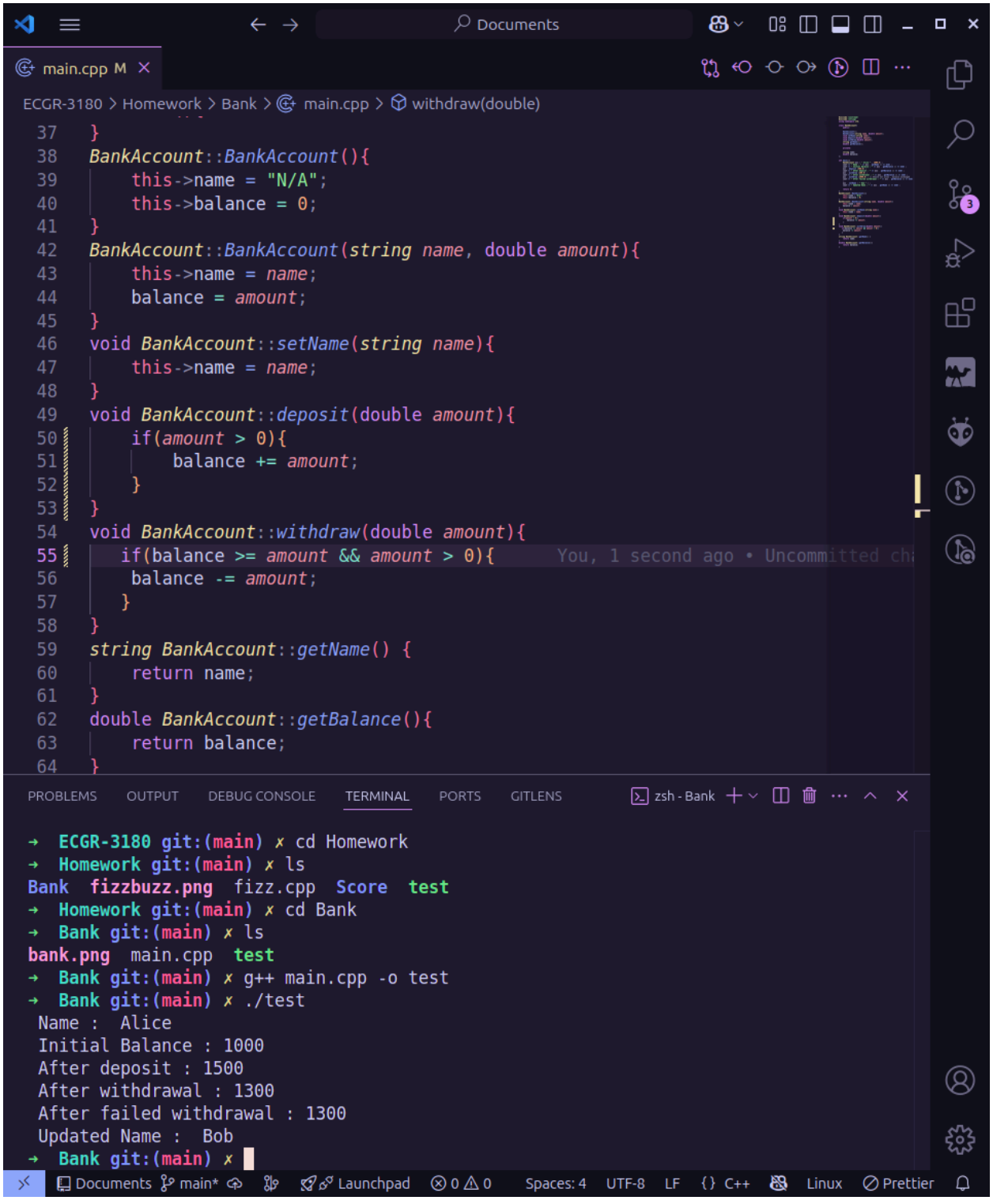
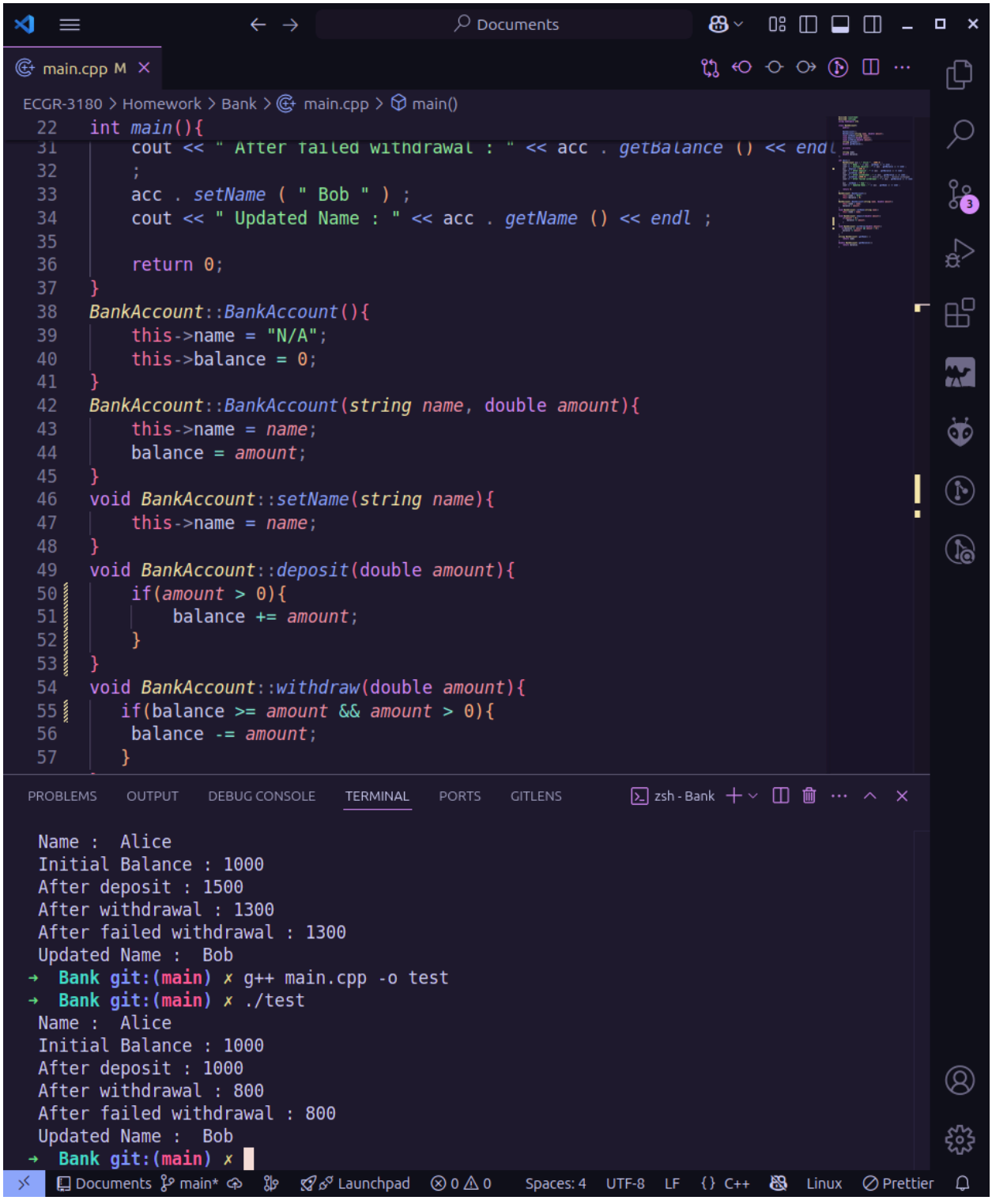
1. In this problem I must make a BankAccount class that holds the name and balance as a private data member. It must have a constructor, getters for the name and balance, and a setter for the name. It must also have a function to deposit money and a function to withdraw money. The withdraw function must prevent overdraft.
2. My solution to this problem didn't require much problem solving except for the withdraw function. For that I just compared the request withdrawal amount to the current balance, if the amount was less than or equal to current balance I would allow transfer to happen if not then the program would do nothing.
3. My non-AI attempt was a if statement to prevent the user from withdrawing more money than they had. I also decided to write an if statement to prevent negative withdrawals and negative deposits.
4. My non-AI attempt was successful so AI was not needed.
5. For the test I used the code provided since it included a test of the withdrawal function. I decided to test the deposit function by changing the amount to -500 instead of 500. The output of that test is in the second picture.

6.


Documents
main.cpp M
ECGR-3180 > Homework > Bank > main.cpp > withdraw(double)
37 }
38 BankAccount::BankAccount(){
39 this->name = "N/A";
40 this->balance = 0;
41 }
42 BankAccount::BankAccount(string name, double amount){
43 this->name = name;
44 balance = amount;
45 }
46 void BankAccount::setName(string name){
47 this->name = name;
48 }
49 void BankAccount::deposit(double amount){
50 if(amount > 0){
51 balance += amount;
52 }
53 }
54 void BankAccount::withdraw(double amount){
55 if(balance >= amount && amount > 0){ You, 1 second ago • Uncommitted ch
56 balance -= amount;
57 }
58 }
59 string BankAccount::getName() {
60 return name;
61 }
62 double BankAccount::getBalance(){
63 return balance;
64 }
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
zsh - Bank
→ ECGR-3180 git:(main) ✗ cd Homework
→ Homework git:(main) ✗ ls
Bank fizzbuzz.png fizz.cpp Score test
→ Homework git:(main) ✗ cd Bank
→ Bank git:(main) ✗ ls
bank.png main.cpp test
→ Bank git:(main) ✗ g++ main.cpp -o test
→ Bank git:(main) ✗ ./test
Name : Alice
Initial Balance : 1000
After deposit : 1500
After withdrawal : 1300
After failed withdrawal : 1300
Updated Name : Bob
→ Bank git:(main) ✗
Documents main* Launchpad 0 0 Spaces: 4 UTF-8 LF C++ Linux Prettier


Documents
main.cpp M
ECGR-3180 > Homework > Bank > main.cpp > main()
22 int main(){
31 cout << " After failed withdrawal : " << acc . getBalance () << endl
32 ;
33 acc . setName (" Bob ") ;
34 cout << " Updated Name : " << acc . getName () << endl ;
35
36 return 0;
37 }
38 BankAccount::BankAccount(){
39 this->name = "N/A";
40 this->balance = 0;
41 }
42 BankAccount::BankAccount(string name, double amount){
43 this->name = name;
44 balance = amount;
45 }
46 void BankAccount::setName(string name){
47 this->name = name;
48 }
49 void BankAccount::deposit(double amount){
50 if(amount > 0){
51 balance += amount;
52 }
53 }
54 void BankAccount::withdraw(double amount){
55 if(balance >= amount && amount > 0){
56 balance -= amount;
57 }
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS
zsh - Bank
Name : Alice
Initial Balance : 1000
After deposit : 1500
After withdrawal : 1300
After failed withdrawal : 1300
Updated Name : Bob
→ Bank git:(main) ✗ g++ main.cpp -o test
→ Bank git:(main) ✗ ./test
Name : Alice
Initial Balance : 1000
After deposit : 1000
After withdrawal : 800
After failed withdrawal : 800
Updated Name : Bob
→ Bank git:(main) ✗
Documents main* Launchpad 0 0 Spaces: 4 UTF-8 LF C++ Linux Prettier