WEB APP REPORT

Tech Stack Comparison

1. Frontend Technology

   1.1 React

   Pros:

   a. Both of our members have taken CSC309 and actively involved in React development. It is the most familiar frontend framework to us.
   b. Performance is good due to its virtual DOM implementation.
   c. Very mature since developed in 2013 and is continuously updating with rich libraries detailed documentations.
   d. Compatible with many decent UI libraries including but not limited to Bootstrap, AntD, and MaterialUI.
   e. Ranked #2 in web frameworks according to 2019 stackoverflow survey.

   Cons:

   a. Library size is large.

   1.2 Vue

   Pros:

   a. Also takes advantage of virtual DOM, thus performance is good.
   b. Library size is small. Thus speed and flexibility are optimal.
   c. Compatible with many UI libraries especially AntD.

   Cons:

   a. Not as mature and popular as React (50% less popular than React) and offers less shared resources.
   b. None of our members hadprevious experience of Vue.

   1.3 Plain HTML&CSS and Vanilla JS

   Pros:

   a. No external libraries thus dependency and complexity is low.

   Cons:

   a. Manual layout formatting is ineffective, tiring and not visually attractive.

2. Backend Technology

   1.1 Python (Flask) + Pytest

   Pros:

   a. Both of our members have years of experience in Python. One member just recently developed a web app using Python Flask as backend.
   b. Our server load is not expected to be high, thus Python's single flow is enough to handle.
   c. Rich, fast, reliable, and understandable modules are available. Well-maintained community.
   d. Fast development due to its simple and compact syntax.
   e. Have built-in Pytest testing framework

Cons:

a. Runtime is relatively slow due to Python's single flow.
b. Fewer resources and less popular compared to Node JS.

1.2 JavaScript (Node JS)

Pros:

a. Good performance compatibility with React as they are all JavaScript libraries.
b. One of the fastest server-side solutions.
c. Both our members have previous exposure to Node JS.
d. Have Mocha testing framework for quick tests.
e. Very popular frameworks according to 2019 stackoverflow report.

Cons:

a. The interactions between middleware are sometimes complicated.
b. Although also rich in 3rd party module quantities, some npm modules have poor quality or even lack of documentation.

1.3 Java + JUnit

Pros:

a. Both of our members have years of experience in Java.
b. Java has some very powerful application frameworks like Spring.
c. In terms of speed, Java is faster than Python since it's a compiled language.
d. Have Junit for Java Unit Testing.
e. Highly reputed multi-purpose high-level language. Has been popular many years.

Cons:

a. Although Java has some very powerful web frameworks, they are all expensive to learn. For this project respectively, the better choice should be a light-weight framework.
b. Less web app related modules due to less popularity of Java in wen dev area.

3. CI/CD

3.1 Github Actions

Pros:

a. Most compatibility and integration with Github repositories since it's developed by Github.
b. Can run a workflow on any Github event (for us: push or PR). When we create PR, we would like to directly test our code on our test server (which is on AWS), thus deployment triggered by PR action is very convenient.
c. It's free.

Cons:

a. Provides less features than Circle CI (more analytics and debugging tools).

3.2 Circle CI

Pros:

a. It has a free version.
b. Faster performance than Github Actions.

Cons:

a. Less integration with Github services

4. Database

4.1 MongoDB

Pros:

a. Both of our members have used MongoDB for CSC309 project.

b. MongoDB is very compatible with Node JS, since data is stored as JSON.

c. Very flexible due to its non-relational feature also speed is fast.

Cons:

a. Relatively poor support for Python.

4.2 MySQL

Pros:

a. Both of members have enough knowledge in MySQL during CSC343.

b. Compatible with most high-level languages (e.g. Java, Python, JavaScript). It can be conveniently supported by a Python library called sqlalchemy to work with Python Flask.

Cons:

a. Requires strict definitions of tables and columns to store data thus less flexible. However, since we don't expect to have many models in this project, it is acceptable.

4.3 Plain text/csv file

Pros:

a. No need to bother with SQLs. Only need basic file I/O operations.

Cons:

a. Since no logical relations are included, querying data can be very complicated and ineffective.

b. Have severe security issues regarding access to data records.

Tech Stack Solution

1. Frontend
   - React.js + Material UI

   Key reason: React is an optimal solution when handling with modularized components. For this project, we design to include dozens of items both in customer cart and manager portal in form-like components and carry out individual operations. In addition, React’s state management feature ease our session checking and item rendering. Also, Material UI provides us with the most convenient way construct simple but coherent and responsive UI components.
2. Backend
   - Python(Flask) + Pytest

   Key reason: Unlike Node.js, Flask’s syntax for writing routes is more compact and readable. Also, Flask allows more customizable error handling and reporting. In addition, Python has built-in texting framework Pytest, which we are very familiar with and easy to use.
3. CI/CD
   - Github Actions

   Key reason: We would prefer high integration with Github services, especially the feature that Github Actions CI/CD can run a workflow on Github PR event, because we want to directly test functionalities on our test server after PR.

4. Database
    - MySQL

    Key reason: Since we don't have many data models, only least amount of relations need to be defined. Moreover, there’s a library called flask-sqlalchemy which adds support for SQLAlchemy which is a Python SQL toolkit (easily supports MySQL).

Summary

The technical stack we decide to use for the web application is React (with MUI) + Flask + MySQL + Github Actions. The main considerations are ease of development, maturity and richness of the tool, and overall integration.