Assignment group 10.3 Summary

Our team are responsible for testing codes. We separated our jobs into two different parts: front-end testing and back-end testing since the other two sub-teams are working on them. Testing helps ensure front-end and back-end teams do accomplish user stories.

Front-end testing(By Yuyang Wang, utorid(wang1423)):

For front-end testing, we use python unittest to verify the code.

The reason why we choose pytest is because the simplicity and tidiness.

It is easy to use python unittest and import useful packages, also, test functions are very clear to testers. For our front-end html files, we imported “BeautifulSoup” package as a useful tool to examine the elements in the file. The package provides many useful functions that help us locate elements and find their relationships instantly. For example, in our front-end test code, get_text() can be used to find whether we contain certain text in the html file. We can use ".tagname" to get the info belongs to that tag, ".parent" can return the parent tag of the current tag, etc.

"front-end.py" contains two test cases for front-end codes, specifically, "index.html".

"test_html_text()" tests whether "index.html" contains the text we want. For example, we want the text of our project's name to be shown on the page. If this test run without error message, then the text is included in the web page.

"test_html_parent()" tests whether some tags are the parent of some other tags. For example, we may want all the tags are inside <div> tags, so that all elements are divided clearly by <div>. The test can do it.

Back-end testing(By Jiaguan Tang, utorid(tangjiag)):

For back-end testing, We have considered two technologies: AVA and Pytest. We have selected to use Pytest in back-end testing.

Ease of development: Both Pytest and AVA are easy to set up and use. Pytest has a simple syntax and provides many built-in fixtures that simplify test development. However, I am more familiar with Pytest than AVA, since Pytest is used in a lot of assignments in other computer science courses.

Maturity of the technology: Pytest has been around for more than a decade and is considered a mature technology. It has a large and active community that provides support, documentation, and plugins. AVA is a relatively new testing framework from 2009, and its ecosystem is not as mature as Pytest.

Domains covered: Pytest is a testing framework for Python, so it's ideal for testing Python-based backend applications. AVA is a testing framework for JavaScript, so it's ideal for testing Node.js-based backend applications. Our backend is built on the Flask framework, a Python-based backend application. Therefore, Pytest is the ideal testing framework for our application

Popularity: Pytest is one of the most popular testing frameworks in the Python ecosystem, with a large and active community of users and contributors. AVA is popular among Node.js developers, but it is not as widely used as Pytest.

Based on the information from Raygun, AVA only has 7% of users in JavaScript unit testing frameworks.

Reference:

Testim. “What Is the Best Unit Testing Framework for JavaScript?” AI-driven E2E automation with code-like flexibility for your most resilient tests. AI-driven E2E automation with code-like flexibility for your most resilient tests, February 18, 2021. https://www.testim.io/blog/best-unit-testing-framework-for-javascript/.

“Unit Testing Frameworks in Python.” Zenesys. Accessed February 17, 2023. https://www.zenesys.com/unit-testing-frameworks-in-python.

o2task. “>Strategic Innovation – the AVAC Framework (Activities, Value, Appropriability, and Change).” O2task's Blog, May 10, 2011. https://o2task.wordpress.com/2010/08/03/strategic-innovation-the-avac-framework-activities-value-appropriability-and-change/.

“Top 6 Best Python Testing Frameworks [Updated 2023 List].” Software Testing Help, January 14, 2023. https://www.softwaretestinghelp.com/python-testing-frameworks/.

Instructions for “front-end.py”:

1. **In order to run tests, you need to install python on your pc.**

https://www.python.org/

2. **An ide is recommended (e.g. Pycharm)**
3. Need to install “BeautifulSoup” package(“pip install beautifulsoup4” or install on your own ide)
4. **Run the file on ide, if no error message prompts in the console, the tests are passed.**

Instructions for backend test“unit_test.py”:

1. In order to run test, you need to install python 3.10 on your pc.

   https://www.python.org/

2. Clone the files from repo:

   https://github.com/csc301-2023-winter/assignment-2-10-3-tangjiag-wang1423.git

3. Open a terminal and cd to the directory you just clone.

   assignment-2-10-3-tangjiag-wang1423

4. Run a command in the terminal to set up the environment.

```
./startup.sh
```

5. Run a command in the terminal to run the test file(**unit_test.py**).

```
pytest backend_with_test/unit_test.py
```