National University of Computer and Emerging Sciences

Req2Design – Automated Requirement-to-Design Analyzer

Team Members: Ali Rizwan — 22I-2447
Muhammed Shahmeer — 22I-1522
Abdul Haq Zulfiqar — 22I-2585

Supervisor: Ma'am Isma Ul Hassan
Co-Supervisor: Sir Owais Idrees
Department: Software Engineering
Date: 21 September 2025
NOTE: Our newname is REQ2Design but the submission is on old name MedMapPK

Contents

1 Introduction

Software engineering heavily relies on precise requirement documentation, as it lays the foundation for every subsequent design, implementation, and testing activity. Poorly documented or ambiguous requirements often lead to project delays, cost overruns, and even complete failure. Despite being critical, requirement documentation is often neglected, rushed, or performed without adherence to standards.

The project **Req2Design** focuses on bridging this gap by automating the creation of structured, standard-compliant documentation. It uses NLP and AI to transform natural-language inputs into well-organized SRS documents, textual use cases, and UML diagrams. Unlike generic note-taking or diagramming tools, Req2Design ensures compliance with **IEEE 830** for SRS documentation, the **Alistair Cockburn template** for use cases, and UML 2.x standards for diagrams.

By providing a guided, semi-automated workflow, Req2Design reduces ambiguity, ensures completeness, and helps students, startups, and software teams save significant time while improving the quality of requirement artifacts.

1.1 Problem Statement

Requirement documentation is widely recognized as one of the most error-prone stages in software development. Ambiguity, inconsistency, and incompleteness in requirements can propagate through later stages, resulting in flawed designs and failed systems.

Current tools for requirement documentation suffer from multiple limitations:

- They are either too manual, relying heavily on user effort, or too rigid, failing to adapt to unstructured requirement input.
- Most lack compliance with recognized standards like **IEEE 830** for SRS or UML guidelines for diagrams, making them unsuitable for academic or professional evaluation.
- Automated tools, where they exist, often stop at generating plain text and fail to enforce completeness, traceability, or clarity.
- Use case generation tools rarely incorporate established templates (e.g., Cockburn's) that are widely taught and expected in academic and professional settings.

These gaps result in documentation that is fragmented, incomplete, or not usable for practical software engineering tasks.

Req2Design directly addresses this challenge by combining natural language processing with standards-based output, ensuring both automation and correctness.

1.2 Motivation

The motivation behind Req2Design lies in three primary observations:

1. **Educational Need:** Students in software engineering programs spend a disproportionate amount of time preparing SRS documents, use cases, and diagrams. This repetitive effort often distracts from higher-level learning goals like analysis and design thinking.

2. **Startup SME Use Cases:** Early-stage companies lack the time or expertise to produce detailed requirement documentation. Automating this process accelerates project kickoff while ensuring clarity.

3. **Quality Improvement:** Manual requirement documentation is prone to oversight. Automating the process enforces completeness, standard adherence, and reduces the risk of missed requirements.

In short, Req2Design saves time, enforces quality, and makes requirement documentation accessible to learners, developers, and organizations alike.

1.3 Solution Overview

Req2Design introduces a four-step automation workflow:

- Intake of requirement data in text or audio form.
- Conversion into structured SRS content following **IEEE 830**.
- Use case documentation in the **Alistair Cockburn template**.
- Automated UML 2.x diagram generation and export.

This modular approach ensures that each phase of requirement documentation is systematically automated while still leaving room for user editing and refinement.

1.4 Stakeholders

- **Students and Researchers:** Use the tool for academic reports and projects.
- **Startups and SMEs:** Accelerate requirement documentation for early-stage projects.
- **Supervisors and Evaluators:** Receive standardized, structured requirement documents for evaluation.
- **Developers:** Utilize consistent requirements for accurate design and implementation.

2 Project Description

2.1 Scope

The scope of this project defines the boundaries of what **Req2Design** will deliver in terms of functionality and coverage. The goal is to automate requirement documentation in a practical, standards-compliant, and academically useful way, while avoiding unnecessary complexity that would make the solution unmanageable within a single Final Year Project. The scope is therefore intentionally **focused yet impactful**, ensuring that the generated documentation is meaningful, useful, and aligned with recognized standards.

The key scope items include:

- **IEEE 830-based SRS:** Req2Design will generate a Software Requirements Specification (SRS) document based on the widely accepted IEEE 830 standard. To keep the scope feasible, only the **most essential sections** will be implemented, including:
  - **Introduction:** Purpose, scope, definitions, and references.
  - **Overall Description:** System perspective, product functions, user characteristics, and general constraints.
  - **Functional Requirements:** Clearly numbered requirements (e.g., FR-01, FR-02) that define what the system must do.
  - **Non-Functional Requirements (NFRs):** Focus on core aspects such as usability, reliability, performance, and portability. Broader categories such as scalability, maintainability, or security are excluded to limit scope.
  - **External Interfaces:** Covering user, hardware, software, and communication interfaces that define how the system interacts with external entities.
- **Use Case Documentation:** Req2Design will generate use cases in a textual format following the **Alistair Cockburn template**. To keep documentation concise and focused, each use case will be limited to the following essential elements:
  - Actors involved.
  - Goal / Brief description.
  - Preconditions.
  - Main flow of events.
  - Postconditions (successful outcomes).

  Alternative and exception flows will not be included in this version to ensure simplicity and feasibility. Editing capabilities will be provided so users can refine generated use cases.

- **UML Use Case Diagram:** Req2Design will automatically generate a UML 2.x Use Case Diagram directly from the textual use cases. The diagrams will comply with UML standards, including rules for system boundaries, actor placement, and associations. This provides users with both textual and visual representations of requirements for clarity.
- **Document Export:** The system will support exporting the generated SRS, use cases, and diagrams into academic-friendly formats such as PDF and DOCX. This ensures the outputs are ready for direct inclusion in university submissions, research papers, or professional documentation.

Out of Scope: To maintain focus and feasibility, several advanced features will not be part of this project:

- Advanced NFR categories (e.g., scalability, maintainability, security).
- Traceability matrices linking requirements to design and testing.
- Feasibility studies or cost-benefit analysis.
- Class diagrams, sequence diagrams, or any design-level diagrams beyond use cases.
- Extended use case flows such as alternative paths and exceptions.

By focusing only on the **core and most essential aspects** of requirements documentation, Req2Design ensures that the generated outputs are both manageable within the project scope and genuinely useful for stakeholders such as students, researchers, and developers.

2.2 Modules

The system **Req2Design** is divided into four main modules. Each module performs a distinct function but works in an integrated manner with the others to achieve the overall goal of automating requirement documentation. The modular approach ensures separation of concerns, scalability, and clear mapping of system functionalities.

2.2.1 Module 1 — Requirement Intake & Preprocessing + SRS (Part 1)

This module is responsible for capturing the raw requirements from the user and converting them into a structured and analyzable form. It forms the entry point of the system.

- **Input Collection:** Requirements can be provided either as plain text (typed or pasted by the user) or as audio recordings.
- **Audio-to-Text Transcription:** If the input is audio, the system employs a speech-to-text engine such as *Whisper* to accurately transcribe user-provided recordings into textual requirements.
- **Preprocessing:** The system applies natural language processing (NLP) techniques to clean the raw text, resolve ambiguities, and identify unclear terms. Where ambiguity is detected, the system prompts the user to clarify, ensuring requirement quality.
- **Initial SRS Generation:** Based on the preprocessed input, the system generates the *Introduction* and *Overall Description* sections of the IEEE 830-compliant SRS. These sections provide an early overview of the system scope, purpose, and context.

2.2.2 Module 2 — SRS (Part 2) Completion & Export

This module expands the partially generated SRS into a complete document, ensuring adherence to IEEE 830 standards.

- **Functional Requirements:** Captures detailed functional requirements in a structured, numbered format (e.g., FR-01, FR-02), clearly describing what the system must do.
- **Non-Functional Requirements:** Documents core NFRs such as usability, reliability, performance, and portability. These are expressed in measurable terms where possible to enhance clarity.
- **External Interfaces:** Specifies how the system will interact with users, hardware, software, and communication interfaces. This ensures that the boundaries of the system are well-defined.
- **Document Export:** The finalized SRS is exported into both PDF and DOCX formats, making it suitable for academic submissions or professional use.

2.2.3 Module 3 — Use Case Documentation

This module focuses on converting the textual requirements into structured use case documentation following Alistair Cockburn's template.

- **Actor and Goal Extraction:** Automatically identifies key actors (users or external systems) and their primary goals from the requirement text.
- **Cockburn Template Application:** Generates textual use cases containing only the most essential elements:
  - Actor(s)
  - Goal / Brief description
  - Preconditions
  - Main flow of events
  - Postconditions
- **Refinement and Editing:** Provides users the ability to edit or refine the generated use cases to ensure correctness and contextual accuracy.

2.2.4 Module 4 — Use Case Diagram Generator

This module automatically creates UML diagrams from the textual use cases to provide a visual representation of the system's functionality.

- **Diagram Generation:** Creates UML 2.x compliant use case diagrams directly from Cockburn-style textual use cases.
- **Validation Rules:** Ensures diagrams follow UML best practices, such as actors being outside the system boundary, proper placement of system boundaries, and correct use of relationships (include, extend, association).
- **Multiple Output Formats:** Allows exporting diagrams in popular formats including SVG, PNG, and PlantUML source code for easy integration with documentation tools.

2.3 Tools & Technologies(Tentative)

- **Frontend:** React/Next.js, Tailwind CSS
- **Backend:** FastAPI or Django
- **Database:** PostgreSQL
- **NLP/AI:** HuggingFace Transformers, spaCy, Whisper
- **Rendering:** PlantUML, Mermaid.js
- **DevOps:** Docker, GitHub Actions, AWS/Heroku

2.4 Work Division

Name	**Registration**	**Primary Modules**	**Shared Responsibilities**
Ali Rizwan	22I-2447	Module 1 & Module 2	UI/UX Design, Testing, Documentation
Muhammed Shahmeer	22I-1522	Module 3 & Module 4	UI/UX Design, Testing, Documentation
Abdul Haq Zulfiqar	22I-2585	Module 2 & Module 3	UI/UX Design, Testing, Documentation

2.5 Timeline

Iteration	Time Frame	Tasks/Modules
01	Sept–Nov 2025	Requirement Intake & Preprocessing + SRS Part 1 (Module 1)
02	Dec–Feb 2025	SRS Part 2 Completion & Export (Module 2)
03	Mar–Apr 2026	Use Case Documentation (Module 3)
04	May–Jun 2026	Use Case Diagram Generator (Module 4), Final Testing & Integration

3 Conclusions & Future Work

Req2Design automates requirement documentation by generating IEEE 830-style SRS, Cockburn-style textual use cases, and UML 2.x Use Case Diagrams. By automating repetitive, error-prone tasks, it ensures consistency, clarity, and saves time for students, startups, and developers.

Future Work: Expansion could include class diagram generation, traceability matrices, feasibility analysis, and automatic generation of code skeletons for end-to-end requirement-to-design workflows.