COMP 2606: Software Engineering
Project Details and Guidelines

Purpose of the Project

The project aims to give students experience of working in small groups and participate in some of the development phases (requirements analysis, design, planning tests and documentation) of a nontrivial software system. It also gives valuable teamwork experience. The project assesses your ability to select a problem analyze and propose solutions then design and plan tests for the software system. It also considers the risks and costs involved in developing the system.

Group Formation and Size

Students are expected to choose their own group members from amongst the students registered for this course. A group size of 4 is recommended. However, exceptions may be made for large complicated projects, pending approval from the course supervisor. Group sizes exceeding 5 will not be permitted.

Group Member Roles and Contributions

Project groups often function more effectively when group members have designated roles. Three roles and responsibilities are:

Project Lead: responsible for keeping the group on task, distributing the workload, meeting deadlines and ensuring smooth group communication and coordination internally as well as accountability with the course coordinator and project requirements.

Documenter Lead: responsible for recording group discussions and decisions, documenting various aspects of the project's progress, and ensuring well-formed reports and project documents are produced.

Designers: responsible for overseeing the collaborative design aspects of software solution, troubleshooting major problems related to design, and ensuring document reviews, merges and proofing are done well.

The entire project team should be engaged in discussions, documentation and development of the project. All members are expected to contribute towards the project. Groups will also have the option to rotate the roles among members for different stages of the project. This will allow members to gain experience through being responsible in different areas of project management.

Project Selection and Proposal

Students are advised to start exploring their options early, and to select a suitable idea that is of interest by the designated date. You can use the Internet for inspiration, but a better idea would be to choose something you always wished to do. Students may propose an idea of their own. You cannot choose something you are already working on. In choosing the project, think about the group's skills, and whether you have (or can obtain) the knowledge to develop that project. Proposed project ideas require approval. Only ideas that have been approved can proceed further.

Managing Project Execution

Project management deals with organizational matters that are needed for effective teamwork. An identified group leader can take the lead in the coordination of project activities, or the group can decide to share coordination responsibilities. Project management includes the following:

- Organizing group meetings and keeping track of deadlines
- Managing shared resources (such as documentation, design documents
- Integrating individual contributions in a coherent manner, and resolving ambiguities and conflicting information including: collating different sections into a project report representing the work of entire group
- Ensuring that the project blog/ website and reports are proofread, collated to ensure consistent layout, font styles, sections, and language style
- Other matters relating to the group's dynamics

Document Development

Groups are encouraged to adopt agile software development methods as the practice is relatively simple to adopt and provides greater project visibility, predictability and flexibility. It also encourages development of high performance teams and early prototype.

A GitHub repository is required for each group project to facilitate project management and keeping track of design changes. The repository will also be used by the lecturer to monitor, review and assess components of each project throughout the course. Note: students should create their own GitHub accounts if they do not have one.

Weekly Meetings and Status Reports

Meetings: Groups are required to meet once every two (2) weeks with the lecturer to provide updates on the progress of their project. Apart from offering advice the job of the lecturer is to check that you are working properly as a group and that you are/can get on target to meet deliverables. The status of your progress (planned vs actual) must be kept up to date on your website. Also, the link to the Weekly Status Report must be shared *prior* to the meeting. **Attendance of *all* group members at *all* meetings count in the assessment**.

Final Project Report

The final project report combines all the reports presented at meetings as well as additional information about the project.

Final Presentation

Your presentation should be approximately 5 minutes long, acting as a pitch to the company you targeted. That is, use your presentation to convince the company that what you are recommending is important and will aid in the profitability of the business.

Your presentation should outline:

a. The problems with the current system that lead you to want to develop and design this software solution.
b. The major functional and non-functional requirements of the proposed system.
c. The overall design of the proposed system
d. The degree to which the proposed system has been tested with specific emphasis on testing user acceptance of the system.
e. The potential risks that were discovered and the strategies to mitigate these risks
f. The cost of the proposed system

You presentation should be original, engaging and flow well.

COMP 2606: Software Engineering
Project Details and Guidelines

Project Details

1. Introduction

a) Problem

b) Goals/Aims of System

c) Target User

2. Requirements

a) User requirements

b) System Requirements

c) Functional requirements

d) Non-functional Requirements

3. Context Model

a) Correct UML notation

b) Basic concepts identified

c) Correct association between concepts

4. Use Cases

a) Simple Use Cases (Correct actors, important use cases shown)

b) Use Case Diagrams

c) Ranking of Use Cases

Appropriate criteria

Appropriate scale

Appropriate separation of importance

d) Expanded Use Cases (top 2 ranked cases)

General format/template

Correct actors

Typical course of events

Alternatives

5. Class Diagram

a) Correct UML notation

b) Basic classes identified

c) Identification of inheritance

d) Basic attributes and methods from other artefacts

e) Correct associations between concepts

6. Risk Management

Risk management strategy (risk, probability, effects, affects, strategy category, strategies)

7. Cost Estimation

a) Specification and justification for cost modelling technique

b) Project cost (breakdown)