Group Project

1. Introduction:

In this part of the course, you will work with a team of 4 members to ideate, design, develop and implement either a web-based system or a web-based game. For simplicity, we will use the term '**web application'** to refer to any of these two options. The goal of this project is:

- Tie up all the topics we learned during the semester
- Utilize these topics to complete a project that you are passionate about
- Conduct online research looking for additional tools, libraries, frameworks and share what we learned with the rest of the class
- Share project milestones with the class
- Practice teamwork skills further
- Improve presentation skills and demoing development work

2. Project requirements

Your project idea MUST satisfy these requirements:

2.1. Client side

The majority of your application's functionality must be accessible through a single-page interface. This will make your application feel more like a traditional desktop application rather than a website. The client side must be implemented using HTML, CSS and JS technologies. You are free to use any open-source libraries in the implementation phase. Keep in mind that your client must support at least Chrome and Firefox.

2.2. Mobile Support

The application must be usable on mobile devices. You may choose to follow a mobile friendly or mobile first development approach.

2.3. Server side

Most of the application logic should be handled by the server side. A good way to achieve this is to think of the client side as a GUI frontend for the server component. Therefore, you will end up relying heavily on JS combined with asynchronous communication with the server via WebSockets, or Ajax. In any scenario, the server side must be implemented using Node.js

2.4. Multiple users support

Your application must support multiple simultaneous users, at minimum it needs to support 2 different users. Therefore, your application should implement at least a very basic user authentication.

2.5. Persistence

Your application needs to demonstrate persistence capabilities. The most common way to implement this requirement is to give users some way of saving and resuming their work.

2.6. User interaction

Clearly, users should be able to interact with the system using typical input (mouse, keyboard, touchscreen, etc.). In addition, users must be able to interact with other users in the system. Depending on your project idea, users may interact with each other in real-time (game, chat, shared drawing canvases, etc.) or not (commenting on other's work, leave or retrieve messages, edit work, etc.). If you doubt that your project satisfies this requirement, check with your TA or the course instructor.

3. Examples for web applications from previous semesters

You are encouraged to be creative and come up with new ideas for this project. If you are having a hard time coming up with new ideas, below is a list of common project concepts from previous semesters. Feel free to adopt one of these ideas and modify it based on your own vision.

- Online image editor: users can either draw or upload and modify images. Multiple users can access the shared editor to work together.
- Server, API and client for data management. Data is stored on the server side and the client communicates with the server using a REST API that you will write. The client-side support two user categories; general user (request data, display visualisations, export data), and data scientist (input, update, delete data).
- A file sharing application with linkable files. Users can upload and manage files and create public links. The site could monitor each link's usage.
- A real-time online video game. Users could play against a computer or against each other. The game doesn't have to be complicated but it should be more advanced than tic-tac-toe. Reversi, Connect-4 or stack-4 would be the minimum expectation. If you can make the next AmoungUs that would be great for your online portfolio 😊
- A small social media application

4. Suggested workflow

You may follow a User-Centered Design (UCD) approach to structure your workflow. Because of the time limit, understandably, you don't need to work with potential users nor stakeholders.

Therefore, you and your teammates can play the role of the stakeholders. A typical UCD process in an academic setup follows these steps:

- Investigation: if you are addressing a specific problem, you need to further investigate this problem by conducting brief research or gathering some primary or secondary data. If you are suggesting improving an existing system, why would your solution be better? If you are introducing a novel system, who are your potential users? How would they benefit from your system? The investigation allows you to gather more information before you ideate.

- Ideating: in this phase, you start to put some ideas on paper, mostly related to your User Experience design and the user task flow. How do you expect your users to interact with the system? What are the supported functionalities? etc. At a later stage of this phase, you might draw some sketches and diagrams illustrating your User Interface (UI) design and your user task flow.

- Prototyping: one of the most common mistakes among developers is to start coding before prototyping. Coding is more time consuming than prototyping and it is very common to waste hours debugging a piece of code that you might completely give up on. It is always wise to use a prototyping tool to try your system before your write a single line of code.

- Development: design your code before you write it. Use diagramming tools to breakdown your project into classes and functions. Map out all the relationships. Follow an Agile software development approach to divide the workload into small light-weight tasks. As a team member, pick the tasks that you are best at. If you see that your skills shine more at front-end work, pick more tasks related to front-end and design. If you are the person who enjoys seeing a black screen with code lines and you hate having to pick colors and images 😊, pick back-end tasks. If you are a good presenter, volunteer to take care of presentations. The point is, use your team's resources wisely. That does not mean some team members have to do more work than others.

- Evaluation: don't waste your time evaluating your system with potential users. Evaluating your system with your team members is sufficient for this project. Take notes or develop a to-do-list during the evaluation process. Finally, address the issues you found in the evaluation.

5. Deliverables

5.1. Project proposal

Before you start working on your project, you need to submit a written proposal. You will submit the proposal to D2L dropbox for a review. Your TA will read the proposal and decide if your project idea meets the minimum requirements for this course component or not. Your TA may give you feedback to improve your proposed idea; either add or remove project components. Once your TA gives you the approval, you can move on to the next task. Your proposal should not exceed 2 pages in length and should not be less than one page (600 – 1000 words). Feel free to add an appendix if you want attach images, tables, sketches, etc. Structure your proposal as follows:

- Introduction: state the overall purpose of the application and the objectives. Take a full paragraph to demonstrate your points. Avoid writing a short and incomplete introduction (ex. 1-2 sentences paragraph style). Keep in mind, writing introductions is one of the most challenging parts of writing (too little and you miss the point, too much and you confuse the reader). By reading this part, your TA should have a full idea of what this project is about.
- User requirements: identify your primary and secondary user groups and their needs. Describe examples of your users from a high-level. Feel free to user persona if you want.
- Functional and implementation requirements: Identify the major tasks and functionalities that your application will support. Give examples.
- Proposed technologies: describe the technologies that you intend to use in your application. List each technology and explain its purpose in your application. (e.g: socket.io: we will use this library to create server side and client side websockets for our real-time game….. Bootstrap: we will use this library to create responsive mobile-first ……)

Your TA will use a binary logic when grading this project component. You will be given full marks for this part of the project upon your TA's approval. In the case of disapproval, your TA will give you feedback and provide you with a new deadline to resubmit the proposal. This new deadline will be relatively short (1-3 days depending on the extend of the feedback). Future project deadlines will not be adjusted based on your new proposal deadline. You are encouraged to submit your proposal earlier than the initial deadline if you want to move faster into the next phase of your project.

5.2. User Interface mockups

The purpose of this milestone is for you to start working on your UX and UI designs for the application. You will submit a recorded video presenting your system design. Your video

presentation should be 5-7 minutes. You may use the Balsamiq provided licence to create wireframe prototypes. Adobe XD is a step-up as it allows you to bring your prototype more towards the high-fidelity side. If you don't wish to use any of these tools, you may use PowerPoint to create these mock-ups and use the hyperlink feature to move between pages. If you are already familiar with another prototyping tool, then feel free to use it. We will create a discussion board on D2L where you post your presentation so other classmates can see it. You are encouraged to watch other team's presentations and provide some feedback. This project component will be graded based on the work quality. Your TA will use a provided rubric to grade your presentation, therefore, you must submit the presentation to the D2L DropBox as well.

5.3. Final project submission

After you move from the mockups phase to the development phase and complete all coding work, you will need to prepare a video demo and a final report. You will zip your project folder (containing all of your coding work) and submit it to D2L. Your project's ReadMe file should explain in detail how to run your application. Your demo will be relatively short where you should focus mainly on demoing your final application. Keep your video between 3-5 minutes in length and post it on final project discussion board on D2L. Once you are done, make yourself a nice beverage and watch other team's demos.

As for the project report, please follow this structure:

- Title page: include the course name, your group members names, a title for the application, if applicable, your course instructor's name and your TA's name.
- Introduction: describe your project objectives and major features and functionalities from a high-level. Write like a senior SENG or CPSC student and avoid short 1-2 sentence descriptions. End the introduction by providing an overview of the report content. This is your final project's report introduction, take it seriously.
- Background: provide a general background related to your application. Think of topics such as why your project is useful, who are your potential users and stakeholders, etc. More importantly, what motivated you to complete this project.
- Project goals: what are the overall goals of your project.
- Project accomplishments: describe what the users are able to accomplish by using your current implementations. Give a few examples.
- Detailed project description: you can think of this section as a brief user manual for your project. The intended audience would be your users. You should provide a description for each task that a user can accomplish accompanied with appropriate screenshots and explanations. Indicate details for all user categories. Implementation details are not needed in this section. Make sure you document all UI elements that you created: screens, dialogs, widgets, etc. provide explanations for all of these elements.
- Mockups vs final project: briefly describe major changes from the mockups stage to the final project. Use screenshots to illustrate your point.

- Project requirements: create a table or a list for all project requirements. Then, provide an explanation of how your project meets or doesn't meet these requirements.
- Technology used: describe which technologies you used in your project, why and how you used them. Provide brief examples. Add some instructions on how to obtain these technologies in order to run your application (technical details).
- Future work: if you were given more time to work on your project, what you would add or change?
- Lessons learned: what are the lessons that you learned from this project and want to pass onto others. (e.g: using this library didn't work, we used this one instead and it helped us in achieving X, Y, Z faster).
- Conclusion and miscellaneous: write one paragraph to conclude your report.
- Page limit: keep your report between 8-12 pages excluding the appendix.

5.4. Submission instructions:

You are required to have all your code related files organized inside a zipped folder titled with your group number. Please add a ReadMe file with a link to your GitHub repository and clear instructions for your TA on how to run your application. Failure at delivering clear instructions in the ReadMe file will result in marks deduction. Submit the zipped folder, final PDF report, and the video demo file to the final project dropbox on D2L before the deadline to avoid late submission penalties. In addition, you are required to post your demo video on the discussion board on D2L.

**Important note: you are required to use Github online repository from the beginning of your development work. You will submit a link of your Github repository in the ReadMe file. This is a crucial deliverable especially in the case of unforeseeable team conflict.*

***In the case where a team member(s) does not contribute to the project development, inform your TA as early as possible so that an investigation will be conducted at the end of the project. This investigation will include: Github repository, peer-review evaluations, and evidence of lack of communications (discord chats, MS teams, etc). Upon concluding the investigation, we will adjust the group grade accordingly.*

6. Additional task, technical presentation

In fifth level courses such as SENG 513, we work with students to improve their research and presentation skills. Over the past two months, we have explored several topics in web-based system development. However, we are far from covering all topics in this context. It is part of your project to

conduct brief research about web-based system topics that may or may not be related to your project. For the purpose of exchanging knowledge and improving your presentation skills, your team is required to give a 15-20 minute presentation about a web-based systems' related topic that was not covered in the lectures during the course. We will use the remaining lectures' time for these presentations, and we will have three presentations per lecture (3 * 10 lectures = 30 presentations which is the total number of groups). There is an Excel sheet containing two tables on D2L with all available topics and some empty spots for other topics that you may want to cover. Each group must pick a topic and a presentation day. There are plenty of topics to choose from, but to assure good topic diversity we are restricting to a maximum of two presentations per topic. If you wish to add a topic that is not listed, feel free to use one of the slots at the bottom of table 2. As for the booking, it will be on a first come first serve basis. Please use table 1 to book your presentation slot. As an audience, you are required to watch these presentations during the lecture time and ask any questions you may have on the discussion board. As a presenter, you should keep an eye on your post and answer your classmates' questions for at least 2 days.

*** Don't delete or modify other group input to this table. All changes are tracked by the SharePoint and any changes to other group input will be found, reversed and the violators will be penalized.*

Summary			
Deliverable	Required Submissions	Deadlines	Grade (total course grade)
Technical presentation	15-20 mins video presentation posted on D2L	Based on group booking	10%
Project proposal	One PDF file submitted to D2L dropbox	13th of March	5%
UI Mockups	5-7 mins recorded presentation posted on D2L and submitted to D2L dropbox for grading	20th of March	10%
Final project submission	Demo posted on D2L, all coding work with a readme file and a link to GitHub repo, PDF final report submitted to dropbox	12th of April	35%

Late submission penalties (applicable to all project milestones):

Students are required to plan their course work properly and utilize their time wisely. In addition, it is the student's responsibility to submit their coursework task using the required file format for that task accordingly (code files, video file, PDF, slides, etc.). Failure at submitting the required file format will result in a complete loss of marks for that course work component. As for any late submissions, failure at meeting a deadline will result in a late submission penalty as follows:

a) 1 to 24 hours late: 20% deduction penalty

b) 24 to 48 hours late: 40% deduction penalty

c) 48 to 72 hours late: 80% deduction penalty

d) More than 72 hours late: student/team will be given 0

Declaration of credits and academic honesty

This project was designed using materials from final student project, SENG 513 course, Winter 2017 by Dr. Pavol Federl, University of Calgary.

End of document