Second Iteration

Team: duckduckrush
Kangwei Ling kl3076
Zefeng Liu zl2715
Kunyan Han kh2931
Xi Yang xy2390

Codebase github repository: https://github.com/kevinkwl/flowey (branch: master, ios, test-reports)

Use cases:
As a user, I want to **register** an account and then use this account to **log in** so that I can use this application.

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
Registration	A user registers a new account for Flowey	User, Flowey App, Flowey server	Device connected to network; User not logged in;	Credentials for the new account stored in the server database

Flow Category	Serial No.	Steps
Main	1	User enter email User enter password User click “Register” button
	2	Flowey server check if email already taken
	3	Flowey server stores the new account’s credentials in database
	4	Registration completed
Exceptions	2a	Taken email Flowey server returns error message to Flowey App Flowey App shows an error message

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
Login	A user logs in to Flowey	User, Flowey App, Flowey server	Device connected to network; User not logged in;	User logged in to Flowey

Flow Category	Serial No.	Steps
Main	1	User enter email User enter password User click “Login” button
	2	Flowey server validate credentials
	3	Flowey server create token and return token to client Flowey App accepts the token and logs the user in
	4	Login completed
Exceptions	2a	Credentials invalid Flowey server returns error message to Flowey App Flowey App shows an error message

As a user, I want to **add transactions** for my personal bills so that I can **keep track of my personal expenses**. These transactions must be **viewable** in some place like a dashboard in the application.

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
Add transaction	A user adds a transaction	User, Flowey App, Flowey server	Device connected to network; User logged in;	New transaction added

Flow Category	Serial No.	Steps
Main	1	User clicks “+” button on “Expenses” scene
	2	Flowey App shows transaction editing scene
	3	User enters transaction amount

		User selects transaction category User selects transaction date User click “Done” button
	4	Flowey server adds the new transaction to database
	5	Flowey App shows the success message Flowey App redirects to the transactions view scene
Exception	3a	User selects a future date Flowey App changes the date back to current date

As a user, I want to be able to add bills that should be **paid by other users or by shared payment**. Bills shared with others need to **be approved** before becoming effective.

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
Split Bills	A user adds bills that should be paid by other users or by shared payment	User, Object users Flowey App, Flowey server	Device connected to network; User logged in; Object users are friends of User; User in “Expenses” scene	New transaction added in all the related users, with correct number of transactions and correct money amount

Flow Category	Serial No.	Steps
Main	1	User clicks “+” button on “Expenses” scene
	2	Flowey App shows transaction editing scene
	3	User enters transaction amount User selects transaction category User selects transaction date
	4	User clicks on the “split with friends” checkbox User clicks on “select friends” User select N friends he/she wants to share bills with
	5	Flowey server adds the new transactions to database, including an ordinary transaction and N-1 “lend” transactions for the issuer, an ordinary transaction and 1 “borrow” transaction for each of other users

	6	Flower server adds the flow relation between users and show them in the “friends” scene
	7	Flowey App shows the success message Flowey App redirects to the “Expenses” scene
Exception	3a	User selects a future date Flowey App changes the date back to current date

As a user, I want to be able to use a friend panel to **send friend request** to other users and **view debt status** between me and my friends. Other users have the right to reject the friend request and stay unconnected. With friends, I want to be able to add bills that should be **paid by other users or by shared payment**. Bills shared with others need to **be approved** before becoming effective.

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
Send friend request	A user sends friend request to another user	User A, User B, Flowey App, Flowey server	Device connected to network; User A logged in; User in “Friends” scene	Friend request sent

Flow Category	Serial No.	Steps
Main	1	User clicks “+” button on “Friends” scene
	2	Flowey App shows friend request scene
	3	User enters a user B’s email
	4	User clicks “Request” button
	5	Flowey server adds the new request to the database
	6	Flowey App redirects to the “Friends” scene
Exception	5a	Account with the entered email not exist Flowey server shows error message
	5b	Account with the entered email already added as friend Flowey server returns message

		Flowey app redirects to the “Friends” scene

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
Accept friend request	A user accepts friend request from another user	User A, User B, Flowey App, Flowey server	Device connected to network; User B logged in; User B clicked on Notification tab	Friend relationship established

Flow Category	Serial No.	Steps
Main	1	User B clicked on accept button of the friend request
	2	The friend request disappear
	3	User B enters friends tab
	4	Flowey App shows the added friend
	5	User A logged in
	6	User A clicked on Friends tab
	7	Flowey App shows friends of A, which contains B

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
Reject friend request	A user rejects friend request from another user	User A, User B, Flowey App, Flowey server	Device connected to network; User B logged in; User B clicked on Notification tab	Friend relationship established

Flow Category	Serial No.	Steps

Main	1	User B do “swipe to left” on the friend request
	2	User B click the reject button
	3	The friend request disappear
	4	User enters friends tab
	5	Flowey App does not show the rejected friend
	6	User A logged in
	7	User A clicked on Friends tab
	8	Flowey App does not show the rejected friend

Use Case Title	Description	Actor(s)	Pre-condition	Post-condition
View debt status	A user checks debt status	User, Flowey App, Flowey server	Device connected to network; User logged in;	User viewed up-to-date debt status

Flow Category	Serial No.	Steps
Main	1	User clicks “Friends” button
	2	Flowey App shows “Friends” scene
	3	Flowey server queries debt status for user
	4	Flowey App shows debt status with respect to each friend
	5	User viewed up-to-date debt status

Equivalence Partitions:

AuthenticationTestCase

1. Register
    a) Register with correct information (email address, password, username).
    b) Register with invalid information, for example, invalid email address, retype wrong

password, and etc.
   c) Register with correct information, but not valid in database. For example, register with an email that has already in the database.

2. Login
   a) Login in with correct email and password, i.e., login in successfully.
   b) Login in with incorrect pairs, for example, the password is wrong for a given email address.

3. Logout
   a) Login out successfully and return to the logining view.
   b) Some error occurs when logging out.

TransactionTestCase

1. Add transactions
   a) Add a valid transaction, for example, borrow, lend, or return money from / to another user, correctly.
   b) Add an invalid transaction, such as, the inputted date is in valid, and so on.

2. Split with friends
   a) Correctly enter information and select friends to split the bill.
   b) Invalid information about splitting the bill, for instance, invalid category, invalid date, and etc.

3. Delete an existing transaction
   a) Successfully delete an existing transaction.
   b) Delete a transaction that is not allowed to do so.

FriendsTestCase

1. New user states
   a) Create a new user, it should have no friends, no friends request
   b) The new user has inconsistent initial data

2. Send friend request
   a) Send friend request to a user that has not already been friend with this user
   b) Send friend request to a user that has already sent request to
   c) Send friend request to non exist user

3. Respond to friend request
   a) Accept friend request from another user
   b) Reject friend request from another user

Test Suite Coverage:

Coverage.py is a tool for measuring code coverage of Python programs. We integrated it with our Flask App so that our test coverage report can be easily obtained using this command,

```
flask test --coverage
```

Every time we run our unit tests in our pre-commit and post-commit processes, this command will be run. The result is stored in the file test_with_coverage.txt in the "test report" branch of our repo. We are still adding more tests to improve the coverage of our testing. The current coverage is around 40%.