**UNIVERSITY INSTITUTE OF COMPUTING**

**PROJECT REPORT**

**ON**

**BANKING MANAGEMENT SYSTEM**

Program Name: BCA

Subject Name/Code: Data Structures(24CAT-151)

**Submitted by: Submitted to:**

**Name:** AMAN **Name:** MONIKA CHOUDHARY

**UID:** 24BCA10462 **Designation:** Assit.prof.

**Section:** 24BCA 7 “A”

ABSTRACT

Introduction:

**This C++ banking application simplifies basic banking operations such as account creation, secure login, balance inquiry, and money transfers. It uses linked lists to manage user data efficiently, allowing dynamic storage and easy access to account information.**

**New users can register with their details, receive a unique user number and account number, and set a password. Existing users can log in securely to check their balance or transfer money to other accounts. The application ensures secure transactions and efficient data handling, providing a streamlined and user-friendly banking experience.**

Technique:

The application is developed using C++ with a focus on object-oriented programming (OOP) principles. It employs the following techniques:

1. **Object-Oriented Design:** The use of classes and objects helps in encapsulating related data and functions, making the code modular and easy to maintain.
2. **Linked List Data Structure:** Linked lists are used for dynamic storage of user accounts, allowing efficient addition, searching, and deletion operations. This structure is ideal for managing user data without the need for fixed-size arrays.
3. **Secure Authentication:** User credentials are securely stored, and password verification ensures that only authenticated users can access their accounts.

System Configuration:

* **OS:** Windows 10 or Linux
* **Processor:** Intel Core i3 (minimum); Core i5 or higher recommended
* **RAM:** 4 GB (minimum); 8 GB recommended
* **Development Environment:** Any C++ IDE (e.g., Visual Studio, Code::Blocks) or Visual Studio Code with a C++ compiler (GCC or Microsoft C++ Compiler)

SUMMARY

Input:

**Main Menu:**

* The user is prompted to choose between:
  + **1:** Existing User
  + **2:** New User
  + **0:** Exit

**For Existing Users:**

* **User Number:** Input your user number (e.g., 1001).
* **Password:** Enter your password for authentication.
* After login, the following options are available:
  + **1:** Check balance.
  + **2:** Transfer money.
  + **0:** Exit.
* **Money Transfer Process:**
  + **Recipient's Account Number:** Enter the receiver's account number (e.g., 33330).
  + **Amount:** Enter the amount to transfer.
  + If balance is insufficient, the user can either re-enter the amount or exit.

**For New Users:**

* **Full Name:** Enter your full name.
* **Mobile Number:** Enter your mobile number.
* **Address:** Enter your address.
* **Password:** Create a password.
* After confirming the details, the system generates a new user number and account number.

**BANKING MANAGEMENT SYSTEM PROCEDURE CODE :-**

#include <iostream>

#include <vector>

#include <string>

class Account {

public:

std::string fullName;

std::string mobileNumber;

std::string address;

std::string password;

int accountNumber;

double balance;

Account(std::string name, std::string mobile, std::string addr, std::string pass, int accNum, double bal)

: fullName(name), mobileNumber(mobile), address(addr), password(pass), accountNumber(accNum), balance(bal) {}

};

class Bank {

private:

std::vector<Account> accounts;

public:

void createAccount(std::string name, std::string mobile, std::string addr, std::string pass, int accountNumber, double initialDeposit) {

accounts.emplace\_back(name, mobile, addr, pass, accountNumber, initialDeposit);

}

void deposit(int accountNumber, double amount) {

for (auto &acc : accounts) {

if (acc.accountNumber == accountNumber) {

acc.balance += amount;

break;

}

}

}

void withdraw(int accountNumber, double amount) {

for (auto &acc : accounts) {

if (acc.accountNumber == accountNumber && acc.balance >= amount) {

acc.balance -= amount;

break;

}

}

}

void displayAccount(int accountNumber) {

for (const auto &acc : accounts) {

if (acc.accountNumber == accountNumber) {

std::cout << "Full Name: " << acc.fullName << ", Mobile Number: " << acc.mobileNumber

<< ", Address: " << acc.address << ", Account Number: " << acc.accountNumber

<< ", Balance: " << acc.balance << std::endl;

break;

}

}

}

};

int main() {

Bank bank;

bank.createAccount("Alice Johnson", "1234567890", "123 Main St", "password123", 1001, 500.0);

bank.createAccount("Bob Smith", "0987654321", "456 Elm St", "mypassword", 1002, 300.0);

bank.deposit(1001, 200.0);

bank.withdraw(1002, 100.0);

bank.displayAccount(1001);

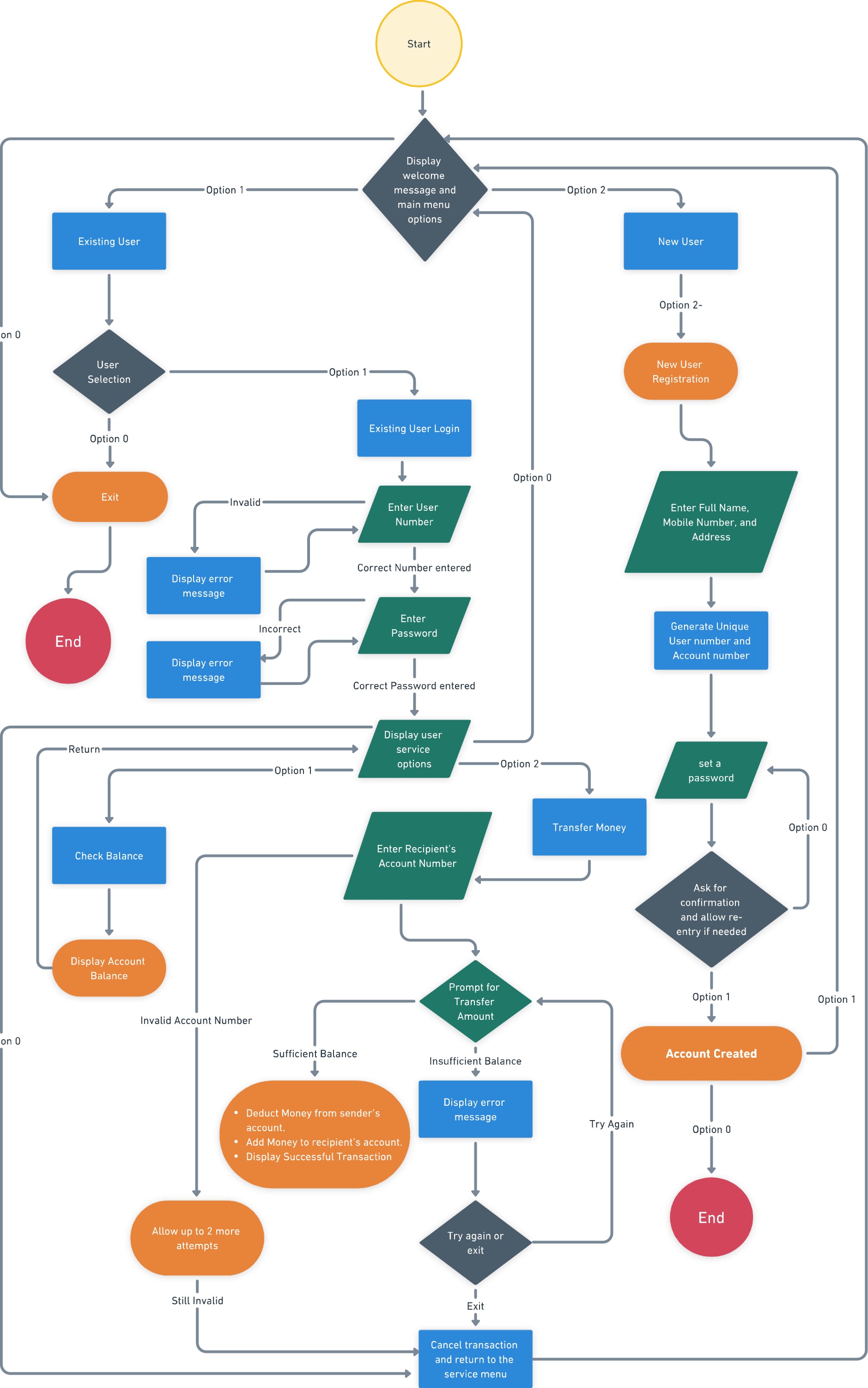
bank.displayAccount(1002);

return 0;

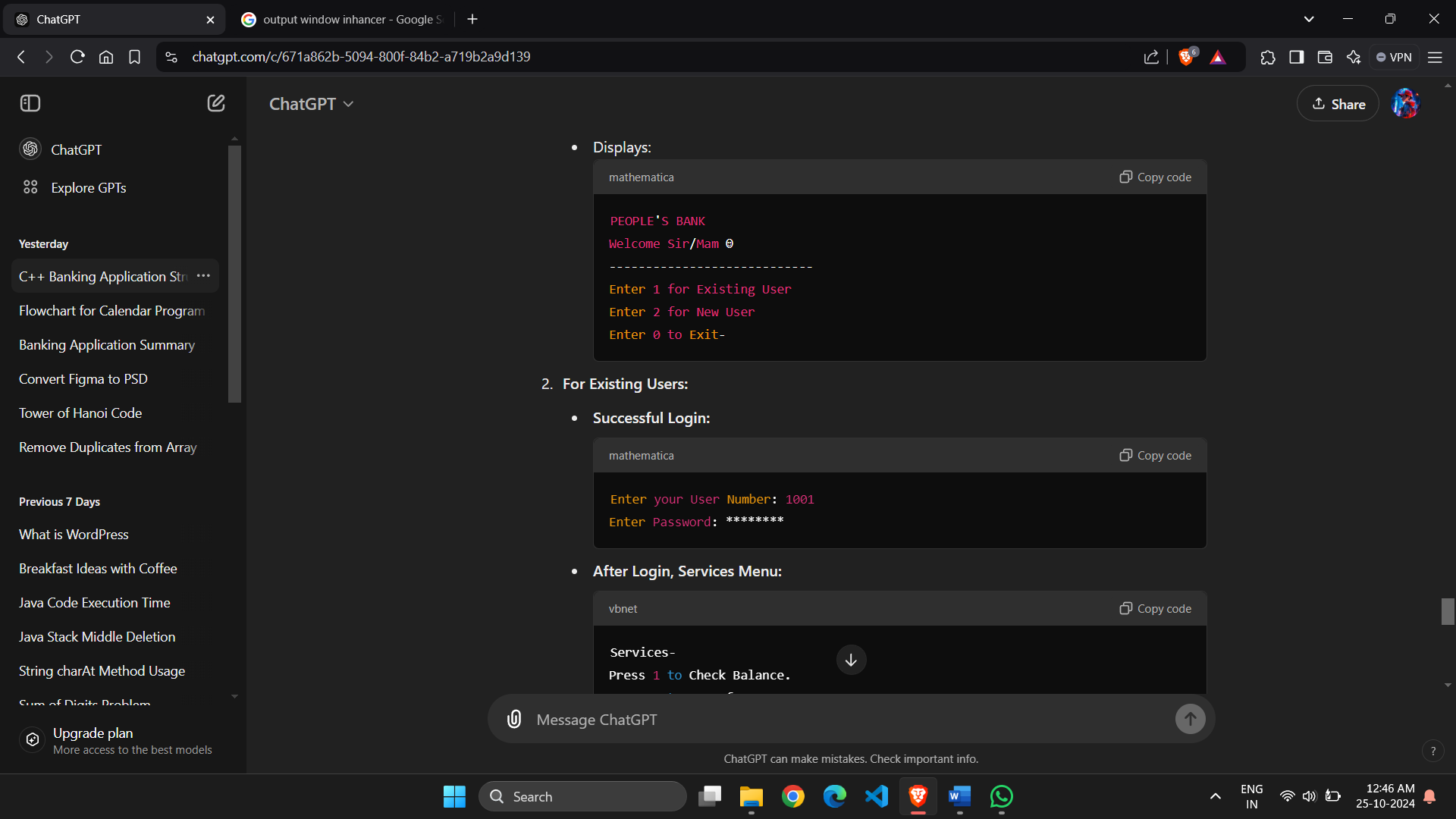
}

Output:

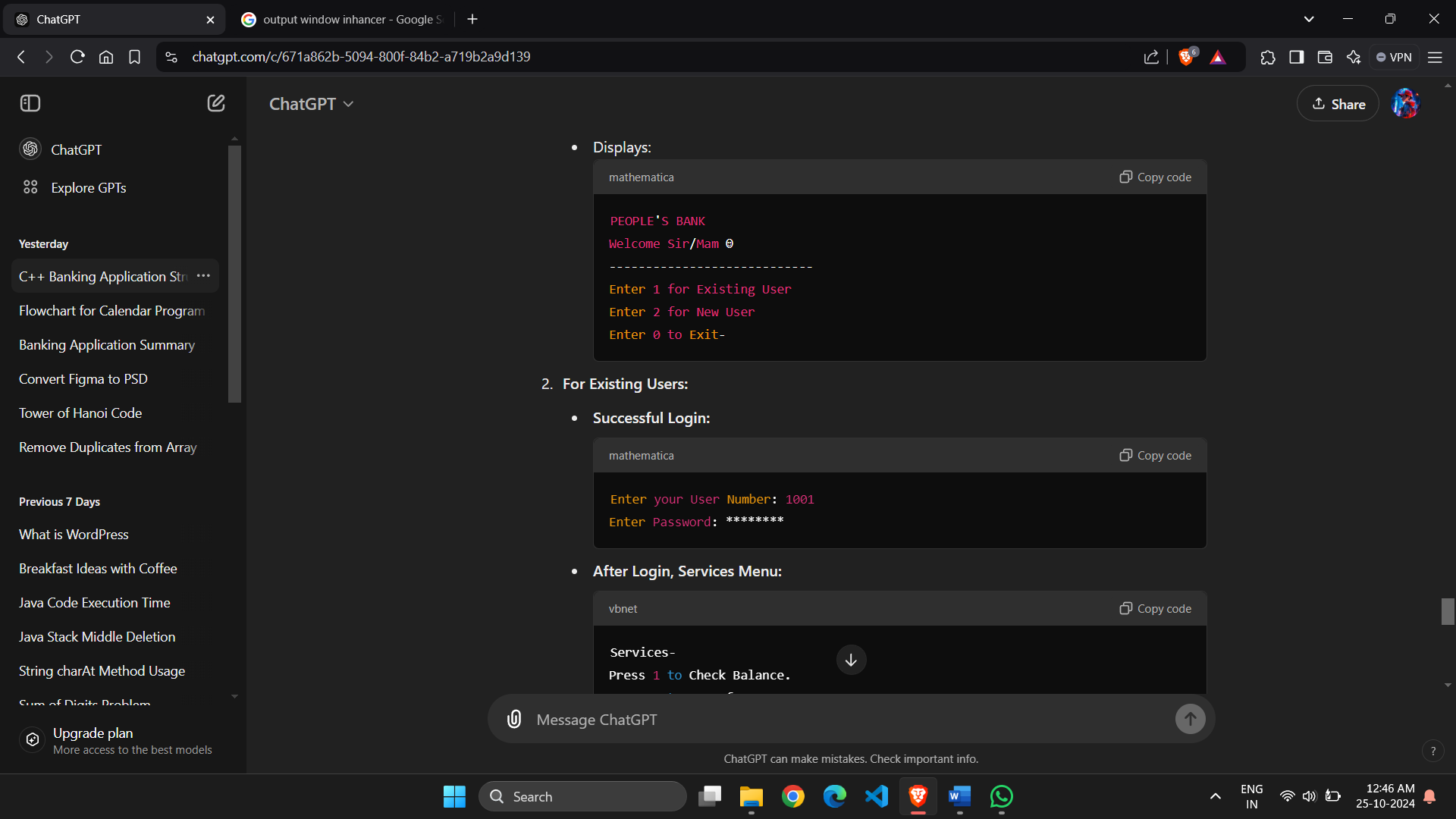
Process:



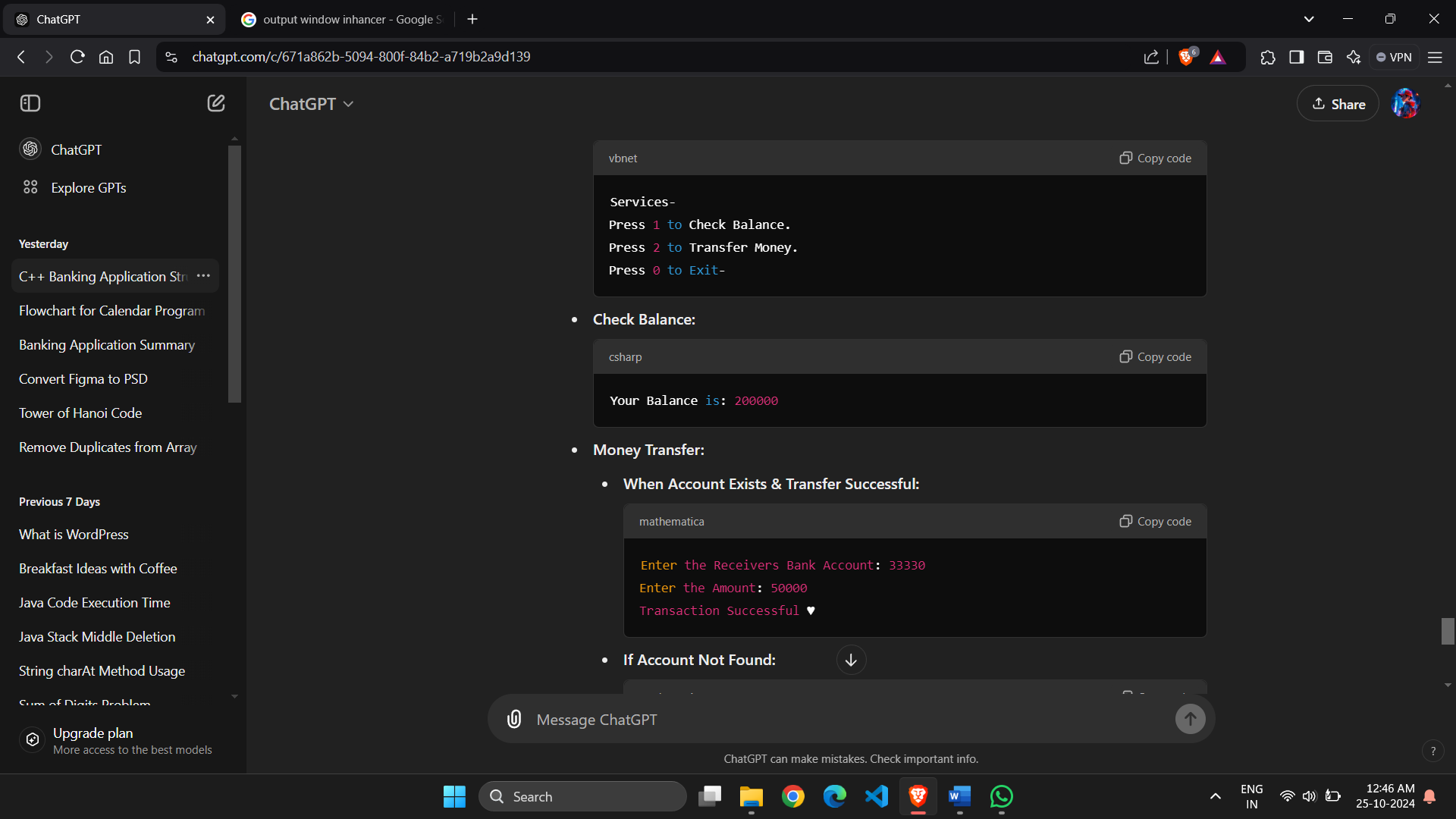
Main Menu:



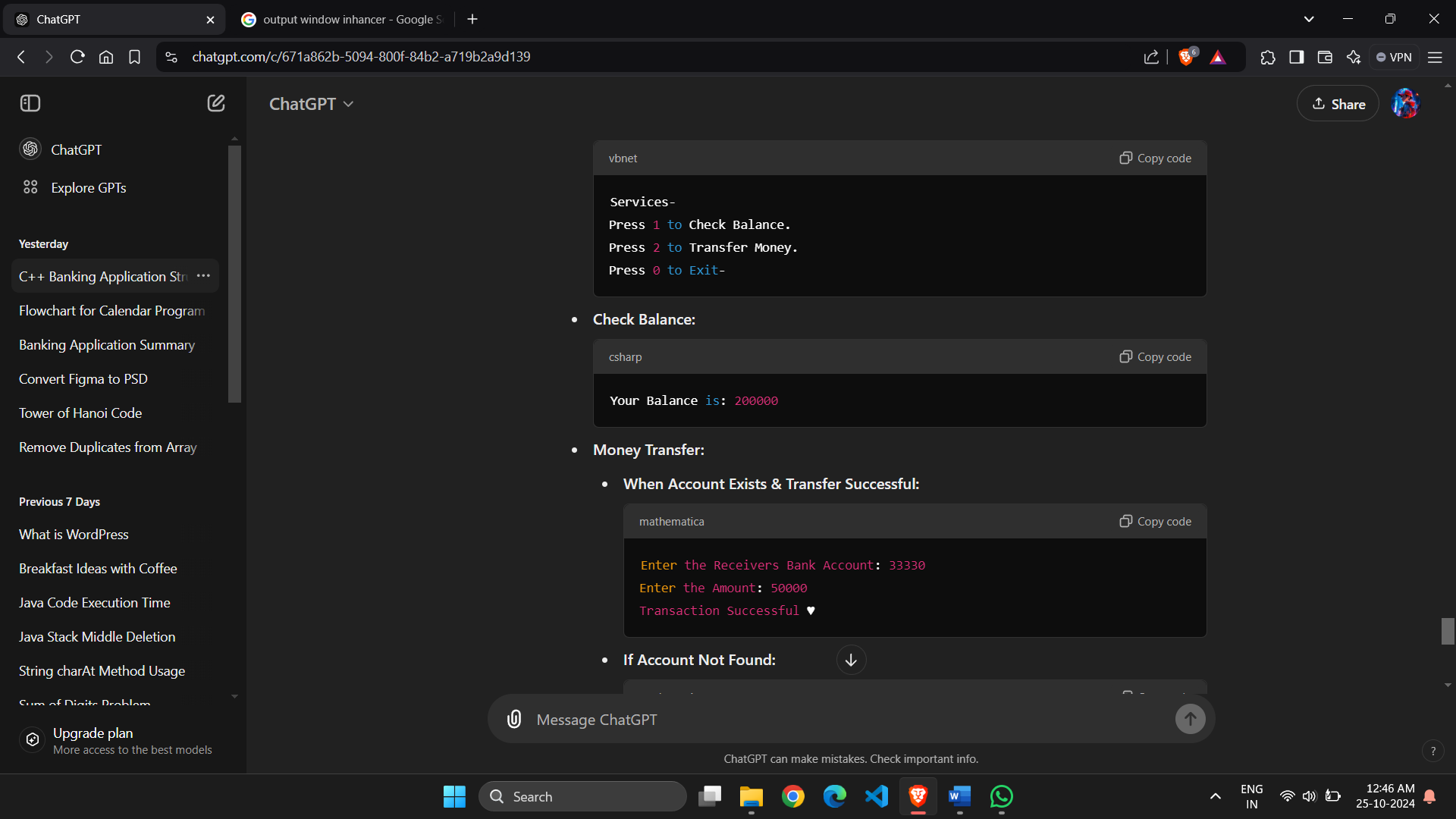
Existing User Login:



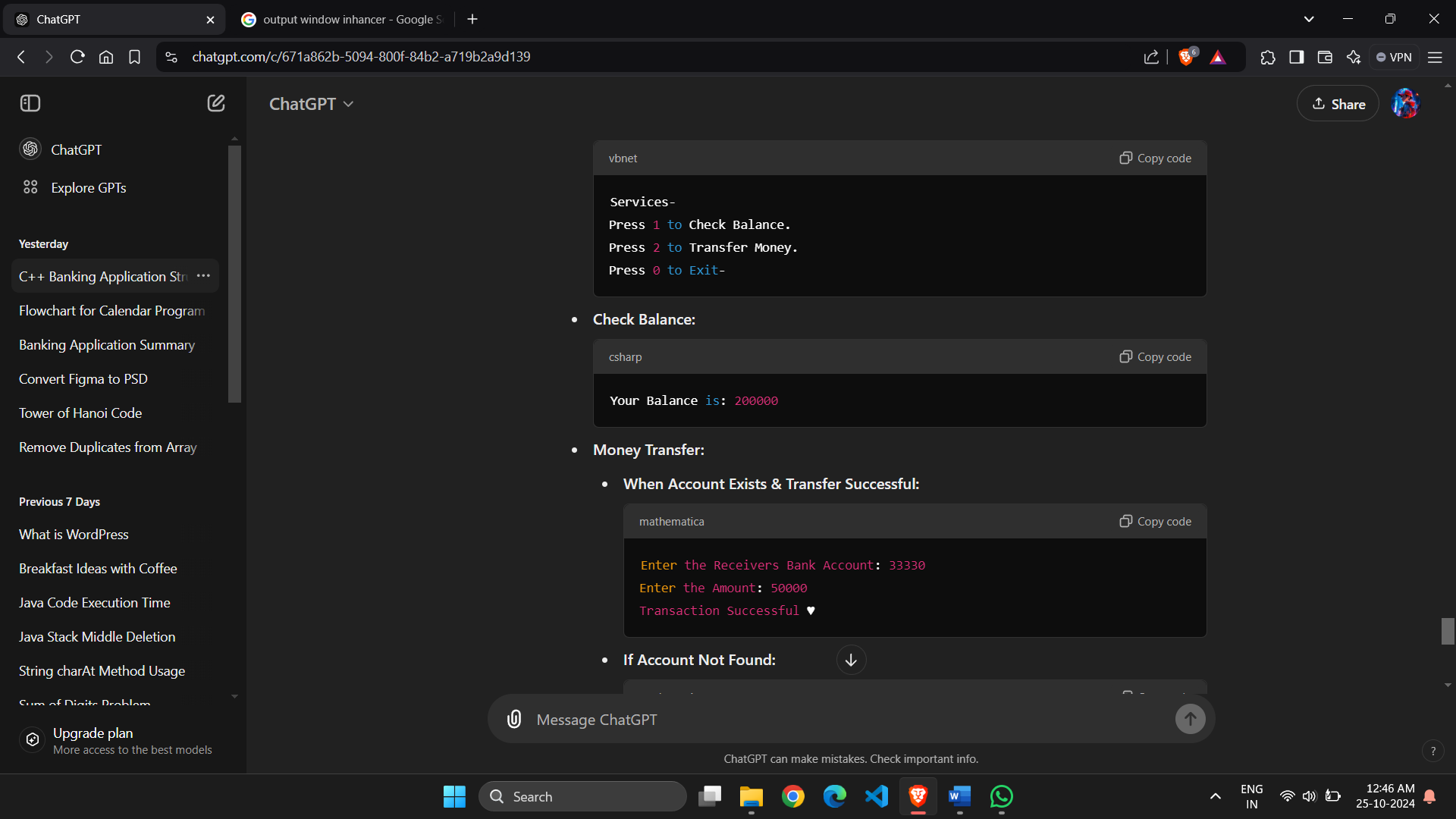
Service Menu:



Check Balance:



Money Transfer:



New Account Creation:

