COMP[39]900 Computer Science/IT Capstone Project
School of Computer Science & Engineering, UNSW
2024 Term 3

Project Proposal Assessment

Assessment

- **Due Date:** Week 3 Sunday (29 September 2024) @ 9:00pm. Late submission attracts a penalty of 5% per day (i.e. 0.5 marks will be deducted per day).
- **Weighting:** 10%
- **Submission:** Submission is via Turnitin on Moodle. Please follow the instructions laid out on Moodle.
- The project proposal should be self-contained. i.e. **no content should be outside of the report and simply linked to**.

Purpose

- The main purpose of your proposal is to
  1. Identify and research the problem you are solving.
  2. Propose a solution based on the research you perform and justify how your solution is unique and overcomes problems you identify in existing systems.
  3. Set out the project management that you will use for the duration of the project (user stories, sprint milestones, project objectives).

Report Quality Requirements (5%)

- The project proposal must adhere to the following **formatting requirements**:
  - Include a title page containing course code, course title, project number and title, team name, each member’s name, email, student ID, role, and project proposal submission date.
  - Be at least 10 pages long using at most 12pt font with reasonable margins and spacing, not including the title page, the table of contents, and references page, and be in PDF format.
  - Include a table of contents and page numbers.
  - Include full references and in-text citations. Use either APA referencing style (https://student.unsw.edu.au/apa) or Harvard referencing style (https://student.unsw.edu.au/harvard-referencing).
- In addition, the report should be easy to read and concise.

Content Requirements (95%)

Your proposal should include the following.

(1) Background (15%)

(a) Clearly identify the **problem domain** and problem(s) being solved. Add a **brief summary** of your solution.

(b) Discuss **at least two existing related works or systems** in your identified problem domain. For each related work:

- Describe the problem their system solves.
- Identify and discuss any strengths of their work that you will take into consideration in your design.
- Identify and discuss any weaknesses or limitations of their work that you will avoid or overcome with your design.
- Make sure to reference all works you discuss following the referencing requirements stated in the formatting requirements.

(2) User stories and sprints (25%)

(a) Include a product backlog of correctly structured user stories, describing the functionality to be delivered, with **screenshots** showing all these user stories defined in **Jira**. A screenshot of your Jira **Backlog** page is sufficient, so long as the entire text of each user story should be readable inside the report.

(b) Identify the **user stories** in scope for the **first sprint** with **screenshots** showing all user stories allocated to the first sprint in **Jira**.

- To satisfy this criterion, you must **allocate a realistic amount of work** to your first sprint. If you have too few user stories allowed to the first sprint, you risk delaying the completion of essential features to stakeholders.
- A realistic amount of work is heavily dependent on your group and the chosen project. Although this is not heavily weighted in this assessment, it will help you greatly in a few weeks if you plan out work estimates well now.
- There are many ways you can ensure that a realistic amount of work is allocated:
  - (i) Assign story points to each user story on Jira and then allocate roughly a third of the story points to the first sprint.
  - (ii) Plan your sprint milestones and align the selected user stories selected with each milestone. If you chose to do this, make sure that each milestone represents roughly the same amount of work.
  - (iii) Create smaller deadlines for a set of features and then align user stories with these deadlines such that you maintain a consistent amount of work during the term.
  - Estimating the time required to complete user stories is an important team skill and it is important to commit to a demanding but manageable workload. Committing to too little work results in your final product will not be as developed as it could be. Overcommitting means you won't finish all your work by its deadline. Both of these scenarios will be taken into account in all assessment tasks within this course.

(c) Define the **start and end dates** for all **sprints** envisaged during the term, and briefly describe your **milestones** for each sprint.

- Note: Ensure that you complete your sprints immediately before the progressive demos in Weeks 5 and 8. This will allow you to hold retrospectives immediately after those demos.
- An appropriate sprint structure would be a 3 sprint structure with the following dates:
  - **Sprint 1**: Week 3 Lab Time → Week 5 Lab Time (2 weeks)

- **Sprint 2:** Week 5 Lab Time → Week 8 Lab Time (2 weeks excluding flexibility week)
- **Sprint 3:** Week 8 Lab Time → Week 10 Lab Time (2 weeks)

(3) Technical Design (55%)

(a) Include a **system architecture diagram** that shows the interactions between the core systems of your application.

- This can be in the form of a **UML diagram**, but a **simpler diagram** containing boxes with object names and **labelled** arrows between them can be just as effective.
- It is **essential** that your diagram contains **sufficient detail** so that the major components and modules of your system are shown. The diagram must also show the nature of the data flow between those components and modules. **It is not sufficient to merely outline the technologies that you plan to use at each layer.** Thus:
    - for web-development and other development focused projects, make sure your diagram clearly demonstrates:
        (i) how different components of the backend interact with each other.
        (ii) how the backend communicates with the frontend.
        (iii) how your software communicates with external databases, APIs, and other services.
    - for machine-learning and research projects, make sure your diagram clearly demonstrates the interactions between your models, the transfer of data to and from where it is stored, how external components such as a user interface interact with a backend that processes requests, and how the backend manages different models.
- **A component or module within the diagram should only be responsible for a small set of uses.** If you find that one component of the backend is handling multiple tasks (calls to the database, parsing information from the frontend etc), consider breaking it down into multiple smaller components.
- Some recommended websites to develop these are **Draw.io** (https://draw.io/) and **LucidCharts** (https://lucid.app/).

(b) Choose **ONE** of the following options**. Choose the option best suited to your project.** Choosing an option not suitable for your project may result in insufficient depth. If multiple options are chosen, only the first option included in the proposal will be marked.

- **OPTION 1: Interface storyboarding** (suitable for **development projects** such as web apps)
    - Develop **storyboards** to illustrate all major system functionality.
    - Not all user stories need to be covered but **all major functionalities must have storyboards**.
    - Mention which **user stories** are covered by each **storyboard** (one storyboard can encompass multiple user stories).
    - The storyboards **do not need to be high fidelity,** however, they should, at minimum:
        (i) Showcase the **design** of the frontend interface, including layout and basic colour schemes

        (ii) Demonstrate the **navigation** between different parts of the application through arrows or similar
        (iii) Clearly show **how each major feature** works and is intended to be used by users.
    - The requirements for your project will constantly evolve throughout the term but with well-researched storyboards, it is highly likely that the majority of features in your project will closely resemble your storyboards.
    - Some recommended websites to develop these are **Figma** (https://www.figma.com/) and **Balsamiq Wireframes** (https://balsamiq.com/wireframes/).

- **OPTION 2** (for machine-learning and research projects involving development of an algorithm or model)
    - Conduct a **literature review** to determine the current research methods relevant to your project.
    - This is **not the same as the background section** of your proposal. This section focuses on:
        i. **existing research** related to your chosen project.
        ii. the **novel methodology used** in this research and how you build on top of that with your own planned project.
        iii. the **evaluation metrics** used to evaluate their models.
        iv. the **limitations** of prior models and how you plan to overcome these.
    - **If there is no research in your specific problem domain** explain why this may be the case. Is it a unique problem that has only arisen recently? Has there been a new technology developed recently that has enabled new development?
        - Even if this is the case, you must still conduct a thorough literature review on adjacent research areas that will help you develop your own work.
    - It is also highly recommended that you add a **proof of concept** or some **storyboarding** that showcases how you envision your research being used and what the outcome of your project will ideally look like.
    - One good starting point for your literature review is to search for your topic on Google Scholar (https://scholar.google.com/) to find well-known papers online in your problem domain. Look at the literature review section of these papers. It is also very useful to read survey papers that establish and explain the state of the art within a specific problem domain.
    - To achieve a high mark for this, you should thoroughly cover **3-5 research papers**.

(c) Include a list of **design justifications** for your planned solution.
- This should include:
    (i) Breaking down the problem you have chosen into more manageable **subproblems.**
    (ii) A summary of how you plan to **solve** each subproblem.

(iii) For the more complex subproblems, brainstorm **alternative solutions** and **justify** why you chose the solution you did rather than the alternatives.
(iv) A discussion how your solution provides novel functionality beyond existing systems wherever relevant in your design justifications.

- Effective design justifications will incorporate research from the background section, identify how decisions will affect users and identify possible limitations in your design and how you will mitigate these.

(d) Clearly communicate how all project **objectives/functionalities** are satisfied by your **user stories**. This can be done through a table which communicates which functionalities are satisfied by which user stories.

Marking Criteria

To achieve the maximum mark for each of the criteria, you must not only include what is listed but also demonstrate research, planning and effective structure.

Category	**Max Mark**
Background (15%)	**1.5**
Clearly identifies the problem domain and problem(s) being solved	0.5
Clear evidence of research into existing systems in the same problem domain and understanding of the benefits and limitations of current software solutions	1
User Stories and sprints (25%)	**2.5**
Product backlog of correctly structured user stories, describing the functionality to be delivered, with screenshots showing all these user stories defined in Jira	1
Identifies user stories in scope for the first sprint with screenshots showing all user stories allocated to the first sprint in Jira	1
Defines the start and end dates for all sprints envisaged during the length of term. Identifies sprint milestones for each sprint	0.5
Technical Design (55%)	**5.5**
Effective system architecture diagram	1
Effective use of storyboarding to illustrate the layout of the application and all major functionalities **OR** Effective literature review covering 3-5 research papers establishing current work in the problem domain, evaluation metrics and limitations	2
Detailed outline of subproblems, list of design justifications, novelty of the solution and the mitigation of limitations	2
Clearly communicate how all project objectives are satisfied by the user stories that are defined	0.5
Report Quality (5%)	**0.5**
Report conforms to the specified formatting requirements and is easy to read	0.5
Total Mark	**10**