CITS5505 Project 2

Brief

For this project, you are required to build a request forum application, which allows users to create accounts, post their own requests and answer other people's requests.

The web application should be styled to be interesting and engaging for a user in the selected context. It should offer several views including:

- [ ] An "introductory" view, describing the context and purpose of the application, and allowing the user to create an account or log in.
- [ ] A "find requests" view, allowing the user to search for and accept requests.
- [ ] A "create requests" view, allowing the user to create requests for other people to answer.

The application should be written using HTML, CSS, Flask, AJAX/Websockets, JQuery, Bootstrap, SQLite. Any additional technologies or frameworks require special approval by the unit coordinator, and will only be approved if you can provide a convincing reason why the proposed project cannot be completed without them.

The creation of the web application should be done in a private GitHub repository that includes a README containing:

1. a description of the purpose of the application, explaining the its design and use.
2. a table with each row containing the i) UWA ID ii) name and iii) Github user name of the group members.
3. a brief summary of the architecture of the application.
4. instructions for how to launch the application.
5. instructions for how to run the tests for the application.
6. The repository should also include a folder called "deliverables". Within the folder there should be 3 short screen-capture videos (.mp4) or image files (.jpeg, .png).
7. These files should be added as the project progresses and each should:
    a. show off the completion of a suitable intermediate deliverable as per the Agile methodology. The video or image should show the feature being demonstrated (audio on videos is allowed, but not required - unless audio is an important part of the feature).
    b. be named the same as a Git tag in the project. That tag should point to the commit where the video/image was made.

Any decisions made about future work should be entered as new Issues for documentation purposes.

Some marks will be assigned individually, looking at evidence from GitHub. There will also be an opportunity during the demonstrations to discuss who did what, and what the challenges were, and marks among the group may be split accordingly.

You are acting as both developers and users so no opportunity to get iterative feedback.

The only requirement is that deliverables should showcase some major new pieces of functionality that a hypothetical user could give feedback on.

Given the structure of the course, a sensible first deliverable would be static HTML and CSS mock-ups of your main pages for your website.

However, it's ultimately up to you to decide what are suitable intermediate deliverables for your project!

Submission

To submit your group project:

1. Make your repository public on GitHub so that the markers can see it and so that the `Insights` tab is enabled.
2. Create a zip file of your complete Flask application:
    a. All source code, with comments and attributions for any external libraries, including your tests directory
    b. A ` `requirements.txt` ` file, listing all packages used. To build the requirements.txt file for your virtual environment, use the command: `pip freeze > requirements.txt` while your virtual environment is active.
    c. The `README.md` file
    d. The `deliverables` folder.
    e. You may include a small database for testing purposes
3. Click on the word "Submission" above and upload the zip file.

The project should be runnable by 1) downloading the code, 2) installing the packages in the requirements.txt file in a new virtual environment, 3) following the instructions in the README.

Do not submit the virtual environment directory, or the .git directory. These are overly large, not required and will result in a penalty if they are submitted.

Submit your zipped file by clicking on the title of the item in the LMS and attaching the file. Only one person per group needs to make this submission.

Assessment

Front-end (35%)

The first part of the project assessment will evaluate the front-end functionality of the application:

1. The frontend must be functional so that the user can easily use the application as intended.
2. The frontend must be implemented using HTML, CSS and Javascript.
3. All external resources used (including pictures, JavaScript libraries, CSS) must be fully referenced.
4. The HTML and CSS must pass the validators: https://validator.w3.org/ and https://jigsaw.w3.org/css-validator/.
5. The website must work on Chrome, Firefox and Microsoft Edge, and render well on mobile devices.
6. There must be a consistent style (via CSS file) for all pages, yet each page should be easily identifiable.

Back-end (40%)

The second part of the project assessment evaluates the back-end functionality of the application. At least the following functionality should be provided:

1. User accounts.
2. The ability to store user interactions and the results of those interactions.
3. The ability to search for previous user interactions.

Agile processes (25%)

The third part of the project assessment will evaluate the Agile process used to create the application. Marks will be given for evidence of Agile development practices, i.e.

1. appropriately sized commits with meaningful messages.
2. planning of short-term goals via the Issues tab.
3. reproducible bug reports via the Issues tab.
4. use pull requests to merge into new features via the Pull Requests tab.
5. exhibiting teamwork by contributing to discussions on the Issues
6. exhibiting teamwork by adding code reviews to other people's Pull Requests.
7. intermediate deliverables, pinpointed with Git tags.