EECS1012 Term Project

This is a teamwork project. The size of the team is required to be minimum 2 and maximum 3 students per team. 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.

In this project, you will develop a Web Application (a website, that serves as an app) through a software development life cycle combining some of the best features of Waterfall and Agile software development models. Please note, our approach does not follow astrict Waterfall or Agile methodology; instead we provide a "taste" of building software in a group. Also note that some phases have milestones and deadlines; and if you fail a milestone, you receive 0 for all following deadlines.

Some definitions in this document:

Milestone: Milestone are effectively progress checks en route to a deadline. Some milestones may require you to complete an online form and other require at least one of your teammates must meet with your TA to review your team's progress. When you meet with TAs, you do not need to have a completed piece of work for that phase;you just need to check in. If you do note complete milestones, you cannot receive a mark for work submitted by the deadlines for that phase. If you are worried about your progress, contact the instructional staff.

Deadline: By this date you must submit the finished version of your phase deliverables to eClass. Any questions you have between Milestones and Deadlines can be emailed to the instructional staff.

TBD: Means "to be determined". If this is written for the date of a phase, that means the date has not yet been finalized. Once finalized, it will be posted on the course announcements and an updated PDF (with the actual deadline instead of 'TBD') will be posted.

Peer evaluations: We will dedicate some classroom time to code reviews and discussions of progress on projects. Stay tuned!

This project isn't supposed to be stressful. Please contact the instructional staff if at any point the project is stressing you out, or if you need any assistance with it. The goal of this projectis to help you meet your classmates (which you will likely have for the next few years), and help you learn how working in a team for CS project is. **Most importantly, it's to help you explore your own creativity and interest in Computer Science.**

The project TA's contact information will be announced on eClass.

Appropriate email titles

For team issues: **'EECS1012 Project – Team Issues'**
For technical questions: '**EECS1012 Project – Technical Question'**

Project Phases:

Phase 1: Requirements + Design

Milestone: Week of September 29 (Progress Check with a TA).

Deadline: October 10. Worth 25% of Project Mark and assumes completed Milestones.

To complete the Milestone:

FIRST, Form a team. Make sure your team size is 2 or 3. No exceptions on team size.

THEN, choose a project Idea. Some ideas are listed at the end of this document. You should be mindful of your teammates' schedules, and as well of their interest in the course and project. Team formation is a very important phase of the project. There willbe challenges in almost every team/project; and only in a good team and with project management skills will you overcome the difficulties. Note that some students will have to drop the course or may find themselves failing in it, and therefore may not contribute to the project towards the end. You will need to carry out your project with or without them. If you are working alone at the end of the semester, the TA can help you figure out ways to lighten your load, so don't worry about people leaving your team. The projects are envisioned by the TA to be completable by 2 people easily, andby 1 person with some assistance and or revisions to the scope of work.

For a project idea, you can choose one of our suggestions or you can suggest your

own and see if the TA approves it. You are required to use the tools and technologies that we specify in this document and use in lectures and labs . No other tools/ technologies are accepted. This means: no java, only javascript, and no libraries/ functions/code that you are not learning about in class without permission from the instructional staff.

If you are having difficulty finding a project partner, complete the following form no later than September 26: https://forms.gle/NZr2QXNJ7RwnWqVt5.

FINALLY, if you find a project partner and have agreed upon an idea, **complete the following form no later than September 29.** Note that only one team member needs to complete this form (but all teammates' information should be included):

https://forms.gle/Tsa3V45obzyFybvg9

Finally, arrange to meet with the instructional team. Many meetings will take place during labs during the week of September 29. You can bring a draft of your project deliverables to your meeting, if you have begun working on them (see below for the details).

To complete the Phase 1 Deliverable:

Create a word document containing three parts:

PART 1: Project Overview. This should be **one page long** and should include:

- **Name of the team.**
- **Members of the team:** first name, last name, and York U email address as well as lecture and lab section of each team member.
- **Title of the project** (note this is not the same as the name of team) plus a short paragraph of describing the project. The number of words in your project description should be between 70 to 120.

PART 2: Software Requirements. This should be **1-2 pages** and include:

- **10 to 15 *Functional (or Software) Requirements***, written as "user stories" and contained in a bulleted list. In general, there should be about 5 requirements for each member of a team. Each "user story" should be up to two lines long and use the following syntax: ***As a <user type>, I want <some goal> so that <some reason>.*** Notice these stories are written in PRESENT-TENSE; there is no use of words like 'will be' or 'should'.

  Some examples of user stories might be as follows:

  As a user, I want to be able to view my bill in HTML format so that it can render in my browser.

  As a user, I want my website to have a green and yellow colour scheme so that it reflects the company's logo.

  As a user, I want a clickable logo in the top left corner of every web page so that I can easily use this to return to the home page.

  To learn more about user stories, read https:// www.pjsrivastava.com/a-short-guide-to-writing-software-requirements. This describes very simple Software Requirements; for your project they can be even simpler. You will however see that even small projects can require a lot of work!

PART 3: Software Design. This section will contain drawings or wireframes to show the design of your interface. The designs should show how each page at frontend will look like, AND how they will interact with one another (navigation with arrows). Your wireframes should be Lo-Fi (no colour, and as concise as possible). You can make them using pencil/pen and paper, or using any application you wish (e.g., Figma). Avoid Hi-Fi wireframes.

You can start learning about wireframe designs 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/

In your document, after requirements, include:

- **One page** that includes **thumbnails of all proposed web pages.** Connect all your thumbnails with **arrows**. All thumbnails must have at least one arrow pointing into them, and at least one arrow pointing away them. This is intended to illustrate how you will move around the website. The arrows must start from the **EDGE of the element or elements** that one clicks to navigate from a given page and the arrows must end at the **EDGE of the web page** or sidebar or popup that opens as a result.

- **Include "close up" images** of 3-10 web pages, Lo-Fi style. Note that you might have popup boxes, or dropdown menus/sidebars. You should draw these as their own small boxes/rectangles/etc. Do not drawn any images atop of your webpage designs to avoid clutter.

- **Title all drawings** that you include in your designs. Refrain from unnecessary detail in your drawings, but provide enough detail so that the instructional team has a sense as to the size of your project and the user's projected experience interacting with it.

All team members should have a good understanding of the requirements and design of the project before you begin programming.

Again, make sure to keep note of any interesting and/or fruitful moments that you have during this phase. In the final phase of the project, you will be asked to deliver a short, professional video (2 to 3 minutes long) that details your learning process, technologies used, learning outcomes, and final result. You can include any reflections in that video

Only one team member needs to submit phase 1 documents, but that person must remember to include a cover page that lists all contributors.

Phase 2: Implementation, Testing and Documentation

Milestone: November 14 (Progress Check with a TA).
Deadline: December 3. Worth 75% of Project Mark and assumes Completed Milestones.

In this phase, you will implement your project, then test and document it. **The implementation will be worth 40% of the project; documentation is worth 20% and tests are worth 15%.**

To complete the Milestone:

Once again we ask that you touch base with a TA during labs. Project meetings will take place during labs during the weeks of November 4 and 11. You can bring a draft of your project deliverables to your meeting, if you have begun working on them (see below for the details).

To complete the Phase 2 Deliverable:

1. implement a **frontend** (A.K.A. client side) and **backend** (A.K.A serverside) for your project using only the following: HTML, CSS, JavaScript, Node.js, jQuery,and optionally React. **Your website appearance should reflect your user stories design in Phase 1.** If you want to change your Phase 1, you must get approval from the TA first, else you will lose marks for P1 and P2 not matching. If you have no server side JS file in your completed work, you will also lose marks.
2. Your code must be **clear** and **properly commented**.
3. Students who use any other technologies or libraries other than what is directly taught in the course will receive 0 in this phase. Consult with the project TA during this phase and before submitting your final product. We should be able to ask where a specific idea for code came from how this relates to a lab/lecture from the course.

To test your project, you must:

Create **vitest testing files** that target at least 3 JS functions. The tests should be useful, and should test unique functions of your project.

To document your project, you must:

Develop a 2 to 3 - minute video and a 2 to 3 page document to show your final product and learning process. **All teammates should participate equally in this phase. If you are not in your team's video demonstration, you will lose marks for this phase. If the video exceeds three minutes, you will lose marks as well.**

Within the video:

- Each member must introduce themselves **clearly**.
- Members must demonstrate the project and explain the unique features of your web

app.
- Members should reflect on '**fruitful moments'** as they relate to lectures, labs, office hours for the project, project group work sessions, peer discussions, self-study, etc. How has this experience helped you learn how to work in a development team?

Within your document create these sections:

- **Roles**. Provide the names and IDs of contributing group members and the role each individual played in the implementation. Also include any information that you feel the instructional team should keep in mind when marking your project.
- **Revised User Stories and Wireframes**. If you had to change the design and user stories you submitted during phase 1, you can update them and resubmit. If everything remained the same, you can include your old Phase 1 here.
- **Learning Outcomes**. Tell us what you learned from your project and how this relates to the course learning outcomes (CLOs). If you do not remember the course CLOs, refer to the course syllabus.
- **Challenges**. In this section, discuss the challenges you faced and how they were overcome. Explain what was learned during the project. List some possible improvements or extensions to your project that you might like to make if you had time.
- **Use of AI**. If you and your team didn't use AI at all to help you with your project, explain to us why. Were you concerned that your brain might atrophy or that the energy use was just too much? Were you concerned about intellectual property considerations and that AI is inhaling all your hard work for its own gain? But if, instead, you did use AI tell us how ... as we really want to know! Did AI to explain code to you or help you optimize? What prompts did you use? Which were successful, which were not? Was one AI better than another for your purposes? Was the energy you invested in AI worth the energy your prompts consumed? Did AI compromise your ability to learn? We are all ears.

Submit your phase 2 deliverables as a ZIP file containing:

— *The **code and tests** you have written with your team;*
— *Your **demo video;***
— ***Your write-up, in a file called 'report.pdf'***

Only one team member needs to submit phase 2 documents, but that person must remember to include a report with everyone's name in it.

Some Sample Suggestions for Project Ideas

Here are some project suggestions. You are encouraged to define your own project too, but you must consult with the TA in advance and get their approval of your idea, before beginning to work on the project.

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 start another round of play. Keep a record ofhow many times someone has guessed the correct number in a row. Congratulate the user on their record when they lose. You determine what X is each round (this requires an input that will be sent to the server side). You may change the game by limiting the number of attempts each user is allowed to guess a given number, too.

2) Mini-Database (definitions)
   Define words for the user. When a user inputs a word, have the server either display information about the word on a 'display area' or let the user know that the definition for that word does not exist in the database. Make a list of all words that are available in the 'database' on the webpage. Make sure user queries are case insensitive (i.e. capitalization should not matter)

3) Password Protected Diary
   Protect information (in the server) on a webpage using a password. Allow information to be displayed only if a user correctly writes a specific password when asked for an input. If user input is correct, display all currently written notes in a diary to a 'display area'. Allow users to input a new entry into the diary, assuming they have authenticated. Also have a button that locks the diary (removes diary information from the webpage).

4) Random Shape Memorization (Simon Says)
   Make a webpage with shapes on it: circle, square, triangle, and rectangle for example (can be any set of shapes). Highlight one of those shapes, and ask the user to click on it. Assuming the user clicks on the correct image, highlight the same shape and one new additional shape and ask the user to click on these 2 inthe correct order. Assuming the user clicks on the correct images in the correct sequence, highlight the sequence again and add one new additional shape to the series. Allow the server to randomly determine which shape will be added to each sequence and save the order of highlighted images on the server-side.

5) Mini-Database of Images

Illustrate an object for a user. Allow a user to input an object they would like displayed on the screen, and have the server respond by sending an image of that object. Display the object on a 'display area'. Present a list of all images/objects that can be illustrated on the side of the webpage. It is a good idea to have all images be related (all animals, all construction building tools, all soccer players, etc.)

Optional: No matter what your project is, you might want to add features from an external API like:

I. Weather Dashboard. Use an API to display information about the client's nearby weather. There are lots of available weather API's.

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

III. Recipe app. Use a recipe API to allow a user to search for recipes.