**Car Rental Management**



Session: 2021 – 2025

**Submitted by:**

Afraz Butt 2021-CS-12

**Supervised by:**

Dr. Muhammad Awais Hassan

Department of Computer Science

**University of Engineering and Technology Lahore Pakistan**

Contents

[1- Abstract 4](#_Toc105004735)

[2- Class Diagram 4](#_Toc105004736)

[3- Domain Model 5](#_Toc105004737)

[4- Sequence Diagrams 6](#_Toc105004738)

[4.1- Create Operation: 6](#_Toc105004739)

[4.2 – View Operation: 7](#_Toc105004740)

[4.3-Update Operation: 7](#_Toc105004741)

[4.4-Delete Operation: 8](#_Toc105004742)

[5- Wireframes 9](#_Toc105004743)

[5.1- Main Form 9](#_Toc105004744)

[5.2(a)-Sign In Form: 9](#_Toc105004745)

[5.2(b)-Sign up Form: 10](#_Toc105004746)

[5.3-Admin Side Wireframes: 10](#_Toc105004747)

[5.3.1 – Main Admin: 10](#_Toc105004748)

[5.3.2 – Add Car: 11](#_Toc105004749)

[5.3.3 – View Car: 11](#_Toc105004750)

[5.3.3.1 – Edit Car: 12](#_Toc105004751)

[5.3.4 – Add Maintenance: 12](#_Toc105004752)

[5.3.5 – Add User: 13](#_Toc105004753)

[5.3.6 – View Bookings(Empty at the time of documentation) 13](#_Toc105004754)

[5.3.7 – End Booking 14](#_Toc105004755)

[5.4 – Customer Side Wireframes: 14](#_Toc105004756)

[5.4.1 – Main Customer Menu: 14](#_Toc105004757)

[5.4.2 – Start Booking: 15](#_Toc105004758)

[5.4.3 – Update Record: 15](#_Toc105004759)

[5.4.4 – View Record: 16](#_Toc105004760)

[Complete Code: 16](#_Toc105004761)

[Class Car: 16](#_Toc105004762)

[Class Booking: 17](#_Toc105004763)

[Class Bus: 18](#_Toc105004764)

[Class Customer: 18](#_Toc105004765)

[Interface IBooking: 18](#_Toc105004766)

[Class Maintenance: 18](#_Toc105004767)

[Class MUser: 19](#_Toc105004768)

[Class MUserDL: 20](#_Toc105004769)

[Class CarDL: 21](#_Toc105004770)

[Class CustomerDL: 23](#_Toc105004771)

[Class BookingDL: 24](#_Toc105004772)

[Form EditCar: 26](#_Toc105004773)

[Form Add Maintenance: 26](#_Toc105004774)

[Form AddUser: 27](#_Toc105004775)

[Form End Booking: 28](#_Toc105004776)

[Form StartBooking: 29](#_Toc105004777)

[Form Update Record: 30](#_Toc105004778)

[Form View Record: 32](#_Toc105004779)

[Form View Car: 32](#_Toc105004780)

[Form Add Car: 33](#_Toc105004781)

[Form Admin Menu: 34](#_Toc105004782)

[Form Customer Menu: 35](#_Toc105004783)

[Form Main: 36](#_Toc105004784)

[Sign In form: 37](#_Toc105004785)

[Sign Up form: 38](#_Toc105004786)

# Abstract

This project aims to modernize the industry of car rentals. This is an important source of income in the auto industry of Pakistan. This project utilizes Windows Forms Framework in .NET framework, association and inheritance and the concept of class interlinkage.

# Class Diagram

Diagram, schematic

Description automatically generated

# Domain Model

Diagram

Description automatically generated

# Sequence Diagrams

## 4.1- Create Operation:

A picture containing diagram

Description automatically generated

## 4.2 – View Operation:

Diagram

Description automatically generated with medium confidence

## 4.3-Update Operation:

A picture containing graphical user interface

Description automatically generated

## 4.4-Delete Operation:

Graphical user interface

Description automatically generated with medium confidence

# Wireframes

## 5.1- Main Form

Graphical user interface

Description automatically generated with medium confidence

## 5.2(a)-Sign In Form:

Graphical user interface, application

Description automatically generated

## 5.2(b)-Sign up Form:

Graphical user interface, application

Description automatically generated

## 5.3-Admin Side Wireframes:

### 5.3.1 – Main Admin:

Text

Description automatically generated

### 5.3.2 – Add Car:

Graphical user interface

Description automatically generated

### 5.3.3 – View Car:

Table

Description automatically generated

### 5.3.3.1 – Edit Car:

Graphical user interface, application, Word

Description automatically generated

### 5.3.4 – Add Maintenance:

Graphical user interface, application

Description automatically generated

### 5.3.5 – Add User:

Graphical user interface

Description automatically generated

### 5.3.6 – View Bookings(Empty at the time of documentation)

Graphical user interface, application

Description automatically generated

### 5.3.7 – End Booking

Graphical user interface, application

Description automatically generated

## 5.4 – Customer Side Wireframes:

### 5.4.1 – Main Customer Menu:

Graphical user interface, text, application

Description automatically generated

### 5.4.2 – Start Booking:

Graphical user interface, application

Description automatically generated

### 5.4.3 – Update Record:

Graphical user interface, application

Description automatically generated

### 5.4.4 – View Record:

Graphical user interface, application, table

Description automatically generated

# Complete Code:

## Class Car:

namespace CarRentalGUI.BL

{

public class Car

{

private string licensePlate;

private string engineNo;

private string chassisNo;

private double kmsReading;

private Maintenance maint;

private string type;

public bool isBooked;

public Car()

{ }

public Car(string licensePlate, string engineNo, string chassisNo, double kmsReading,string type,double maintReading)

{

this.licensePlate = licensePlate;

this.engineNo = engineNo;

this.chassisNo = chassisNo;

this.kmsReading = kmsReading;

this.maint = new Maintenance(maintReading);

this.type = type;

isBooked = false;

}

public string LicensePlate { get => licensePlate; set => licensePlate = value; }

public string EngineNo { get => engineNo; set => engineNo = value; }

public string ChassisNo { get => chassisNo; set => chassisNo = value; }

public double KmsReading { get => kmsReading; set => kmsReading = value; }

public string Type { get => type; set => type = value; }

public Maintenance Maint { get => maint; set => maint = value; }

public virtual string busB()

{

return null;

}

}

}

## Class Booking:

namespace CarRentalGUI.BL

{

public class Booking: IBooking

{

private Car isCurrentlyBook;

private Customer isBookCustomer;

private string bookCode;

private DateTime date;

// private DateTime endDate;

public Booking(Car isCurrentlyBook, Customer isBookCustomer,DateTime date)

{

this.isCurrentlyBook = isCurrentlyBook;

this.isBookCustomer = isBookCustomer;

BookCode = null;

this.date = date;

}

public Car IsCurrentlyBook { get => isCurrentlyBook; set => isCurrentlyBook = value; }

public string BookCode { get => bookCode; set => bookCode = value; }

public Customer IsBookCustomer { get => isBookCustomer; set => isBookCustomer = value; }

public DateTime Date { get => date; set => date = value; }

public static string randomCodeGenerator()

{

Random random = new Random(); // generating a random string

StringBuilder str = new StringBuilder();

char ch ;

for(int i = 0; i < 7;i++)

{

double number = random.NextDouble();

int check = Convert.ToInt32(Math.Floor(25 \* number)); // getting a random string bw 0 and 25

ch = Convert.ToChar(check + 65); // converting floor int value to character for appending in string

// builder

str.Append(ch);

}

return Convert.ToString(str);

}

public double endBooking(double price)

{

DateTime end = DateTime.Today;

double rentalP = 0.0d;

TimeSpan t= end.Subtract(this.date);

string diff = t.ToString();

diff = (end - this.date).TotalDays.ToString();

rentalP = price \* (Convert.ToDouble(diff) + 1.0);

return rentalP;

}

}

}

## Class Bus:

namespace CarRentalGUI.BL

{

public class Bus:Car

{

public override string busB()

{

return "Bus is Booked.";

}

}

}

## Class Customer:

namespace CarRentalGUI.BL

{

public class Customer

{

protected string name;

protected string idCard;

// protected string phone;

protected string address;

public Customer(string name)

{

this.name = name;

}

public Customer(string name,string idCard,string address)

{

this.name = name;

this.idCard = idCard;

this.address = address;

// this.phone = phone;

}

public string Name { get => name; set => name = value; }

public string IdCard { get => idCard; set => idCard = value; }

public string Address { get => this.address; set => this.address = value; }

}

}

## Interface IBooking:

public interface IBooking

{

double endBooking(double price);

}

## Class Maintenance:

namespace CarRentalGUI.BL

{

public class Maintenance

{

private double oilChange;

private string comments;

public Maintenance()

{

oilChange = 3000;

comments = "fine";

}

public Maintenance(double value)

{

oilChange = value;

comments = "fine";

}

public Maintenance(double oilChange,string comments)

{

this.oilChange = oilChange;

this.comments = comments;

}

public double OilChange { get => oilChange; set => oilChange = value; }

public string Comments { get => comments; set => comments = value; }

}

}

## Class MUser:

namespace CarRentalGUI.BL

{

public class MUser

{

public MUser(string username)

{

this.username = username;

}

public MUser(string username, string password, string role)

{

this.username = username;

this.password = password;

this.role = role;

}

public MUser(string username, string password)

{

this.username = username;

this.password = password;

}

public MUser()

{

username = "";

password = "";

role = "";

}

private string username;

private string password;

private string role;

public string Username { get => username; set => username = value; }

public string Password { get => password; set => password = value; }

public string Role { get => role; set => role = value; }

public bool isAdmin()

{

if (role == "ADMIN")

return true;

return false;

}

public bool isCustomer()

{

if (role == "CUSTOMER")

return true;

return false;

}

}

}

## Class MUserDL:

namespace CarRentalGUI.DL

{

public class MUserDL

{

private static List<MUser> user = new List<MUser>();

public static List<MUser> User { get => user; } //read only attribute

public static MUser getReference(string name, string pass)

{

foreach (var i in User)

{

if (name == i.Username && pass == i.Password)

{

return i;

}

}

return null;

}

public static void addintoList(MUser m)

{

User.Add(m);

}

public static void writeInFile()

{

string path = "dataUser.txt";

StreamWriter file = new StreamWriter(path);

foreach (MUser i in user)

{

file.WriteLine(i.Username + "," + i.Password + "," + i.Role);

file.Flush();

}

file.Close();

}

public static void LoadFromFile()

{

string path = "dataUser.txt";

StreamReader file = new StreamReader(path);

if (File.Exists(path))

{

string item = "";

while ((item = file.ReadLine()) != null)

{

string[] record = item.Split(',');

string usernameA = record[0];

string passwordA = record[1];

string role = record[2];

MUser obj = new MUser(usernameA, passwordA, role);

User.Add(obj);

}

file.Close();

}

}

}

}

## Class CarDL:

namespace CarRentalGUI.DL

{

public class CarDL

{

private static List<Car> cars = new List<Car>();

public static List<Car> Cars { get => cars; set => cars = value; } // lambda operator getter setter

public static Car fetchUnBook(string type)

{

foreach(var fetchCar in cars)

{

if (type == fetchCar.Type && fetchCar.isBooked == false)

return fetchCar;

}

return null;

}

public static void writeCarinFile() // write operation in file

{

string path = "dataCars.txt";

StreamWriter file = new StreamWriter(path);

foreach(Car writeCars in cars)

{

file.WriteLine(writeCars.LicensePlate+","+writeCars.EngineNo + "," +

writeCars.ChassisNo + "," +writeCars.KmsReading + "," +writeCars.Type+","+writeCars.Maint.OilChange);

file.Flush();

}

file.Close();

}

public static void readCarFromFile() // Read Operation in file

{

string path = "dataCars.txt";

StreamReader file = new StreamReader(path);

if(File.Exists(path))

{

string item = "";

while((item = file.ReadLine()) != null)

{

string[] record = item.Split(',');

string licensePlate = record[0];

string engineNo = record[1];

string chassisNo = record[2];

double kmsReading = Convert.ToDouble(record[3]);

string type = record[4];

double maintReading = Convert.ToDouble(record[5]);

Car c = new Car(licensePlate,engineNo,chassisNo,kmsReading,type,maintReading);

if(!isCarAlreadyExist(c))

addCar(c);

}

file.Close();

}

}

public static void addCar(Car c)

{

cars.Add(c);

}

public static void clearList()

{

cars.Clear();

}

public static bool isCarAlreadyExist(Car c)

{

foreach(Car car in cars)

{

if(c.LicensePlate == car.LicensePlate)

{

return true;

}

}

return false;

}

public static void deleteCar(Car deleted)

{

cars.Remove(deleted);

}

public static Car getAcar(string licensePlate)

{

foreach(Car c in cars)

{

if(licensePlate == c.LicensePlate)

{

return c;

}

}

return null;

}

public static void updateCar(Car prev,Car updated)

{

int count = 0;

foreach(var selectCar in cars)

{

if(prev.LicensePlate == selectCar.LicensePlate)

{

cars.Remove(prev);

cars.Insert(count, updated);

break;

}

count++;

}

}

}

}

## Class CustomerDL:

namespace CarRentalGUI.DL

{

public class CustomerDL

{

private static List<Customer> customers = new List<Customer>();

public static List<Customer> Customers { get => customers; }

public static void addCust(Customer c)

{

customers.Add(c);

}

public static void writeCustomerFromFile()

{

string path = "dataCustomer.txt";

StreamWriter file = new StreamWriter(path);

foreach(var c in customers)

{

file.WriteLine(c.Name+","+c.IdCard + "," +c.Address);

file.Flush();

}

file.Close();

}

public static void readCustomerfromFile()

{

string path = "dataCustomer.txt";

StreamReader file = new StreamReader(path);

if (File.Exists(path))

{

string item = "";

while((item = file.ReadLine()) != null)

{

string[] record = item.Split(',');

string name = record[0];

string idCard = record[1];

string address = record[2];

Customer cust = new Customer(name, idCard, address);

if(!isCustomerAlreadyExist(cust))

{

addCust(cust);

}

}

file.Close();

}

}

public static void editCust(Customer prev, Customer updated)

{

int count = 0;

foreach (Customer c in customers)

{

if (prev.Name == c.Name)

{

customers.Remove(prev);

break;

}

count++;

}

customers.Insert(count, updated);

}

public static bool isCustomerHaveDetails(Customer c)

{

if (c.Address == null || c.IdCard == null)

{

return false;

}

return true;

}

public static bool isCustomerAlreadyExist(Customer c)

{

foreach (var f in customers)

{

if (c.Name == f.Name && c.Address == f.Address && c.IdCard == f.IdCard)

{

return true;

}

}

return false;

}

public static Customer fetchCustomer(Customer c)

{

foreach (var fetchCust in customers)

{

if (fetchCust.Name == c.Name)

{

return fetchCust;

}

}

return null;

}

public static Customer getCustomerByName(string name)

{

foreach(var fetchCust in customers)

{

if (fetchCust.Name == name)

return fetchCust;

}

return null;

}

}

}

## Class BookingDL:

namespace CarRentalGUI.DL

{

public class BookingDL

{

private static List<Booking> books = new List<Booking>();

public static List<Booking> Books { get => books; set => books = value; }

public static void addIntoList(Booking b)

{

books.Add(b);

}

public static void removeBooking(Booking b)

{

books.Remove(b);

}

public static Booking getDetails(string bookCode)

{

foreach (Booking b in books)

{

if(bookCode == b.BookCode)

{

return b;

}

}

return null;

}

public static void writeBookinginFile()

{

string path = "dataBookings.txt";

StreamWriter file = new StreamWriter(path);

foreach(var b in books)

{

file.WriteLine(b.IsCurrentlyBook.LicensePlate + "," +b.BookCode + "," +b.IsBookCustomer.Name + "," +b.Date.Day + ";" +b.Date.Month + ";" +b.Date.Year);

file.Flush();

}

file.Close();

}

public static bool isBookingAlreadyExist(string bookCode)

{

foreach(Booking b in books)

{

if (b.BookCode == bookCode)

return true;

}

return false;

}

public static void readBookingFromFile()

{

string path = "dataBookings.txt";

StreamReader file = new StreamReader(path);

if(File.Exists(path))

{

CarDL.readCarFromFile();

CustomerDL.readCustomerfromFile();

string item = "";

while((item = file.ReadLine()) != null)

{

string[] record = item.Split(',');

Car isBookCar = CarDL.getAcar(record[0]);

isBookCar.isBooked = true;

Customer isBookCustomer = CustomerDL.getCustomerByName(record[2]);

string [] recordOfDate = record[3].Split(';');

DateTime date = new DateTime(int.Parse(recordOfDate[2]), int.Parse(recordOfDate[1]), int.Parse(recordOfDate[0]));

Booking b = new Booking(isBookCar, isBookCustomer, date);

b.BookCode = record[1];

if (!isBookingAlreadyExist(b.BookCode))

{

books.Add(b);

}

}

}

file.Close();

}

}

}

## Form EditCar:

public partial class EditCar : Form

{

internal Car prev;

public EditCar(Car prev)

{

InitializeComponent();

this.prev = prev;

textBox1.Text = prev.LicensePlate;

}

private void btnAdd\_Click(object sender, EventArgs e)

{

try

{

Car updated = new Car(textBox1.Text, textBox2.Text, textBox3.Text, Convert.ToDouble(textBox4.Text), comboBox1.Text, Convert.ToDouble(textBox4.Text));

CarDL.updateCar(prev, updated);

CarDL.writeCarinFile();

CarDL.clearList();

CarDL.readCarFromFile();

MessageBox.Show("Updated Successfully!");

this.Close();

}

catch(Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void EditCar\_Load(object sender, EventArgs e)

{

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Form frm = new AdminMenu();

frm.Show();

this.Hide();

}

## Form Add Maintenance:

namespace CarRentalGUI.Forms

{

public partial class FrmAddMaintenance : Form

{

internal Car check;

public FrmAddMaintenance()

{

InitializeComponent();

}

private void txtLP\_TextChanged(object sender, EventArgs e)

{

check = CarDL.getAcar(txtLP.Text);

if (check != null)

{

txtLP.ReadOnly = true;

lblOC.Visible = true;

lblComments.Visible = true;

txtOilChange.Visible = true;

txtComments.Visible = true;

}

}

private void FrmAddMaintenance\_Load(object sender, EventArgs e)

{

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Form frm = new AdminMenu();

this.Hide();

frm.Show();

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

check.Maint.OilChange = Convert.ToDouble(txtOilChange.Text);

check.Maint.Comments = txtComments.Text;

}

catch(Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void FrmAddMaintenance\_KeyDown(object sender, KeyEventArgs e)

{

}

}

## Form AddUser:

public partial class FrmAddUser : Form

{

public FrmAddUser()

{

InitializeComponent();

}

private void lblback\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

this.Hide();

Form frm = new AdminMenu();

frm.Show();

}

private void btnCreate\_Click(object sender, EventArgs e)

{

MUser user = new MUser(txtUsername.Text,txtPassword.Text,comboBox1.Text);

MUserDL.addintoList(user);

MessageBox.Show("User Added Successfully!");

Form frm = new AdminMenu();

this.Hide();

frm.Show();

}

private void FrmAddUser\_Load(object sender, EventArgs e)

{

}

}

## Form End Booking:

public partial class FrmEndBooking : Form

{

internal Booking b;

internal double price;

internal double fPrice;

public FrmEndBooking()

{

InitializeComponent();

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void txtBookCode\_TextChanged(object sender, EventArgs e)

{

string code = txtBookCode.Text;

b = BookingDL.getDetails(code);

if(b != null)

{

txtPrice.Visible = true;

txtBill.Visible = true;

txtBill.ReadOnly = false;

lblBill.Visible = true;

label4.Visible = true;

btnEndBooking.Visible = true;

}

}

private void txtPrice\_TextChanged(object sender, EventArgs e)

{

try

{

DateTime dt = DateTime.Now;

price = Convert.ToDouble(txtPrice.Text);

fPrice = b.endBooking(price);

txtBill.Text = Convert.ToString(fPrice);

}

catch(Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void btnEndBooking\_Click(object sender, EventArgs e)

{

BookingDL.removeBooking(b);

b.IsCurrentlyBook.isBooked = false;

BookingDL.writeBookinginFile();

MessageBox.Show("Booking removed successfully.");

Form frm = new AdminMenu();

frm.Show();

this.Hide();

}

private void FrmEndBooking\_Load(object sender, EventArgs e)

{

BookingDL.readBookingFromFile();

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Form frm = new AdminMenu();

this.Hide();

frm.Show();

}

}

## Form StartBooking:

public partial class FrmStartBooking : Form

{

internal Customer c;

internal Car check;

public FrmStartBooking(Customer c)

{

InitializeComponent();

this.c = c;

}

private void label3\_Click(object sender, EventArgs e)

{

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Form frm = new CustomerMenu(c);

this.Hide();

frm.Show();

}

private void FrmStartBooking\_Load(object sender, EventArgs e)

{

CarDL.readCarFromFile();

txtCustName.Text = c.Name;

txtCustID.Text = c.IdCard;

txtCustID.ReadOnly = true;

txtCustName.ReadOnly = true;

}

private void btnCheck\_Click(object sender, EventArgs e)

{

check = CarDL.fetchUnBook(cmbCategories.Text);

if (check != null)

{

lblCarPlate.Visible = true;

txtCarDetails.Visible = true;

txtCarDetails.Text = check.LicensePlate;

txtCarDetails.ReadOnly = true;

btnBook.Visible = true;

if (!CustomerDL.isCustomerAlreadyExist(c))

{

c.IdCard = txtCustID.Text;

c.Address = txtCustAddress.Text;

}

else

{

MessageBox.Show("Customer already exist, will use previous details.");

txtCustID.Text = c.IdCard;

txtCustAddress.Text = c.Address;

txtCustID.ReadOnly = true;

txtCustAddress.ReadOnly = true;

}

}

else

MessageBox.Show("Car not available in given categories. Please try again.");

}

private void btnBook\_Click(object sender, EventArgs e)

{

Booking b = new Booking(check, c, DateTime.Today);

b.BookCode = Booking.randomCodeGenerator();

BookingDL.addIntoList(b);

BookingDL.writeBookinginFile();

MessageBox.Show("Booked Successfully.Your code for this booking is "+b.BookCode);

this.Hide();

Form frm = new CustomerMenu(c);

frm.Show();

}

}

## Form Update Record:

public partial class FrmUpdateRecord : Form

{

internal Customer c;

private bool isUpdated = true;

public FrmUpdateRecord(Customer c)

{

InitializeComponent();

this.c = c;

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

if (!isUpdated)

{

Form frm = new CustomerMenu(c);

this.Hide();

frm.Show();

}

}

private void FrmUpdateRecord\_Load(object sender, EventArgs e)

{

try

{

CustomerDL.readCustomerfromFile();

Customer check = CustomerDL.fetchCustomer(c);

c = check;

txtCustName.Text = c.Name;

txtCustName.ReadOnly = true;

txtCustID.Text = c.IdCard;

}

catch(Exception exp)

{

MessageBox.Show(c.Name);

}

//txtCustID.ReadOnly = true;

}

private void btnUpdate\_Click(object sender, EventArgs e)

{

try

{

isUpdated = false;

c.IdCard = txtCustID.Text;

if (!isNumberIDCardCorrect(c.IdCard) || !isCorrectIDCard(c.IdCard))

throw new Exception("Incorrect data pertinent to ID card entered.");

c.Address = txtCustAddress.Text;

Customer updated = new Customer(txtCustName.Text, txtCustID.Text, txtCustAddress.Text);

CustomerDL.editCust(c, updated);

CustomerDL.writeCustomerFromFile();

CustomerDL.readCustomerfromFile();

Form frm = new CustomerMenu(c);

frm.Show();

this.Hide();

}

catch(Exception exp)

{

isUpdated = true;

MessageBox.Show(exp.Message);

}

}

private bool isCorrectIDCard(string idCard)

{

foreach(char c in idCard)

{

if (c < 48 || c > 57)

return false;

}

return true;

}

private bool isNumberIDCardCorrect(string idCard)

{

int count = 0;

foreach(char c in idCard)

{

count++;

}

if(count == 13)

return true;

return false;

}

}

## Form View Record:

public partial class FrmViewBooking : Form

{

public FrmViewBooking()

{

InitializeComponent();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Form frm = new AdminMenu();

frm.Show();

this.Hide();

}

public void dataBind()

{

bookingsGV.DataSource = null;

foreach(Booking b in BookingDL.Books)

{

string licensePlate = b.IsCurrentlyBook.LicensePlate;

bookingsGV.DataSource = BookingDL.Books.Select(c => new { licensePlate, c.BookCode }).ToList();

}

}

private void FrmViewBooking\_Load(object sender, EventArgs e)

{

BookingDL.readBookingFromFile();

dataBind();

}

}

## Form View Car:

public partial class ViewCar : Form

{

public ViewCar()

{

InitializeComponent();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

Car prev = (Car)ViewCarGV.CurrentRow.DataBoundItem; //type casting

if(ViewCarGV.Columns["Edit"].Index == e.ColumnIndex)

{

Form frm = new EditCar(prev);

this.Hide();

frm.ShowDialog();

dataBind();

frm.Hide();

this.Show();

}

else if(ViewCarGV.Columns["Delete"].Index == e.ColumnIndex)

{

CarDL.deleteCar(prev);

dataBind();

MessageBox.Show("Deleted Successfully!");

}

}

private void label3\_Click(object sender, EventArgs e)

{

}

public void dataBind()

{

ViewCarGV.DataSource = null;

ViewCarGV.DataSource = CarDL.Cars;

ViewCarGV.Refresh();

}

private void ViewCar\_Load(object sender, EventArgs e)

{

CarDL.readCarFromFile();

ViewCarGV.DataSource = CarDL.Cars;//introspection

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Form frm = new AdminMenu();

this.Hide();

frm.Show();

}

}

## Form Add Car:

public partial class AddCar : Form

{

public AddCar()

{

InitializeComponent();

}

private void btnAdd\_Click(object sender, EventArgs e)

{

try

{

Car c = new Car(textBox1.Text, textBox2.Text, textBox3.Text, Convert.ToDouble(textBox4.Text), comboBox1.Text, Convert.ToDouble(textBox4.Text));

CarDL.addCar(c);

CarDL.writeCarinFile();

CarDL.readCarFromFile();

MessageBox.Show("Car Added Successfully!");

this.Hide();

Form frm = new AdminMenu();

frm.Show();

}

catch(Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Form frm = new AdminMenu();

this.Hide();

frm.Show();

}

private void AddCar\_Load(object sender, EventArgs e)

{

CarDL.readCarFromFile();

}

}

## Form Admin Menu:

public partial class AdminMenu : Form

{

public AdminMenu()

{

InitializeComponent();

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void addCarToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Hide();

Form frm = new AddCar();

frm.Show();

}

private void backToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Hide();

Form frm = new FormMain();

frm.Show();

}

private void viewCarToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Hide();

Form frm = new ViewCar();

frm.Show();

}

private void addUserToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Hide();

Form frm = new FrmAddUser();

frm.Show();

}

private void addMaintenanceToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Hide();

Form frm = new FrmAddMaintenance();

frm.Show();

}

private void viewBookingsToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Hide();

Form frm = new FrmViewBooking();

frm.Show();

}

private void endBookingToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form frm = new FrmEndBooking();

this.Hide();

frm.Show();

}

private void addCustomerToolStripMenuItem\_Click(object sender, EventArgs e)

{

}

private void AdminMenu\_Load(object sender, EventArgs e)

{

}

}

## Form Customer Menu:

public partial class CustomerMenu : Form

{

internal Customer c;

public CustomerMenu(Customer c)

{

InitializeComponent();

this.c = c;

}

private void bookCarToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form frm = new FrmStartBooking(c);

this.Hide();

frm.Show();

}

private void CustomerMenu\_Load(object sender, EventArgs e)

{

CustomerDL.readCustomerfromFile();

bool check = CustomerDL.isCustomerHaveDetails(c);

if(!check)

{

Form frm = new FrmUpdateRecord(c);

frm.Show();

this.Close();

}

}

private void backToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form frm = new FormMain();

this.Hide();

frm.Show();

}

private void viewPerToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form frm = new FrmViewRecord(c);

this.Hide();

frm.Show();

}

private void updatePersonalRecordsToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form frm = new FrmUpdateRecord(c);

this.Hide();

frm.Show();

}

}

## Form Main:

public partial class FormMain : Form

{

public FormMain()

{

InitializeComponent();

MUserDL.LoadFromFile();

}

private void lblHeader\_Click(object sender, EventArgs e)

{

}

private void SignIN\_Click(object sender, EventArgs e)

{

Form f1 = new SignIn();

f1.Show();

this.Hide();

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void checkBox2\_Click(object sender, EventArgs e)

{

//this.Hide();

//Form f1 = new SignUp();

//f1.Show();

}

private void checkBox1\_Click(object sender, EventArgs e)

{

//this.Hide();

//Form f1 = new SignIn();

//f1.Show();

}

private void SignInForm\_Load(object sender, EventArgs e)

{

}

private void checkBox2\_CheckedChanged(object sender, EventArgs e)

{

this.Hide();

Form f1 = new SignUp();

f1.Show();

}

private void checkBox1\_CheckedChanged\_1(object sender, EventArgs e)

{

this.Hide();

Form f1 = new SignIn();

f1.Show();

}

private void FormMain\_Load(object sender, EventArgs e)

{

CustomerDL.readCustomerfromFile();

}

}

## Sign In form:

public partial class SignIn : Form

{

internal string role;

//internal Customer c;

public SignIn()

{

InitializeComponent();

}

private void lblMessage\_Click(object sender, EventArgs e)

{

}

private void btnCheck\_Click(object sender, EventArgs e)

{

MUser p = MUserDL.getReference(txtUserName.Text, txtPassword.Text);

if(p != null)

{

role = p.Role;

if(role == "ADMIN")

{

Form frm = new AdminMenu();

frm.Show();

this.Hide();

}

else if(role == "CUSTOMER")

{

Customer c = new Customer(p.Username);

CustomerDL.addCust(c);

// Customer check = MUserDL.fetchCustomer(c);

Form frm = new CustomerMenu(c);

frm.Show();

this.Hide();

}

}

}

private void btnBack\_Click(object sender, EventArgs e)

{

Form frm = new FormMain();

this.Hide();

frm.Show();

}

private void SignIn\_Load(object sender, EventArgs e)

{

}

}

## Sign Up form:

public partial class SignUp : Form

{

public SignUp()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

MUser p = new MUser(textBox1.Text, textBox2.Text, "CUSTOMER");

MUserDL.User.Add(p);

MUserDL.writeInFile();

MUserDL.LoadFromFile();

MessageBox.Show("Registered successfully.");

this.Close();

Form frm = new FormMain();

frm.Show();

}

private void btnback\_Click(object sender, EventArgs e)

{

this.Hide();

Form f1 = new FormMain();

f1.Show();

}

private void SignUp\_Load(object sender, EventArgs e)

{

}

private void btnback\_Click\_1(object sender, EventArgs e)

{

Form frm = new FormMain();

frm.Show();

this.Hide();

}

private void SignUp\_Load\_1(object sender, EventArgs e)

{

}

}