Report

Summary of Analysis

As suggested by the professor, this assignment can be seen as a preparation for our project and a chance to familiarize ourselves with some key tools. Thus, our goal is to choose technologies that would help smooth this simple checkout development. At the same time, these technologies should function well for our group project.

Our project will focus on machine learning algorithms to analyze companies' annual reports in order to provide insights for investors. From our understanding and our partner's requirements, these functions will be carried out on the web so the users can easily view firm information and make a thoughtful investment.

First and foremost, we decide to use Python as it is a programming language that supports machine learning with a mature and healthy ecosystem - practical packages and the community. Therefore, this is likely to be the language we are going to deploy, and it is helpful to refresh our memories of Python and prepare ourselves for the upcoming ML team project.

After we decide to code in Python, we think of libraries/frameworks for web development. Since this assignment is a simple checkout page, we choose Flask, a light-weight web framework that suits our smaller projects of the scale better. In terms of frontend, we focus on building a clean and straightforward website, so we go with Bootstrap, a simple, easy HTML, CSS, and JS library.

As beginners of CI/CD workflow, Github Actions is not the most powerful tool, but it reduces learning costs for installation, customization, and configuration.

To perform unit tests, Pytest is easy but powerful. Pytest is friendly for beginners, so we can quickly pick up the tool. Additionally, the rich choices of plugins allow it to realize complex and advanced testing (e.g., application, database).

Although our website now only needs a simple check out calculation function, when we consider some standard features that we should extend in the future, such as login, post comments, etc., it requires a database to manage such information. Thus, we decide to integrate a database with our website. Databases will allow us to keep track of each users' specific information. Additionally, storing information in a database is more secure in the sense that users will not be able to interact or view the data stored in the database. Last but not least, the database enables manipulation with information, such as save, add, remove, search, etc. As we perceive it, these functions are a must-have for our partner and the final users.

As a result, Bootstrap (frontend), Python Flask(back-end), Pytest, Keruto (CI/CD), and SQLite Database satisfied our requirements well. Each of them has its vital function in a machine learning project that needs to deal with large amounts of data and display different contents to different users.

Detailed Comparison

Frontend

Chosen technology: Bootstrap

Key reasons:

The main reason we choose Bootstrap is that what we asked for is a simple checkout page.Though react and angular are very popular and provide a much clearer framework in terms of regulating JS, we don't need to that fancy interact with users. Thus, a simple, easy HTML, CSS, and JS library is good enough for us.

	Pros	**Cons**
Bootstrap	● Easy to use, as a result, time-saving ● Got the support of a strong community ● Highly responsive structure and style	● If developers require customized design, they'll have to spend extra time on coding due to lots of style overrides or rewriting files
React	● Easy to start ● Has a large community ● Better performance in terms of site load time (due to virtual DOM)	● Need to be familiar with lots of third party packages ● No predefined way to structure your app
Angular	● Slightly smaller popularity compared to React, but still large ● Productive (No need to search for libraries) ● Angular tends to perform better overall for task-based benchmark tests	● Hard to learn at first ● Difficulty scaling (due to two-way data binding)

Backend

Chosen technology: Python Flask

Key reasons:

In terms of programming language, we decided to use Python as it supports machine learning projects, so it'll be a chance to prepare ourselves for the upcoming ML team project. Though Django is in advantage when it comes to bigger projects with more functionalities, we choose Flask, a light-weight web framework that suits our smaller projects of the scale better.

	Pros	Cons
Python Flask	• We're familiar with this technology already • Performs slightly better since it's smaller and fewer layers • Got the support of a strong community • Python is the popular programming language and supports ML projects	• More workload when adding a new functionality
Python Django	• Got the support of a strong community • Python is the popular programming language and supports ML project	• Somewhat complicated, need a deep learning
Java Spring	• Plain Old Java Objects make it a lightweight framework (advantage in developing web applications) • Flexibility for configuration (either XML or Java-based annotations)	• Sophisticated and hard to learn • Parallel mechanisms leads to confusion when choosing features

CI/CD

Chosen technology: Github Actions

Key reasons:

Github Actions is not the most powerful CI/CD tool, but it reduces learning costs for beginners in installation, customization, and configuration. Also, as a default feature integrated in Github, it works well with Github repository which further reduces the potential problem in compatibility.

	Pros	**Cons**
Github Actions	● Actions is integrated in Github so it is easy to keep track of and it supports Github very well ● It is open source so anyone can publish actions and can use other's published actions ● Actions allow multiple config files to separate different components	● It is relatively new so it is less stable and may continue to experience changes ● The openness may become a problem in business ● Actions is a bit slow
CircleCI	● CircleCI is cloud based, so there is no dedicated server to administer and maintain ● CircleCI checks dependencies and will install them by default ● It has a free plan, even for business ● Have YAML file as a config ● CircleCI run on all languages ● Fast authorization with GitHub or Bitbucket	● Cloud-based service means there may be issues in service outages, security, and privacy ● Customization may need extra efforts ● CircleCI charges for most versions of Ubuntu as well as the MacOS ● It supports only a few languages "out of the box"
Jenkins	● Jenkins requires a server so it is better for stability, security and privacy ● Jenkins is a self-contained ready to run out-of-the-box, with packages for Windows, Mac OS X and other Unix-like operating systems ● With numerous plugins, it is easy to customize ● Jenkins has a large user base so there is a good community for learning	● A server will need constant maintenance, which results in additional expenses. ● It needs installation of all dependent Jenkins tools and plugins ● There is a learning curve for configuration / customization

Testing

Chosen technology: Pytest

Key reasons:

Pytest is simple but powerful. Pytest is friendly for beginners so we can quickly pick up the tool. Though it has limited compatibility, the rich choice of plugins allows it to realise complex and advanced testing (e.g. application, database) and work well with programs written in different frameworks. As we see in the future, databases will be an important and necessary component, we can still deploy Pytest without moving to another technology.

	Pros	**Cons**
Pytest	● Pytest is simple and flexible in writing test cases ● Pytest is easy to learn as it hides details in implementation ● There are many plugins for easy extension ● Pytest is compatible with unittest ● Pytest also supports complex testing, e.g. application and libraries	● This is a third party framework ● Pytest uses its own special routines to write tests, which limits its compatibility
Unittest	● Unittest is packaged with Python so no need to install	● The format must be strictly followed so there is a learning curve ● Unittest is not compatible with other test framework ● Unittest does not have many plugins
Nose	● Nose is simple and flexible in writing test cases ● Nose is easy to learn as it hides details in implementation ● There are some plugins	● This is a third party framework ● Nose has been in maintenance mode for years and will likely cease

Database:

Chosen technology: SQLite

Key reasons:

In this assignment, we are only asked to build a simple checkout without loading or saving data, so it seems unnecessary to deploy a database at this point. However, if we take the possibility for extension, such as to allow different users to log in and see their unique shopping cart, it would be more efficient to manage the information using a database. Databases will allow us to display different contents to different people. Especially when our site content is likely to change quickly, a database can reduce human administration.

Additionally, storing information in a database is more secure in the sense that users will not be able to directly interact or view the information in the backend. They have to follow our rules for them to interact. There are also other measures to secure databases, such as authentication so that only authenticated users will be granted access to the data.

Last but not least, the database enables basic and advanced manipulation with information, such as save, add, remove, search, etc. If customers are looking for products, the server can pull out related data by querying the database. Then, that information is inserted into parts of a pre-set website and displayed to customers. It's fast, and it also ensures a good-looking, consistent presentation.