this page the admin can be able to delete the record of an customer. Admin page can be only accessed by admin because it has unique name and password. Now if user want to come out from this program so he/she can be able to exit by just typing 3 for exit in the main menu.

1. **Customer Page:**

This is the page where customer can access the functionalities of the system after successfully registering in the program. Also, from this page user shall be able to change his information like password phone number etc. This page will look like

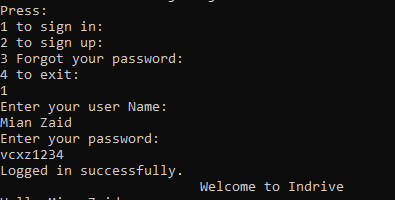


Here this is menu based program so menu is shown in which the user can access these three functionalities which is first to travel if the user press travel so he/she can be able to choose the type of ride from another menu shown after that in which if he/she select any ride type so the program will show another menu in which they can be able to select their current location and the desired location and then if both are different then the user will be shown another menu in which they can be able to select driver according to the price and after they select the driver they can be shown the message to enjoy their ride and after that the program will again ask the user if he/she want to travel or add amount in the account or exit so if he/she press exit they will come out of the program. Now if the user entered the second functionality which is add amount in the account so from here, they can be able to add the amount in their account by using their valid visa card number so here in this functionality it will first validate the visa card number, then the user can be able to add amount which he/she want. After adding the amount, the user balance will update and the same menu will be shown according to which user can choose travel, add amount, or exit so if he/she want another ride they can chose travel otherwise they can choose exit which will terminate the program.

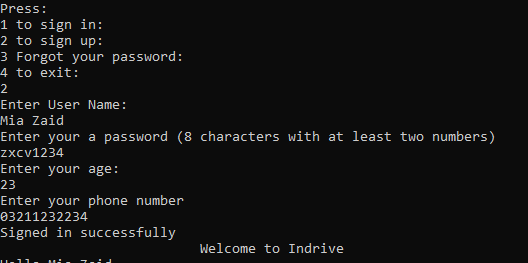
**Outputs Screenshots:**

Now here we can add all the outputs screenshots as was required for the report.

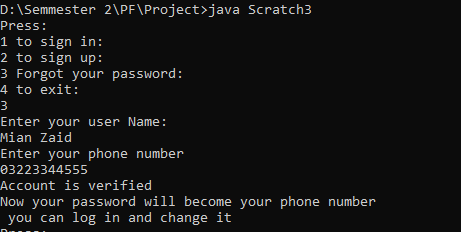
**1 Sign-in Page:**

****

**2 Sign Up page:**

****

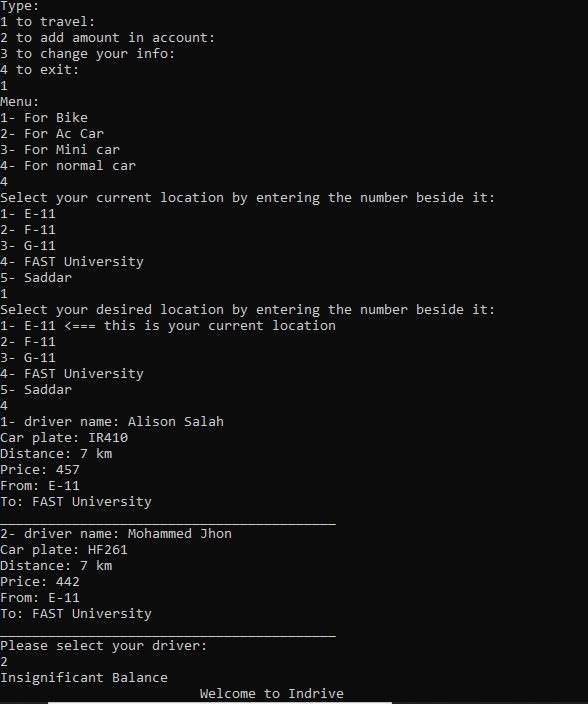
**3 Forgot Password:**

****

**4 After logged in the account the user will be shown 2nd page**

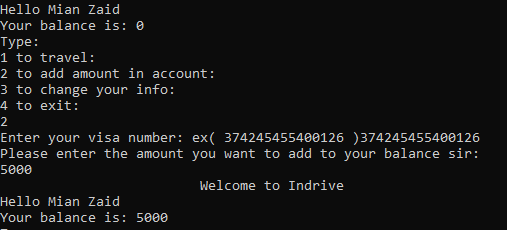
**If user pressed for travel:**

If 1 is pressed the menu will be shown to user in which he/she can be able to select ride type and locations and then they can be able to travel if there is amount in their account otherwise the message will shown to them which is insignificant balance.

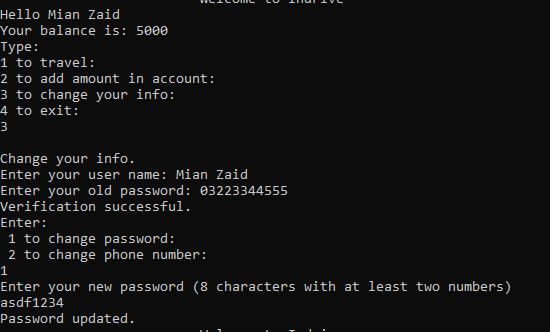
****

**5 Add amount:**

If user pressed add amount so here he/she can be able to add amount in their account by using their valid visa card number.

****

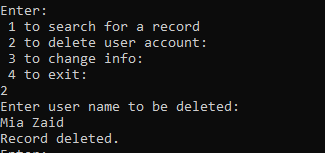
**6 Change Info:**

****

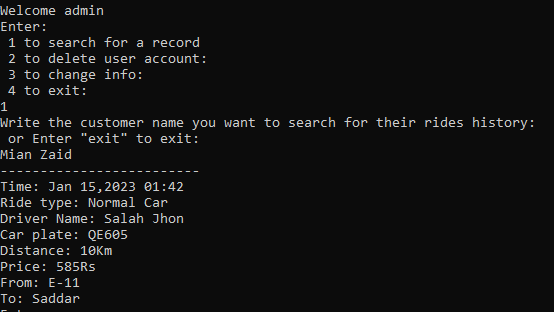
**7 Admin Page:**

Admin can be able to search for record and delete the record.

**Delete Record:**

****

**Search Record:**

****

**Modules Used in the Program:**

**1 Main Method:**

First of all, when the program executes the main method will execute at first In main method first of all it will go to first page if first page is validated then it will go to second page which is for travel, add amount. Also, in main method one other method is called which is delete info like if the data has been updated so it will delete the old previous file and the new temp file will be replaced with it.

public static void main(String[] args) {  
 if(firstPage())  
 SecondPage();// second page containing traveling and adding amount and edit menu  
 if(deleteInfo){  
 // delete the previous file and replace it with the temp that has been edited  
 deleteFile1(accounts,tempFile);  
 deleteInfo = false;  
 }

**2 First Page:**

This module is used for login page which is sign-in, sign-up and change password portion as well so in this module Admin can also access this page to login due to that adminPage() is also called in this module also the sign in method is also called in this module. This module is used to validate user name and password. Also if the user is already in the system so he can also change his password with the help of this module.

public static boolean firstPage () {  
 String userPassword;  
 boolean admin = false;  
 boolean passed = false;  
 do {  
 do{  
 try {  
 System.out.println("Press: \n1 to sign in: \n2 to sign up: ");  
 userInputInt = Integer.parseInt(inputString());  
 if(userInputInt > 2 || userInputInt <= 0)  
 System.out.println("Wrong input.");  
 else  
 break;  
 } catch (Exception e) {  
 System.out.println("Error is "+e.getMessage());  
 }  
 }while (true);  
 switch (userInputInt) {  
 // sign in portion  
 case 1->{  
 System.out.println("Enter your user Name:");  
 userInputString = inputString();  
 System.out.println("Enter your password:");  
 userPassword = inputString();  
 if (userInputString.equals("admin") && userPassword.equals("admin")){ //admin login check  
 admin = true;  
 adminPage();// method called  
 } else if(!signIn(userInputString, userPassword)||admin) // method called  
 System.out.println("Wrong username or password.");  
 else{  
 System.out.println("Logged in successfully.");  
 passed = true;  
 }  
 }  
 // sign up portion  
 case 2->{  
 try {  
 System.out.println("Enter User Name:");  
 userInputString = inputString();  
 System.out.println("Enter your a password (8 characters with at least two numbers)");  
 userPassword = inputString();  
 // method called  
 if (signUp(userInputString,userPassword)){  
 passed = true;  
 }  
 }  
 catch (Exception e){  
 System.out.println(e.getMessage());  
 }  
 }  
 // change password portion  
 default -> {  
 return false;  
 }  
 }  
 }while (!passed);  
 return passed;  
}

**3 Second Page:**

This module is used after the user is being signed-in the system so this module will show four options in which user can be able to travel, add amount in account, change info, and exit. So, for travelling user have to add current location and desired location. Also in this module we used arrays to store different locations. VisaNumber validation will also done in this module. RandomDriverGenerator module is called here also checkVisa is called here. Because if the user want to add amount to his account so he can add it through visa card validation. Also if the user wants to change their info from here so it can also be done and the previous old file will be deleted and the new temp file will be converted to new file. So for this deleteFile method is also called in this module.

public static void SecondPage(){  
 // end the loop and exit the program  
 boolean exit = false;  
 while (!exit){  
 setBalance(userNameFinal);//change the balance in each iteration  
 //Names of locations  
 String [] locations = {"E-11","F-11","G-11","FAST University","Saddar"};  
 System.out.println("\t\t\t Welcome to Indrive");  
 System.out.println("Hello "+ userNameFinal);  
 System.out.println("Your balance is: "+userBalance);  
 boolean secondPageTrue = true;  
 do{  
 try {  
 System.out.println("Type: \n1 to travel: \n2 to add amount in account: \n3 to change your info:\n4 to exit: ");  
 userInputInt = Integer.parseInt(inputString());  
 if(userInputInt > 4 || userInputInt <= 0)  
 System.out.println("Wrong input.");  
 else  
 secondPageTrue = false;  
 } catch (Exception e) {  
 System.out.println("Error is "+e.getMessage());  
 }  
 }while (secondPageTrue);  
 switch (userInputInt) {  
 case 1 -> {  
 do {  
 try {  
 System.out.println("Menu: \n1- For Bike \n2- For Ac Car \n3- For Mini car \n4- For normal car");  
 userInputInt = Integer.parseInt(inputString());  
 if (userInputInt > 4 || userInputInt <= 0)  
 System.out.println("Wrong input.");  
 else  
 break;  
 } catch (Exception e) {  
 System.out.println("Error is " + e.getMessage());  
 }  
 } while (true);  
 switch (userInputInt) {  
 case 1 -> {  
 rideType = 0.5;  
 rideTypename = "bike";  
 }  
 case 2 -> {  
 rideType = 2;  
 rideTypename = "Ac car";  
 }  
 case 3 -> {  
 rideType = .8;  
 rideTypename = "Mini";  
 }  
 case 4 -> {  
 rideType = 1;  
 rideTypename = "Normal Car";  
 }  
 default -> {  
 rideType = 1;  
 rideTypename = "normal car";  
 }  
 }  
 // current place of customer  
 int currentLocation = 0;  
 do {  
 try {  
 System.out.println("Select your current location by entering the number beside it: ");  
 for (int i = 0; i < locations.length; i++) {  
 System.out.println((i + 1) + "- " + locations[i]);  
 }  
 currentLocation = Integer.parseInt(inputString()) - 1;  
 if (currentLocation + 1 > 5 || currentLocation + 1 <= 0)  
 System.out.println("Wrong input.");  
 else  
 break;  
 } catch (Exception e) {  
 System.out.println(e.getMessage());  
 }  
 } while (true);  
 // destination of customer  
 int desiredLocation = 0;  
 do {  
 try {  
 System.out.println("Select your desired location by entering the number beside it: ");  
 for (int i = 0; i < locations.length; i++) {  
 if (i == currentLocation) {  
 System.out.println((i + 1) + "- " + locations[i] + " <=== this is your current location");  
 } else {  
 System.out.println((i + 1) + "- " + locations[i]);  
 }  
 }  
 desiredLocation = Integer.parseInt(inputString()) - 1;  
 if (desiredLocation + 1 > 5 || desiredLocation + 1 <= 0)  
 System.out.println("Wrong input.");  
 else if (desiredLocation == currentLocation)  
 System.out.println("You cannot travel to the same place.");  
 else  
 break;  
 } catch (Exception e) {  
 System.out.println(e.getMessage());  
 }  
 } while (true);  
 randomDriverGenerator(currentLocation, desiredLocation, locations); // method called  
 }  
 case 2 -> {  
 long visaNumber = 0;  
 boolean validVisa = false;  
 do {  
 try {  
 System.out.print("Enter your visa number: ex( 374245455400126 )");  
 visaNumber = Long.parseLong(inputString());  
 if (checkVisa(visaNumber)) // checks if visa card is valid or not  
 validVisa = true;  
 else  
 System.out.println("Invalid card.");  
 } catch (Exception e) {  
 System.out.println("Wrong input.");  
 }  
 } while (!validVisa);  
 if(checkVisa(visaNumber)) {  
 String userName = userNameFinal;  
 int balance1 = 0;  
 boolean amountCheck = false;  
 do {  
 try {  
 System.out.println("Please enter the amount you want to add to your balance sir: ");  
 balance1 = Integer.parseInt(inputString());  
 amountCheck = true;  
 } catch (Exception e) {  
 System.out.println("Wrong input.");  
 }  
 } while (!amountCheck);  
 String balance2 = Integer.toString(balance1);  
 editBalance(userName, balance2, balance, tempFile2); // method called  
 } else {  
 System.out.println("Your visa is invalid.");  
 }  
 }  
 case 3 -> {  
 System.out.println("\nChange your info.");  
 System.out.print("Enter your user name: ");  
 userInputString = inputString();  
 System.out.print("Enter your old password: ");  
 String password = inputString();  
 if(checkForPassChange(userInputString,password,accounts,accounts,tempFile)){  
 System.out.print("");  
 }  
 else  
 System.out.println("Wrong name or password.");  
 }  
 default -> exit = true;  
 }  
 if(deleteBalance){  
 //delete the previous file and replace it with the temp that has been edited  
 deleteFile1(balance,tempFile2);  
 deleteBalance =false;  
 }  
 }  
}

**4 Admin Page:**

This module is made for admin and in this module admin can view and delete the records of the customer. In this module showRideHistory module is called. In this module we used switch methods with two cases first if admin is searching the customer and second if the admin is deleting the customer record so for this purpose deleteRecord() method is also called in this module. CheckIfRecordExist method is also called in this module so it can check if the record of user is present or not.

public static void adminPage(){  
 do {  
 do {  
 try {  
 System.out.println("Enter: \n 1 to search for a record \n 2 to delete user account: \n 3 to exit:");  
 userInputInt = Integer.parseInt(inputString());  
 } catch (Exception e) {  
 System.out.println("Invalid input.");  
 userInputInt = 0;  
 }  
 } while (userInputInt < 1 || userInputInt > 3);  
 switch (userInputInt) {  
 case 1 -> {  
 System.out.println("Write the customer name you want to search for their rides history: \n or Enter \"exit\" to exit: ");  
 String toBeSearch = inputString();  
 // method called  
 if (CheckIfRecordExist(toBeSearch, history, "", "history")) {  
 showRidesHistory(toBeSearch); // method called  
 } else if (toBeSearch.equals("exit")) {  
 break;  
 } else {  
 System.out.println("This user name is not found");  
 }  
 }  
 case 2 -> {  
 System.out.println("Enter user name to be deleted: ");  
 String username = inputString();  
 if (CheckIfRecordExist(username, accounts, "", "accounts")) {  
 deleteRecord(username, accounts);  
 System.out.println("Record deleted.");  
 } else {  
 System.out.println("This user is not found");  
 }  
 }  
 default -> System.exit(1);  
 }  
 } while (true);  
}

**5 CheckIfRecordExist:**

This module is used to check the record of the customer(user) if it exists or not while the admin is searching for it or searched anywhere else. In this module we used if else and while statement to check if the user record is in the file or not so we can search it from file. We can search it with the help of username and phone number.

public static boolean CheckIfRecordExist(String userName, File fileToCheck, String phoneNumber, String Searchin) {  
 boolean exist = false;  
 if (Searchin.equals("accounts")){  
 try {  
 try (Scanner reader = new Scanner(fileToCheck)) {  
 String userName1;  
 String phoneNumber1;  
 String password;  
 String dummy;  
 int foundCounter = 0;  
 while (reader.hasNext()) {  
 userName1 = reader.nextLine();  
 password = reader.nextLine();  
 phoneNumber1 = reader.nextLine();  
 dummy = reader.nextLine();  
 if (userName1.equals(userName) || phoneNumber1.equals(phoneNumber)) {  
 exist = true;  
 foundCounter++;  
 }  
 }  
 reader.close();  
 if (exist) {  
 System.out.println(foundCounter + " records were found.");  
 }  
 }  
 } catch (FileNotFoundException e) {  
 System.out.println(e.getMessage());  
 input.nextLine();  
 }  
 return exist;  
 }  
 else if (Searchin.equals("history")){  
 try {  
 Scanner reader = new Scanner(fileToCheck);  
 while (reader.hasNext()){  
 String userNameHere = reader.nextLine();  
 reader.nextLine();  
 reader.nextLine();  
 reader.nextLine();  
 reader.nextLine();  
 reader.nextLine();  
 reader.nextLine();  
 reader.nextLine();  
 reader.nextLine();  
 if(userNameHere.equals(userName)){  
 return exist = true;  
 }  
 }  
 }catch (Exception e){  
 System.out.println(e.getMessage());  
 }  
 }  
 return exist;  
}

**6 ShowRideHistory:**

This module can be used to search the ride history of the customers. In this module we used do-while and if statements to search for the ride history of the user by the help of username.

public static void showRidesHistory (String username){  
 do {  
 try{  
 Scanner reader = new Scanner(history);  
 while (reader.hasNext()){  
 String userNameHere = reader.nextLine();  
 if(userNameHere.equals(username)){  
 System.out.println("-------------------------");  
 System.out.println("Time: "+reader.nextLine());  
 System.out.println("Ride type: "+reader.nextLine());  
 System.out.println("Driver Name: "+reader.nextLine());  
 System.out.println("Car plate: "+reader.nextLine());  
 System.out.println("Distance: "+reader.nextLine()+"Km");  
 System.out.println("Price: " + reader.nextLine()+"Rs");  
 System.out.println("From: "+reader.nextLine());  
 System.out.println("To: "+reader.nextLine());  
 }  
 }  
 reader.close();  
 break;  
 }catch (FileNotFoundException e){  
 System.out.println(e.getMessage());  
 input.nextLine();  
 }  
 } while (true);  
}

**7 deleteRecord:**

This module is used to delete the record of the user. This module will first ask about the user name and then file from which to delete and then it can delete the record of the user from that file so as we have 2 files so first it will deal with accounts file then it will deal with the balance file.

public static void deleteRecord (String userName,File fileToCheck){  
 try {  
 Scanner count = new Scanner(fileToCheck);  
 int counter = -4;  
 while (count.hasNextLine()){  
 count.nextLine();  
 counter++;  
 }  
 count.close();  
 Scanner readerAccount = new Scanner(fileToCheck);  
 Scanner readerBalance = new Scanner(balance);  
 String []arrayAccount = new String[counter];  
 String []arrayBalance = new String[counter / 2];  
 String readbyScanner;  
  
 //deal with deleting record form accounts file  
 for (int i = 0; i < counter ; i++) {  
 readbyScanner = readerAccount.nextLine();  
 if(readbyScanner.equals(userName)){  
 readerAccount.nextLine();  
 readerAccount.nextLine();  
 readerAccount.nextLine();  
 arrayAccount[i] = readerAccount.nextLine();  
 }  
 else {  
 arrayAccount[i] = readbyScanner;  
 }  
 }  
 readerAccount.close();  
  
 PrintWriter writerAccounts = new PrintWriter(fileToCheck);  
 for (String i:arrayAccount) {  
 writerAccounts.println(i);  
 }  
 writerAccounts.close();  
  
 //deal with deleting record from balance file  
 for (int i = 0; i < counter / 2; i++) {  
 readbyScanner = readerBalance.nextLine();  
 if(readbyScanner.equals(userName)){  
 readerBalance.nextLine();  
 arrayBalance[i] = readerBalance.nextLine();  
 }  
 else {  
 arrayBalance[i] = readbyScanner;  
 }  
 }  
 readerBalance.close();  
  
 PrintWriter writerBalance = new PrintWriter(balance);  
 for (String i:arrayBalance) {  
 writerBalance.println(i);  
 }  
 writerBalance.close();  
 } catch (Exception e){  
 System.out.println(e.getMessage());  
 }  
}

**8 editBalance:**

This module is used to edit the balance of the user this module deals with file handling to edit the balance of the user in the files and update them. This module will first check the username if the username validate it will write it on the file then it will check the change in amount and will write it on the new file.

public static void editBalance(String username, String change, File source , File target) {  
 try {  
 Scanner reader = new Scanner(source);  
 PrintWriter writer = new PrintWriter(target);  
 String userName1;  
 String toChange;  
 while (reader.hasNext()){  
 userName1 = reader.nextLine();  
 toChange = reader.nextLine();  
 if (userName1.equals(username)){  
 writer.println(username);  
 long balance = Integer.parseInt(toChange) + Integer.parseInt(change);  
 writer.println(balance);  
 } else {  
 writer.println(userName1);  
 writer.println(toChange);  
 }  
 }  
 reader.close();  
 writer.flush();  
 writer.close();  
 deleteBalance = true;  
 } catch (FileNotFoundException e){  
 System.out.println(e.getMessage());  
 input.nextLine();  
 }  
}

**9 CheckForPassChange:**

This module is used to change the password or phone Number of the user while he will be doing in edit info. In this module first it will validate the new password of user with old password if it exist then record exist message will show other wise if user want to change password or phone number then user can be asked to enter new password if this password is already taken so they can be asked to do it again. Also if user want to change phone number same phenomenon will occur.

public static boolean checkForPassChange(String userName, String password, File fileToCheck, File source , File target) {  
 boolean exist = false;  
 try {  
 Scanner reader = new Scanner(fileToCheck);  
 String userName1;  
 String password1;  
 while (reader.hasNext()) {  
 userName1 = reader.nextLine();  
 password1 = reader.nextLine();  
 if (userName1.equals(userName) && password1.equals(password)) {  
 System.out.println("Record found.");  
 exist = true;  
 do {  
 try {  
 System.out.println("Enter: \n 1 to change password: \n 2 to change phone number: ");  
 userInputInt = Integer.parseInt(inputString());  
 }catch (Exception e){  
 System.out.println("input invalid");  
 userInputInt = 0;  
 }  
 }while (userInputInt > 2 || userInputInt < 1);  
 switch (userInputInt) {  
 case 1 -> {  
 do {  
 System.out.println("Enter your new password (8 characters with at least two numbers)");  
 String change = inputString();  
 if (checkPasswordStrength(change)) {  
 editRecordOfAccount(userName, change, source, target, "password");  
 break;  
 }  
 }  
 while (true);  
 }  
 case 2 -> {  
 do {  
 System.out.println("Enter the new phone number: ");  
 String change = checkPhoneNumber();  
 if (!CheckIfRecordExist("", accounts, change, "accounts")) {  
 editRecordOfAccount(userName, change, source, target, "phone");  
 break;  
 } else {  
 System.out.println("this phone number was used before");  
 }  
 } while (true);  
 }  
 }  
 break;  
 }  
 }

**10 editRecordOfAccount:**

In this module user can be able to edit the record of the Account in which there are two variables password and phone. In this module if-else-if are used to know what to edit and then that data has been edited from that file.

public static void editRecordOfAccount(String username, String change, File source , File target, String whatToEdit) {  
 try {  
 Scanner reader = new Scanner(source);  
 PrintWriter writer = new PrintWriter(target);  
 String userName1;  
 String password1;  
 String phoneNumber;  
 String dummy;  
 while (reader.hasNext()) {  
 userName1 = reader.nextLine();  
 password1 = reader.nextLine();  
 phoneNumber = reader.nextLine();  
 dummy = reader.nextLine();  
 if (whatToEdit.equals("password")){  
 if (userName1.equals(username)) {  
 writer.println(username);  
 writer.println(change);  
 writer.println(phoneNumber);  
 writer.println(dummy);  
 } else {  
 writer.println(userName1);  
 writer.println(password1);  
 writer.println(phoneNumber);  
 writer.println(dummy);  
 }  
 } else if (whatToEdit.equals("phone")){  
 if (userName1.equals(username)) {  
 writer.println(username);  
 writer.println(password1);  
 writer.println(change);  
 writer.println(dummy);  
 } else {  
 writer.println(userName1);  
 writer.println(password1);  
 writer.println(phoneNumber);  
 writer.println(dummy);  
 }  
 }  
 }  
 reader.close();  
 writer.flush();  
 writer.close();  
 deleteInfo = true;  
 } catch (FileNotFoundException e) {  
 System.out.println(e.getMessage());  
 input.nextLine();  
 }  
}

**11 Sign-in:**

This module is used for the user to get sign-in into their account. In this if statement has been used to validate if the user is already in the system or not with the help of username and password. If the username and password validates so the user can sign-in to his already existing account.

public static boolean signIn(String userName , String password) {  
 try {  
 try (Scanner reader = new Scanner(accounts)) {  
 String userName1;  
 String password1;  
 while (reader.hasNext()) {  
 userName1 = reader.nextLine();  
 password1 = reader.nextLine();  
 if (userName1.equals(userName) && password1.equals(password)) {  
 userNameFinal = userName;  
 setBalance(userName); // method called  
 return true;  
 }  
 }  
 }  
 } catch (Exception e) {  
 System.out.println(e.getMessage());  
 input.nextLine();  
 }  
 return false;  
}

**12 setBalance:**

This module is used to set balance of the user. This module will help to get balance from the text file which was stored in the file and show it in the program where called. This module can read the balance data from the file with the help of reader class.

public static void setBalance(String username) {  
 // the row balance from txt file  
 try {  
 try (Scanner reader1 = new Scanner(balance)) {  
 while (reader1.hasNext()) {  
 if (reader1.nextLine().equals(username)) {  
 userBalance = Integer.parseInt(reader1.nextLine());  
 }  
 }  
 }  
 } catch (Exception e) {  
 System.out.println(e.getMessage());  
 }  
}

**13 signUp:**

This module is used to add new user into the system with the help of new username and password. This module is signing new user into the system and also this module will add the data of new user to the file so for this reason this module will use both accounts and balance file in it also this module can validate the age of the user to confirm that user must be above 18 years.

public static boolean signUp (String userName , String password) throws IOException {  
 // create the communication tanel  
 FileOutputStream accountsIn = new FileOutputStream("accounts.txt",true);  
 PrintWriter writer = new PrintWriter(accountsIn);  
 FileOutputStream balanceIn = new FileOutputStream("balance.txt",true);  
 PrintWriter balanceWriter = new PrintWriter(balanceIn);  
 System.out.println("Enter your age:");  
 do {  
 try {  
 userInputInt = Integer.parseInt(inputString());  
 if (userInputInt < 18){  
 System.out.println("to have an account you must be over 18");  
 return false;  
 }else break;  
  
 }catch (Exception e){  
 System.out.println("please enter age in integers");  
 }  
 }while (true);  
 System.out.println("Enter your phone number");  
 String phoneNumber = checkPhoneNumber();  
  
 //this "if" statement checks if the username has been already taken  
 if(!CheckIfRecordExist(userName,accounts,phoneNumber,"accounts")) {  
 if (checkPasswordStrength(password)) {  
 userNameFinal = userName;  
 System.out.println("Signed in successfully");  
 writer.println(userName);  
 writer.println(password);  
 writer.println(phoneNumber);  
 writer.println("----------------------------");  
 balanceWriter.println(userName);  
 balanceWriter.println("0");  
 writer.close();  
 balanceWriter.close();  
 return true;  
 } else {  
 return false;  
 }  
 } else {  
 System.out.println("This user name or phone number is already taken.");  
 return false;  
 }  
}

**14 CheckPasswordStrength:**

This module is used to check the strength of the password which user will enter. This module will check and ensure that user enter a strong password to secure his account. Due to that reason this module will ask the user to also use integer in his password if he only used letters. This module will check this all with the help of if statement.

public static boolean checkPasswordStrength(String password){  
 int counter = 0 ;  
 int lettersCounter = 0;  
 for (int i = 0 ; i < password.length() ; i++){  
 if (Character.isDigit(password.charAt(i))){  
 counter++;  
 }  
 else if(Character.isLetter(password.charAt(i))){  
 lettersCounter++;  
 }  
 if (password.length()>= 8 && counter >= 2 && lettersCounter >= 2){  
 return true;  
 }  
 }  
 System.out.println("Please choose another password:");  
 return false;  
}

**15 randomDriverGenerator:**

This module is used to generate random drivers using multi dimensional arrays so this module can generate the drivers if the user entered his current and desired locations which was already stored in array and also the drivers names are also stored in array so this module can generate the drivers according to price destination and carplate number. RideAccepting method is also called in this module. This module work with the help of arrays.

public static void randomDriverGenerator(int currentLocation , int desiredLocation , String[] locations) {  
 // The distance between the locations as a 5 x 5 matrix  
 int [][] distances = {  
 {0,3,5,7,10},  
 {3,0,2,4,7},  
 {5,2,0,4,8},  
 {7,4,6,0,3},  
 {10,7,4,3,0}  
 };  
 String [] names = {"Ahmed" , "Mark" , "Mohammed", "Salah" , "Kashif" , "Omar" , "Alison", "Samer","Jhon", "Hasan" , "Hael" , "Khaled"};  
 int distance = distances[currentLocation][desiredLocation];  
 int numberOfAvailableDrivers = 2 + (int)(Math.random() \* 5);  
 String[][] array = new String[numberOfAvailableDrivers][6];  
 for (int i = 0; i < numberOfAvailableDrivers; i++) {  
  
 // this line will generate a random first and second name for the drivers out of the "names" array  
 String name = names[(int) (Math.random() \* names.length )] + " " + names[(int) (Math.random() \* names.length )];  
 System.out.println((i + 1) + "- driver name: " + name );  
  
 // this line will generate a random plate number with 2 characters and 3 numbers  
 String carPlate = randomCarPlateGenerator(); // method called  
 System.out.println("Car plate: " + carPlate);  
 System.out.println("Distance: " + distance + " km");  
  
 // this line will generate the price according to the distance and the price pre KM and add extra charge to simulate the different drivers  
 int price = (int)((distance \* pricePerKM) \* (1.1 + Math.random() \* 2) \* rideType);  
 System.out.println("Price: " + price);  
 System.out.println("From: " + locations[currentLocation]);  
 System.out.println("To: " + locations[desiredLocation]);  
 System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");  
 array[i][0] = name;  
 array[i][1] = carPlate;  
 array[i][2] = distance + "" ;  
 array[i][3] = price+"";  
 array[i][4] = locations[currentLocation];  
 array[i][5] = locations[desiredLocation];  
 }  
 int driverNum;  
 do {  
 try {  
 System.out.println("Please select your driver: ");  
 driverNum = Integer.parseInt(inputString().trim()) - 1;  
 if(driverNum+1 > numberOfAvailableDrivers || driverNum + 1 <= 0)  
 System.out.println("Wrong driver selected. Select again");  
 else  
 break;  
 } catch (NumberFormatException e) {  
 System.out.println(e.getMessage());  
 }  
 } while (true);  
 rideAccepting(array, driverNum); // method called  
}

**16 RideAccepting:**

This module will check balance and other accepting process (i.e. add ride record and change balance). This module check the balance of the user to accept their ride also this module will store the time of the ride. And if everything is accepted then this module will show the message to enjoy the ride for user. In this module we used if else in the arrays to check the balance. This module used looping.

public static void rideAccepting(String[][] array , int driverNum){  
 for (int i = 0; i < array.length; i++) {  
 FileOutputStream records;  
 for (int j = 0; j < 6; j++) {  
 try {  
 records = new FileOutputStream("record.txt",true);  
 PrintWriter writer = new PrintWriter(records);  
 if(i == driverNum){  
 //check balance  
 if(userBalance < Integer.parseInt(array[i][3])){  
 System.out.println("Insignificant Balance");  
 break;  
 } else {  
 // store the time of ride  
 long yourMilliSeconds = System.currentTimeMillis();  
 SimpleDateFormat sdf = new SimpleDateFormat("MMM dd,yyyy HH:mm");  
 Date resultDate = new Date(yourMilliSeconds);  
 if (j < 1) {  
 System.out.println("Enjoy Your Ride <3 <3 ^\_+ ");  
 editBalance(userNameFinal,"-"+array[i][3],balance,tempFile2);  
 writer.println(userNameFinal);  
 writer.println(sdf.format(resultDate));  
 writer.println(rideTypename);  
 }  
 writer.println(array[i][j]);  
 }  
 writer.close();  
 }  
 } catch (FileNotFoundException e) {  
 System.out.println(e.getMessage());  
 input.nextLine();  
 }  
 }  
 }  
}

**17 randomCarPlateGenerator:**

This module is used to generate the random carPlate number for the drivers used in the system so these carPlate numbers will use with those cars and with the help of these carPlate numbers the user can recognize the car. This module used for loop to create random carPlate numbers for drivers.

public static String randomCarPlateGenerator (){  
 StringBuilder carPlate = new StringBuilder();  
 for (int i = 0; i < 2; i++) {  
 carPlate.append((char) (65 + (int) (Math.random() \* 20)));  
 }  
 for (int i = 0; i < 3; i++) {  
 carPlate.append((int) (Math.random() \* 10));  
 }  
 return carPlate.toString();  
}

**18 inputString:**

This module is just used as one place for all strings inputs.

**19 DeleteFile1:**

This module is used to delete the previous file and replace it with the temp that has been edited.

First this module will delete the previous file then this will rename the temp file to the previous file which was the destination file.

public static void deleteFile1(File toDelete,File destination){  
 File dummyFile = new File(toDelete.getName());  
 toDelete.delete();  
 destination.renameTo(dummyFile);  
}

**20 CheckPhoneNumber:**

This module is used to check and validate the phone number of the user with the help of if-else statements. This module will check the characters of the phone number and compare it with the data given to it. If the phone number validate it will accept it otherwise it will ask again.

public static String checkPhoneNumber (){  
 boolean valid = false;  
 String number = "";  
 while (!valid){  
 number = inputString();  
 if (number.length() == 11 && number.charAt(0) == '0'&& number.charAt(1) == '3'){  
 valid = true;  
 for (int i = 0; i < number.length(); i++) {  
 if (!Character.isDigit(number.charAt(i))){  
 valid = false;  
 }  
 }  
 }  
 else  
 System.out.println("Number is invalid try again");  
 }  
 return number;  
}

**21 Dummy code to validate visa:**

The below whole code is used to validate the visa card number of the user to add amount to their account.

public static boolean checkVisa(long number) {  
 return (getSize(number) >= 13 && getSize(number) <= 16) &&  
 (prefixMatched(number, 4) || prefixMatched(number, 5) ||  
 prefixMatched(number, 37) || prefixMatched(number, 6)) &&  
 ((sumOfDoubleEvenPlace(number) + sumOfOddPlace(number)) % 10 == 0);  
 }  
 // Get the result from Step 2  
 public static int sumOfDoubleEvenPlace(long number) {  
 int sum = 0;  
 String num = number + "";  
 for (int i = getSize(number) - 2; i >= 0; i -= 2) {  
 int number2 = Integer.parseInt(num.charAt(i) + "") \* 2;  
 if(number2 < 9)  
 sum += number2;  
 else  
 sum += number2 / 10 + number2 % 10;  
 }  
 return sum;  
 }  
  
 // Return sum of odd-place digits in number  
 public static int sumOfOddPlace(long number) {  
 int sum = 0;  
 String num = number + "";  
 for (int i = getSize(number) - 1; i >= 0; i -= 2) {  
 sum += Integer.parseInt(num.charAt(i) + "");  
 }  
 return sum;  
 }  
 // Return true if the digit d is a prefix for number  
 public static boolean prefixMatched(long number, int d) {  
  
 return getPrefix(number, getSize(d)) == d;  
 }  
 // Return the number of digits in d  
 public static int getSize(long d) {  
 String num = d + "";  
 return num.length();  
 }  
  
 */\*\* Return the first k number of digits from number. If the  
 number of digits in number is less than k, return number. \*/* public static long getPrefix(long number, int k) {  
 if (getSize(number) > k) {  
 String num = number + "";  
 return Long.parseLong(num.substring(0, k));  
 }  
 return number;  
 }  
}