Second iteration

Team Name:
TeamOfFour

Team Member:
Jialin Zhao (jz2862)
Linlin Tian (lt2667)
Xi Chen (xc2403)
Zhou Xingchen (xz2721)

https://github.com/manMadeLogic/YourChef/

User case:

1.
Title: Registration
Actor: customer
Description: As a customer, I want to **sign up**/**sign in** using my username/passcode and build personal taste profile so that my information can be saved for future.
Conditions of satisfaction: I can successfully sign up, create username and password, create the user profile and log in.
Basic flow:
A user registers a username with an email address which has not been used: store the user information form in our database.
A user logs in using an already sign-up username with correct passcode: alert success information, give the user account access to the resources according to the user group and redirect to the restaurant list page.
Alternate flows:
A user registers using an email address or username that has already been used: alert an error message and redirect to the login page.
A user registers using a username that contains special characters: alert an error message and redirect to the register page.
A user logs in using a username that does not exist in our database: alert an error message and redirect to the login page.
A user logs in using a wrong password that does not match the username: alert an error message and redirect to the login page.

2.
Titile: Restaurant menu
Actor: Customer

Description: As a customer, I want to get **restaurants menu** so that I can see the dishes to order.
Conditions of satisfaction: Once I open a restaurant page I want to see the up-to-date menu.
Basic flow:
A user opens a page of a valid restaurant: redirect to the menu page of the restaurant and show the menu contents.
Alternate flows:
A user opens a page of an un-opened restaurant name: alert an error message and reload the page.

3.
Title: Add to cart
Actor: customer
Description: As a customer, I want to add dishes to cart and see what I put in the cart, limited to 10 dishes.
Conditions of satisfaction: When I click add, the dishes are added in the cart.
After refresh, the cart will show dishes I put in the cart.
Basic flow:
A user clicks the 'x' button after the dish: add it to the cart.
A user refreshes the page and sees what's in the cart: redirect to the cart list page with dish list user created.
Alternate flows:
Add dish when there are already 10 dishes in the cart: cart stays the same and a browser alert.

4.
Title: Registration
Actor: restaurant manager
Description: As a restaurant, I also want to sign up/sign in and build up my **restaurant profile** so that it can be viewed by customers.
Conditions of satisfaction: I can successfully sign up, create username and password, create the restaurant profile and log in.
Acceptance:
A restaurant manager who has already posted their information can look at their information and can edit. And also, customers are also able to see their posted restaurant information online.

Basic flow:
A user registers a username with an email address which has not been used: store the user information form in our database. Then redirect to the page requiring restaurant name. After fill into the restaurant name which has not been used: stored the automatically-searched address info in the database.
A user logs in using an already sign-up username with correct passcode: alert success information, give the user account access to the resources according to the user group and redirect to the restaurant list page.
Alternate flows:
A user registers using an email address or username that has already been used: alert an error message and redirect to the login page.
A user registers using a username that contains special characters: alert an error message and redirect to the register page.
A user registers using a restaurant name that has no matched address: alert an error and redirect to the page requiring manually input the address.
A user logs in using a username that does not exist in our database: alert an error message and redirect to the login page.
A user logs in using a wrong password that does not match the username: alert an error message and redirect to the login page.

5.
Titie: Modify menu
Actor: restaurant manager
Description: The owner can add dishes to its restaurant. And also, the owner account is able to do some modification, such as sold out or change the price.
Basic flow:
The web application provides a box area filled with all the dishes that have been saved in the database.
When the user wants to add dishes to the database, then input the dish name and price in the editor box and then click ‘Add’ button.
Alternative flow:
No alternative flow for this use case.

Test plan

https://github.com/manMadeLogic/YourChef/blob/master/Test/SampleUser.py

https://github.com/manMadeLogic/YourChef/blob/master/Test/TestRegistration.py

1. Registration

Test usernames with special characters:

Equivalence partitions:

1. Username is a nonempty string with no special character: should return the success status
2. Username contains special symbols such as / and $: should return the error status
3. Username is empty: should return the error status

Boundary cases:

1. Usernames such as “xc-6666”, should return the error status

If a user wants to register by an empty username or a username with special characters, the server will not allow it and give an error message. The test will pass when it returns false for insert_fail_users in SampleUsers.py.

https://github.com/manMadeLogic/YourChef/blob/master/Test/TestUser.py

2. User

Test getting a user information

Equivalence partitions:

1. If the user information is correct, it should match the information in the database and sha256_crypt verify will return true: should return the success status
2. if the user information is not correct: should return the error status

Boundary cases:

1. if the user information is partly correct but some information was lost: should return the error status

https://github.com/manMadeLogic/YourChef/blob/master/Test/TestRegistration.py

3. Test register:

When a user register using legal userId, username, and email that hasn’t registered before, the server inserts a record to the database and return true. The test will pass when it returns true for insert_users in SampleUsers.py.

Equivalence partitions:

1. When a user register using legal userId, username, and email that hasn’t registered before, the server inserts a record to the database and return true: should return the success status
2. When one of the userId/username/email is illegal, registration fails.: should return the “illegal” error status
3. When one of the userId/username/email is registered before, registration fails.: should return the “used before” error status

Boundary cases:

1. When some of the userId/username/email is illegal and some are registered before: should return illegal status

4. Test login:

When the password can be verified to be the same for the user_name, the server returns true. If not, the server returns false. The test will pass if the corresponding condition holds.

Equivalence partitions:

1. Username and password are valid: should return success status with login tokens
2. Username does not exist: should return the error status
3. The password is not correct: should return the error status

Boundary cases:

1. When some of the userId/username/email is illegal and some are registered before: should return illegal status

https://github.com/manMadeLogic/YourChef/blob/master/Test/TestAddDish.py

5. Add dish to cart

Test add dish:

When adding a dish to the cart that has less than 10 dishes, it stores in the session. The test will pass if the result session contains both the dishes before and the newly added dish.

When adding a dish to the cart that has 10 dishes, it doesn't store in the session and return false. The test will pass if the function returns false and result in session contains only the dishes before.

Equivalence partitions:

1. The dish number is less than 10: should return a valid status
2. The dish number is more than 10: should return the error status

Boundary cases:

1. The dish number is exactly 10: should return the valid status

https://github.com/manMadeLogic/YourChef/blob/master/Test/TestManageDish.py

6. Modify menu

Test add dish:

equivalence partitions for dish name:

1. Dish name is a nonempty string with only letters and spaces: valid equivalence class
2. Dish name contains special symbols such as / and $: invalid equivalence
3. Dish name is empty: invalid equivalence

Boundary cases:

1. Dish name is the same as the one added before: invalid equivalence

equivalence partitions for restaurant name:

1. The restaurant name is a nonempty string with only letters and spaces: valid equivalence class
2. The restaurant name contains special symbols such as / and $: invalid equivalence
3. The restaurant name is valid but not exist in the database: invalid equivalence
4. The restaurant name is empty: invalid equivalence

7. Restaurant name nearby search function

We need to test whether the function can handle any inputs allowed by the format without letting the program crash before return value.

equivalence partitions:

1. There is no matched result according to the latitude, longitude, and name: return an error status

2. there is one or more result according to the latitude, longitude, and name, return the first matched value: return a valid result

Coverage

How you measured the branch coverage achieved by your test suite.
Coverage tool:

1. configure Coversall + python-coveralls + coverage
2. create .coverage file to stream coverage output to the Coversall io
3. in .travis.yml, add lines:

```
script:
  - coverage run --source=YourChef.userHelper Test/TestUser.py
  - coverage report
  - coverage run --source=YourChef.registration Test/TestRegistration.py
  - coverage report
after_success:
  - coveralls
```

Report are stored in https://github.com/manMadeLogic/YourChef/tree/master/report
We achieve 100% in YourChef/registration.py and 50% in YourChef/userHelper.py