UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

COMPUTER SCIENCE

COS 301 Software Engineering

Capstone Project: Demo 1 Instruction

S Baror, A Schreuder, M Mohamad Al, LM Burgess, T Chesane & A Street

1 Introduction

The First Capstone Demo is scheduled for Monday, 03 June 2024. Please ensure to book **one demo slot**. For Demo 1, a **minimum of three use cases** of your capstone project should be implemented. As much as we want you to show your system's wireframe and requirements, we would also like to see **some** implementation. Show the working components of your system for the Demo.

As discussed during the capstone presentation in one of our lectures, all teams should choose a specific time slot to meet on BLACKBOARD COLLABORATE for 5 to 15 minutes for a DAILY STANDUP.

This meeting, which is mandatory for all team members must be recorded, is a DAILY. Its purpose is to provide a brief update on your project progress, identify any obstacles, and plan your tasks for each day. Not attending this meeting may result in a delay in your project's progress. Please refer to the Guest Lecture on Agile principles and practices to implement this process correctly.Use this link to confirm your meeting time `https://forms.office.com/r/M6G7LQzq7t?origin=lprLink`

COS 301 lecturers will evaluate this session weekly and use this to scale team and individual students mark for each demo. Team leader, ensure your team follow through.

Be aware that you have multiple clients for your Capstone project. Schedule a meeting with your industry client as soon as possible. You are expected to meet with your industry client regularly. This can be weekly or bi-weekly on a medium (Google Meets, Discord, etc.) of your choice. You are expected to meet the client's needs as the project develops and react to any feedback.

The project owner—the industry client—is the primary client. The other clients are the COS 301 Lecturers and the CS Department Project Committee. The team's project is expected to address each client's requirements and develop towards the grand presentation on Project Day.

Your industry mentor grades you throughout the Demos. The team lead must forward them the link to grade your team. **It is the responsibility of the team lead to ensure your industry mentor grades you on the Demo days or the day after the Demo. Here is**

the link for the Mentor to grade you: `https://forms.gle/AARikW1wSkg8BagMA`

no marks from your industry mentor will be added once the marks are published.

Finally, when an unsolvable dispute, issue or challenge arises with your team member, client or industry mentor - you **must** email the staff immediately. Delaying these disputes may impact your product/project's final output. COS301 does not allow any student to have a 'free ride'. Note: Only doing documentation for a demo is considered a 'free ride', and you will fail.

2 In person Demo + Video Recordings of your Demo

1. For Demo 1, you are expected to:

   - In-person Demo on campus - IT 4-66.
   - The Video demo is compulsory to show the progress of your project. It also serves as a contingency if your project is to be considered for the Software Engineering Excellence Award; the sponsors need to see your project progress, showing that you employed SE principles at the beginning of your project). The maximum length of the video recording should be 15 minutes. The naming convention should be in the form **"TeamName-Demo1"**. Arriving at the live presentation and showing the video is not acceptable. We expect you to speak and explain your product. Your mark will be deducted if you choose to play a video instead of speaking.

     Here is the link to submit the video: `https://forms.gle/hFoG95DrufUZb1338` Upload the recorded video using the link provided above and add the same Demo to your team's profile.
   - Your video demo record should show the following:
     - *2 min* - Overview of the system that highlights the user stories of the features of the system, focusing on the ones you have implemented
     - *6 min* - Video that shows your Demo consisting of 3 working use cases.
     - *3 min* - Show unit testing of the components of your system that have been developed. *For this Demo and the subsequent demos, all testing must be **Automated tests** for both unit, integration and another testing of implemented use cases (if any).*
     - *1 min* - Github branching and workflow
     - 5-10 min demo of what you did or what your feature required. The Demo has to be live on the latest code in a release.
     - A proof and show a consistency of work and commitment of each member's contribution in their branch.
     - Show also your team's project management through Projects/Cards/Issues
   - Your GitHub landing page (README) should have the following:
     - Short description of your project - This description should also appear in the about field in your **GitHub** repository and the README. The description should follow the following format: "TeamName - ProjectName - Project Description".
     - A link to a your recorded demo video named **"TeamName-Demo1"**.
     - a link to the Functional Requirements (SRS) documents with all the sections described in Section 3.1.

  - A link to your **GitHub Project Board** - Project management tool.
  - Short profile description of each member of your team.
  - Individual professional profile of each team member, including a link to LinkedIn visible in your GitHub.

  Your GitHub repository should show the following:

  - Git structure (mono repo),
  - Git organization and management,
  - Branching strategy, and
  - Code Quality badges
    (a) Code Coverage — Coveralls, Codecov, SonarCloud
    (b) Build — AppVeyor, GitHub Actions, CircleCI, Drone, Netlify, Sonar Tests (SonarCloud), Travis
    (c) Requirements — requires.io
    (d) Issue tracking — GitHub Issues
    (e) Monitoring — NodePing, PingPong, Security Headers, Uptime Robot

3 Deliverable

All appropriate deliverables and your use of the tools specified in Section 1 will be evaluated for every Demo. The relevant artefacts should always reflect the details and current state of the project under development. The Master branch should always reflect a working version of your system. You must merge into the master before each Demo. This is equivalent to submitting an assignment in another module.

3.1 Requirements specifications

The main deliverable for Capstone Demo 1 is the base SRS for your project. It would help if you used the mini-project SRS as a guideline (other examples are given below). **You don't need to have a full or detailed SRS document**. Rather, an incremental approach as applied to **Agile methods** is advised; therefore, you will need to keep updating the SRS document over time. Put old sections which have changed in an appendix at the end of the document. The document should contain the following subsections:

- Introduction
- User Stories / User Characteristics
- Functional Requirements (For each Demo, specify the implemented features)
- Service Contracts
- Class diagram
- Architectural requirements
  - Quality Requirements
  - Architectural Patterns
  - Design Patterns
  - Constraints
- Technology Requirements

3.1.1 Introduction

Explain the vision and objectives. State the business need for the application and summarise the project's scope.

3.1.2 Class Diagram

Illustrate the software solution using a valid and correctly formatted UML class diagram. Ensure correct use of association, aggregation, composition and related concepts such as multiplicity.

3.1.3 User stories/ user characteristics

List all intended users and explain for what purpose each would use the system.

3.2 Functional requirements

3.2.1 Use Cases

Draw high-level use case diagrams for your system's use cases. Number them for traceability purposes. Ensure use cases are done using a service contract approach, as this will greatly assist in designing tests for your system. https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/

3.2.2 Requirements

Specify functional requirements to satisfy the use cases. Formulate them in terms of requirements and sub-requirements.

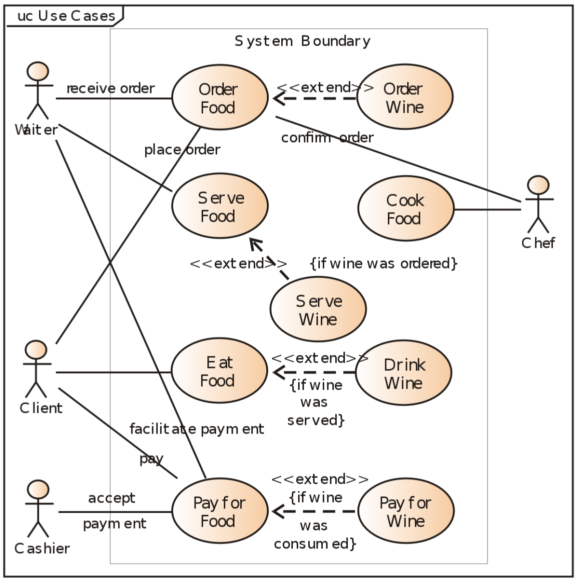
3.2.3 Subsystems

Assign the requirements to subsystems. in Figure 1. A use case diagram with the required functionality in the form for each concrete use case includes and extends relationships to lower-level use cases. Use `<<include>>` to specify sub-functionality for a given functionality and `<<extend>>` if the sub-functionality is optional. Show the required functionality within a system boundary as actions (ovals). The actors (stick figures) are people or other systems requesting or delivering services (labels on the connection line between an actor and an action).

Group the requirements in logical modules that can be implemented as inter-dependent subsystems with low coupling and high cohesion.

3.3 Quality Requirements

Specify and quantify each of the system's quality requirements. Examples of quality requirements include performance, reliability, scalability, security maintainability, and usability. This will be revised and extended for demo 2.



https://en.wikipedia.org/wiki/Use_case_diagram

Figure 1: An example of a UML use case diagram

4 Base Features

A base set of features is expected from all groups in COS301 to exist within their project. **They do not count for marks and do not count towards your total use cases.** These are things which you may have done in prior modules. Your project is expected to have all of these by Demo four and To ensure you earn the maximum marks possible.

4.1 Expected Features

(a) (Demo 1) Registration and Login - Your system must have a fully functioning basic registration and login at all times (if it makes sense for your system). Do not demo this to us unless asked. OAuth and more advanced login/registration techniques CAN count as features.

(b) (Demo 1) Basic Themes - Switching between light and dark mode is neither a feature nor a wow feature... You must have this as the bare minimum in your system for accessibility. More advanced customization for themes can be considered a feature.

(c) (Demo 1) Form validation

(d) (Demo 2) –to review– If your application is mobile-oriented, PWA would be desired?

(e) (Demo 3) SSL / TLS - When it comes time to host your system, the system must be protected with TLS certificates. A system with a lack of security is a bad system.

5 Assessment rubric

Item	Marks
Live Demo mark	
Live show (Demo) of at least three working use cases	30
Unit testing	10
Presentation & Demo (soft skill)	10
Other Deliverable (Documentations)	
Functional requirements & user stories/ user characteristics	10
Use case diagram	20
Class diagram	10
Industry mentor's grade	10
Total	100

5.1 Working prototype

You should implement your system in such a way that you always have a working prototype of the system in your git master branch AT ALL TIMES. The features that have not yet been implemented should be mocked. We expect at least three low-level use cases to be implemented and tested for this Demo with an automated testing framework.

6 Project Client

It would help if you had regular discussions/meetings with your project owner (client). Seek their approval of all artefacts that we require for your project to stand out on Project Day. Ensure your clients are aware of your GitHub Repository.

7 Demo 1 preparation checklist

In preparation for the First Capstone Demo, here are a few pointers for the preparation of your Demo 1:

(a) Your team should have completed a meeting with the client for requirements clarification, attended by at least one COS 301 staff member.

(b) Email the clients/mentors to join the Demo at the demo slot booked by your team. The link for your client to join the Demo will be published on ClickUp at least 5-days before the Demo; it is the team's responsibility to ensure that the link is forwarded to their client.

(c) The meeting sessions with the lecturers are to discuss the project's progress and planned project completion before the next meeting. This session is mandatory and should happen at least one time every fortnight. The link to the COS 301 calendar is on ClickUp. Select a meeting slot with the lecturer's Mentor's name to book. **(Again, meeting booked without team name will be cancelled)**

(d) Ensure you have a prototype (working components) of 3-use cases of the system.

(e) Setup a session with your lecturer mentor, industry mentor and client at a minimum of three times per month for adequate preparation for each Demo.

(f) TeamLead, look through the Demo 1 instruction rubric (this document) for a detailed guide to the Demo.

Best wishes.

7.1 Extra Tips

- Further presentation tips
  (a) Use PowerPoint, google slides or Canva (`https://www.canva.com/`) to present the demo
  (b) Start strong with the live Demo of the system early in the presentation
  (c) Mention what has changed with the project and the documentation (agile) and where these changes can be found (from demo 2).
  (d) Have all your members present something for the Demo (for maximum presentation mark)
  (e) Be ready to forward links to your repository, SRS and testing files to the markers on the day
  (f) Time yourselves the day before to ensure you can make it through all of your content
  (g) Ensure your diagrams correlate with how you learned to design them from COS301 and COS214 (consider draw.io for diagrams as visual paradigm can be a headache)

7.2 Groups from 2022 for reference

(a) https://github.com/COS301-SE-2022/Pure-LoRa-Tracking
(b) https://github.com/COS301-SE-2022/The-Au-Pair
(c) https://github.com/COS301-SE-2022/Japanese-Writing-Evaluator
(d) https://github.com/COS301-SE-2022/CryptoHub
(e) https://github.com/COS301-SE-2022/ReverseHand
(f) https://github.com/COS301-SE-2022/Map-out-game-reserves-using-aerial-photographs

7.3 Groups from 2023 for reference

(a) https://github.com/COS301-SE-2023/AI-Photo-Editor
(b) https://github.com/COS301-SE-2023/Blue-Skies
(c) https://github.com/COS301-SE-2023/Domain-Pulse-A-Sentiment-Analysis-Platform
(d) https://github.com/COS301-SE-2023/Avalanche
(e) https://github.com/COS301-SE-2023/WriteToPdf