- *Section 5 (Conclusion and Outcomes)* – for final insights into deliverables and benefits.

- **Developers and Test Engineers** should focus on:
  - *Section 2.2.3 (External Interface Requirements)* – covering user, hardware, and software interfaces.
  - *Section 2.2.4 (Non-Functional Requirements)* – covering performance, security, and safety considerations.
  - *Section 3.4 (Tools and Technology Used)* – for implementation environment and dependencies.
- **Project Managers** should focus on:
  - *Section 2.2.1.3 (Project Scope)* – defining boundaries and deliverables.
  - *Section 2.3 (Cost Analysis)* – for resource allocation and feasibility.
  - *Section 4 (Methodology and Outcomes)* – for workflow and milestone planning.
- **Researchers and Academicians** should consult:
  - *Section 1.4 (Literature Survey & Research Gaps)* – to understand novelty and prior work.
  - *Section 3.4 (Tools and Technology Used)* – for implementation choices.
  - *Bibliography/References* – for academic grounding.

2.2.1.3 Project Scope

The scope of *CodeCodez* defines the boundaries, objectives, deliverables, and overall vision of the project. This section highlights what the system will achieve, what it will not cover, and how it fits into the broader context of software engineering and AI-driven automation.

1. Project Overview

CodeCodez is designed to automate the **end-to-end generation of production-ready software projects** from natural language specifications. It integrates multiple specialized Large Language Models (LLMs) with a **hierarchical task decomposition engine**, enabling systematic breakdown of user requirements into modular subtasks such as frontend, backend, testing, and documentation. The system aims to reduce **manual coding efforts**, accelerate **software delivery cycles**, and enhance **developer productivity** by handling repetitive or boilerplate tasks while leaving high-level design decisions to human developers.