System Architecture

Programming Language

We had a choice between Java and Python and we went with Python as our programming language as most of the team is familiar with it and has used it in their own time and some used it in their coops.

Framework

Since our language of choice was python we had two popular frameworks to choose from Django and Flask, they are both written in python and are open source. After doing our research, we decided to go with the Django framework rather than flask because Django supports MVT(Model-View-Template) pattern which is slightly different from the MVC(Model-View-Controller) pattern. On the other hand, the flask doesn't support the MVC pattern(1) and would require us to design it first which in turn would add more time to development cycle.

A quick explanation of how MVT in Django works. In MVC we have a model, view and controller whereas, in Django MVT, we have a model, view and template. The controller part which handles interactions between a model and view is all handled by Django itself. Also slightly confusing part between these two patterns Is how they handle view and controllers. In Below Diagram you can see that View in MVT is not the same as View from MVC because the View in MVC handles how the page is presented to the user but in MVT it’s the template that handles that part and View in MVT is not entirely same as controller from MVC but it act as one to some degree but most of the controller part is handled by Django itself.

Diagram

Description automatically generated

MVC and MVT difference diagram

Django MVT pattern Diagram.

Diagram

Description automatically generated

Another few reasons we went with Django and not flask, is because Django has some important features that are supplied out of the box with it like -

ORM

-comes with ORM(Object-relational-mapping) that supports many relational databases such as Oracle, MySQL, SQLite etc and we needed MySql.

-Admin

-It’s one of the most powerful thing that comes with Django, it’s a functional admin panel. It will read metadata from our models and provide us with a quick, model-centric interface.

-Authentication

Django comes with a user authentication system that handles authorization and authentication of user accounts, groups, and permissions. Nowadays most if not all apps, and webpages require some form of authentication and authorization.

Database

For the database, we chose one of the popular databases, which is MySQL(Open-Source). Another reason is that most of the team members have some experience with it from one of our previous semesters (Database Systems) and that would reduce a big chunk of the learning curve. Also as stated above it is fully supported by Django.

1) <https://ieeexplore.ieee.org/document/8901656>