**Bank Console Application**

**Introduction**

The Bank Console Application is a simple Python console-based application that allows users to manage their accounts, view balances, send money, and perform other basic banking operations. The application uses an Excel file to store account information and transaction history.

**Components**

The application consists of four main components:

1. **bank\_app.py**: Contains the **BankApp** class, which handles the core functionality of the banking application.
2. **console\_ui.py**: Contains functions for displaying the console user interface menus and prompts.
3. **excel\_storage.py**: Contains the **ExcelStorage** class, which handles loading and saving account data to an Excel file.
4. **main.py**: Contains the main execution loop and integrates the other components together.

**Features**

The application provides the following features:

1. Create Account: Users can create a new account with a username and password. The initial balance is set to 1000.
2. Update Password: Logged-in users can update their account password by providing the old password and a new one.
3. Login Account: Users can log in to their account using their username and password.
4. View Balance: Logged-in users can view their account balance.
5. Send Money to Other Account: Logged-in users can send money to another account, provided the recipient exists and there is sufficient balance.
6. Logout: Logged-in users can log out of their account.

**Usage**

To use the Bank Console Application, follow these steps:

1. Save the contents into their respective files: **bank\_app.py**, **console\_ui.py**, **excel\_storage.py**, and **main.py**.
2. Make sure all four files are in the same directory.
3. Run the **main.py** file from the command line:

cssCopy code

python main.py

1. The application will start, displaying the menu options based on the user's login status.
2. Interact with the console to create accounts, log in, view balances, send money, and more.

**Documentation**

**BankApp Class**

**BankApp** class is responsible for managing user accounts and performing banking operations. The following methods are available:

* **\_\_init\_\_(self, excel\_filename)**: Initializes the BankApp instance with an Excel file for account storage.
* **save\_accounts(self)**: Saves the accounts to the Excel file using the **ExcelStorage** object.
* **create\_account(self, username, password)**: Creates a new account with a given username and password, setting the initial balance to 1000.
* **update\_password(self, old\_password, new\_password)**: Updates the password for the logged-in user, provided the old password is correct.
* **login\_account(self, username, password)**: Logs in to an account with the provided username and password.
* **view\_balance(self)**: Displays the balance for the currently logged-in account.
* **send\_money(self, to\_username, amount)**: Sends money from the currently logged-in account to another account if the recipient and balance are valid.
* **logout(self)**: Logs out of the currently logged-in account.

**Console UI Functions**

**console\_ui.py** contains functions for displaying the console user interface menus and prompts.

* **display\_menu\_logged\_out()**: Displays the menu options when the user is not logged in.
* **display\_menu\_logged\_in()**: Displays the menu options when the user is logged in.

**ExcelStorage Class**

**ExcelStorage** class is responsible for loading and saving account data to an Excel file.

* **\_\_init\_\_(self, filename)**: Initializes the **ExcelStorage** object with the given Excel filename.
* **load\_accounts(self)**: Loads the accounts from the Excel file if it exists, or returns an empty dictionary if not.
* **save\_accounts(self, accounts)**: Saves the accounts to the Excel file using a DataFrame.

**Main Execution Loop**

**main.py** contains the main execution loop, which integrates the other components (**bank\_app.py**, **console\_ui.py**, and **excel\_storage.py**) together.

* The **main()** function contains the main execution loop that displays the appropriate menu based on the user's login status, takes user input, and calls the corresponding methods of the **BankApp** class.
* The **if \_\_name\_\_ == "\_\_main\_\_":** block ensures that the **main** function is called when the script is run directly.

**Dependencies**

The application requires the following Python libraries:

* pandas: Used for manipulating account data and saving it to an Excel file.

**Limitations**

* The application does not support advanced features such as interest calculation, loans, or investment options.
* The application is not designed for high concurrency or large-scale use.
* There is no encryption or secure storage for user passwords, making it unsuitable for real-world use.