**10.code**

#include <iostream>

#include <vector>

#include <string>

using namespace std;

class Bike {

private:

string bikeID;

string model;

bool isAvailable;

double pricePerHour;

public:

Bike(string id, string mdl, double price)

: bikeID(id), model(mdl), isAvailable(true), pricePerHour(price) {}

string getBikeID() const { return bikeID; }

string getModel() const { return model; }

bool getAvailability() const { return isAvailable; }

double getPricePerHour() const { return pricePerHour; }

void bookBike() { isAvailable = false; }

void returnBike() { isAvailable = true; }

};

class User {

private:

string userID;

string name;

public:

User(string id, string nm) : userID(id), name(nm) {}

string getUserID() const { return userID; }

string getName() const { return name; }

};

class Booking {

private:

string bookingID;

User user;

Bike &bike;

int hours;

double cost;

bool isCompleted;

public:

Booking(string id, User usr, Bike &bk, int hrs)

: bookingID(id), user(usr), bike(bk), hours(hrs), cost(hrs \* bk.getPricePerHour()), isCompleted(false) {}

string getBookingID() const { return bookingID; }

Bike &getBike() { return bike; }

bool isCompletedStatus() const { return isCompleted; }

void completeBooking() {

isCompleted = true;

bike.returnBike();

}

void displayDetails() const {

cout << "Booking ID: " << bookingID << endl

<< "User: " << user.getName() << endl

<< "Bike: " << bike.getModel() << endl

<< "Hours: " << hours << endl

<< "Cost: ₹" << cost << endl

<< "Status: " << (isCompleted ? "Completed" : "Ongoing") << endl;

}

};

class BikeBookingSystem {

private:

vector<Bike> bikes;

vector<Booking> bookings;

public:

void addBike(string id, string model, double price) {

bikes.emplace\_back(id, model, price);

}

void bookBike() {

if (bikes.empty()) {

cout << "No bikes available in the system!" << endl;

return;

}

string userID, userName;

int hours, bikeChoice;

cout << "Enter User ID: ";

cin >> userID;

cout << "Enter User Name: ";

cin.ignore();

getline(cin, userName);

cout << "Available Bikes:" << endl;

for (size\_t i = 0; i < bikes.size(); ++i) {

cout << i + 1 << ". Bike ID: " << bikes[i].getBikeID()

<< ", Model: " << bikes[i].getModel()

<< ", Price per Hour: ₹" << bikes[i].getPricePerHour()

<< ", Availability: " << (bikes[i].getAvailability() ? "Available" : "Booked") << endl;

}

cout << "Select the bike by entering the corresponding number: ";

cin >> bikeChoice;

if (bikeChoice < 1 || bikeChoice > bikes.size()) {

cout << "Invalid selection. Try again!" << endl;

return;

}

if (!bikes[bikeChoice - 1].getAvailability()) {

cout << "The selected bike is already booked. Please choose another bike." << endl;

return;

}

cout << "Enter number of hours to book the bike: ";

cin >> hours;

User user(userID, userName);

string bookingID = "BK" + to\_string(bookings.size() + 1);

bikes[bikeChoice - 1].bookBike();

bookings.emplace\_back(bookingID, user, bikes[bikeChoice - 1], hours);

cout << "Bike booked successfully!" << endl

<< "Booking ID: " << bookingID << endl

<< "Total cost: ₹" << hours \* bikes[bikeChoice - 1].getPricePerHour() << endl;

}

void completeBooking() {

string bookingID;

cout << "Enter Booking ID to complete: ";

cin >> bookingID;

for (auto &booking : bookings) {

if (booking.getBookingID() == bookingID && !booking.isCompletedStatus()) {

booking.completeBooking();

cout << "Booking completed successfully." << endl;

return;

}

}

cout << "Booking ID not found or already completed." << endl;

}

void showAllBikes() const {

for (const auto &bike : bikes) {

cout << "Bike ID: " << bike.getBikeID()

<< ", Model: " << bike.getModel()

<< ", Price per Hour: ₹" << bike.getPricePerHour()

<< ", Availability: " << (bike.getAvailability() ? "Available" : "Booked") << endl;

}

}

void showAllBookings() const {

for (const auto &booking : bookings) {

booking.displayDetails();

cout << "----------------------" << endl;

}

}

};

int main() {

BikeBookingSystem system;

int choice;

system.addBike("B001", "Yamaha FZ", 50.0); // ₹50 per hour

system.addBike("B002", "Honda Shine", 100.0); // ₹100 per hour

system.addBike("B003", "Royal Enfield", 150.0); // ₹150 per hour

do {

cout << "\nBike Booking System Menu:" << endl;

cout << "1. Show All Bikes" << endl;

cout << "2. Book a Bike" << endl;

cout << "3. Complete a Booking" << endl;

cout << "4. Show All Bookings" << endl;

cout << "5. Exit" << endl;

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

system.showAllBikes();

break;

case 2:

system.bookBike();

break;

case 3:

system.completeBooking();

break;

case 4:

system.showAllBookings();

break;

case 5:

cout << "Exiting system. Goodbye!" << endl;

break;

default:

cout << "Invalid choice. Please try again." << endl;

}

} while (choice != 5);

return 0;

}