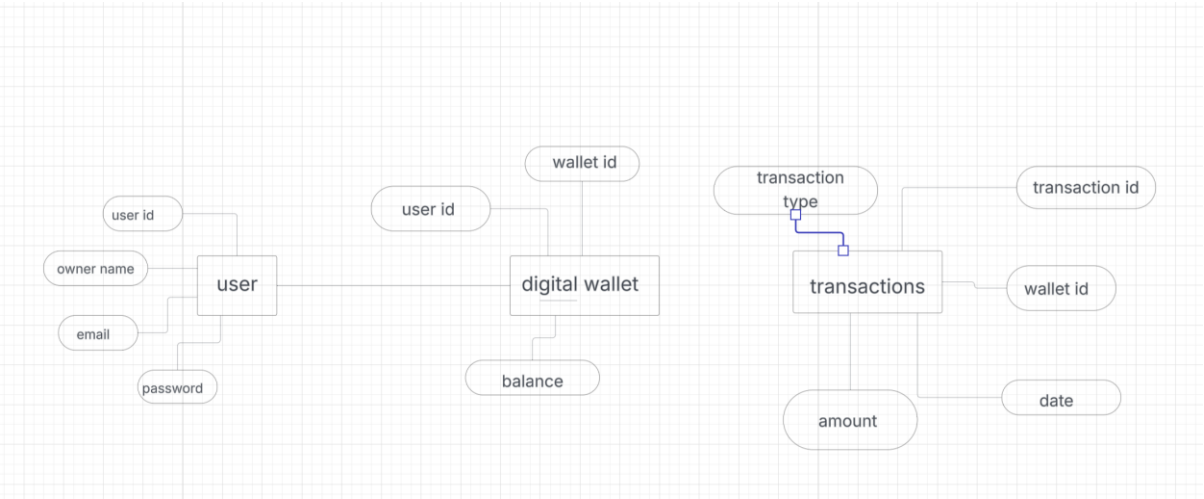
ER DIAGRAM:



1. Introduction

The Digital Wallet project is a C++ application designed to facilitate the management of personal finances through a digital wallet interface. As digital transactions become increasingly prevalent, the need for secure and efficient financial management tools has grown. This project aims to provide users with a simple yet effective way to manage their money, track their spending, and save their financial data.

1.1 Purpose

The primary purpose of the Digital Wallet application is to allow users to:

- Add funds to their wallet.
- Spend money while ensuring they do not exceed their balance.
- Save their wallet data to a file for future reference.
- Display wallet details in a user-friendly manner.

1.2 Importance

Digital wallets are essential in today's economy, providing convenience and security for transactions. This project contributes to the growing field of financial technology (FinTech) by offering a basic yet functional digital wallet solution.

2. Project Objectives

The objectives of the Digital Wallet project include:

- Developing a robust C++ application that implements core wallet functionalities.
- Ensuring user-friendly interactions through console-based input and output.
- Implementing error handling to manage invalid operations gracefully.
- Providing data persistence through file operations.

3. Key Features

3.1 User Management

- Each wallet is associated with an owner's name, allowing for personalized management of funds.

3.2 Balance Management

- Users can set an initial balance when creating their wallet.
- The application ensures that the balance cannot go negative, promoting responsible financial management.

3.3 Polymorphism

- The application utilizes polymorphism to allow the addition of funds using both double and integer types, enhancing flexibility in user input.

3.4 File Operations

- Users can save their wallet data to a text file, ensuring that their financial information is preserved between sessions.

4. Technical Implementation

4.1 Programming Language

- The application is developed in C++, leveraging its object-oriented features for better code organization and reusability.

4.2 Key Classes

4.2.1 DigitalWallet Class

- **Attributes:**
  - **`std::string ownerName`**: Stores the name of the wallet owner.
  - **`double balance`**: Represents the current balance in the wallet.
- **Methods:**
  - **Constructor**: Initializes the wallet with the owner's name and an initial balance.
  - **addMoney(double amount)**: Adds a specified amount to the wallet if it is positive.
  - **addMoney(int amount)**: Overloaded method to add an integer amount to the wallet.
  - **spendMoney(double amount)**: Deducts a specified amount from the wallet if sufficient balance exists.
  - **saveToFile(const std::string& filename)**: Saves the wallet's owner name and balance to a specified file.

4.2.2 Friend Function

- **displayWalletDetails**: A friend function that allows access to private members of the **`DigitalWallet`** class to display wallet information.

4.3 User Interaction

The main function serves as the entry point for the application, guiding users through the following steps:

1. **Creating a Wallet**: Users can create a new wallet instance by providing their name and an initial balance.
2. **Displaying Wallet Details**: The application displays the current wallet owner and balance.

3. **Adding Money**: Users can add money to their wallet using both double and integer values.
4. **Spending Money**: Users can spend money, with checks in place to prevent overspending.
5. **Saving Data**: The wallet's state can be saved to a text file for future reference.

4.4 User Feedback

The application provides console messages to inform users of successful operations or errors, enhancing the user experience.

5. Error Handling

The application incorporates robust error handling to ensure:

- Only positive amounts can be added or spent.
- Spending does not exceed the current balance.
- File operations handle potential errors, such as inability to open a file.

6. Data Persistence

The wallet data is saved to a text file (**wallet_data.txt**) in a simple format:

```
RunCopy code
1Owner: John Doe
2Balance: 190
```

This allows users to retrieve their wallet information in subsequent sessions.

7. Development Environment

7.1 Tools Used

- **C++ Compiler**: Clang is used for compiling the application.
- **IDE**: Visual Studio Code provides a user-friendly environment for development.

- **Debugger**: LLDB is utilized for debugging the application.

7.2 Configuration Files

- **settings.json**: Configures compiler paths, warning settings, and other preferences for the C++ runner.
- **c_cpp_properties.json**: Defines include paths and IntelliSense settings for code completion.
- **launch.json**: Configures the debugger settings for running the application.

8. Testing and Validation

8.1 Test Cases

The application is tested through various scenarios, including:

- Adding money (both double and integer).
- Spending money with sufficient and insufficient balance.
- Saving and retrieving wallet data.

8.2 Validation

User inputs are validated to ensure that only valid operations are performed, with appropriate error messages displayed for invalid actions.

9. Future Enhancements

The project can be expanded with additional features, such as:

- **User Authentication**: Implementing a secure login system to protect user data.
- **Integration with Online Payment Systems**: Allowing users to make online purchases directly from their wallet.
- **Mobile Application Version**: Developing a mobile-friendly version for greater accessibility.
- **Enhanced User Interface**: Creating a graphical user interface (GUI) for improved user experience.

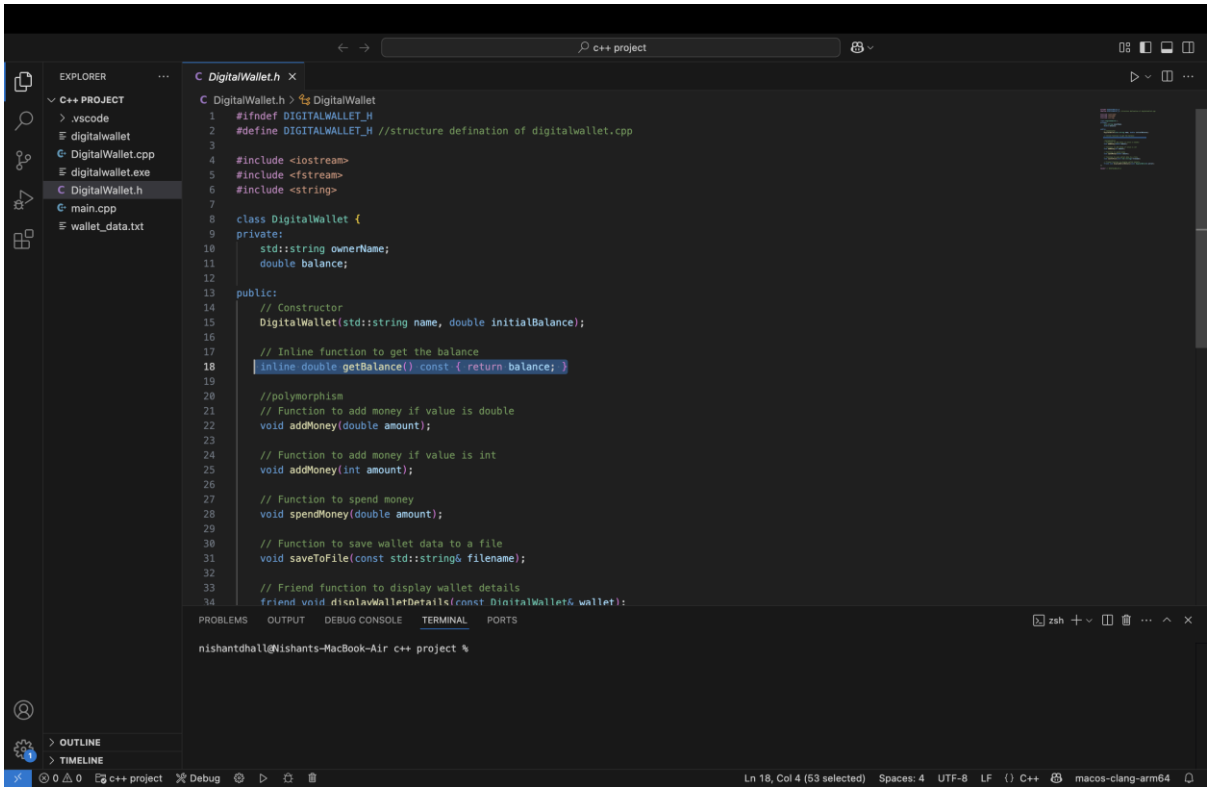
10. Conclusion

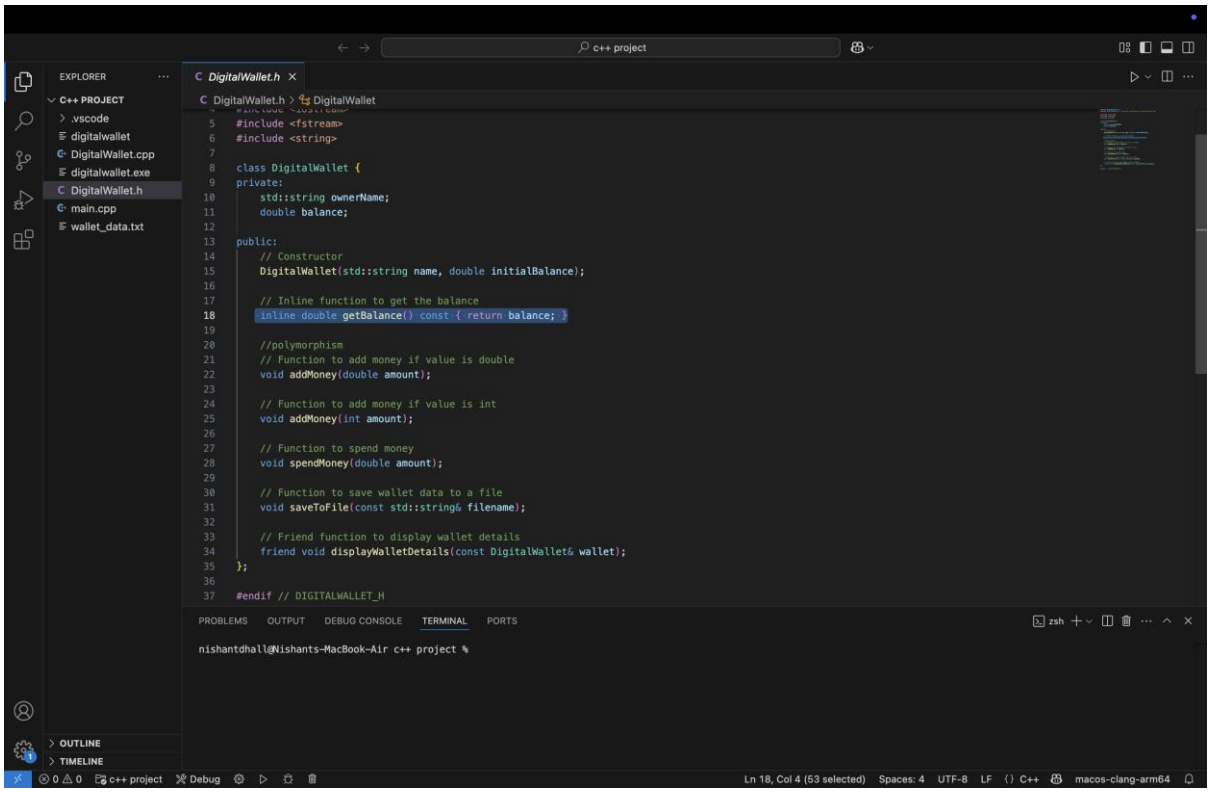
The Digital Wallet project successfully demonstrates the core functionalities of a digital wallet application using C++. It highlights the importance of financial management tools in the digital age and sets the foundation for future enhancements and developments in the FinTech space.

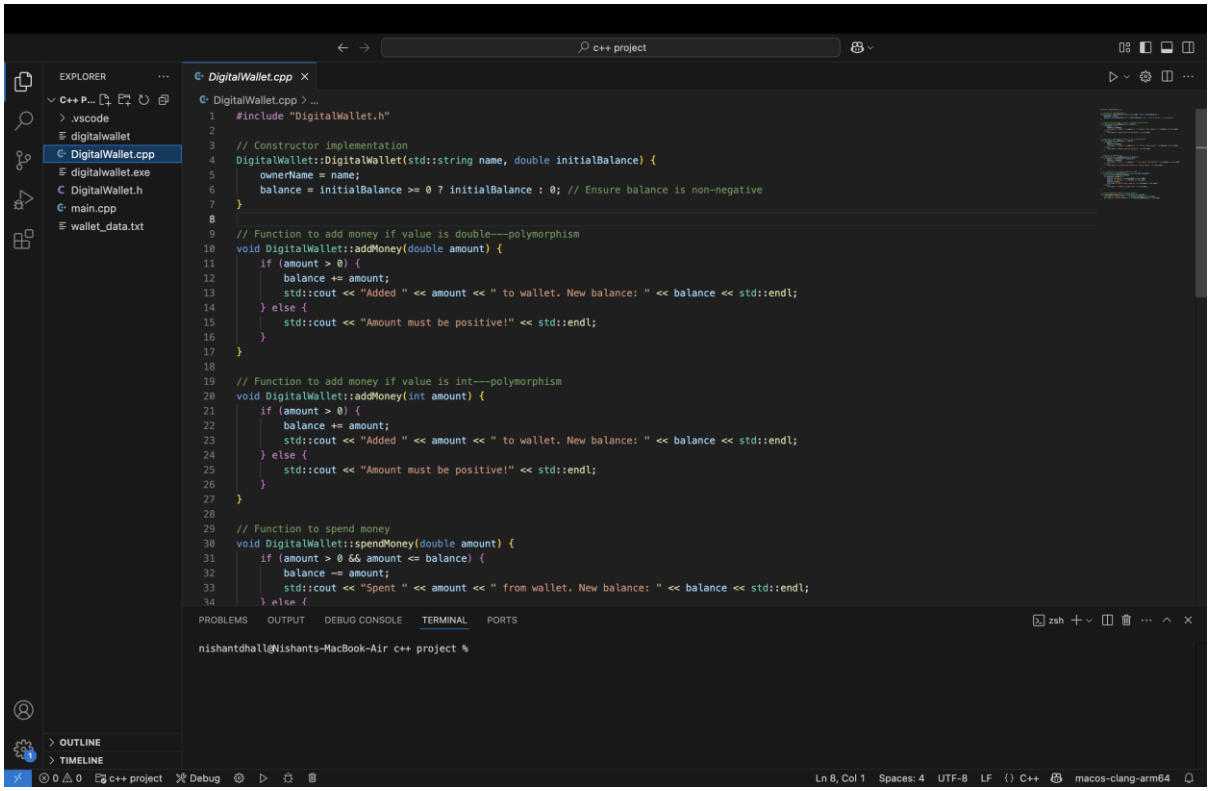
11. References

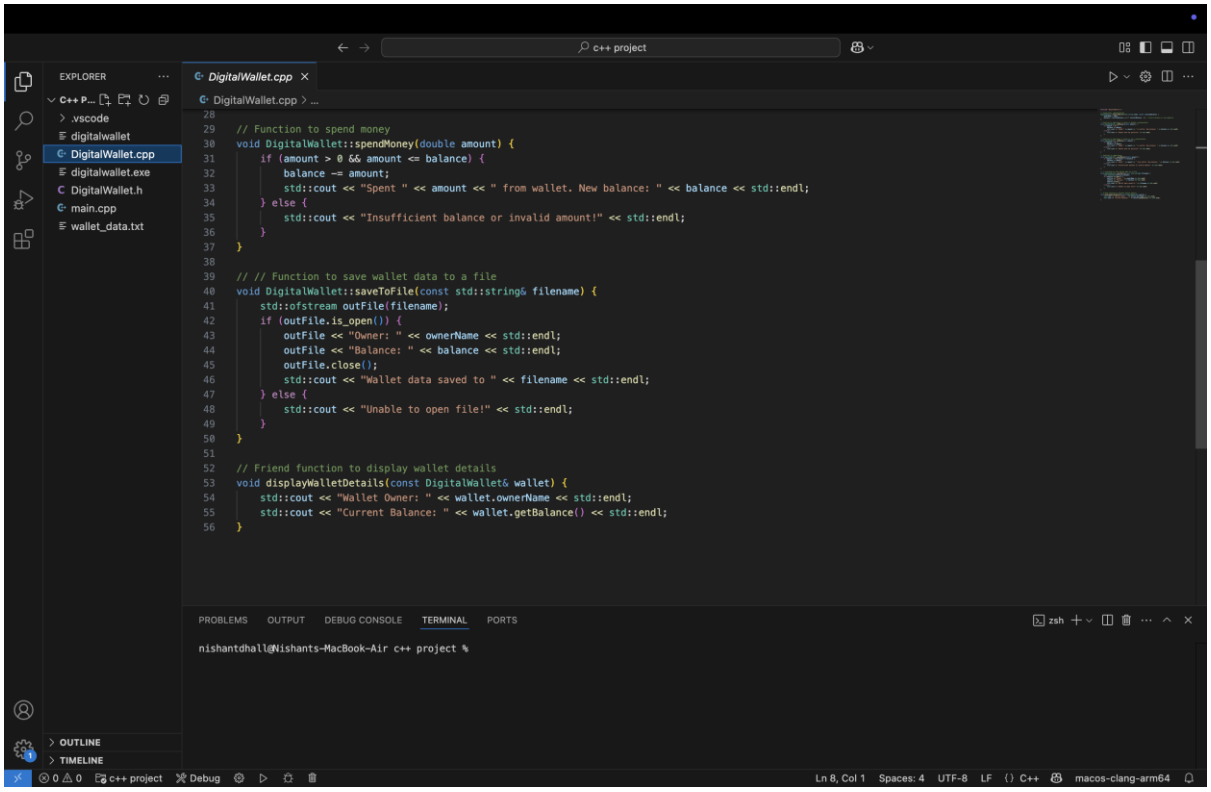
- C++ programming resources, including textbooks and online tutorials.
- Documentation for libraries used in file handling and console I/O operations.

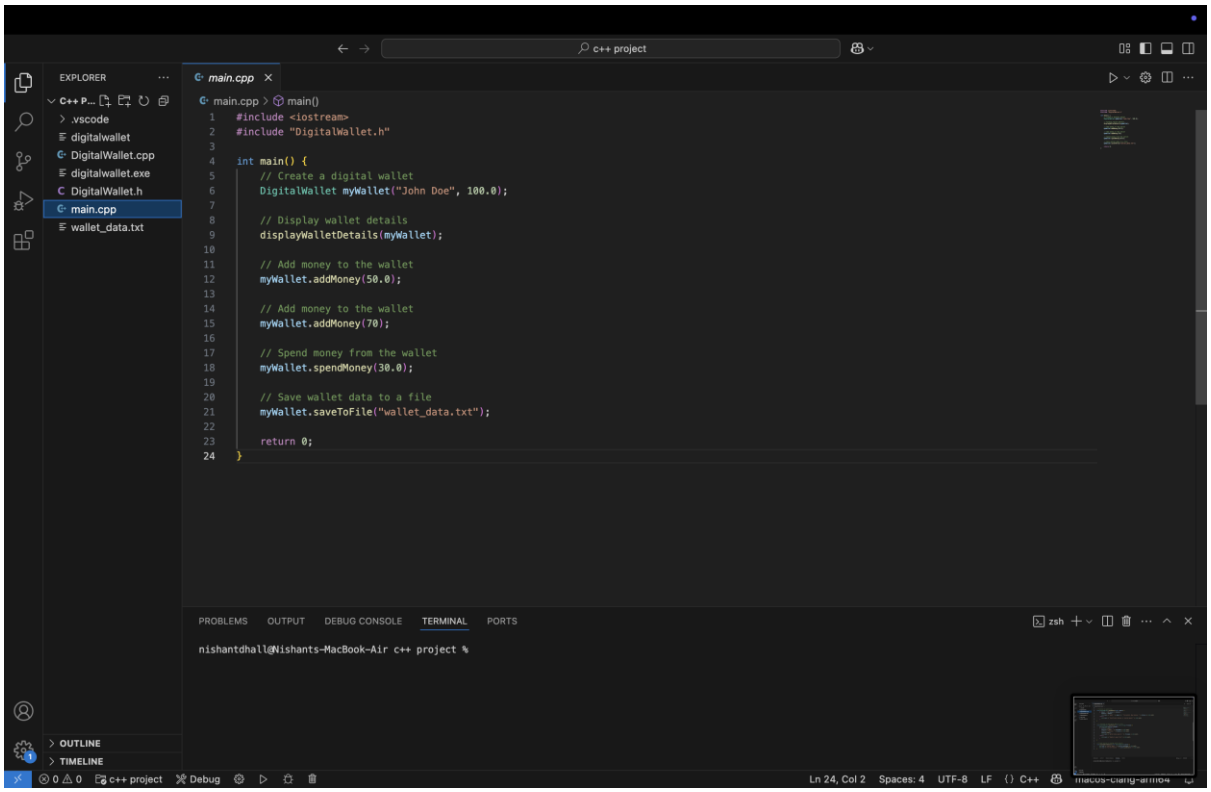
Code snippit:

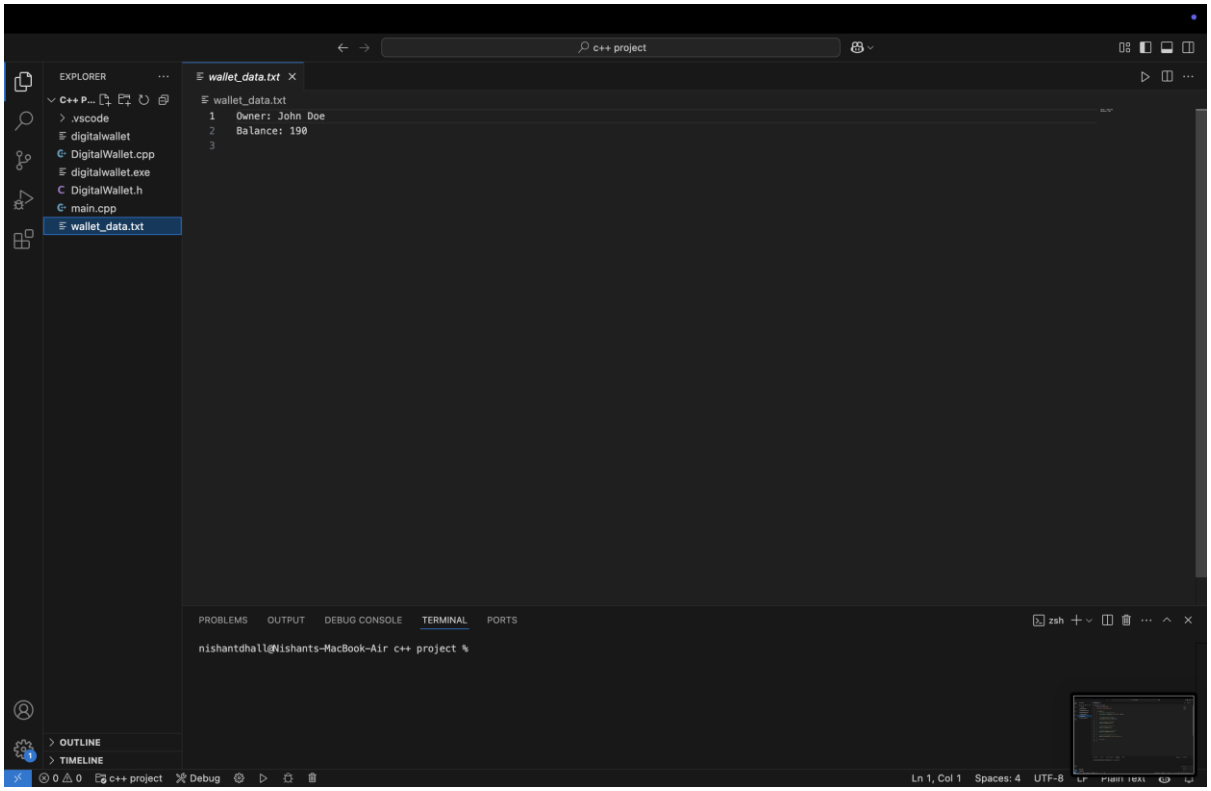

c++ project
EXPLORER
C++ PROJECT
.vscode
digitalwallet
DigitalWallet.cpp
digitalwallet.exe
DigitalWallet.h
main.cpp
wallet_data.txt
DigitalWallet.h
DigitalWallet.h > DigitalWallet
#ifndef DIGITALWALLET_H
#define DIGITALWALLET_H //structure defination of digitalwallet.cpp
#include <iostream>
#include <fstream>
#include <string>
class DigitalWallet {
private:
std::string ownerName;
double balance;
public:
// Constructor
DigitalWallet(std::string name, double initialBalance);
// Inline function to get the balance
inline double getBalance() const { return balance; }
//polymorphism
// Function to add money if value is double
void addMoney(double amount);
// Function to add money if value is int
void addMoney(int amount);
// Function to spend money
void spendMoney(double amount);
// Function to save wallet data to a file
void saveToFile(const std::string& filename);
// Friend function to display wallet details
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh
nishantdhall@Nishants-MacBook-Air c++ project %
OUTLINE
TIMELINE
Ln 18, Col 4 (53 selected)
Spaces: 4
UTF-8
LF
C++
macos-clang-arm64


c++ project
EXPLORER
C++ PROJECT
.vscode
digitalwallet
DigitalWallet.cpp
digitalwallet.exe
DigitalWallet.h
main.cpp
wallet_data.txt
DigitalWallet.h
DigitalWallet.h > DigitalWallet
#include <fstream>
#include <string>
class DigitalWallet {
private:
std::string ownerName;
double balance;
public:
// Constructor
DigitalWallet(std::string name, double initialBalance);
// Inline function to get the balance
inline double getBalance() const { return balance; }
//polymorphism
// Function to add money if value is double
void addMoney(double amount);
// Function to add money if value is int
void addMoney(int amount);
// Function to spend money
void spendMoney(double amount);
// Function to save wallet data to a file
void saveToFile(const std::string& filename);
// Friend function to display wallet details
friend void displayWalletDetails(const DigitalWallet& wallet);
};
#endif // DIGITALWALLET_H
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh
nishantdhall@Nishants-MacBook-Air c++ project %
OUTLINE
TIMELINE
Ln 18, Col 4 (53 selected)
Spaces: 4
UTF-8
LF
C++
macos-clang-arm64


c++ project
EXPLORER
C++ P...
.vscode
digitalwallet
DigitalWallet.cpp
digitalwallet.exe
DigitalWallet.h
main.cpp
wallet_data.txt
DigitalWallet.cpp
DigitalWallet.cpp > ...
#include "DigitalWallet.h"
// Constructor implementation
DigitalWallet::DigitalWallet(std::string name, double initialBalance) {
ownerName = name;
balance = initialBalance >= 0 ? initialBalance : 0; // Ensure balance is non-negative
}
// Function to add money if value is double---polymorphism
void DigitalWallet::addMoney(double amount) {
if (amount > 0) {
balance += amount;
std::cout << "Added " << amount << " to wallet. New balance: " << balance << std::endl;
} else {
std::cout << "Amount must be positive!" << std::endl;
}
}
// Function to add money if value is int---polymorphism
void DigitalWallet::addMoney(int amount) {
if (amount > 0) {
balance += amount;
std::cout << "Added " << amount << " to wallet. New balance: " << balance << std::endl;
} else {
std::cout << "Amount must be positive!" << std::endl;
}
}
// Function to spend money
void DigitalWallet::spendMoney(double amount) {
if (amount > 0 && amount <= balance) {
balance -= amount;
std::cout << "Spent " << amount << " from wallet. New balance: " << balance << std::endl;
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh
nishantdhall@Nishants-MacBook-Air c++ project %
OUTLINE
TIMELINE
Ln 8, Col 1
Spaces: 4
UTF-8
LF
C++
macos-clang-arm64


c++ project
EXPLORER
C++ P...
.vscode
digitalwallet
DigitalWallet.cpp
digitalwallet.exe
DigitalWallet.h
main.cpp
wallet_data.txt
DigitalWallet.cpp
DigitalWallet.cpp > ...
// Function to spend money
void DigitalWallet::spendMoney(double amount) {
if (amount > 0 && amount <= balance) {
balance -= amount;
std::cout << "Spent " << amount << " from wallet. New balance: " << balance << std::endl;
} else {
std::cout << "Insufficient balance or invalid amount!" << std::endl;
}
}
// // Function to save wallet data to a file
void DigitalWallet::saveToFile(const std::string& filename) {
std::ofstream outFile(filename);
if (outFile.is_open()) {
outFile << "Owner: " << ownerName << std::endl;
outFile << "Balance: " << balance << std::endl;
outFile.close();
std::cout << "Wallet data saved to " << filename << std::endl;
} else {
std::cout << "Unable to open file!" << std::endl;
}
}
// Friend function to display wallet details
void displayWalletDetails(const DigitalWallet& wallet) {
std::cout << "Wallet Owner: " << wallet.ownerName << std::endl;
std::cout << "Current Balance: " << wallet.getBalance() << std::endl;
}
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh
nishantdhall@Nishants-MacBook-Air c++ project %
OUTLINE
TIMELINE
Ln 8, Col 1
Spaces: 4
UTF-8
LF
C++
macos-clang-arm64


c++ project
EXPLORER
C++ P...
.vscode
digitalwallet
DigitalWallet.cpp
digitalwallet.exe
DigitalWallet.h
main.cpp
wallet_data.txt
main.cpp
main.cpp > main()
#include <iostream>
#include "DigitalWallet.h"
int main() {
// Create a digital wallet
DigitalWallet myWallet("John Doe", 100.0);
// Display wallet details
displayWalletDetails(myWallet);
// Add money to the wallet
myWallet.addMoney(50.0);
// Add money to the wallet
myWallet.addMoney(70);
// Spend money from the wallet
myWallet.spendMoney(30.0);
// Save wallet data to a file
myWallet.saveToFile("wallet_data.txt");
return 0;
}
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh
nishantdhall@Nishants-MacBook-Air c++ project %
OUTLINE
TIMELINE
0 0
c++ project
Debug
Ln 24, Col 2
Spaces: 4
UTF-8
LF
C++


c++ project
EXPLORER
C++ P...
.vscode
digitalwallet
DigitalWallet.cpp
digitalwallet.exe
DigitalWallet.h
main.cpp
wallet_data.txt
wallet_data.txt
wallet_data.txt
Owner: John Doe
Balance: 190
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
zsh
nishantdhall@Nishants-MacBook-Air c++ project %
OUTLINE
TIMELINE
0 0
c++ project
Debug
Ln 1, Col 1
Spaces: 4
UTF-8