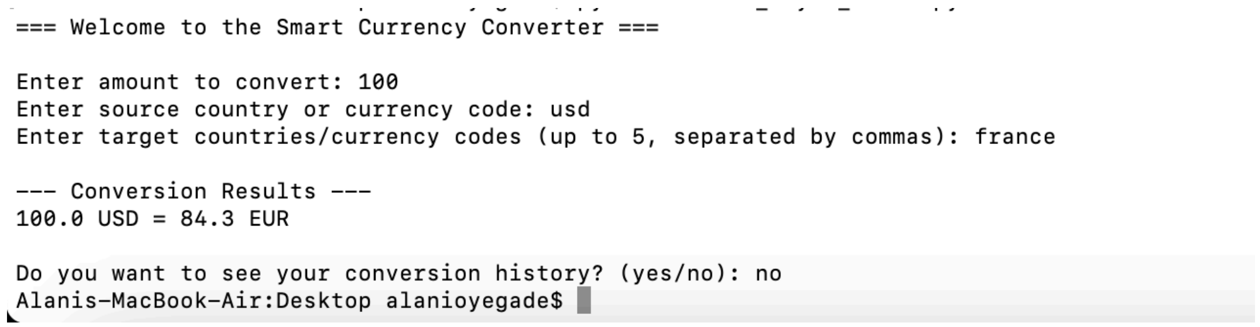
Smart Currency Converter Module 3

The Smart Currency Converter is a command-line based application that allows users to convert money from one currency to another using live exchange rates. The program is designed to be simple, accurate, and user-friendly. Users can enter an amount, choose a source country or currency code, and select one or more target currencies to convert into. The system retrieves real-time exchange rate data from an online API to ensure accurate conversions. The program also validates user input to prevent errors such as negative numbers, invalid text entries, or unsupported currency codes. If an error occurs, the program displays a clear message and allows the user to try again without crashing.

This application demonstrates proper variable management, uses meaningful variable names, and applies appropriate data types such as floats, strings, lists, and dictionaries. The system is structured clearly to maintain readability and follow best coding practices.

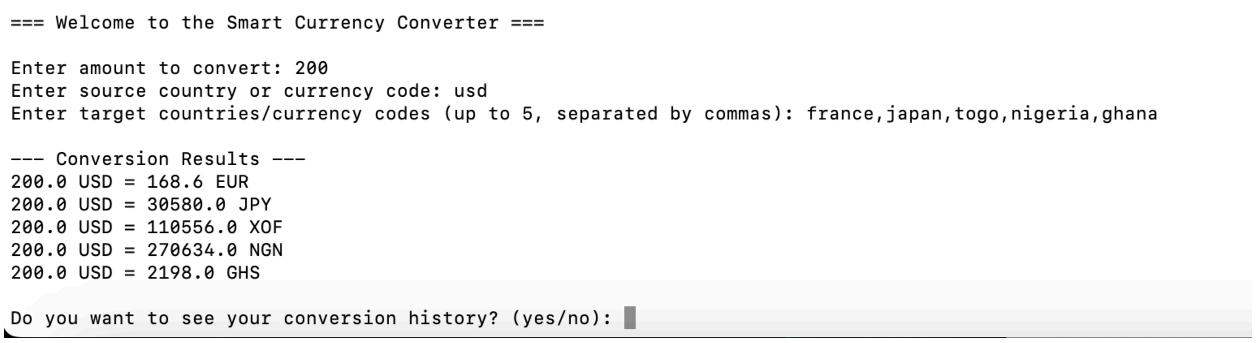
These are the Screenshots showing the program meeting all functional requirements:

Valid Conversion:



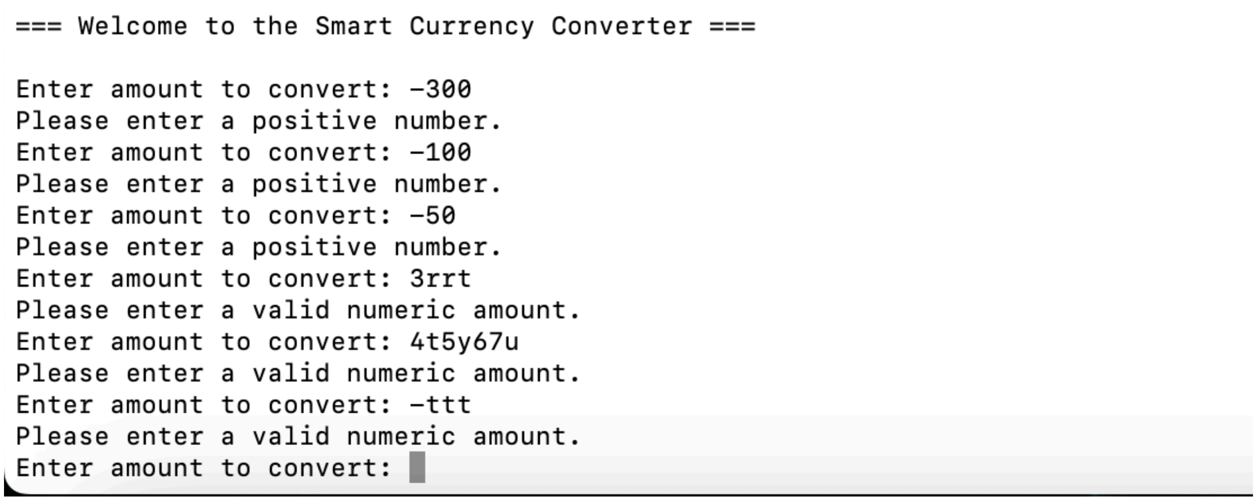
```
=== Welcome to the Smart Currency Converter ===

Enter amount to convert: 100
Enter source country or currency code: usd
Enter target countries/currency codes (up to 5, separated by commas): france

--- Conversion Results ---
100.0 USD = 84.3 EUR

Do you want to see your conversion history? (yes/no): no
Alanis-MacBook-Air:Desktop alanioyegade$ 
```

Multiple Currency Conversions:



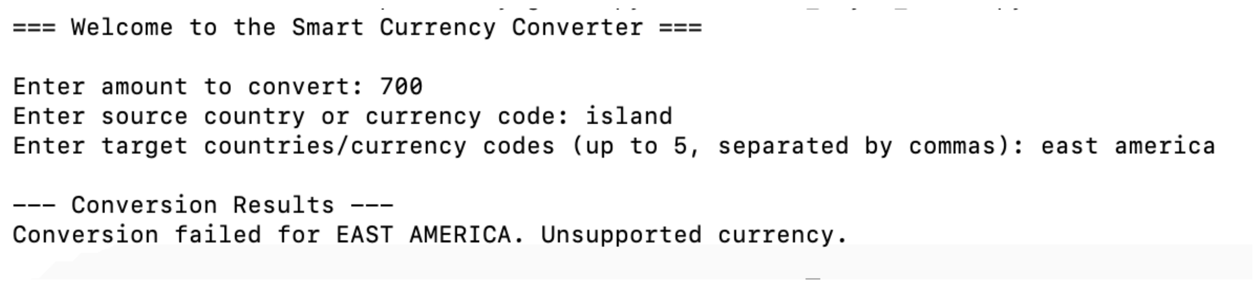
```
=== Welcome to the Smart Currency Converter ===

Enter amount to convert: 200
Enter source country or currency code: usd
Enter target countries/currency codes (up to 5, separated by commas): france,japan,togo,nigeria,ghana

--- Conversion Results ---
200.0 USD = 168.6 EUR
200.0 USD = 30580.0 JPY
200.0 USD = 110556.0 XOF
200.0 USD = 270634.0 NGN
200.0 USD = 2198.0 GHS

Do you want to see your conversion history? (yes/no):
```

Input Validation:



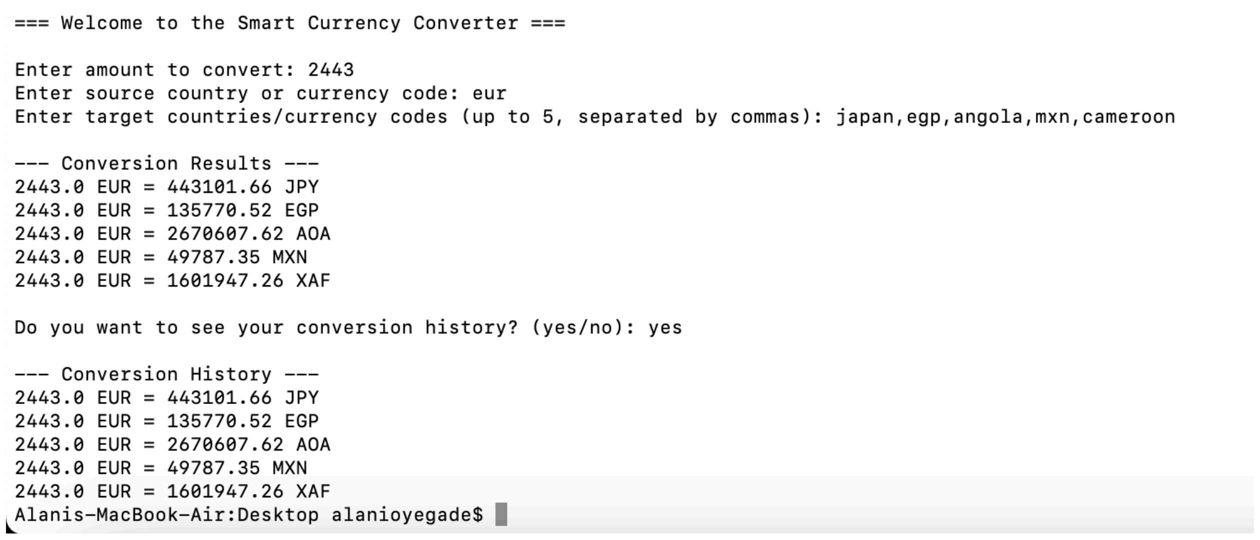
```
=== Welcome to the Smart Currency Converter ===

Enter amount to convert: -300
Please enter a positive number.
Enter amount to convert: -100
Please enter a positive number.
Enter amount to convert: -50
Please enter a positive number.
Enter amount to convert: 3rrt
Please enter a valid numeric amount.
Enter amount to convert: 4t5y67u
Please enter a valid numeric amount.
Enter amount to convert: -ttt
Please enter a valid numeric amount.
Enter amount to convert:
```

Unsupported Currency:



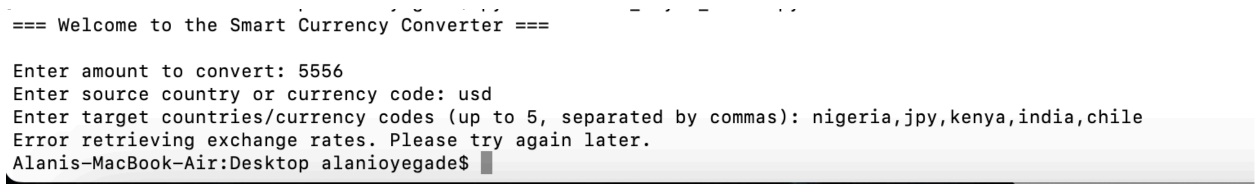
```
=== Welcome to the Smart Currency Converter ===

Enter amount to convert: 700
Enter source country or currency code: island
Enter target countries/currency codes (up to 5, separated by commas): east america

--- Conversion Results ---
Conversion failed for EAST AMERICA. Unsupported currency.
```

Session History Feature:



```
=== Welcome to the Smart Currency Converter ===

Enter amount to convert: 2443
Enter source country or currency code: eur
Enter target countries/currency codes (up to 5, separated by commas): japan,egp,angola,mxn,cameroon

--- Conversion Results ---
2443.0 EUR = 443101.66 JPY
2443.0 EUR = 135770.52 EGP
2443.0 EUR = 2670607.62 AOA
2443.0 EUR = 49787.35 MXN
2443.0 EUR = 1601947.26 XAF

Do you want to see your conversion history? (yes/no): yes

--- Conversion History ---
2443.0 EUR = 443101.66 JPY
2443.0 EUR = 135770.52 EGP
2443.0 EUR = 2670607.62 AOA
2443.0 EUR = 49787.35 MXN
2443.0 EUR = 1601947.26 XAF
Alanis-MacBook-Air:Desktop alanioyegade$
```

API Error Handling:



```
=== Welcome to the Smart Currency Converter ===

Enter amount to convert: 5556
Enter source country or currency code: usd
Enter target countries/currency codes (up to 5, separated by commas): nigeria,jpy,kenya,india,chile
Error retrieving exchange rates. Please try again later.
Alanis-MacBook-Air:Desktop alanioyegade$
```

Updated Documentation:

The documentation for the Smart Currency Converter continues to provide all the information needed for both users and developers to understand and use the program successfully.

User Manual:

The program is run in the terminal or interpreter. Users are prompted to enter the amount of money they want to convert, followed by the source country or currency code. Users can then enter up to five target countries or currency codes at the same time. The program will fetch live exchange rates from an online API and display the converted amounts clearly for each target currency.

The program also keeps a (session conversion history), which allows users to review all conversions performed during that run. Users can choose whether or not to view the history. The program handles errors gracefully, such as negative amounts, invalid numeric input, unsupported country names, or invalid currency codes, and provides clear messages to guide the user in correcting any issues.

Technical Guide:

The program is organized into **several main components**:

1. **Input and Validation Module:**
    - Collects user input for the amount, source currency, and target currencies.
    - Ensures the amount is numeric and positive.
    - Converts country names to currency codes when necessary.

2. **Conversion Engine:**
    - Implemented as a custom class “CurrencyConverter”, which represents an unusual data type.
    - Handles fetching live exchange rates from the API.
    - Performs currency conversions and manages session history.

3. **Output Module:**
    - Displays conversion results for each target currency.
    - Optionally displays a session history of all conversions.

The program uses meaningful variable names and clear, structured code with comments explaining each step. Error handling is included at all stages to prevent crashes and ensure data integrity.

Future Expansion:

The program can be extended to support additional countries, save history between sessions, or include a graphical user interface. The modular design makes these future improvements straightforward.

Appendix – Example Scenarios:

1. User converts 100 USD to NGN, ZAR, and EUR: program fetches live rates and displays results for all three currencies.
2. User enters a negative amount: program displays an error and prompts for a positive number.
3. User enters an unsupported currency code: program informs the user and skips that conversion.
4. The user chooses to view session history: program displays all conversions made in the current session in an organized list.