COMP3900/9900
Project Deliverables and Assessment

This document describes group formation, assessments, and other project course deliverables.

- Group Formation: A group will be five (5) students in size, with all members being in the same lab (unless there is no possibility to have 5 group members because of the size of the lab, in which case a group of 4 may be accepted with the mentor's permission). Please do register your group on the WebCMS3 course website under the Groups page and specify group members as well as the Scrum Master (as Admin in WebCMS3 Groups terminology).

- Roles: Each group should have a Scrum Master and four (or three) Developers. Their responsibilities are discussed in the lecture. Note that these roles are for accountability. We expect that all members should be involved in coding. Scrum Master may contribute marginally less coding efforts, e.g., 10% less, if he/she will administrate GitHub (having a Maintainer role) and Jira accounts for the team.

- GitHub: Each group should use their GitHub classroom repository to store and maintain the entire code-base used for their project, and ensure they keep commit history accurate to represent contributions made by each team-member.

Work Diary

Each student should maintain a work diary (call it z0001234.txt if your student zID is z0001234) and check the file into GitHub.

From Week 1 onwards, we expect that you will update this file every week with your new contributions to the team and the project. Check the file into GitHub at least once a week (can be more frequent).

The content of the file should be brief but clear. For example, it should be like:

```
Week 1

Group formed. I created the Jira & GitHub accounts. Together with John
Smith, I wrote the user story & sprints section of the proposal. I also
found and discussed with the team all available software tools and
libraries that we can use for the project.

Week 2

I wrote the first version of hello.c, api.h, api.c and drafted a design for
the Web service API between the client and the server – all by myself.
```

You should also include in this diary the following information (if applicable) about the project progress:

- what was planned for the period since the last work diary entry
- what was finished
- what were the main technical and non-technical obstacles and how they were overcome (or what was tried and did not work)
- what was not finished, why this happened, how this affects the overall project progress and what adjustments (if any) are needed so the success of the project is not endangered
- what is planned for the next period (e.g., next week)

We expect that other members of the team may access this file to know where others are up to, but you **should not modify your peer's work diary.**

The Proposal

- Project Proposal (due Monday Week 4 @ 9.00am, 21 June) (10%): Students will choose a project from a list of possible projects with a given description and project objectives. They will produce a proposal that describes: the background for their project, the product backlog to be used for satisfying all project objectives, an initial sprint backlog, user storyboards, and the system architecture. The requirements and criteria for the proposal are described below.

- Please make sure you also take a look at the WebCMS Proposal assessment submission page for submission instructions, and follow those submission instructions.
- Students may request to undertake a custom project only if 8 distinct project objectives of similar technical depth & scope to existing projects are clearly defined with the request. A clear project description should also be included with the request. Such requests are subject to mentor approval and amendment, and are to be sought prior to week 2 by filling in and agreeing to the terms on the custom project form that can be found at https://rebrand.ly/aje7pe4 .

- The proposal should be self-contained, (ie: ***no content*** should be outside of the report and simply linked to), and follow the following *formatting requirements*:

  a) Include a title page containing: project title; a nominated group name; each member's name, email, student ID, role; proposal submission date.

  b) Be at least 5 pages long (at most 12pt font with reasonable margins & spacing), not including the title page and references page, and be in PDF format that is readable with Acrobat.

  c) Include a table of contents, and page numbers.

  d) Include references and use either APA (https://student.unsw.edu.au/apa) or Harvard (https://student.unsw.edu.au/harvard-referencing) referencing style.

- The proposal should include:

  a) Background (10%)

     - Clearly identifies the problem(s) being solved.
     - Identifies existing work or systems in the same problem domain, and their drawbacks.

b) User stories & Sprints (50%):

- Product backlog of correctly structured user stories, describing the functionality to be delivered, with screenshots showing all these user stories defined in Jira. The entire text of each user story should be readable inside the report.

- Defines the start and end dates for all sprints envisaged during term.

  - Note: ensure that the sprints you define allow you to undertake a demo in each of weeks 5 and 8; as well as a retrospective before each of the labs in weeks 7 and 9.

- Identifies user stories in scope for the first sprint with screenshots showing all user stories allocated to the first sprint in Jira. The entire text of each user story should be readable inside the report.

- Clearly communicates how all project objectives are satisfied by user stories that are defined.

- Describes how some of the defined user stories provide novel functionality compared to existing systems.

c) Technical depth, scale, report formatting (40%)

- Report conforms to the formatting requirements specified above; and is easy to read.

- Interface and flow diagrams (Storyboards for user stories)

  - Storyboards should be developed to illustrate the system functionality, and how users interact with the system. One storyboard can cover multiple user stories. All user stories should be covered by these storyboards.

- System Architecture

  - A clear description showing the presentation, business and data layers in the system, and what each layer contains.

  - A clear description of the external actors (eg: user types) and how they interact with the system.

  - A clear description of the technologies/languages planned for use (eg: mysql, sql server, msmq, .NET, java, etc), including all third party functionality planned to be used (eg: clouds/services/APIs/libraries/code).

- Proposal is marked out of 10, contributes 10% towards the final mark for the project, and is marked according to the marking criteria below.

Project Proposal Marking Criteria

Category	Max Mark	Team Mark	Comments
Background (10%) **Problem domain, existing work/systems, and their drawbacks**	**1**		
Clearly identifies problem(s) being solved	0.5		Marks considered separately for this category
Clear evidence of research as to existing work or systems in the same problem domain and what their drawbacks are.	0.5		Marks considered separately for this category
User stories and sprints (50%)	**5**		
Product backlog of correctly structured user stories, describing the functionality to be delivered, with screenshots showing all these user stories defined in Jira	1.5		Marks considered separately for this category
Defines the start and end dates for all sprints envisaged during the length of term	0.5		Marks considered separately for this category
Identifies user stories in scope for the first sprint with screenshots showing all user stories allocated to the first sprint in Jira	1		Marks considered separately for this category
Clearly communicates how all project objectives are satisfied by user stories that are defined	1		Marks considered separately for this category
Describes how some of the defined user stories to be implemented provide novel functionality compared to existing systems	1		Marks considered separately for this category
Technical depth, scale, report formatting / readability (40%)	**4**		
Report conforms to the formatting requirements specified for the Proposal in this Project Deliverables and Assessment document; and is easy to read	0.5		Marks considered separately for this category
Includes clearly a detailed software architecture diagram	1.5		Marks considered separately for this category
Interface and flow diagrams	2		Marks considered separately for this category
Total Mark (out of 10)	**10**		

The Progressive Demos

- Progressive Demo A (Week 5 Lab Time) (2.5%)
  and Progressive Demo B (Week 8 Lab Time) (2.5%):

  Progressive demonstrations provide an opportunity to showcase user stories that have not yet been demonstrated, and how well you have developed functionality to support these.

- Marking criteria:
  a) Completed stories to be demonstrated are shown in Jira and described, with these stories having the correct status in Jira (Done) (1%)
  b) Demonstrates the functionality used to support each completed story (1%)
  c) Keeps to 12 minutes or less (0.5%)

- Group members absent for a progressive demo receive 0 marks for that entire demo.

The Retrospectives

- Retrospective A (Week 7 Lab Time) (2.5%)
  and Retrospective B (Week 9 Lab Time) (2.5%):

  A retrospective is a reflective activity, where the team meets to think about their team work process over the past sprint, and discuss: what went well, what didn't go so well, and what the team should try to over the next sprint to improve their work process. Additional aspects are also described in the marking criteria below. This meeting should follow soon after the sprint demo (usually in the same day). The deliverable for each retrospective should be emailed by the team scrum-master to your mentor and also submitted through WebCMS (*make sure the document submitted on WebCMS is the same one sent over email*).

- Marking criteria:

  a) Retrospective report has columns or sections to describe what went well, what didn't go so well, and things 'to try' next sprint to improve the team's work process (1.5%). (Empty sections/columns must have an explanation)

  b) A team member assigned responsibility for attempting to enforce or follow up on each item on the 'to try' list (0.75% for Retrospective A, 0.5% for Retrospective B) (If an item in the to-try list is not assigned a member - this must have an explanation)

  c) An outline of how effective 'things to try' from the previous retrospective were at improving the team work process (only for Retrospective B) (0.25%)
  (If this outline is empty, there should be an explanation for why it is empty)

  d) Date, time, and team members present or absent at the retrospective (0.25%)

- Group members absent from the retrospective meeting according to members present/absent list, receive 0 marks for the retrospective.

- Please make sure you also take a look at the WebCMS retrospective submission pages for submission instructions, and follow those submission instructions.

The Software Quality

- Software Quality (Monday Week 10 @ 10.00pm) (20%): The submission for this assessment should include your entire codebase that you've developed for the project. This assessment is mainly for the scale and technical depth of the delivered implementation; the correctness of the implementation; its value or its novelty; its performance (e.g., is it too slow for its intended usages); clarity of your code, its design (including interface design), its structure and its organization; and ease of use. The marking criteria to be followed is shown below.

- ***NOTE: The final complete system your team submits will need to be able to be built/compiled (if language used supports this), configured, setup, run, be usable and work on the one of the 2 environments below:***

  1. *The CSE vlab machines: https://vlabgateway.cse.unsw.edu.au/*

     OR

  2. *On the Lubuntu 20.4.1 LTS virtual machine image as described here: https://rebrand.ly/xvsoiua*

     If using this second virtual machine option to host your system, do NOT include the virtual machine as part of your submission, but rather specify in your report that you are using this virtual machine option.

     We will take the software artefacts and setup scripts/instructions you submit and use them with this virtual machine.

- Please make sure you also take a look at the WebCMS Software Quality assessment submission page for submission instructions, and follow those submission instructions.

  If you have any issues making your submission through WebCMS, or if your submission exceeds 100MB:

  1. make sure you create a zip file of your submission with file name:
  <TeamName>FinalSoftwareQuality.zip
  (where <TeamName> is replaced with your team's name)

  2. Use the command line to push this zip file to your team's GitHub classroom repository by the deadline for this assessment.

  3. Follow instructions on the WebCMS submission page to submit a zip file that includes a Readme.txt plain text file. This text file should mention that you have uploaded your final submission to your team's GitHub classroom account on time (commit history should reflect this), and include a link in this text file to your submission on GitHub. Also email your mentor to let them know that you've taken the GitHub approach to submission.

  Note: If your submission exceeds 100MB, you will need to use git-lfs to push your submission to your team's GitHub account: https://git-lfs.github.com/

Software Quality Marking Criteria

Category	Max Mark	Team Mark	Notes
Technical Depth and Novelty (45%)	**9**		
Implementation far from completion	2.3		Mark falls within one of the categories to the left
Complete implementation according to the scope of all project objectives without solving technical challenges	4.5		
Complete implementation and solving some technical challenges	6		
Completed with some degree of technical novelty	7.5		
Completed, with good degree of technical novelty, and functional novelty	9		
Correctness and Performance (30%)	**6**		
Unacceptable performance, buggy even with a few tests	1.5		Mark falls within one of the categories to the left
Overall correct but slow	3		
Overall correct and efficient	4		
No issues during demo and project testing (by the assessors)	5		
Robust and excellent performance	6		
Code Style, Structure, and Readability (12.5%)	**2.5**		
Messy code structures, difficult to read	0.7		Mark falls within one of the categories to the left
Readable but not organized	1.3		
Code is well structured and readable with some documentation	1.7		
Well structured and easy to read with ample documentation	2.1		
Easy to read, well documented, and demonstration of excellent coding style and practice	2.5		
Interface and Usability (12.5%)	**2.5**		
Primitive interface and difficult to use	0.7		Mark falls within one of the categories to the left
Poor interface design but still usable	1.3		
Generally good design with usability issues on some use cases	1.7		
Generally good design and ease to use in all aspects	2.1		
Professional interface design and excellent usability	2.5		
Total Mark (out of 20)	**20**		

The Report

- Project Report (Monday Week 10 @ 10:00pm) (20%): The project report should be prepared according to the instructions below and should at least include the following information:

1. Using the similar format as the project proposal, your report should be at least 15 pages long, excluding the title and references pages, and be in PDF format that is readable with Acrobat on the CSE workstations.
2. Title page similar to that of the project proposal, with the project submission date.
3. Table of contents and page numbers.
4. Overview - Architecture / design of the overall system & functionalities.
5. Descriptions of the functionalities developed by the team and how they map/address all project objectives.
6. Proper references and brief descriptions of ALL third-party functionalities (clouds/services/APIs/libraries/code) used by the team, with justification for their use and discussion how their licensing terms impact (or don't impact) this project.
7. Implementation challenges: descriptions of any tricks, non-trivial algorithms, special architecture/design, etc.
8. User documentation/manual: how to build, setup, configure, and use your system and functionalities.
9. Use either APA (https://student.unsw.edu.au/apa) or Harvard (https://student.unsw.edu.au/harvard-referencing) referencing style.

The marking criteria that follows will be used for the report.

Report Marking Criteria

Category	Max Mark	Team Mark	Comments
Overview (10%)	**2**		
Fails to present the overall picture (design and architecture) of the project	0.4		Mark falls within one of the categories to the left
Provides vague/insufficient design and architecture descriptions	0.8		
Provides clear design and architecture, but has weaknesses or technical issues with them	1.2		
Provides clear and correct design and architecture	1.6		
Provides concise and professional presentation of design and architecture	2		
Functionalities and Implementation Challenges (50%)	**10**		
Clearly deficient, lack of any useful details	2		Mark falls within one of the categories to the left
"Thin" results, lacking intellectual engagement, lack of justifications	4		
Several functionalities of the software not coherently linked	6		
Solid, coherent work, linking all the functionalities together into a consistent story. Good description on solving difficult technical, research, or implementation issues	8		
Outstanding, coherent and consistent functionalities; discussion of third party functionality licensing; and description on solving difficult technical, research, or implementation issues	10		
User Document/Manual (30%)	**6**		
Insufficient / incorrect instructions to compile, build, setup or use the software	1		Mark falls within one of the categories to the left
Unclear instructions but can still follow to build and run the software	2		
Easy to follow to build and setup. Some functionality documentation, but not enough information to cover all the functionality usages	4		
Complete and correct instructions	5		
Professional and concise instructions (correct and complete)	6		
Document Presentation, Title Page, References (10%)	**2**		
Impedes document reading or missing sections	0.4		Mark falls within one of the categories to the left
Poor formatting and document structure	0.8		
Poor judgement with respect to layout and possible padding	1.2		
Minor issues, but overall high quality	1.6		
Professional, easy to read and high quality presentation (such as layout and design)	2		
Total Mark (out of 20)	**20**		

The Demonstration / Presentation

- Project Demonstration/Presentation (During Week 10 Lab) (20%): Each group should prepare to give a live 15-18 minute presentation about the final outcome of the project. This should include a demonstration of the system built , as well as a presentation of the other project aspects described in the demonstration/presentation marking criteria below.

  Each member should contribute to one part of the demonstration/presentation (ie: each member should speak during the demonstration/presentation).

  Any member that is **absent or does not speak** during the demonstration/presentation **receives 0 marks** for the demonstration/presentation assessment.

  Please note that UNSW community, but also outside experts (e.g., from prospective employers) may be invited by comp3900/9900 staff to project demos.

  We adopt guidelines similar to the CSE Honours thesis system which have been adjusted to better fit COMP3900/9900 course which is project focused.

Demonstration/Presentation Marking Criteria

Category	Max Mark	Team Mark	Notes
Technical Quality and Completeness of the Project as Demonstrated (70%)	**14**		
Complete, fully functional, correct and coherent demonstration/presentation by all team members, covering all project objectives.	6		Marks considered separately for this category
User interfaces are well designed and working without issues	4		Marks considered separately for this category
High technical quality, demonstrating excellent engineering practice, and solid methodology	4		Marks considered separately for this category
Structure and Delivery of the Demo/Presentation (30%)	**6**		
Demonstration is well prepared, and confidently and professionally delivered	2		Marks considered separately for this category
Demonstration is well structured with evidence of good team work	2		Marks considered separately for this category
Q and A handled well	1.5		Marks considered separately for this category
Adherence to demo/presentation time requirements	0.5		Marks considered separately for this category
Total Mark (out of 20)	**20**		

The Peer Assessment

- Peer Assessment (Friday Week 10 @ 10.00pm) (20%): Each member's contributions to the project are evaluated based on 3 components:
  (1) the participation records from GitHub, Jira, and Lab Progress / Demonstrations;
  (2) your claimed contributions in your GitHub diary;
  (3) a rating of each member from their peers (i.e., Peer Assessment).

For Peer Assessment, each member must submit a text file (namely peers.txt). Each line of this file contains the zID of your group member, followed by a space, followed by an integer score from 0 to 10 (full mark). Please do not include yourself in the file. The score is to indicate the relative efforts and contributions to the effort, from your perspective. For example, a sample file looks like:

```
z1000123 8
z1234567 8
z1234123 10
z2468123 4
```

If you believe that the first two members contributed well to the project, and the third member is the critical contributor, whilst the last person has done unacceptable work for the project. Remember, you are not one of them on the list.

Note that the scores received for a group will be aggregated and consolidated. Furthermore, the deviation of the scores matter (whilst the actual scores do not). For example, every member having the same score of 7 is the same as every member having the same score of 3. Similarly, it will have the same effect to the score of every member of the group if the above sample file is changed to:

```
z1000123 4
z1234567 4
z1234123 5
z2468123 2
```

If extreme scores are obtained within a group, records on GitHub and Jira will be used to substantiate these scores. Therefore, please keep the GitHub and Jira accounts for at least a few weeks after you receive the course grade from COMP3900/9900.