Software Development Lifecycle Project
Course: Software Engineering
Instructor: Mr HO Mafua
Academic Term: 2025/2026

Project Overview

This project requires you to apply the complete Software Development Lifecycle (SDLC) to develop a functional software application. You will work through all phases from initial planning to deployment, documenting your process and delivering both the software product and comprehensive documentation.

Project Objectives

By completing this project, you will:

1. Apply SDLC phases to a real-world software problem
2. Create professional software documentation
3. Practice requirements analysis and system design
4. Implement software using best coding practices
5. Conduct systematic testing and quality assurance
6. Experience deployment and maintenance planning

Project Phases & Deliverables

Phase 1: Planning & Requirements Analysis

Deliverables:

1.1 Project Charter
Project title and brief description
Project objectives and scope
Stakeholder identification
Success criteria
Estimated timeline and resources
1.2 Requirements Specification Document
Introduction and purpose
Target users/stakeholders
Functional requirements (minimum 10-15)
Use case descriptions or user stories
Priority levels (Must-have, Should-have, Could-have)
Non-functional requirements
Performance requirements
Security requirements
Usability requirements
Compatibility requirements
Constraints and assumptions
Glossary of terms
Format: PDF document, 5-10 pages

Phase 2: System Design
Deliverables:
2.1 System Architecture Document
Architecture overview diagram
Technology stack selection with justification
System components and their interactions
Data flow diagrams
Deployment architecture
2.2 Database Design
Entity-Relationship Diagram (ERD)
Database schema with tables, fields, and data types
Relationships and constraints
Normalization explanation (which normal form and why)
2.3 User Interface Design
Wireframes or mockups for main screens (minimum 5 screens)
Navigation flow diagram
UI/UX design rationale
Accessibility considerations
Tools Suggested: Lucid chart, Draw.io, Figma, Balsamiq, or similar
Format: PDF document with embedded diagrams, 8-15 pages

Phase 3: Implementation
Deliverables:
3.1 Source Code
Complete, functional codebase
Code organization and structure
Adherence to coding standards and conventions
Comments and inline documentation
Version control usage (Git commits showing progress)
Evaluation Criteria:
Functionality: Does it work as specified?
Code quality: Readability, modularity, efficiency
Best practices: Error handling, input validation
3.2 Implementation Report
Technologies and tools used
Key algorithms or design patterns implemented
Challenges faced and solutions
Deviation from original design (if any) with justification
Format:
Source code: GitHub/GitLab repository link or ZIP file
Report: PDF document, 3-5 pages

Phase 4: Testing & Quality Assurance
Deliverables:
4.1 Test Plan Document
Testing strategy and approach
Types of testing to be performed
Testing environment setup
Test schedule
4.2 Test Cases
Minimum 20 test cases covering:
Functional testing
Boundary testing
Negative testing
Integration testing
Test case format: Test ID, Description, Preconditions, Steps, Expected Result, Actual Result, Status (Pass/Fail)
4.3 Test Report
Test execution summary
Bugs found and their severity
Bug tracking and resolution status
Screenshots or evidence of testing
Overall quality assessment
Format: Excel/Google Sheets for test cases, PDF for plan and report, 5-8 pages total

Phase 5: Deployment & Documentation
Deliverables:
5.1 Deployment Guide
System requirements (hardware/software)
Installation instructions (step-by-step)
Configuration settings
Troubleshooting common issues
Uninstallation procedure
5.2 User Manual
Getting started guide
Feature descriptions with screenshots
Frequently Asked Questions (FAQ)
Support contact information
5.3 Maintenance & Future Enhancements
Known limitations
Planned future features
Maintenance schedule recommendations
Backup and recovery procedures
Scalability considerations
Format: PDF documents, 10-15 pages total

Phase 6: Project Presentation
Requirements:
Duration: 10-15 minutes per team + 5 minutes Q&A
Live demonstration of working software
Overview of SDLC phases and key decisions
Challenges and lessons learned
Professional presentation slides

Evaluation Criteria:
Clarity and organization
Demonstration effectiveness
Team participation
Q&A responses

Submission Guidelines
Format Requirements:
All documents in PDF format unless otherwise specified
Use consistent formatting (fonts, headers, page numbers)
Include cover page with project title, team members, date
Use proper grammar, spelling, and professional language
Include table of contents for documents over 5 pages
File Naming Convention:
TeamName_PhaseNumber_DocumentName.pdf
Example: TeamAlpha_Phase1_Requirements.pdf
Submission Method:
Specify: Learning Management System, Email, GitHub Classroom, etc.

Evaluation Criteria
All deliverables will be evaluated based on:
Completeness (40%): All required elements are present and thoroughly addressed
Quality (30%): Professional presentation, clear writing, accurate technical content
Technical Correctness (20%): Proper application of SDLC principles and software engineering best practices
Creativity & Initiative (10%): Original thinking, going beyond minimum requirements, innovative solutions

NOTE:
1.All work must be original
2.Properly cite any external resources, libraries, or code snippets used
3.Collaboration within teams is expected; collaboration between teams is prohibited
4.Plagiarism will result in zero points and disciplinary action

Recommended Tools:
Version Control: Git, GitHub/GitLab
Documentation: Microsoft Word, Google Docs, LaTeX
Design: Lucid chart, Draw.io, Figma
Project Management: Trello, Asana, Jira
Development: (Specify based on your course stack)