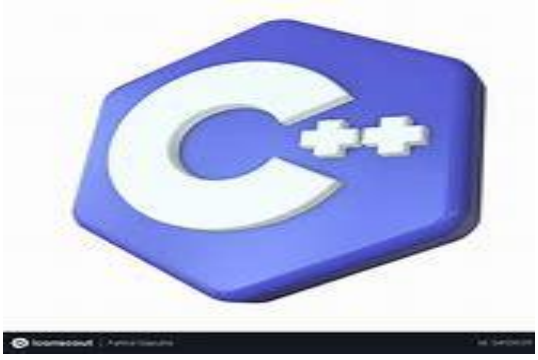
LOVELY PROFESSIONAL UNIVERSITY

CAP-281: OBJECT ORIENTED PROGRAMMING

CA:2 PROJECT REPORT

SUBMITTED BY:	SUBMITTED TO:
Name: SAHIB CHOUHAN REG.NO: 12411026 ROLL NO: RD2411A03 NAME: GARGI SHARMA REG.NO:12403355 ROLL NO: RD2411A024 SECTION: D2411[G1]	MRS. Yamini Bhardwaj



PROJECT TITLE:

COURIER TRACKING SYSTEM – TRACK PACKAGES AND UPDATE DELIVERY STATUS

The Courier Tracking System is designed to efficiently manage parcel deliveries by allowing users to add, track, and update shipments. It enhances transparency, reduces delivery uncertainties, and improves logistics management.

Developed in C++ using object-oriented programming (OOP) principles, the system features unique tracking ID generation, secure login, and receipt generation. This project aims to streamline courier operations, ensuring accuracy and convenience for both service providers and customers.

KEY FEATURES OF THE PROJECT:

- User Authentication: Secure login system to prevent unauthorized access.
- Unique Tracking ID Generation: Each parcel is assigned a randomly generated tracking ID for easy identification.
- Courier Management: Users can add new couriers with sender and receiver details.
- Real-time Tracking: Allows users to track shipments by entering the tracking ID.
- Status Update: Delivery status can be updated as packages move through different stages.
- Receipt Generation: Automatically generates a detailed receipt for each transaction.
- User-Friendly Menu: A structured menu makes navigation simple and intuitive.
- Data Handling: Uses vectors to store courier records efficiently.

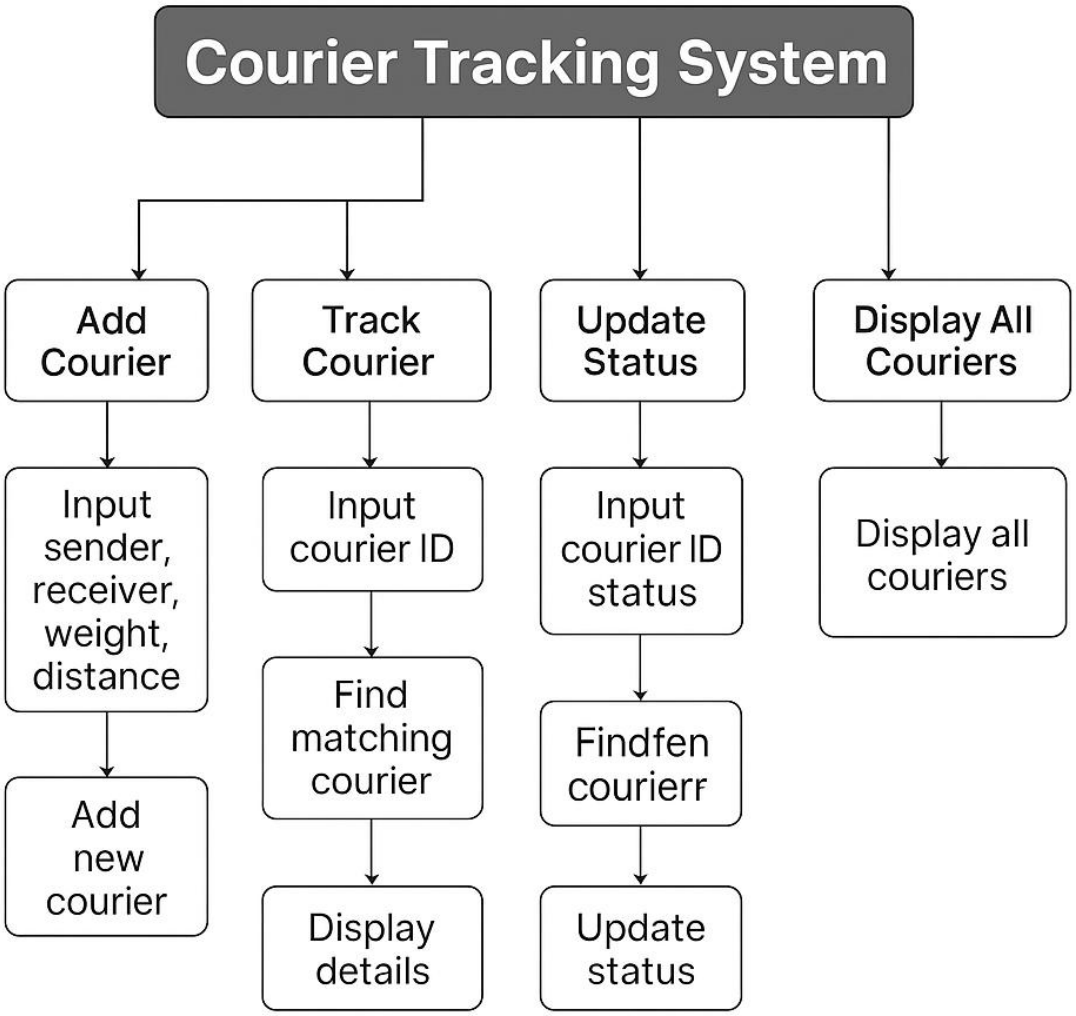
KEY CONCEPTS USED IN ABOVE PROJECT:

- Object-Oriented Programming (OOP): Utilizes classes and objects to structure the program efficiently.
- Encapsulation: Protects data by keeping it private within the class and using public methods for controlled access.
- Constructors: Automatically initialize object properties when a new courier entry is created.
- Vectors: Used for dynamic storage and management of multiple courier records.
- Random Number Generation: Generates unique tracking IDs using the rand() function.
- Loops and Conditional Statements: Used for user interaction, menu navigation, and courier tracking.
- Functions and Modular Programming: Splits tasks into separate functions for better code organization and reusability.

FUNCTIONALITY OF THE PROJECT:

1. **Login System:** The program starts by prompting the user for a secure login.
2. **Adding a Courier:** Users can enter sender details, receiver details, weight, and distance to generate a new courier entry with a unique tracking ID.
3. **Tracking a Shipment:** Users can input a tracking ID to retrieve the current status and details of the parcel.
4. **Updating Courier Status:** Allows modification of the delivery status as the parcel progresses.
5. **Displaying All Couriers:** Lists all courier entries along with their details and statuses.
6. **Receipt Generation:** Automatically prints a receipt with all relevant courier information.
7. **Menu System:** A well-structured menu guides users through different functionalities for an intuitive experience.

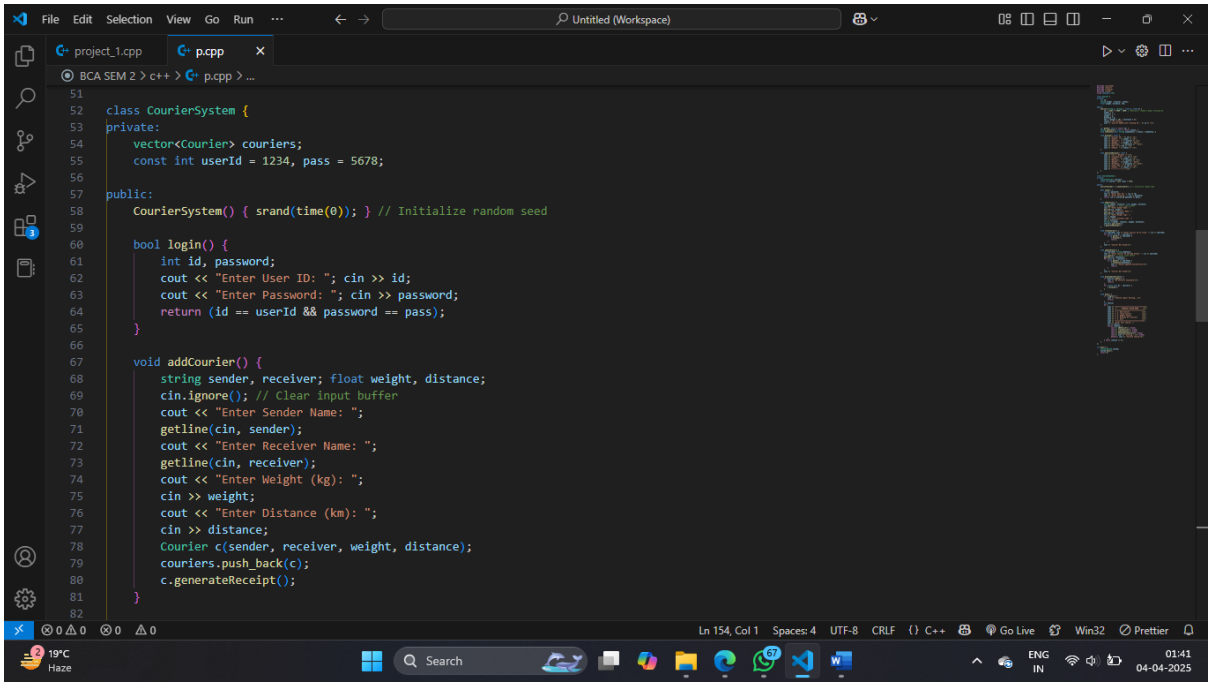
DIAGRAMATICAL REPRESENTATION FOR THE WORKFLOW OF CODE:



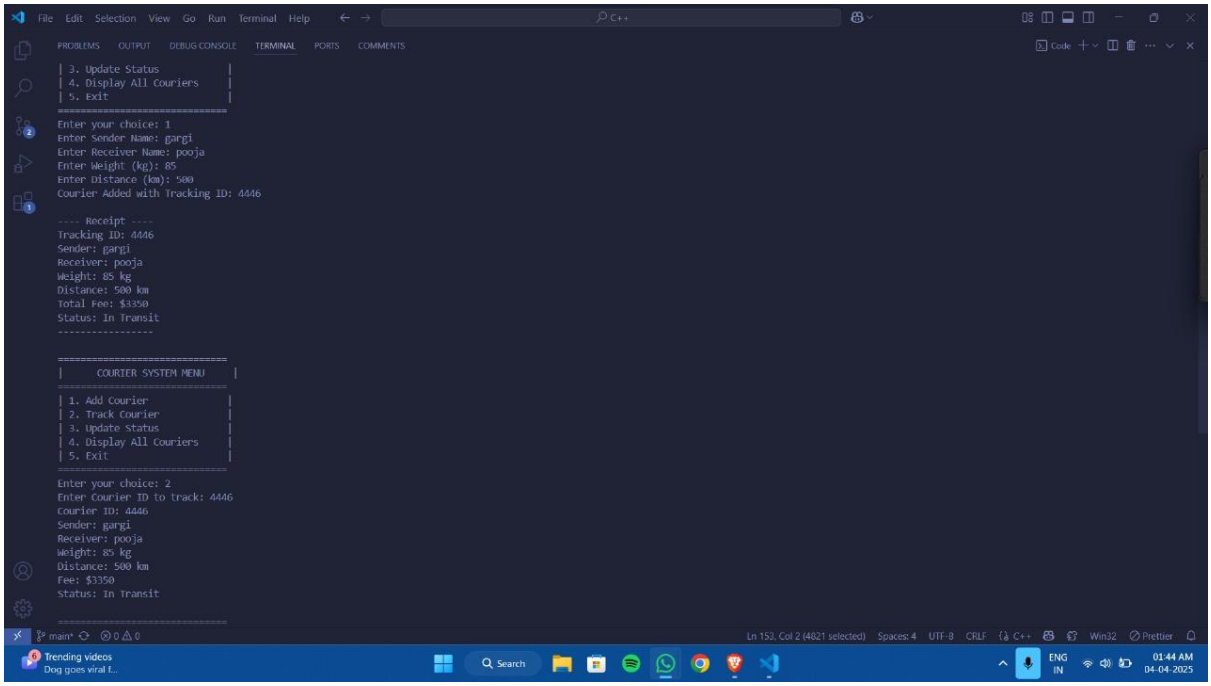
CODE:

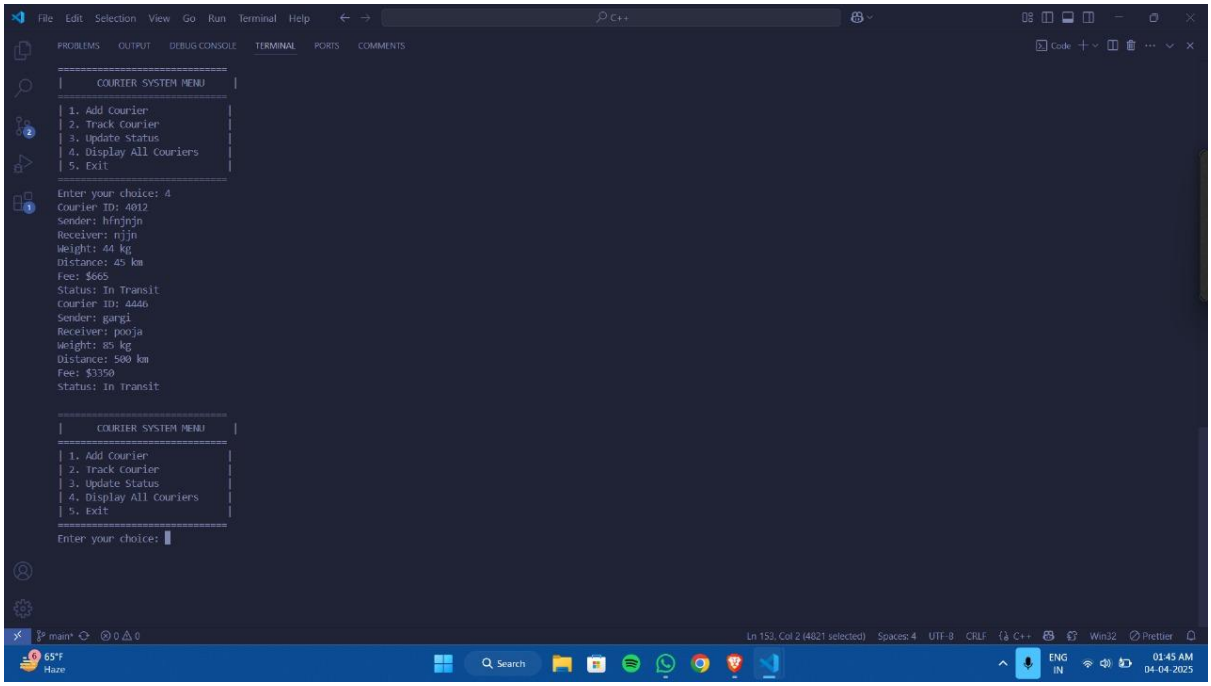
```
class CourierSystem {
    void displayAllCouriers() {
    }

    void menu() {
        if (!login()) {
            cout << "Invalid Login! Exiting...\n";
            return;
        }
        int choice;
        do {
            cout << "\n============================\n";
            cout << "|      COURIER SYSTEM MENU      |\n";
            cout << "==================== (const char [33])"| 1. Add Courier              |\n"
            cout << "| 1. Add Courier              |\n";
            cout << "| 2. Track Courier            |\n";
            cout << "| 3. Update Status            |\n";
            cout << "| 4. Display All Couriers     |\n";
            cout << "| 5. Exit                     |\n";
            cout << "============================\n";
            cout << "Enter your choice: ";
            cin >> choice;
            switch (choice) {
                case 1: addCourier(); break;
                case 2: trackCourier(); break;
                case 3: updateStatus(); break;
                case 4: displayAllCouriers(); break;
                case 5: cout << "Exiting...\n"; break;
                default: cout << "Invalid choice!\n";
            }
        } while (choice != 5);
    }
```

```
#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
using namespace std;

class Courier {
private:
    int id;
    string sender, receiver, status;
    float weight, distance, fee;

public:
    Courier(string s, string r, float w, float d) {
        id = rand() % 9000 + 1000; // Generate a random 4-digit tracking ID
        sender = s;
        receiver = r;
        weight = w;
        distance = d;
        fee = (weight * 10) + (distance * 5);
        status = "In Transit";
        cout << "Courier Added with Tracking ID: " << id << "\n";
    }

    int getId() const { return id; }
    string getStatus() const { return status; }
    void setStatus(const string &newStatus) { status = newStatus; }

    void display() const {
        cout << "Courier ID: " << id << "\n";
        cout << "Sender: " << sender << "\n";
        cout << "Receiver: " << receiver << "\n";
```


File Edit Selection View Go Run
Untitled (Workspace)
project_1.cpp
p.cpp
BCA SEM 2 > c++ > p.cpp > ...
class CourierSystem {
private:
vector<Courier> couriers;
const int userId = 1234, pass = 5678;
public:
CourierSystem() { srand(time(0)); } // Initialize random seed
bool login() {
int id, password;
cout << "Enter User ID: "; cin >> id;
cout << "Enter Password: "; cin >> password;
return (id == userId && password == pass);
}
void addCourier() {
string sender, receiver; float weight, distance;
cin.ignore(); // Clear input buffer
cout << "Enter Sender Name: ";
getline(cin, sender);
cout << "Enter Receiver Name: ";
getline(cin, receiver);
cout << "Enter Weight (kg): ";
cin >> weight;
cout << "Enter Distance (km): ";
cin >> distance;
Courier c(sender, receiver, weight, distance);
couriers.push_back(c);
c.generateReceipt();
}
Ln 154, Col 1 Spaces: 4 UTF-8 CRLF C++ Go Live Win32 Prettier
19°C
Haze
Search
ENG
IN
01:41
04-04-2025

OUTPUT:


File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
| 3. Update Status |
| 4. Display All Couriers |
| 5. Exit |
Enter your choice: 1
Enter Sender Name: gargi
Enter Receiver Name: pooja
Enter Weight (kg): 85
Enter Distance (km): 500
Courier Added with Tracking ID: 4446
---- Receipt ----
Tracking ID: 4446
Sender: gargi
Receiver: pooja
Weight: 85 kg
Distance: 500 km
Total Fee: $3350
Status: In Transit
COURIER SYSTEM MENU
| 1. Add Courier |
| 2. Track Courier |
| 3. Update Status |
| 4. Display All Couriers |
| 5. Exit |
Enter your choice: 2
Enter Courier ID to track: 4446
Courier ID: 4446
Sender: gargi
Receiver: pooja
Weight: 85 kg
Distance: 500 km
Fee: $3350
Status: In Transit
Trending videos
Dog goes viral f...
Search
ENG
IN
01:44 AM
04-04-2025


File Edit Selection View Go Run Terminal Help
C++
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
Code
==========
| COURIER SYSTEM MENU |
==========
| 1. Add Courier |
| 2. Track Courier |
| 3. Update Status |
| 4. Display All Couriers |
| 5. Exit |
==========
Enter your choice: 4
Courier ID: 4012
Sender: hfnjnjn
Receiver: njjn
Weight: 44 kg
Distance: 45 km
Fee: $665
Status: In Transit
Courier ID: 4446
Sender: gargi
Receiver: pooja
Weight: 85 kg
Distance: 500 km
Fee: $3350
Status: In Transit
==========
| COURIER SYSTEM MENU |
==========
| 1. Add Courier |
| 2. Track Courier |
| 3. Update Status |
| 4. Display All Couriers |
| 5. Exit |
==========
Enter your choice:
main* 0 0
Ln 153, Col 2 (4021 selected) Spaces: 4 UTF-8 CRLF C++ Win32 Prettier
65°F
Haze
Search
ENG
IN
01:45 AM
04-04-2025