**HOMEWORK WEEK2**

**Decision Making App,**

**Group Members:**

Phoenix Null

[Vania Beatriz](https://app.slack.com/team/U03GD9P27QC)

[Ziling](https://app.slack.com/team/U03G81XNQ21) Guo

Helen Braczek

This week’s homework will be purely Project based. You need to work as a group and the homework will be submitted by one of the members of your group. List the other members of your group in the document, so that your instructor can mark every student.

1. Your group needs to decide what kind of project you are going to work on and lock in your decision.
2. You need to submit a free style paper that describes your project on a high level. Please cover the following questions:
3. **What are you building?**

A make a browser-style decision making application to choose a movie to watch.

1. **What does it do or what kind of problem does it solve?**

Helps indecisive users select a movie to watch.

1. **What are the key features of your system?**

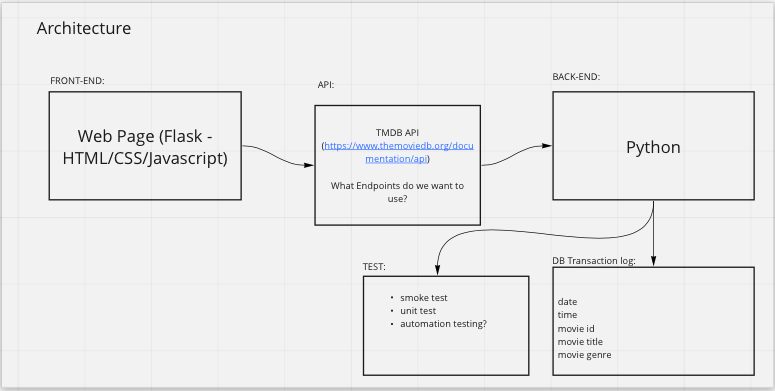
**Essential:**

* API returns a random movie.
* Ability to spin again for a new movie.

**Bonus features** (once basic code works)**:**

* Spinning wheel to show code is doing something.
* A button to select genre to filter results.
* Recommended ways to watch the movie.

1. **Provide a sample architecture diagram of your system**:



1. **Describe the team approach to the project work** how are you planning to distribute the workload, how are you managing your code, how are you planning to test your system.

**Distribution of workload:**

* Split tasks up into user-stories to divide amongst the 4 of us.
* Weekly stand-ups.
* Github/Google docs/ Miro board/ and Slack to aid remote collaboration.

**Manage code:**

* Use GitHub to raise tickets to keep track of user stories and tasks. Each task has its own branch then added incrementally to the main branch.
* Pull requests and code reviews to ensure code does not break or have any bugs.

Ideal workflow:

Work on own branch/es -> test own code, if ok -> commit and push changes -> create a pull request on GitHub -> everyone else can review and understand -> approve the pull requests -> merge to ‘main’ project.

**Testing:**

* Testing our code within GitHub to ensure it works when merged with each other's code.
* Regression testing to ensure the code works after any changes are made.
* Ensure we are getting a response from the API and the system is working.
* Testing the buttons work and go to the right page.
* Test that it doesn’t crash.
* UnitTesting.
* Automated testing, e.g. PyTest.
* End-to-end testing: https://playwright.dev/