EECS1012 Term Project

Instructions:

This is a teamwork project. The size of the team is required to be minimum three and maximum four students when you start the project. We highly encourage you to choose the teammate(s) from the same lab you are enrolled in. You cannot be in a group with students that are in a different class section than you (M to N).

In this project, you will develop a Web Application through a software development life cycle combining some of the best features of Waterfall and Agile models; simplified in four phases below each of which demonstrate a milestone by the corresponding deadline. Please note, our approach does not follow a strict Waterfall or Agile methodology, this is more of a "taste" of building software. Also note that some phases have more than one deadline; and if you fail a Pass/Fail deadline, you receive 0 for all following deadlines. In other words, pass/fail deadlines are crucial and should not be underestimated.

What the deadlines mean:

Deadline 1: By this date, you must have attended one of the office hours for the given Phase, and present your progress for that given phase to the TA. To clarify it's a PROGRESS CHECK, you do not need to have anywhere near a completed piece of work. If you do not pass the Sanity Check (progress check), you will not be able to receive a grade for Deadline 2. If you are worried about how much work should be done for Deadline 1, email the TA.

Deadline 2: By this date you must submit your completed work for the phase on eClass. There is no office hour prior to this deadline. Any questions you have between the Deadline 1 office hours and this Deadline, can be emailed to the TA (with appropriate email title).

TBD: To be determined. The date has not yet been finalized, and will be updated onto eClass + a notification on the course announcements of the update when the update is made.

Phase 0: Team Formation

Deadline: Feb 17 (No office hour, just submit on eClass); Grading: Pass or Fail.

A- Start a discussion with your classmates (in particular, with those from your own lab) and see what common Web-Application project you are interested in. You should also be mindful of your teammates schedule as well as their passion to the course and project. Team formation is a very important phase of the project. There will be challenges in almost every team/project; and only in a good team and with project management skills you could overcome the difficulties. Note that some students will have to drop the course or may find themselves failing in it, and therefore not contributing in the project towards the end; and you are still in charge of carrying out your project with or without them. That's why the Team Formation stage is really important.

For the project ideas, you could choose one of our suggestions at the end of this document, or you could suggest your own and see if we approve it. Note that you are required to use the exact tools and technologies that we specify in this document and/or use in lectures and labs. No other tools/technologies are accepted. This also includes libraries that are not taught directly in the course unless that library is specifically approved by project TA.

B- Make sure your team size is three or four. We do not make any exceptions, so teams of smaller than three or bigger than four students are not accepted.

C- Pick a professionally acceptable name for your team.

D- Submit the following information to the TA via eClass:

- Name of the team
- Members of the team: name, last name, and York U email address as well as lecture and lab section of each team member
- Title of the project plus a short paragraph of describing it. Number of words in your project description should be between 70 to 120 words.

If you fail in Phase 0, you will not be allowed to do the project. Hence, you may want to start communicating with the project TA as soon as possible to make sure you are on the right track. Hence, try to go to their office hour as early as possible. Their schedule is posted in eClass. *{make sure to record any interesting and/or fruitful moment that you have; in the final phase of the project, you need to deliver a kind-of professional video, yet very short (2 to 3 minutes), regarding your learning process, technologies used, 1012's learning outcomes, and the project development.}*

Phase 1. Requirements Definition

Deadline 1: Mar 6; Sanity Check; Weight: Pass or Fail.

Deadline 2: Mar 11; Weight: 20% of the project if passed Deadline 1. (3% by peer evaluation) If you do not have a Git account, you are required to make one by this date. The project should be developed in Git.

In this phase, you should describe the project in further details –in a simple yet organized way. Note: to guide you that the scope of your project is appropriate for this course, we have provided you with some estimated quantitative numbers in this document. But each project is different, and you need to regularly consult with the TAs during their office hours to make sure the scope of your project is appropriate. In this phase, you need to complete three steps:

A- Open the document you submitted in Phase 0 (You will be submitting an updated P0 for P1).

B- Improve the project description based on the comments that you received from the project TA during their office hours for this phase. You have the option of adding one more paragraph of 70-120 words here if you deem it necessary.

C- Then, define 10 to 15 ***functional requirements*** of your project as a bullet list. To learn more about Software Requirements Definition, start with this reading: https://www.pjsrivastava.com/a-short-guide-to-writing-software-requirements . Note that the Requirements Definition in this reading is simplified, and you may need to simplify it even further for your project. Afterall, the scope of your project in EECS1012 should be a tiny one compared to commercial projects. The goal is to learn the net-centric software development process in a modest level. You will probably note that even a small project needs a lot of work. Hence, from that reading, all you need for your 10-15 requirements is 10-15 statements like this:

- Users can view their bill in HTML format

Each statement should be up to two lines, typically, ~10-40 words; and you should provide 10-15 of these statements, called requirements. All quantitative numbers would depend to the project you are working on. Hence, consult your project with the project TA during their office hours before submitting your Requirement Definitions.

All team members should have a great understanding of the requirements definition of their project so that they can answer any potential questions we may ask in Tests, Quizzes, or Exams.

{make sure to record any interesting and/or fruitful moment that you have; in the final phase of the project, you need to deliver a kind-of professional video, yet very short (2 to 3 minutes), regarding your learning process, technologies used, 1012's learning outcomes, and the project development.}

Phase 2. Design

Deadline 1: Mar 6, Sanity Check; Weight: Pass or Fail.

Deadline 2: Mar 11; Weight: 20% of the project if passed Deadline 1. (3% by peer evaluation)

In this phase, you should use wireframes to show the design of your interface: how each page at frontend (client side) will look like (content and functionality) and how they will interact with one another (navigation). Your wireframes are required to be Lo-Fi. You could do it using pencil/pen and paper or using any application you wish (e.g., Figma); but you still should avoid Hi-Fi wireframes in this project. You can start learning about wireframe design from the following links:

- https://selftaughtcoders.com/from-idea-to-launch/lesson-19/5-guidelines-for-creatingweb-application-wireframes/
- https://www.justinmind.com/wireframe
- https://careerfoundry.com/en/blog/ux-design/how-to-create-your-first-wireframe/

Your submission should have 3-10 layouts of your project interface pages, depicted by wireframes. All quantitative numbers would depend to the project you are working on. Hence, consult your project's design phase with the project TA during their office hours, before submitting your Design. All team members should have a great understanding of the design of their project so that they can answer any potential questions we may ask in Tests, Quizzes, or Exams.

{make sure to record any interesting and/or fruitful moment that you have; in the final phase of the project, you need to deliver a kind-of professional video, yet very short (2 to 3 minutes), regarding your learning process, technologies used, 1012's learning outcomes, and the project development.}

Phase 3. Implementation and Testing

Deadline 1: TBD, Sanity Check; Weight: Pass or Fail.

Deadline 2: TBD; Weight: 35% of the project if passed Deadline 1. (5% by peer evaluation)

In this phase, you implement the frontend (client side) as well as the backend (server side) of your project using the following technologies, `HTML`, `CSS`, `JavaScript`, `Node.js`, `jQuery`, and/or `React`. Your implementation should be based on your design phase and should be tested using

`Mocha.js` and `Chai.js`. Students who use any other technologies or libraries --other than what is directly taught in the course-- will receive 0 in this phase. Further discussion/details will be provided along the term. Again, continuously consult with the project TA during this phase as well and before submitting your final product. TAs should be able to ask where a specific idea for code came from, and it should be referenceable to a lab/lecture from the course. All team members should have a great understanding of the design of their project so that they can answer any potential questions we may ask in Tests, Quizzes, or Exams.

{make sure to record any interesting and/or fruitful moment that you have; in the final phase of the project, you need to deliver a kind-of professional video, yet very short (2 to 3 minutes), regarding your learning process, technologies used, 1012's learning outcomes, and the project development.}

Phase 4. Demonstration

Deadline: TBD (No office hour, just submit on eClass); Weight: 25% of the project if delivered reasonable work by Deadline 2 of Phase 3. (4% by peer evaluation)

In this phase, you develop one single 2–3-minute video to show your learning process both in EECS1012 as well as in this project based on what you recorded in previous phases, as well as demonstration of your final product. Make sure you list name of your team and all teammates in the beginning of your video. **All teammates should participate in this phase too, which includes being in the video**. **The quality of your video should aim for a possible publication on YouTube. If you are not in the video, it will be assumed you did not contribute to the video presentation, so you won't receive marks for this phase.**

So, in summary, your phase 4 MUST have the following 4 things – but the order is up to you:

1. Introduce yourselves **clearly**,
2. Demo the project and explain all functionality as if a user was to use it (meaning you don't have to explain **HOW the functions work**),
3. Describes some **fruitful moments** you had throughout the term such as in lectures, labs, exclusive sessions for the project, peer discussions, self-study, etc.
4. **Final notes** of your learning process in EECS1012, and what you feel may have impacted you for your future (knowledge or experiences you've gained).

Aim as if it was publishable to YouTube, and again in 2-3 minutes. This presentation is also a serious phase and is a representation of how well you can communicate your ideas and work to others in a verbal/visual manner—an important skill in the field of computer science.

Team:	Team Formation
RD1:	Requirements Definition (Stage 1)
D1:	Design (Stage 1)
RD2:	Requirements Definition (Stage 2)
D2:	Design (Stage 2)
IT1:	Implementation and Testing (Stage 1)
IT2:	Implementation and Testing (Stage 2)

Demo: Demonstration

Some Sample Suggestions for Project Ideas:

Here are some project suggestions. You can define your own project too as long as you consult with the TA in advance and get their approval of your idea.

1) What number am I thinking of? (between 1 and X, as many times as you can guess in a row)
   Server chooses a number between 1 and X. If the user guesses the number correctly, notify them and keep guessing between 1 and X. Keep a record of the high score for how many times someone has guessed the correct number in a row. (new number every time between 1 and X, after the user guesses correctly/incorrectly). Congratulate them on their record when they lose (not including 0). You determine the X (this requires an input that will be sent to server side)

2) What number am I thinking of? (between 1 and X, and user also inputs how many tries you have)
   Server chooses a number between 1 and X. If the user guesses the number right, congratulate them. Also have the webpage ask the user for how many attempts they must guess the number. Once they guess the number correctly, add their name to a list which is displayed on the webpage (ask user for name once they win).

3) Mini-Database (definitions)
   When a user inputs a word to search for, the server either displays information on a 'display area' – or lets the user know that it does not contain the definition for that word. List of all words that are available on the 'database' should be listed on the side of the webpage (capitalization should be made to not matter (regex)).

4) Mini-Database (images)
   User inputs an object they would like displayed on the screen, and the server will send an image (URL perhaps) to the client, which will display the object on a 'display area'. List of all images/objects available on 'database' should be listed on the side of the webpage. It is a good idea to have all these images be of a similar topic (all be animals, or all be construction building tools, or all be soccer players, etc.)

5) Password Protected Diary
   Have a button that displays information (written in the server) on the webpage. For this information to display however, you must correctly write the password when asked for an input. If input is correct, a 'display area' will display all currently written notes in the diary. To input a new entry into the diary, have a way to input information that is saved by sending the new info to the server and will display next time the webpage is loaded up. Also have a button that locks the diary (removes the diary information from the webpage).

6) Random Number Memorization

Webpage will display a set of random numbers, one after another, starting with 1 random number. These numbers are saved on the server side (where the randomization of the number is also done), and the user must input them one by one in the order they are displayed. Each time that the user guesses the current order of random numbers, the webpage states they are correct, and adds a new random number to this set. Each time a new random number is added to the set, the webpage will again display the whole list of random numbers (not just the newest one).

For example, webpage displays (1) which was randomized on the server side. User inputs 1. Webpage now displays (1), and then (4). User inputs 1, and then 4. Webpage now displays (1), and then (4), and then another random number, etc.

7) Random Shape Memorization

Webpage has 4 shapes on it: circle, square, triangle, and rectangle for example (can be any 4 shapes). The webpage will highlight one of those shapes, and the user must click on it (highlight meaning make it clear that the shape is to be selected). If the user clicks on the correct image, the server will highlight the same shape, and one new additional shape. The user must now click on those 2, in the correct order. The server randomly determines which random shape will be highlighted and saves the order of highlighted images in an array on its server-side JS.

(OPTIONAL, does not count as a main project idea, this is just a side portion you can consider adding) No matter what the main functionality of your project is, you might want to add a side feature like any of the following features to your project:

I. Weather Dashboard. Use an API to display information about the client's nearby weather. Location can be found using Browser API's and there are lots of available weather API's. You can make requests using Axios, an industry standard HTTP request package for JS.

II. Stock market app. Use an API to display information about stock information. Toggle on and off information about each stock.

III. COVID-19 app, showing local case count and other available information. User can toggle on an off information that they want to see. Graphs and charts are to be used.

IV. Use an API to show recipes to the user. User can search for recipes, and the API will return relevant information.