SER315 Software Enterprise: Project Document - Fall 2025, Session A

Alexandra Mehlhase

August 25, 2025

1 Overview & Setup

This document contains essential information for your project. Review it as a team so you don't miss any requirements. The document is subject to change; the instructor will notify you of adjustments.

You'll design a software system based on the Requirements document on Canvas. Requirements may change (to simulate real-world clients). You'll evaluate requirements and exchange design feedback with peer teams during the course.

Design in Software Engineering is usually a group effort. The best designs can usually be achieved as a group and through talking about different designs.

- Groups will be formed on Aug 25th, you will see them on Canvas.
- You should create two Slack channels and invite me @amehlhas:
  - *groupname_gen* for internal communication.
  - *groupname_inst* for group questions.
- Contact me via Slack for individual or group questions.

2 Expectations & Tools & Deliverable Requirements

2.1 Expectations

- **Time**: 7 hours/week outside class (excludes assignments/prep).
- **Meetings**: At least two meetings per week (full team or subteams).
- **Teamwork**: Discuss all design decisions as a team; avoid strict divide-and-conquer.

- **Communication**: Synchronous or asynchronous is fine; maintain regular Slack check-ins (or other tools).
- **Grading note**: You'll be graded on being a good, prepared team member as well as on project artifacts.

2.2 Tools

- **UML Tool**: Astah (free for students), LucidChart, or Mermaid (other tools please ask). I personally use Astah or Mermaid.
- **Task Management (optional)**: Trello or similar tools to organize tasks is advised, so you can easily assign and track your progress.

2.2.1 AI

You can use AI to assist with the design, but ensure you discuss and create a consistent design. Understand the concepts, pay attention to detail, and do not rely solely on AI. Use your own knowledge and understanding.
Reference where AI was used and include important prompts in an appendix in your document.

2.3 Deliverable Requirements

- Submit one PDF per deliverable on Canvas (group submission). There are 4 deliverables; two are instructor-graded (2 & 4) with individual survey in which I ask you questions about your participation and your team members' participation (link will be on Canvas); two are peer-reviewed (1 & 3) so you can see other groups design and help them improve their work.

  Due dates are on Canvas.

- Include the following sections in each deliverable:
  1. **Team Reflection**:
     - How do you see your progress?
     - What went well?
     - What can you improve on for the next Deliverable?
     - Did you have any problems?
     - A plan of what you would like to do differently during the next iteration?
     - A link to a short screencast of 2-3 minutes explaining your status and your decisions. Upload this screencast to YouTube and add the link here. I use OBS but any tool that can record your screen is fine.

2. **Individual Reflection**: Short paragraph from each member individually (with name) on contributions to designs, documentation, discussions, etc. This does not have to be long but is supposed to give me an overview what each individual team member did, **will give you an opportunity to reflect on your contribution** and your team members an idea how you see your contribution.
3. **Meeting Overview**: Summary of meetings held, including dates, how long the meeting was, who attended, and short meeting minutes.
4. **Design**: All diagrams/documentation required for the specific deliverable (see 3); ensure high resolution and include current versions.
5. **Review Section**: Main feedback received (from instructor/peers) and what you changed based on it (required in Deliverables 2-4).

3 Software Design

In this course, you will iteratively design and implement a software system, following a structured process. Each deliverable builds on the previous one, allowing you to refine your design based on feedback.
Note on progression: Each deliverable builds on the previous one. Del 1 helps you understand what your system needs to do. Del 2 helps you structure how it will do it. Del 3 refines your design with patterns. Del 4 brings it all together.

- **Deliverable Overview**:
  1. **Deliverable 1**: High-level class diagram, Use case diagram, design principles and constraints.
  2. **Deliverable 2**: Architecture, activity diagram, UI sketch, updated previous work.
  3. **Deliverable 3**: Design patterns, initial code, updated previous work.
  4. **Deliverable 4**: Final implementation, final documentation, reflection.

Deliverable Details

Deliverable 1: Understanding the domain

- **High-level class diagram:** Show the main classes/entities and their relationships.
- **Use case diagram:** Show main actors and system functionalities.
- **Design principles:** Identify which design principles (e.g., from SOLID, KISS) you plan to apply and where you might use them.
- **Design constraints:** Identify design constraints for the system that you should take into consideration.

Focus on the overall structure and rationale. Details will be refined in later deliverables.

Deliverable 2: Structuring the system

- **Updated Deliverable 1:** Revise based on feedback, including your Design Principles/Constraints part (make sure you apply your principles)
- **Architectural design:** Now that you understand your system's components, choose and illustrate an architecture (e.g., layered, client-server) that fits your system. Include a diagram, a brief description, and your reasoning.
- **Activity diagram:** Illustrate a key user workflow (from one Use Case).
- **UI sketch:** Provide a rough sketch of the work flow from your AD.

The UI sketch can be hand-drawn or digital; focus on layout and main features, not visual polish.

Deliverable 3: Refining and adding details

- **Updated Deliverable 2:** Revise based on feedback.
- **Revised GUI design:** Include a better more detailed GUI design.
- **Design patterns:** Identify and integrate at least two design patterns into your design.
- **Initial implementation:** Provide a simple Java implementation (command-line is fine) that demonstrates your architecture and use of design patterns.

The code should show structure and integration of patterns, not full functionality.

Deliverable 4: Final phase

- **Updated Deliverable 3:** Revise based on feedback.
- **Final implementation:** Complete the main features from your Activity Diagram of your system (you do not have to have a full implementation of all requirements), ensuring consistency with your design. No GUI needed.
- **Reflection:** In your final video, discuss your design choices, trade-offs, and lessons learned.

Note: Each deliverable should include all previous diagrams and documents, updated as needed. Ensure consistency across all artifacts.

4 Project Peer Review

Here is a basic overview for grading. Detailed grading rubrics will be shown on Canvas for Deliverable 2 and 4.

- Each student will review another team's project twice during the course - I will assign these when the time comes.
- Provide constructive feedback to help the other group improve.
- Use a provided template for the review.
- Check syntax, consistency, overall design, and everything else you can think of.
- Your review should be constructive and help the other group create a better design.

5 Grading

- Project component: 30% of the overall grade.
- Grading breakdown:
  - 50% for the second deliverable + individual survey.
  - 50% for the fourth deliverable + individual survey + overall project process and outcome.
- Rubrics for individual and team performance:
  - Consistent activity and participation.
  - Application of module concepts.
  - Reviewing other groups and applying feedback.
  - Consistency in diagrams and documentation.
  - Delivery of a minimally viable product.
  - Adherence to requirements documentation standards.