Ethics Dashboard (1): Requirements Report

Introduction:

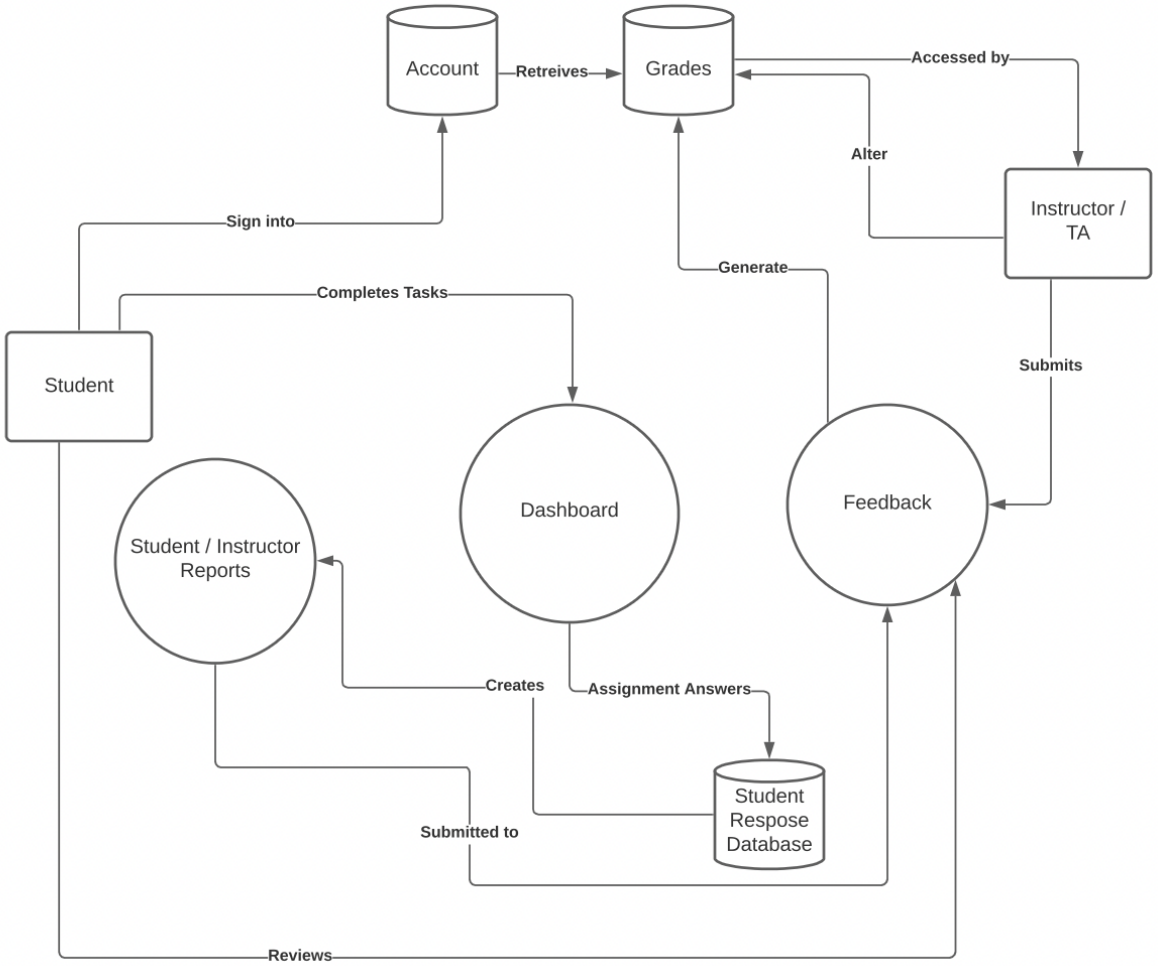
We will be creating an Ethics dashboard as a resource for students in an applied ethics course to further their learning, provide extra help where needed, and submit their work. After submitting their work, the professor or the teaching assistant should be able to give the student a grade on the ‘My Progress’ page and add any comments necessary. Additionally, there will be a database which will hold the information of each student, along with their grades and submissions. With this project, there are three main user groups we will be focusing on. The main target group is students and the application will be tailored mainly towards them. With this site, students should be able to create at least 4 dashboards, have access to additional readings, answer practice questions and submit their work. Secondly, the teaching assistants for the course should be able to view the students' grades, mark their work, be able to perform any necessary modifications, and should be able to view the database and the information in it, unlike the students. Finally, the professor of the course needs to have access to this application much like the teaching assistants, but must also have administrative privileges, such as being able to override any decisions taken by the teaching assistant if deemed necessary.

Data Flow Diagrams

DFD: Level 0

This level 0 data flow diagram shows the three largest components of the system as a whole. Starting with students, they will be able to receive case studies and content to practice on from the process, Ethics Dashboard, which is also the general system. The students will also be able to submit their answers to the case studies and practice questions to the Ethics Dashboard system. The third component is the course instructor(s) and TA(s). They will be able to view student progress through the Ethics Dashboard. They will also be able to comment and give feedback on the students’ work through the Ethics Dashboard.

DFD: Level 1



The level 1 data flow diagram shows a total of three processes and three databases. The students will be able to sign into their accounts on the Ethics Dashboard system (peer testing #1), which will also be linked to the Grades database (peer testing #2). The students will also be able to complete tasks on their dashboards (peer testing #1). Their assignment answers will then be stored in the student response database which will be used to create the student/instructor reports page (peer testing #1). The student / instructor reports will then be submitted to the feedback process which students will be able to view and instructors / TAs will be able to submit their feedback and comments on the students' work to this process (peer testing #2). Instructors and TAs will also have access to the Grades database and they will be able to alter the database as well (peer testing #2). Lastly, the feedback process will generate a grade for the students' work and will insert it into the Grades database (peer testing #2).

Functional Requirements for each Milestone

Milestone 1: Peer Testing #1 - Basic Functionalities (Student side)

- Basic skeleton of the website(HTML) and the linked pages should be set up. This means each ethical theory tab (Utilitarianism, Deontology, Virtue Ethics, and Care Ethics) should be linked to the navigation of the main Dashboard page.
- Basic styling using CSS should be set up and consistent through all pages
- Account login pages are connected to a sample database
- Users are able to input and submit forms (assignments)

Milestone 2: Peer Testing #2 - Advanced Functionalities (Admin side)

- Instructors and TAs have access to their own portals
- Admins can add feedback and grades to the student assignment submissions
- Admins have access to altering and accessing student grades through the grades database
- Students can view feedback and grades from Instructors
- Student and instructor reports page implemented

Milestone 3: Final Product

- Testing of all features that work together and have been implemented
- Entire system is functioning as it should and includes all functional and non-functional requirements
- Databases are working and are manageable for the future and maintenance

Testing Strategies

General Testing Strategies:

We are going to utilize functional testing strategies to verify that the application behaves correctly and the way we expect it to. This will be mostly accomplished through manual testing of inputs and evaluating the outputs to deem whether it is correct or not. Some automated testing will be made use of, particularly for server side scripting such that there are no unknown security vulnerabilities or bad output. After functional testing is completed and deemed correct, we will perform nonfunctional testing to determine if the application is responsive, has reasonable loading times, if it is compatible with older builds, and if it can handle high loads.

Frontend Testing:

The frontend is composed of HTML and CSS, as these technologies are static, there is little testing required apart from manual testing of links, visual verification that styles are

appropriate and consistent, and so on. The HTML will be developed correctly such that document object model manipulation and access does not break the flow when applying scripts.

Backend Testing:

Our backend is split into three testable segments: The client-side scripting, server-side scripting, and database access and manipulation. For the client side, we will attempt to employ ESLint, a JavaScript code linter. Our client-side scripting will require relatively little active testing compared to other backend components, since the extent of client-side scripting will be to accomplish checkboxes, radio buttons, pre-request text box handling, and so on. ESLint provides an extension to regular IDE warnings and suggestions, reducing bad practice and thus reducing vulnerability, errors, and slowness.

Server-side scripting and database access and manipulation will be handled with a mix of manual and automatic testing practices. GET and POST strings will be fed into the server-side script and sent to the database, then returned. Because security and correctness of database results are of utmost importance, we will place a large emphasis on the database access and manipulation portion of testing. This means attempting to throw junk input as well as attempting to conduct injection attacks, to verify that the system behaves as required.

Continuous Integration:

Continuous integration will allow us to create numerous builds of the system without breaking - or at least severely breaking the system - through a very modular file format. Each new page of the application is a new HTML file with no effect on the others, with styles extracted from a static CSS library. The scripting will be handled mostly on a file-to-file basis, and the database is kept separate from everything. This means all components of the system are only tenuously linked, and will not break each other if one file is in error. As well, with such frequent commits, any issues will be trivial and small in nature, since not much work can be done between commits. If we did not adopt continuous integration in this manner, we would have to sift through large files or collections of files with no idea where the problem is.

Regression Testing:

Regression testing will be easily conducted due to the method of continuous integration we are employing, due to the triviality of changes made per commit. While we are conducting testing as we develop, this may not be appropriate after more functionality is added, and we will need to conduct even more comprehensive testing on previous functionality in our program. Say, our HTML may be correctly formatted when static, but after modifying, for example, a text box, it may break. This requires us to expand upon our testing from just one form to a mix of multiple, as these technologies are interacting.

Non-functional Requirements and Environmental Constraints

Functional

The dashboard itself is required to record any student input, store student data, load any previously saved work, calculate the student response and should be able to download the student data into a proper csv file. The support of multiple users is important and security should be a present factor when building the dashboard. It is required for us to design the dashboard in a way that the student stays engaged and remains a degree of academic honesty.

Non-Functional

Hosted by: https://www.okanaganbcwebhost.ca because of this we decided to use mySQL to avoid any hosting constraints.. Web hosting and domain name defined by client as well as we are constrained to using a canada based server for student accessibility. This does not affect our tech stake greatly and actually helped us narrow down our choices for frontend and backend options. 1.95GBs of space but is flexible in terms of upgrades, stated by the client that this should be enough space for the database that is being used. Power consumption is handled by the client and college, not an issue for the development team. Although power consumption should be rather low as the main consumption would be the client database. Future maintenance and development by future students and peers, thus code should be well documented and easy to understand. For the team this would be maintenance and accessibility should be a concern but not a high priority. Mobility and scalability should be a priority as not one device will be used for the dashboard.

Tech Stack

Frontend

The team choices of tech stack were based on technology we are familiar with, popularity of tech, documentation and support, and whether or not the server host requires the use of a specific tech or doesn't even support the use of another. Our client is very flexible with the technology we use so the frontend language design of choice is HTML, CSS and JavaScript. Our reasoning is that it's fairly easy to develop while still supporting many of the features that the client requested. CSS and JavaScript are easy to implement into HTML and do not require a lot of documentation or prior knowledge. Maintainability and updates are easy to document and implement due to the simple nature of HTML and its support of stylesheets, client-side and server-side scripts.

Backend

In our backend we looked at options like Django which is a high level web framework based on python which allows for fast development and design. Although it is relatively new

to the team, the extensive documentation and learning materials make Django ideal for this project. It is also free to use and is open source which means it is quite flexible for many web development projects. Security features are built into Django ensuring that the database remains smooth and streamlined

Database

We will be using mysql and sql to integrate and manipulate the database into the dashboard. The reasoning behind this is because it is fairly simple to understand and there is tons of documentation and support for troubleshooting.

Peer Review Feedback

- Is there any plan on using a CSS framework?

As of right now, after the requirements presentation we have begin to explore and test the advantages and disadvantages of css framework and may change to using one depending on the progress of the project

- Were there any other contenders when choosing the back end language? Pros and cons?

Django and PHP were the two contenders. PHP is more popular and likely to be known by maintainers and we are familiar with it. Django is up and coming and perhaps more flexible but none of us know it. We are currently revising our choice of using Django, perhaps thinking of

- Will the students be able to submit their responses to the prompts at any time during the creation of the case study or will there be checkpoints for saving and submitting?

We plan on implementing a save function for the student so they can return to work at a later time if they choose.

- Since students are your main focus of the user base and you are tailoring to them, and since the vast majority of students have access to cell phones, would you consider a mobile version of the dashboard as a non-functional requirement?

Yes, since students are a major user group of the system, we are tailoring the site to them but also to the instructors and TAs as they will be accessing the system regularly as well. In terms of a mobile version, we are going to make sure the website can be viewed properly on any device size using a responsive design model. We have discussed a possible mobile app for this system with the client who mentioned if this were to happen, it would be happening way down the line (so probably will not be implemented by us).

- It was mentioned that the project would be hosted on an Apache server. Will you be configuring that yourselves?

There should be some configuration involved with apache on our part but we are not sure how much or the steps that need to be taken to do this. More expiration will be done in this area as the project progresses.

- Will you be using a CSS framework such as bootstrap or bulma? Why or why not?

As of right now we began to do some prototype testing with bulma as it seems to be a more effective tech stack option for the project. We decided to go with bulma because it is fairly easy to learn and implement into our project

- Using a JavaScript library for front end UI (ReactJs, Angular, ...) could be a great idea to support future maintenance and expandability when trying to add new data to the app, would this be a possible option for the project?

After reading this question, the team has decided to review our decision for the front end framework. We realized that it might make the design nicer and easier to implement. So we are currently looking into using jQuery and Bulma for our front end.

- Are there any plans to implement automated tests?

Yes, automated tests will be appropriate for high-volume text inputs such as GET and POST requests. This will likely be employed in that situation due to the vast diversity of inputs that would need to be checked.

- Do you think using a non-standard backend technology will make it harder or easier to have long term support for the project?

Upon review and discussion with the team, we have come to realize that using unfamiliar backend technology would actually make the project a bit harder to maintain - or at least a bit harder to find people with the appropriate skills in order to maintain the site. Because of this, the team has decided to rethink our back end technology and perhaps use PHP which is a bit more standard and familiar in the industry.

- How will the frontend (HTML & JS) interact with the backend technology?

Html will facilitate requests to the backend through user visible elements, such as ticking boxes or entering text and sending that to the db, or login, etc.. Javascript will likely handle dynamic content then send that to be validated or stored by the backend.

- In testing the CSS, are there some manual testing strategies that will be/should be implemented?

Yes, our method of testing CSS will be to manually check that the styles are appropriate for our application, as well as consistent across the entire application. We will verify that the CSS changes appropriately according to any dynamic changes to the webpage, such as adding a textbox after clicking a button not breaking a wrapper that it should be contained in.

- Do you see there being any issues with scalability/hosting capacity?

As of right now we don't see any issues with scalability/hosting because the database is relatively small and most of the hosting capacity will be dealt with by the client. If an issue does arise the client has no problem with assisting with increasing the hosting capacity.

- The DFD level 1 only shows Instructors / TA as being able to access or modify students grades, does the professor have any other control over the system such as modifying course content on the dashboard through the client?

In discussion with the client it seems that the maintenance role will include adding course content into the dashboard. However the professor should be able to have access to the questions that are being displayed to the students through their admin portal.

- There is no indication of how the course content will be uploaded to the database, will this be done only by your clients or will you be developing a system in order to take care of that

For testing purposes, we will be adding course content while we are implementing the features however, there will also be a way for the instructors to add case studies and assignment questions through their portal