#include<iostream>

using namespace std;

void Run() {

Start();

}

int main() {

while (true)

{

Run();

}

return 0;

}

void Menu()

{

system("cls");

cout << "\n ======= MAIN MENU ======= " << endl;

cout << " ADMIN 1 " << endl;

cout << " GUEST 2 " << endl;

cout << " ========================= " << endl;

}

void Start() {

Menu();

cout << "\n Enter your selection : ";

char select = 0;

cin >> select;

if (select == '1')

{

system("cls");

cin.ignore();

cin.clear();

cout << "\n ========= LOG IN =========" << endl;

char\* username = new char[100]{};

cout << "\n Enter username : ";

cin.getline(username, 100);

char\* pass = new char[100]{};

cout << " Enter password : ";

cin.getline(pass, 100);

if (isAdmin(username, pass))

{

while (true)

{

system("cls");

cout << "\n =======================" << endl;

cout << " CARS 1 " << endl;

cout << " RENTS 2 " << endl;

cout << " ADD CAR 3 " << endl;

cout << " DELETE CAR 4 " << endl;

cout << " MAIN MENU 5 " << endl;

cout << " =======================" << endl;

cout << "\n Enter your selection : ";

char s = 0;

cin >> s;

if (s == '1')

{

auto cars = LoadData();

ShowAllCars(cars, global\_car\_count);

cout << endl;

system("pause");

}

else if (s == '2')

{

auto users = LoadUsers();

system("cls");

ShowAllUsers(users, global\_user\_count);

}

else if (s == '3')

{

cars = ReadFromFile();

car\_count = global\_car\_count;

AddCar();

WriteCarsToFile(cars, car\_count);

cars = ReadFromFile();

cout << "\n Car was successfully added!" << endl;

Sleep(3000);

}

else if (s == '4')

{

system("cls");

cout << "\n ========== DELETING A CAR ========== " << endl;

auto cars = LoadData();

ShowAllCars(cars, global\_car\_count);

int id = 0;

cout << "\n Enter ID of the car : ";

cin >> id;

RemoveCarbByID(id);

}

else if (s == '5')

{

break;

}

else

{

system("cls");

cout << "\n Incorrect Input! " << endl;

Sleep(3000);

}

}

}

}

else if (select == '2')

{

while (true)

{

system("cls");

auto cars = LoadData();

ShowAllCars(cars, global\_car\_count);

cout << "\n Enter Car ID : ";

int id = 0;

cin >> id;

auto car = getCarById(cars, global\_car\_count, id);

if (car == nullptr)

{

system("cls");

cout << "\n There is not a car with this ID." << endl;

Sleep(3000);

break;

}

else

{

system("cls");

if (car->isFull)

{

cout << "\n You cannot select this car." << endl;

cout << " Car is full." << endl;

return;

}

else

{

system("cls");

ShowCar(car);

cout << "\n Enter start date" << endl;

int start\_day = 0;

cout << " Day : ";

cin >> start\_day;

int start\_month = 0;

cout << " Month : ";

cin >> start\_month;

int start\_year = 0;

cout << " Year : ";

cin >> start\_year;

cout << "\n Enter end date" << endl;

int end\_day = 0;

cout << " Day : ";

cin >> end\_day;

int end\_month = 0;

cout << " Month : ";

cin >> end\_month;

int end\_year = 0;

cout << " Year : ";

cin >> end\_year;

cin.ignore();

cin.clear();

int total\_days = end\_day - start\_day;

cout << "\n Total Payments : " << total\_days \* car->pricePerDay << "$" << endl;

cout << "\ Enter your own info" << endl;

cout << " Enter name and surname : " << endl;

char\* fullname = new char[100]{};

cin.getline(fullname, 100);

User\* user = new User{ car->id,fullname,car->model,\_\_DATE\_\_ };

car->isFull = true;

cout << "\n Your rent process was completed successfully!" << endl;

WriteCarsToFile(cars, global\_car\_count);

auto users = LoadUsers();

users = AddUser(users, global\_user\_count, user);

WriteUsersToFile(users, global\_user\_count);

}

}

}

}

else

{

system("cls");

cout << "\n Incorrect Input! " << endl;

Sleep(3000);

}

}

int car\_count = 3;

int id = 3;

Car\*\* cars = new Car \* [car\_count] {

new Car{ 1,new char[] {"Bentley Continental"},false,350,Date{0,0,0},Date{0,0,0} },

new Car{ 2,new char[] {"Mercedes 63 AMG"},true,280,Date{0,0,0},Date{0,0,0} },

new Car{ 3,new char[] {"Rolls Royce Ghost"},false,550,Date{0,0,0},Date{0,0,0} },

};

int user\_count = 1;

User\*\* users = new User \* [1]{

new User{2,new char[] {"Rafiq Rafiqli"},new char[] {"Mercedes 63 AMG"},\_\_DATE\_\_}

};

Car\*\* LoadData() {

Car\*\* car\_s = nullptr;

if (isExist("cars.bin")) {

car\_s = ReadFromFile();

}

else {

WriteCarsToFile(cars, 3);

car\_s = ReadFromFile();

}

return car\_s;

}

User\*\* LoadUsers() {

User\*\* user\_s = nullptr;

if (isExist("users.bin")) {

user\_s = ReadUsersFromFile();

}

else {

WriteUsersToFile(users, 1);

user\_s = ReadUsersFromFile();

}

return user\_s;

}

bool isAdmin(const char\* username, const char\* pass) {

if (strcmp(username, "admin") == 0) {

if (strcmp(pass, "admin") == 0) {

return true;

}

else {

system("cls");

cout << "\n Password is incorrect!" << endl;

Sleep(3000);

return false;

}

}

else {

system("cls");

cout << "\n Username is incorrect!" << endl;

Sleep(3000);

return false;

}

}

void ShowCar(const Car\* car) {

cout << "\n =========CAR INFO==========" << endl;

cout << " ID : " << car->id << endl;

cout << " MODEL : " << car->model << endl;

cout << " Rented : ";

if (car->isFull) {

cout << "YES" << endl;

}

else {

cout << "NO" << endl;

}

cout << " Price per day : " << car->pricePerDay << "$ " << endl;

cout << " Date start : " << car->start.day << "/" << car->start.month << "/" << car->start.year << endl;

cout << " Date end : " << car->end.day << "/" << car->end.month << "/" << car->end.year << endl;

}

void ShowAllCars(Car\*\* cars, int count) {

system("cls");

for (size\_t i = 0; i < count; i++)

{

ShowCar(cars[i]);

}

}

void ShowUser(const User\* user) {

cout << "\n RENT INFO : " << user->fullname << " " << user->car\_id << " " << user->car\_model << " " << user->takeCarDate << endl;

}

void ShowAllUsers(User\*\* users, int count) {

for (size\_t i = 0; i < count; i++)

{

ShowUser(users[i]);

}

cout << endl;

system("pause");

}

Car\* getCarById(Car\*\* cars, const int& count, const int& id) {

for (size\_t i = 0; i < count; i++)

{

if (cars[i]->id == id) {

return cars[i];

}

}

return nullptr;

}

User\* getUserByFullname(User\*\* users, const int& count, const char\* fullname) {

for (size\_t i = 0; i < count; i++)

{

if (strcmp(users[i]->fullname, fullname) == 0) {

return users[i];

}

}

return nullptr;

}

User\*\* AddUser(User\*\* users, int& count, User\* newuser) {

auto newusers = new User \* [count + 1]{};

for (size\_t i = 0; i < count; i++)

{

newusers[i] = users[i];

}

newusers[count] = newuser;

users = newusers;

newusers = nullptr;

count++;

return users;

}

// Tasks

//Rent Car proyektine

//

//add car funksiyasi yazmaq

//

//isFull field problem hell elemek

//

//remove Car ozelliyi

int GetLatestCarID()

{

cars = ReadFromFile(); // to get car count

return cars[car\_count - 1]->id;

}

Car\* GetNewCar()

{

system("cls");

cout << "\n ========== ADDING NEW CAR ========== " << endl;

int i = GetLatestCarID();

int id\_of\_car = i + 1;

id++;

cin.ignore();

cin.clear();

char\* model = new char[100] {};

cout << "\n Enter model of the car : ";

cin.getline(model, 100);

bool isFull = false;

double price = 0;

cout << " Enter price of the car : ";

cin >> price;

cout << "\n Enter start date" << endl;

int start\_day = 0;

cout << " Day : ";

cin >> start\_day;

int start\_month = 0;

cout << " Month : ";

cin >> start\_month;

int start\_year = 0;

cout << " Year : ";

cin >> start\_year;

cout << "\n Enter end date" << endl;

int end\_day = 0;

cout << " Day : ";

cin >> end\_day;

int end\_month = 0;

cout << " Month : ";

cin >> end\_month;

int end\_year = 0;

cout << " Year : ";

cin >> end\_year;

Car\* new\_car = new Car{ id\_of\_car, model, isFull, price, start\_day, start\_month, start\_year, end\_day, end\_month, end\_year };

return new\_car;

}

void AddCar()

{

Car\* new\_car = GetNewCar();

Car\*\* new\_cars = new Car \* [car\_count + 1]{};

for (int x = 0; x < car\_count; x++)

{

new\_cars[x] = cars[x];

}

new\_cars[car\_count] = new\_car;

cars = new\_cars;

new\_cars = nullptr;

car\_count++;

global\_car\_count++;

}

int GetIndexOfCarByID(int id)

{

cars = ReadFromFile(); // to get car count

for (int x = 0; x < car\_count; x++)

{

if (cars[x]->id == id)

{

return x;

}

}

return -1;

}

void RemoveCarbByID(int id)

{

cars = ReadFromFile(); // to get car count

int index = GetIndexOfCarByID(id);

if (index != -1)

{

Car\*\* new\_cars = new Car \* [car\_count - 1]{};

for (int x = 0; x < index; x++)

{

new\_cars[x] = cars[x];

}

for (int y = index + 1; y < car\_count; y++)

{

new\_cars[y - 1] = cars[y];

}

cars = new\_cars;

new\_cars = nullptr;

car\_count--;

global\_car\_count--;

WriteCarsToFile(cars, car\_count);

cars = ReadFromFile();

cout << "\n Car was successfully deleted!" << endl;

Sleep(3000);

}

else

{

cout << "\n There is not a car with this ID." << endl;

Sleep(3000);

}

}

bool isExist(const char\* filename) {

FILE\* file;

fopen\_s(&file, filename, "rb");

if (file != nullptr) {

fclose(file);

return true;

}

return false;

}

//3

// 1

// 5 - model data

// bool

//double

// 10 12 2022

// 12 12 2022

void WriteCarsToFile(Car\*\* cars, int car\_count) {

FILE\* file;

fopen\_s(&file, "cars.bin", "wb");

fwrite(&car\_count, sizeof(int), 1, file);//car count

for (size\_t i = 0; i < car\_count; i++)

{

fwrite(&cars[i]->id, sizeof(int), 1, file);// car id

int l = strlen(cars[i]->model);

fwrite(&l, sizeof(int), 1, file); //model length

fwrite(cars[i]->model, sizeof(char), l, file);

//bool

fwrite(&cars[i]->isFull, sizeof(bool), 1, file);

//double

fwrite(&cars[i]->pricePerDay, sizeof(double), 1, file);

//start

fwrite(&cars[i]->start.day, sizeof(int), 1, file);

fwrite(&cars[i]->start.month, sizeof(int), 1, file);

fwrite(&cars[i]->start.year, sizeof(int), 1, file);

//end

fwrite(&cars[i]->end.day, sizeof(int), 1, file);

fwrite(&cars[i]->end.month, sizeof(int), 1, file);

fwrite(&cars[i]->end.year, sizeof(int), 1, file);

}

fclose(file);

}

int global\_car\_count = 0;

Car\*\* ReadFromFile() {

int count = 0;

FILE\* file;

fopen\_s(&file, "cars.bin", "rb");

fread\_s(&count, sizeof(int), sizeof(int), 1, file);

global\_car\_count = count;

auto cars = new Car \* [count] {};

for (size\_t i = 0; i < count; i++)

{

int id = 0;

fread\_s(&id, sizeof(int), sizeof(int), 1, file);

int l = 0;

fread\_s(&l, sizeof(int), sizeof(int), 1, file);

char\* model = new char[l + 1]{};

fread\_s(model, l, sizeof(char), l, file);

bool isFull = false;

fread\_s(&isFull, sizeof(bool), sizeof(bool), 1, file);

double pricePerDay = 0;

fread\_s(&pricePerDay, sizeof(double), sizeof(double), 1, file);

Date start;

fread\_s(&start.day, sizeof(int), sizeof(int), 1, file);

fread\_s(&start.month, sizeof(int), sizeof(int), 1, file);

fread\_s(&start.year, sizeof(int), sizeof(int), 1, file);

Date end;

fread\_s(&end.day, sizeof(int), sizeof(int), 1, file);

fread\_s(&end.month, sizeof(int), sizeof(int), 1, file);

fread\_s(&end.year, sizeof(int), sizeof(int), 1, file);

cars[i] = new Car{ id,model,isFull,pricePerDay,start,end };

}

fclose(file);

return cars;

}

int global\_user\_count = 0;

User\*\* ReadUsersFromFile() {

int count = 0;

FILE\* file;

fopen\_s(&file, "users.bin", "rb");

fread\_s(&count, sizeof(int), sizeof(int), 1, file);

global\_user\_count = count;

auto users = new User \* [count] {};

for (size\_t i = 0; i < count; i++)

{

int id = 0;

fread\_s(&id, sizeof(int), sizeof(int), 1, file);

int l1 = 0;

fread\_s(&l1, sizeof(int), sizeof(int), 1, file);

char\* fullname = new char[l1 + 1]{};

fread\_s(fullname, l1, sizeof(char), l1, file);

int l2 = 0;

fread\_s(&l2, sizeof(int), sizeof(int), 1, file);

char\* model = new char[l2 + 1]{};

fread\_s(model, l2, sizeof(char), l2, file);

int l3 = 0;

fread\_s(&l3, sizeof(int), sizeof(int), 1, file);

char\* date = new char[l3 + 1]{};

fread\_s(date, l3, sizeof(char), l3, file);

users[i] = new User{ id,fullname,model,date };

}

fclose(file);

return users;

}

void WriteUsersToFile(User\*\* users, int user\_count) {

FILE\* file;

fopen\_s(&file, "users.bin", "wb");

fwrite(&user\_count, sizeof(int), 1, file);

for (size\_t i = 0; i < user\_count; i++)

{

fwrite(&users[i]->car\_id, sizeof(int), 1, file);

int l1 = strlen(users[i]->fullname);

fwrite(&l1, sizeof(int), 1, file);

fwrite(users[i]->fullname, sizeof(char), l1, file);

int l2 = strlen(users[i]->car\_model);

fwrite(&l2, sizeof(int), 1, file);

fwrite(users[i]->car\_model, sizeof(char), l2, file);

int l3 = strlen(users[i]->takeCarDate);

fwrite(&l3, sizeof(int), 1, file);

fwrite(users[i]->takeCarDate, sizeof(char), l3, file);

}

fclose(file);

}

struct Date {

int year;

int month;

int day;

};

struct Car {

int id;

char\* model;

bool isFull = false;

double pricePerDay;

Date start;

Date end;

};

struct User {

int car\_id;

char\* fullname;

char\* car\_model;

const char\* takeCarDate;

};