Case Study: Online Electronics Store (AWE Electronics - Online)

The *AWE Electronics* is an Electronics Store on Glenferrie Road. The store owners want to expand its operation and have a new online presence, where the customers can browse the store catalogue, order goods, and get them delivered home.

AWE Electronics currently only has a physical store. Normally, only customers from surrounding suburbs visit the store, having limited reach. With the growing trend of online shopping, the owners have decided to expand its operation through an online store, aiming to reach customers Australia-wide.

Having researched online stores of a similar scale, the owners devised an initial set of ideas of what processes they would like to have covered with the new online store. In particular, the new online store shall support creating customer accounts, browsing store catalogue, managing shopping cart, creating invoices and receipts for customers, handling payments, managing goods packaging and shipment. In order to get a better overview of the online store operation, the new online store system shall also support basic statistics about the goods sold over various periods (a day, a week, a month, a year, or year to date) etc. The owners also want to be able to change the types of goods/products available for sale as new products become available.

Other online stores have incorporated further features into their system (e.g., discounts, loyalty programs etc). The owners of *AWE Electronics*, however, decided not to consider any of these features for now, but these may be options in the future.

In order to facilitate a tender process, the owners of *AWE Electronics* are now faced with the task of writing up a more detailed specification that clearly states the goals and requirements for the online store to be developed.

Note: This project is relatively small and relatively well-defined, yet may have several surprises.

Final Project: Online Electronics Store System Requirements (HK2 2025)

This project is your capstone effort to design, build, test, and present a full-stack, enterprise-level system. You must function as a unified Software Engineering team, addressing the needs of both internal staff and the external customer base of the Online Electronics Store. The project is divided into three distinct phases to ensure a complete lifecycle approach.

This document outlines the requirements for the Software Engineering final project. Teams of two students will design, implement, and test an end-to-end system for an **Online Electronics Store** over a **6-week period**, structured around the mandatory **3-Tier Architecture** and professional software engineering practices

Project Schedule & Milestones

Phase	Duration	Presentation/Submission	Deliverable Focus	Points (Out of 10)
Part 1: Analysis	2 Weeks	**Nov 7** (Live Presentation)	Requirements, Project Plan, UI/UX Design (The Vision)	**2.5**
Part 2: Design	2 Weeks	**Nov 21** (Live Presentation)	System Architecture, UML Models, Database Schema (The Blueprint)	**2.5**
Part 3: POC	2 Weeks	**Dec 5** (Video Demo & Final Submission)	Implementation, Testing, Final Documentation (The Product)	**5.0**

Required Technology Stack

Component	Requirement	Justification/Notes
Database	**MSSQL Server** (Single Instance)	All applications must connect to the *same* database.
Core Languages	**C# / .NET Framework** or **.NET Core**	Mandatory for Staff/Agent applications and all core logic (BLL/DAL).
Architecture	**3-Tier Model**	Strict separation of Presentation, Business Logic, and Data Access Layers.
Development	**Git / GitHub**	Mandatory for collaborative version control, development, and code sharing.

Part 1: Requirements Analysis and Planning (Total: 2.5 Points)

This phase is focused on the **discovery and definition** of the system. You must prove you understand the problem, define the solution's scope and boundaries, document all client needs, and visualize the complete user experience through prototyping. The live presentation serves as a gate to validate your core vision before any technical design begins.

Requirements Specification

You must thoroughly document all system requirements. This includes:

- **Functional Requirements:** Detail every core action the system must perform. This includes back-office functions such as **Product/Inventory CRUD** (Create, Read, Update, Delete), **Order Processing**, and **Staff Login**, as well as customer-facing functions like **Product Browsing, Shopping Cart Management, Checkout,** and **Online Payment**.
- **Non-Functional Requirements:** Define the essential qualities of the system, such as **Security** (e.g., secure login, input validation), **Performance** (e.g., fast database response time), and **Usability**.

UI/UX Design and Prototyping

Translating requirements into a user-centric design is mandatory. This stage directly addresses the **Mockup/Prototype** scoring criterion.

- **Low-Fidelity Wireframes:** Use **Balsamiq** (or an equivalent tool) to create conceptual wireframes for the most critical screens, focusing on content, information hierarchy, and user flow.
- **High-Fidelity Mockups/Prototypes:** Using **Figma** (or equivalent), develop detailed, high-fidelity prototypes of at least five core screens (e.g., the Staff Product Management screen, the Customer Checkout process, and the B2C Product Detail page).

Part 1 Detailed Points

Deliverable	Required Tasks	Points
Project Management Plan	Detail **Project Organization**, **Lifecycle Model** (e.g., Scrum), and **Risk Analysis**. Must justify team roles.	**0.75**
Requirements Specification	Document comprehensive **Functional** (e.g., CRUD operations, Stock Report, Order Status) and **Non-Functional** requirements (e.g., Security, Performance).	**1.0**
UI/UX Prototyping	1. **Wireframes (Low-Fidelity):** Use **Balsamiq** for basic layout. 2. **Mockups (High-Fidelity):** Use **Figma** for detailed visual design for 5 **core screens**.	**0.50**
UML Use Case Model	Create and present the graphical model showing all actors (Staff, Customer, etc.) and system boundaries.	**0.25**
Live Presentation (Nov 7)	Present and defend the **Requirements**, **Project Plan**, and **UI/UX design choices**.	**0.0** (Scored within above categories)

Part 2: Object-Oriented Design (Total: 2.5 Points)

This phase is the **technical blueprint**. You translate the validated requirements into a structured, layered, and language-agnostic design. You must define the data structure (Database), the architectural layers (3-Tier), the software components (Classes), and

model how they communicate (Sequence Diagrams). The live presentation ensures architectural integrity before any implementation effort starts.

Phase 2 focuses on translating the functional needs into a solid, structured technical design, addressing the **Architecture** and **Design** sections of the final report.

System Architecture and Technology

You must propose and justify a clear **System Architecture** by:

- Adhering to the mandatory **3-Tier Architecture** model (Presentation, Business Logic, and Data Access Layers).
- Defining your **Technology Stack**, which must include **C#** for the logic and **MSSQL Server** for the database, and detailing the chosen frameworks for the UI (e.g., Windows Forms, ASP.NET Core MVC). A strong rationale for your choices is required.

Object-Oriented and Data Modeling

All core system components must be modeled using Unified Modeling Language (UML) and data diagrams.

- **Static Model (Class Diagram):** Create a **UML Class Diagram** showing the major software entities (e.g., Product, Order, Customer) and their associations, attributes, and operations.
- **Database Design:** Produce a clear **Entity-Relationship Diagram (ERD)** or schema diagram showing all tables, primary/foreign keys, and relationships. This must be accompanied by the **SQL script** file containing the CREATE TABLE statements and **20-30 sample records** for testing.
- **Dynamic Model (Sequence Diagrams):** Develop **UML Sequence Diagrams** for 3–5 critical use cases (e.g., "Customer Places Order," "Accountant Creates Goods Delivery Note") to model object interaction across the defined 3-Tiers.

Finally, include a **Traceability Matrix** linking your initial functional requirements to the classes and database tables created in this design phase.

Part 2 Detailed Points

Deliverable	Required Tasks	Points
System Architecture	Design and present the **3-Tier Model** diagram. Provide a detailed, written **rationale** for the architecture and the selection of all technologies (C#, .NET, MSSQL).	**0.50**
Database Design	Design and document the **ERD** (or schema diagram). Submit the SQL **script file** with **CREATE TABLE** statements and 20-30 sample **INSERT** records.	**0.50**
UML Static Model	Develop a detailed **UML Class Diagram** correctly modeling core entities (e.g., Product, Order, Customer) with attributes, methods, and relationships.	**0.50**
UML Dynamic Model	Develop **3-5 UML Sequence Diagrams** for critical business logic (e.g., *Accountant Creates Goods Delivery Note*) showing the flow across the UI -> BLL -> DAL tiers.	**0.50**
Traceability Matrix	A matrix linking all **Functional Requirements** to their corresponding **Design Components** (Classes, Database Tables).	**0.25**
Live Presentation (Nov 21)	Present and defend the **3-Tier Model**, explain the **UML Models**, and justify the **Database Schema**.	**0.25**

Part 3: POC Implementation & Final Submission (Total: 5.0 Points)

This is the **delivery phase**, focusing on coding, quality assurance, and final documentation. You must implement the functional core of the system across two platforms and prove its reliability through systematic testing. The final video demonstration serves as the functional acceptance test for the client.

This final phase focuses on delivering a working, high-quality solution ready for deployment, encompassing all practical skills acquired in the labs.

Core System Implementation (50% of Software Grade)

You must deliver **two interconnected applications** demonstrating the core functions:

- **Staff Desktop Application:** A **Windows Forms** application used for internal business tasks, including a secure login function and at least four primary features (e.g., Inventory CRUD, viewing Goods Received/Delivery Notes).
- **Web/Cloud Application:** An **ASP.NET Web Forms or MVC/Core** application for staff or agents (must also include a login) for remote management.
- **Shared Database:** Both applications **must connect to the same MSSQL database** using the established 3-Tier architecture.

Quality Assurance and Source Control

Quality is paramount, and the process is judged as much as the final code:

- **Source Control (Git/GitHub):** All source code must be managed in a shared **GitHub repository** showing collaborative commits and history from all team members throughout the project duration. The source code must also adhere to **Coding Conventions**.
- **Unit Testing:** A separate **Unit Test Project** is mandatory. This includes developing test cases for core business logic methods (e.g., calculations, validation) by applying test techniques such as **Equivalence Partitioning (EP)** and **Boundary Value Analysis (BVA)**.
- **Test Documentation:** Submit a formal **Test Plan** with a matrix linking implemented test cases back to your original use cases.

Part 3 Detailed Points

Deliverable	Required Tasks	Points
Staff Desktop Application	**Windows Form Project** (C#, .NET) for internal staff (e.g., inventory, goods notes). Must implement a **login** and demonstrate at least **4 main functions**.	**1.00**
Agent Web Application	**ASP.NET Web Forms or MVC/Core Project** (C#, .NET) for agents/remote staff. Must implement a **login** and access shared data using the **3-Tier** model.	**1.00**
Testing & Quality	A dedicated **Unit Test Project** applying **EP/BVA** to core logic (e.g., stock validation, financial calculations). Submit **Test Plan** and test traceability.	**1.00**

Deliverable	Required Tasks	Points
Source Control & Code Quality	Project hosted on **GitHub** with a visible, **collaborative commit history**. Code must adhere to **Coding Conventions**.	**1.00**
Final Report & Demo	Submit the complete, final, **bookmarked PDF Report** adhering to the outline. Submit a high-quality **Video Recording Demo** showcasing both applications.	**1.00**

High Distinction (HD) / Extra Credit (Max: +1.0 Point)

B2C E-commerce Website/Mobile App (Extra Point): To achieve High Distinction, you must build the **Customer-Facing E-commerce Frontend**. This can use any technology (e.g., PHP, React/NodeJS) but **must connect to the same MSSQL database** via an appropriate access layer (***API or class library is preferred***). The system should feature a customer experience that integrates with a form of **Digital Wallet** or online payment (e.g., PayPal, Momo, bank transfer simulation).

Final Report: Submit the complete **Final Report (PDF)** incorporating all three parts, strictly following the required outline, and including bookmarks for navigation.

Final Project Marking Guide: Online Electronics Store System

This rubric is based on a 10 **-point scale** for assessing team performance across three major project milestones.

Part 1: Requirements Analysis and Planning (Total: 2.5 Points)

This phase assesses foundational planning and user experience design.

Item	Criteria for Excellence (≥ 0.75)	Criteria for Competence (≥ 0.40)	Criteria for Minimal Pass (≥ 0.25)	Score
Project Management Plan	**Clear and correct** plan (organization, lifecycle, risk analysis). Strong defense in presentation.	Good plan, but lacking depth or detail; some areas of the presentation are unclear.	Minimal, incomplete plan submitted; team struggles to defend structure.	0.75
Requirements Specification	**Comprehensive, correct, and sufficient** Functional (e.g., all CRUDs, reports) and Non-Functional Requirements (e.g., strong security focus).	Requirements are present but vague, inconsistent, or miss critical aspects (e.g., no clear non-functional goals).	Minimal or significantly incorrect list of core requirements submitted.	1.0
UI/UX Prototyping	Both **High-Fidelity Mockups (Figma)** and **Wireframes (Balsamiq)** are complete, professional, and visually defended in presentation.	Mockups/Wireframes present but lack professional detail, clarity, or consistency; presentation is unclear.	Only one format (Mockup or Wireframe) submitted, or artifacts are non-functional/incomplete.	0.50

Item	Criteria for Excellence (≥ 0.75)	Criteria for Competence (≥ 0.40)	Criteria for Minimal Pass (≥ 0.25)	Score
UML Use Case Model	Correct and sufficient graphical model showing actors and system boundaries.	Diagram present but contains minor errors or inconsistencies.	No use case diagram submitted.	0.25
Total Points for Part 1				2.5

Part 2: Object-Oriented Design (Total: 2.5 Points)

This phase assesses the technical blueprint and data modeling prior to coding.

Item	Criteria for Excellence (≥ 0.75)	Criteria for Competence (≥ 0.40)	Criteria for Minimal Pass (≥ 0.25)	Score
System Architecture	**Clear 3-Tier Model** with strong, logical rationale for the chosen technology stack (C#, .NET, MSSQL). Defense during presentation is convincing.	Architecture is present but rationale is weak or minor architectural errors exist.	No architecture diagram or design contains critical flaws (e.g., direct DB access from UI).	0.50
Database Design	**Correct ERD** and SQL **script** (CREATE TABLE + 20-30 INSERT records) demonstrating normalization and integrity.	Database design presents, but contains minor errors (e.g., missing indices, poor naming) or is missing sample data.	No database design or the schema contains significant data integrity flaws.	0.50
UML Static Model	Detailed, correct **UML Class Diagram** showing all core attributes, methods, and relationships.	Class diagram is present but is not detailed (e.g., missing methods) or has minor structural errors.	No class diagram submitted.	0.50

Item	Criteria for Excellence (≥ 0.75)	Criteria for Competence (≥ 0.40)	Criteria for Minimal Pass (≥ 0.25)	Score
UML Dynamic Model	3-5 **Sequence Diagrams** clearly and correctly model complex cross-tier interactions (UI, BLL and DAL) for critical use cases.	Diagrams are simplistic or contain minor flaws in sequencing or tier interaction modeling.	Less than 3 diagrams submitted or they contain major logical errors.	0.50
Traceability & Presentation	Complete Traceability Matrix linking all requirements to design components. Presentation is clear and organized.	Matrix is incomplete or minor presentation issues exist.	No traceability matrix submitted.	0.50
Total Points for Part 2				**2.5**

Part 3: POC Implementation & Final Submission (Total: 5.0 Points)

This phase assesses final functional delivery, code quality, and documentation.

Item	Criteria for Excellence (≥ 0.8)	Criteria for Competence (≥ 0.5)	Criteria for Minimal Pass (≥ 0.25)	Score
Staff Desktop Application	**Fully functional** WinForm Project. Implements secure **Login** and $\mathbf{4+}$ **main functions** (e.g., Inventory CRUD, Reports) with no critical bugs.	Runnable, but with minor bugs or only basic implementation of the core functions.	Non-functional, crashes frequently, or fails to connect to the DB/3-Tier structure.	1.00
Agent Web Application	**Fully functional** Web/MVC Project. Implements secure **Login** and successfully uses the **3-Tier model** to access shared data.	Runnable, but with minor bugs, or the 3-Tier separation is violated (e.g., poor DAL/BLL use).	Non-functional or severe bugs prevent core data access.	1.00
Testing, Unit Test	**Thorough Unit Test Project** (EP/BVA	Tests cover minimal paths, EP/BVA	No test project, or tests do not	1.00

Item	Criteria for Excellence (≥ 0.8)	Criteria for Competence (≥ 0.5)	Criteria for Minimal Pass (≥ 0.25)	Score
	applied) for critical business logic (e.g., calculations, validation). All tests pass. Complete Test Plan/Traceability.	vaguely applied, or 1-2 minor tests fail.	cover business logic.	
Source Control (GIT/Code)	**GitHub** shows **consistent, collaborative history** from both team members. Code adheres to **thorough coding conventions**.	History is sparse or only one member contributed significantly. Conventions are present but inconsistent.	No Git/poor commit history. No or minimal coding conventions.	1.00
Final Report & Demo	**Complete, bookmarked PDF Report** with no errors. High-quality **Video Demo** clearly and professionally showcasing all features.	Report is mostly complete but has minor errors. Video is acceptable but lacks polish (e.g., poor audio/visuals).	Report is incomplete or contains critical errors. Video is missing or unusable.	1.00
Total Points for Part 3				**5.0**