Software Engineering Project

Objectives

Students will apply skills gained in software engineering course to develop a prototype software system in specific domain. Students will work in group to document software requirements.

- Students will analyze requirement specifications and prepare analysis documentation.
- Students will become more experienced and confident in software development and be able to carry out group projects including planning, scheduling, milestone definition, etc.
- Students will learn how to identify and document requirements.
- Students will be able to prepare UML diagrams related to their projects.
- Students will become skilled in writing precise system documentation for maintainers.

Purpose and Method

The purpose of the course project is to provide the students with the knowledge of software engineering methodology and the skills to apply it.

Teamwork and Project Management

Each team consists of 3 to 5 students working on the same project. *Teamwork is **required** since team work is an integral part of software development*

All team members must take part in all project activities and **none** of the activities should be done exclusively by one student. Each team member or a pair must be *responsible* for all aspects of development required for the features they own.

Report Format

A report is submitted by the whole team and will be graded as a whole, and points will be split among the team members according to the declared contributions

A report must have a *cover page* containing:
. the course title,
. group number,
. project title,
. . submission date, and
. all team-member names.
Negative points will be assigned to reports missing- or having an incomplete cover page.

The **second page** of each report must detail the **breakdown of individual contributions** to the project (use more pages if necessary)—. **Each student** should provide an **itemized list** of his or

her contributions to components of the report, such as: requirements specification (use cases and non-functional requirements) etc and other: any other relevant contribution.

Report Section

1. Customer Problem Statement

   A minimum 3-page description about your project. The description should *not* be written from the developer's perspective, describing the features of the planned system. Rather, put yourself into a customer's role, and write your problem statement as if your imagined customer would write it! —Describe the problem that your customer is facing and his or her suggestions about how a software system could help.

2. System Requirements

a. Enumerated Functional Requirements
   Extract the requirements from the customer's narrative and list them in a table, one row per requirement. The first column shows a unique label "REQ-*x*". The second column shows briefly describes the requirement.

b. Enumerated Nonfunctional Requirements
   List, prioritize, and describe the Nonfunctional Requirements. The non-functional requirements numbering should continue the functional requirements list.

3. Functional Requirements Specification
   Derive the use cases based on the requirements from Section 2 and Section 2 above..
   a. Stakeholders
      Identify anyone and everyone who has interest in this system (users, managers, sponsors, etc.). Stakeholders should be humans or human organizations.
   b. Actors and Goals
      Identify the *roles* of people or devices that will directly interact with the system, their *types* (initiating vs. participating) and the *goals* of the initiating actors.
   c. Use Cases
      i. Use case Description
         For **all** use cases that you can think of (based on your System Requirements), write a *brief* or *casual* text description. List explicitly the requirements that each use case responds to.
      ii. Use Case Diagram
         Draw the use case diagram with all the use cases. Indicate the relationships, such as <<include>> and <<extend>>.
   d. System Sequence Diagrams
      Draw the system sequence diagrams for the **few most important** use cases selected above.
   e. Class Diagram and Interface Specification

Show all classes and their associations. Only indicate attributes and operations.
If you cannot fit the class diagram on one page, or it looks too cluttered, create one "overview" class diagram showing all classes and their relationships, but for each class show only a single compartment with the class name (leave out attributes and operations).
Then on subsequent pages show partial class diagrams, with three compartments and all the attributes and operations of a class. Make sure to indicate in diagrams and describe in text how partial diagrams fit into the overall class diagram.
Independently of the class diagram, write down class specification in UML notation. For every class, specify data types of all attributes and operation signatures. **Define** the meaning of each class, operation, and attribute in plain language.