Who Did What

Changxing Wang came up with the idea of making a program that can help users to split bills wisely with others. Then, all three of the members wrote the algorithms or the whole program, the basic structure of the program and the functions the group needs to write.

Jiatian wrote the main menu function, which is an interface that allows a user to interact with the program. The main menu function also let other functions run. Jiatian also wrote the file functions that save splitmate names and transactions as well as loading them. She came up with the idea of exporting the transaction history in a csv file. Finally, Jiatian was also responsible for the poster design.

Zequn made the inform prompt. He wrote the api of currency part. The api support several kinds of currency. The function to check the status of currency rate was also written. Zequn wrote the Confirmation box function which could be used for several times in the program. He wrote the ‘view transaction history’ function to allow the user to view the transaction the history.

Changxing Wang wrote the function to borrow and lend a certain amount of money among users and the paying system. It properly recorded the amount which person should pay to which. And how their debt should be paid or balanced. In the function, the maximum amount of money a person could pay cannot exceed the amount he owes. He also wrote a function to check the payee for a specific user.

To the bug fixing, all three members of the group have participated in the troubleshootings and improvements activities. Major bugs have been fixed.

Code Explanation

TigaSplit has 6 functions: add a bill, make a payment, add a splitmate, view summary, view transaction history, and export transactions to CSV.

**Main Menu**

**Input: In the main menu, user need to enter numbers matching the functions to operate the program.**

The main menu of TigaSplit is the function main\_menu(), which is called when a user is trying to run the program. The function can call different other functions depend on user input. For example, if user input=1, the add\_bill() function will be called.

**Output: the coordinated functions**

**Add a Bill**

**Input: Lender name, Borrower name, Bill information, description**

When the user makes this choice, the add\_bill() function will be called. Before adding a bill, the function will check if there are any split mate names. This progress involve two functions, add\_splitmate() and select\_splitmate(). If there is no existed name in the global list “splitmates”, the add\_splitmate() function will be called (explained below). The select\_splitmate() function will check if there are any existed names in the program and let the user select who are the borrowers and lenders.

Add\_bill() will then call the check\_currency\_exchange\_rates\_status() function, which checks if the exchange rate api can work. If confirmed, the check currency function will then call the load\_currency\_exchange\_rates\_from\_api() function, which can request and use the exchange rate api to convert other currencies to USD. Finally, the add\_bill() function will print the result user entered, including the borrower, lender, amount and description.

**Output: Confirmation of the bill, the bill information will be written into the file.**

**Add a Splitmate**

**Input: Users name**

If the user makes this choice, the add\_splitmate() function will be called. (This function can also be called when the user chooses to add a bill). The add\_splitmate() function asks user to input a splitmate name for both lenders and borrowers. If the new name duplicates, the program will ask the user to try again. The new name entered will then be written in a file called “Splitmate.txt”, which is opened and written by the function save\_splitmate(). There is also a load\_splitmate() function that is used to load the names saved in the file, so when the user reopens the program, he or she can still find the names entered. Then, the select\_splitmate() function will check and provide users the names to select.

**Output: Confirmation of the name which indicate it has been added.**

**Make a Payment**

**Input: payer selection, payee selection, payment amount.**

In the “make a payment” function. The program starts by choosing the payer and payee with select\_splitmate function. The program will first tell the user how much the payer owe to the payee. During the process of the payment, the program will ask user the amount of the payment and then the program uses the number the user entered to edit the status of the splitmates. After entering the information, the user will get he information box of the transaction details. Finally, the user can choose Yes or No to admit the payment or cancel it.

**Output: Confirmation of the payment which indicate it has been added.**

**View Summary**

**Input: user selection (user enter ‘1’)**

The View Summary function is for view the status of every splitmates’ debt situations. Like A owes B 100 USD. In this function, it will call another function generate\_summary which is really doing the work to calculate the debt status. In the generate\_summary, the program use several for loop and a lot of ‘if’ to calculate who is the final borrower and who is the final lender, then get the final summary.

**Output: The summary of bills.**

**View Transaction History**

**Input: user select (user enter ‘1’) to show the information.**

In this function, the program will read the data from the transactions file and arrange them in clear format. We get the length of the longest sentence to make the the form clear and be in order. Finally, the form of transaction history will show up clearly.

**Output: The transaction history**

**Check Currency exchange rates status**

**Input: the 3-letter currency code user entered.**

This function is for check the avaliability of the Currency api. It will call load\_currency\_exchange\_rates\_from\_api() function and try to access to the api. If the Api is not working( maybe no internet connection), the program will handle the error and skip this step, using USD in all transactions in the program.

**Output: currency exchange rate and the program will convert them automatically according to the currency code.**

**Inform and Confirm functions.**

**Input: Confirmation (Usually “Enter”, “Y(Yes)”) from the user or Cancellation (“N(No)”)**

The function use lines to “draw” simple boxes to become the inform and confirm boxes. This make the program more beautiful and clearer to use.

For the confirm function, the user could enter Y(es) to confirm the process or enter N(o) to withdraw it.

**Output: Confirmation of information from the program or starting point of the specific function for user to reenter the information.**

**Export transaction CSV**

**Input: Confirmation (Usually “Enter”, “Y(Yes)”) from the user or Cancellation (“N(No)”)**

This function is for exporting the transaction history to a CSV file which could be used to build charts or just for data storage. The CSV file will be also readable which can just easily to read by people.

**Output: a CSV file with transaction history.**

**View Transaction in Chart**

**Input:** name of the user file

The user will first enter the name of the file without CSV extension. Then, the function view\_chart() will be called in the main\_menu() function. The view\_chart() function will use matplotlib to draw a chart that shows the amount and description. The payment will usually be negative. If the user wants to go back to the main menu, he or she only needs to simply press “ENTER” and the main\_menu() function will be called again.

**Output**: A chart that shows transaction history

**Email User the Transaction**

**Input: Users Email address.**

In this function I use **smtplib** api — it is a SMTP protocol client. Source code: Lib/**smtplib**.py. The **smtplib** module defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon.

In this program I use outlook365 school email as the host of this service. The email was sent from my email address and will be automatically send to the user.User will be allowed to type in their email address. A message with the information in transaction.txt will be sent to the users. The attributes (transaction titles, From, To, etc.) is editable. And the sender’s email address could be modified to any email addresses. The sender also has to provide credentials for this service. At the end of the function the program will quit smtp service and back to the main menu.

**Output: User’s automated email messages.**