Software Engineering Project's Specifications Summer Semester 2024/2025

Course Code	1902372
Course coordinator and instructors	**Course coordinator:** Dr. Hamad Alsawalqah. **Instructors:** Dr. Muhammed Abu Shariah
Weight	100 points to be considered as 20% of total course weight. 50% of points will be deducted from the tasks that were not submitted on time.
Submission Deadline	The project final submission deadline is **Sunday 17, August 2025**. No submissions are allowed and accepted after this date. You must submit Tasks 1, 2 and 3 according to the deadlines in the course syllabus (also will be announced on eLearning).
Project Working Mode	Teamwork, whereby students are required to work in self-organized groups consisting of 5 members
Presentation	Groups will present their projects using **ENGLISH language**. Project Groups' presentations date and time will be announced later.
Presentation Duration	20 minutes
Project Topic	**Each group MUST select one of the topics listed at the end of this document**
Required Product	• **Prototype:** use any prototyping tool (e.g., https://www.fluidui.com/) or any IDE just to show the project's user interface screens. • **Implementation of the Frontend.** • **Documentation:** according to the template in the next page • **Backend Implementation (Bonus)**
Analysis and Design method (SAD)	**Student must follow Object Oriented analysis and design approach**
Design and Modeling tools	Use professional tools to draw your system models e.g.: Rational Rose. We will have online training lectures. Also, you can use open-source tools: StarUML, lucidchart, Acceleo, ArgoUML, BOUML, Eclipse UML2 Tools, Umbrello UML Modeller, Frame UML, AmaterasUML, Dia, Xholon

Documentation Template: the following sections must be included in your document:

Cover page (project title, names of students, supervisor, and date)

Executive Summary

Table of Contents

List of Figures

List of Tables

1.0 Project Initiation

- **1.1 Project Overview (**Provides a general introduction to the project)
- **1.2 Problem Definition**
  - **1.2.1 Current Situation and Problem Statement (**Clearly states the problem)
  - **1.2.1 Issues (**Identifies specific issues in the current situation)
  - **1.2.2 Objectives for each issue** (Lists objectives addressing the identified issues)
  - **1.2.3 Requirements (**that must be included in all proposed systems**)**
  - **1.2.4 Constraints (**Defines limitations affecting the project**)**
- **1.3 Feasibility Study**
  - **1.3.1 Technical Feasibility**
  - **1.3.2 Operational Feasibility**
  - **1.3.3 Economic Feasibility**
  - **1.3.4 Schedule Feasibility**
  - **1.3.5 Legal Feasibility**
- **1.4 Recommended Solution and Expected Project Deliverables (**Outlines the proposed solution and what will be delivered)
- **1.5 Local and Global Impact of the Proposed Solution (**Discusses the impact of the solution)
- **1.6 Naming Conventions and Definitions (Terms, Acronyms, and Abbreviations)**

2.0 Project Management plan

- **2.1 Project Organization (**Describes the structure of the project team)
- **2.2 Roles and Responsibilities (**Assigns specific roles and responsibilities)
- **2.3 Software Process Model (**Defines the process model to be followed and justifies your selected model)
- **2.4 Tools and Techniques (**Lists tools and techniques to be used)
- **2.5 Work breakdown** (for each task: Describes Project Tasks, Deliverables and Milestones, Resources needed (Skills, HW and SW), Dependencies and Constraints**)**
- **2.6 Assigning Team Members to Tasks**
- **2.7 Project Schedule (Gantt chart and PERT diagram)**
- **2.8 Risk Analysis and Plans**

2.9 Monitoring, Reporting, and Controlling Mechanisms (Describes how the project will be monitored and controlled)

3.0 Software Requirements Specifications (SRS)

3.1 System Stakeholders and Requirement Sources

- ✓ Identify all user types, clients, maintainers, and any third parties.
- ✓ Describe how requirements were gathered (interviews, documents, observations, etc.).

3.2 User Requirements and Stories

- ✓ Informal user-level needs described in natural language.
- ✓ Include user stories or problem scenarios.
- ✓ Link each requirement to relevant stakeholders where applicable.

3.3 Use Case Model

- ✓ Include Use Case Diagram(s).
- ✓ Provide an overview or list of all identified use cases with brief descriptions.

3.4 Functional Requirements Specification

- ✓ Detail the system-level functional requirements.
- ✓ Organize them by modules or use cases if preferred.
- ✓ Ensure each requirement is uniquely numbered and testable.

3.5 Use Case Descriptions

- ✓ Provide a textual format for each use case.
- ✓ Include: actors, preconditions, main steps, alternate flows, and exceptions.

3.6 Non-Functional Requirements

3.6.1 Performance Requirements

3.6.2 Dependability Requirements (reliability, availability)

3.6.3 Security Requirements

3.6.4 Usability Requirements

3.6.5 Operational and Environmental Requirements

3.6.6 Maintainability and Supportability Requirements

3.7 Data Requirements

- ✓ Define business data entities and describe data storage requirements.
- ✓ Include format constraints, volume expectations, or dependencies.
- ✓ May reference ER diagrams or object models from the design section.

4.0 Analysis and Design

4.1 System Architecture Overview

- ✓ Describe the overall architecture of the system. Include architectural style (e.g., layered, microservices, client-server), main components, and their interactions.
- ✓ Provide a high-level diagram showing major components and data flow between them.

4.2 Behavioral Modeling

- ✓ Activity Diagrams for core processes or workflows
- ✓ Sequence Diagrams for specific scenarios (e.g., login, transaction)
- ✓ Explain each diagram briefly.

4.3 Structural Modeling

- ✓ Define the static structure of the system using Class Diagrams.
- ✓ Classes should include attributes, methods, and relationships (association, inheritance, etc.).

4.4 Component and Deployment Design

- ✓ Describe how the system is broken into components or modules.
- ✓ If relevant, include a deployment diagram showing how software components are distributed across hardware nodes.

4.5 Graphical User Interface (GUI) Design

- ✓ Include mockups or wireframes of key screens.
- ✓ Explain the user interaction flow and how usability principles are applied.

4.6 Object to ER Mapping (if required)

- ✓ Explain how object-oriented classes map to the relational database schema.
- ✓ Discuss how relationships (e.g., inheritance, aggregation) are translated into tables, foreign keys, and join tables.

4.7 Physical Database Design (if required)

- ✓ Provide a detailed schema of the database including tables, fields, types, constraints, and keys.
- ✓ Ensure the schema supports the functional and data requirements specified earlier.

5.0 Implementation

5.1 Graphical User Interface Implementation (Required) (Describes GUI implementation)

5.2 Database Implementation (Bonus) (Details database implementation)

5.3 Other Components Implementation (if needed)

6.0 User Manual

6.1 System Requirements

6.2 How to Install (if needed)

6.3 How to Use the System (Steps + Screenshots)

6.4 Known Issues or Limitations

7.0 References: books and tools

Format of the documentation:

1. **Submit soft copy of the report in MS Word format.**
2. **Font Type:** Times New Roman
3. **Font Size:** 12 points
4. **Spacing:** Double Spacing
5. **Horizontal Alignment:** Justify
6. **Margins: (Top:** 2cm**, Bottom:** 2cm**, Right:** 2cm**, Left:** 2cm**)**

Details about project topics:

1. HEALTHGUARD: REMOTE HEALTH MONITORING AND ALERT SYSTEM FOR THE ELDERLY

Description:

HealthGuard is a mobile and web-based health monitoring system tailored for elderly patients. It enables seniors to **record vital signs** (such as blood pressure, glucose, heart rate, and temperature), **track medication adherence**, and **stay connected with healthcare providers** in real time. The system analyzes collected health data to detect abnormalities and automatically notifies doctors or caregivers of potential emergencies. The app aims to support independent living while ensuring medical professionals can intervene early when necessary.

MAIN FEATURES

VITAL SIGNS LOGGING & MONITORING

- Manual input or optional integration with Bluetooth-enabled health devices
- Daily logs for:
  - Blood pressure
  - Heart rate
  - Blood glucose
  - Body temperature
- Visual trends/charts for users and doctors

MEDICATION ADHERENCE TRACKER

- Personalized medication schedule with dosage, times, and instructions
- Alerts and reminders for each dose
- Missed medication alerts sent to caregiver/doctor

REAL-TIME ALERTS & ESCALATION

- System flags out-of-range vital signs based on thresholds
- Immediate alerts sent to doctor, caregiver, or emergency services
- One-tap "Request Assistance" button for the user

DOCTOR DASHBOARD

- Secure portal for physicians to:
  - View patient health logs and trends
  - Set alert thresholds per patient
  - Communicate with patients via in-app messaging or video
  - Send instructions or update medication schedules

⚕CAREGIVER ACCESS (OPTIONAL)

- Designated family members can view health summaries
- Receive alerts for emergencies or missed meds

🔐 SECURITY AND PRIVACY

- Role-based access (patient, doctor, caregiver)
- End-to-end encryption of medical data
- Compliant with health data privacy standards (e.g., HIPAA-ready design)

2. LOCALEASE: SMART LOCAL BUSINESS DIRECTORY AND SERVICES HUB

DESCRIPTION:

LocalEase is a **web and mobile application** that empowers communities by creating a centralized, smart directory of **local businesses and service providers**. It goes beyond basic listings by offering **real-time interaction**, **intelligent recommendations**, **scheduling**, and **user reviews**. The platform enables small and informal businesses (like tailors, tutors, barbers, mechanics, home cooks, etc.) to create a digital presence, while users can **discover, book, review, and communicate** with them in one place. The project is especially valuable in areas where small businesses lack access to online exposure or modern customer management tools.

KEY FEATURES

SMART BUSINESS DIRECTORY

- Searchable database of businesses filtered by:
  - Category (e.g., food, repair, tutoring)
  - Proximity
  - Availability (open/closed now)
- Business profiles include services offered, pricing, images, contact info

AI-POWERED DISCOVERY & RECOMMENDATIONS

- Personalized business suggestions based on:
  - User location and preferences
  - Previous bookings or searches
  - Seasonal/trending services
- Popularity-based ranking and featured listings

BOOKING & SCHEDULING SYSTEM

- Users can request services or book appointments
- Businesses set working hours, slots, and availability
- Real-time updates and calendar integration for both sides

COMMUNICATION TOOLS

- In-app chat or messaging between users and businesses
- Automated responses for common queries (e.g., pricing, working hours)
- Optional voice call link or WhatsApp integration

☆ REVIEWS & RATINGS

- Verified customers can rate businesses and leave feedback
- Star ratings and keyword-based review summaries
- Option for business replies to reviews

BUSINESS DASHBOARD

- Service providers can:
  - Manage bookings
  - Update service details and availability
  - View customer insights and trends

NOTIFICATIONS & PROMOTIONS

- Push/email/SMS notifications for confirmed bookings, promotions, or schedule changes
- Businesses can post deals or discounts (e.g., 10% off on first booking)

MULTILINGUAL SUPPORT & ACCESSIBILITY

- Local language support
- Simplified layout for low-literacy or elderly users
- Offline mode with caching (optional)

3. SKILLSWAP: PEER-TO-PEER LEARNING AND MICRO-TEACHING PLATFORM

DESCRIPTION:

SkillSwap is a web and mobile platform that enables users to **learn and teach skills in short, focused sessions**. Whether it's learning basic graphic design, conversational Spanish, or coding a website, users can offer micro-courses or 1-on-1 sessions in exchange for time, ratings, or money. Unlike formal MOOCs, SkillSwap focuses on **real-time knowledge exchange** between peers, with features like scheduling, rating, payment handling, and AI-based skill matching. It democratizes education and fosters local or global communities of mutual learning.

KEY FEATURES

USER PROFILES AND SKILL BOARDS

- Each user can be both a **learner and teacher**
- Profile includes:
  - Skills offered

    - Skills wanted
    - Availability
    - Reviews and ratings
- Users can post micro-class offers (e.g., "Teach JavaScript basics – 30 mins")

SMART MATCHING SYSTEM

- AI/ML algorithms suggest potential peer matches based on:
    - Interests
    - Skill level
    - Availability and location (optional)
- Recommends trending or in-demand skill sessions

📅 SESSION SCHEDULING AND MANAGEMENT

- Users book each other for 1:1 or group sessions
- Calendar and time zone support
- Reschedule, cancel, or auto-match sessions
- Option for live video, in-person, or pre-recorded material

🎥 INTEGRATED LEARNING TOOLS

- In-app video calling with screen sharing (or Zoom/Google Meet integration)
- File sharing (e.g., slides, PDFs)
- Whiteboard tools or collaborative notes (optional)

💰 REWARDS AND CREDITS SYSTEM

- Time-based credit system: e.g., teach for 1 hour → earn 1 credit
- Or monetary model (paid classes with Stripe/PayPal integration)
- Leaderboard or badges for active contributors

📝 RATINGS AND FEEDBACK

- After each session, both users give:
    - A star rating
    - Short feedback
    - Tags (e.g., “helpful,” “needs improvement,” “patient”)

🔔 NOTIFICATIONS AND COMMUNITY

- Alerts for upcoming sessions, new offers, feedback
- Discussion forums or skill-specific groups
- Challenges and contests (e.g., “Teach 3 people this week”)

SAFETY AND MODERATION

- Report and block features
- AI flagging of inappropriate behavior or content
- Option for ID verification (for premium or high-frequency users)