INTRODUCTION

1.1 Project Overview

Background and Motivation

In the rapidly changing world of software development, programmers are usually stuck doing the same old groundwork prior to even starting the actual development. Activities such as organizing directory hierarchies, initializing modules, environment configuration, and boilerplate code writing waste precious time and brain power. While AI-driven tools and LLMs (Large Language Models) have come out as mighty aids in coding code snippets, they lack in creating cohesive, production-ready projects that reflect the design and depth of actual software. There is a wide chasm between code generation and project generation, and it is just this gap that this project CodeCodez aims to fill.

Project Vision and Purpose

CodeCodez is designed as an intelligent, end-to-end system that automates the generation of full software projects from structured natural language input. Rather than stopping at simple function generation, the goal is to deliver fully structured repositories, complete with modular code, documentation, setup configurations, testing frameworks, and deployment readiness. This vision redefines the role of AI from a passive helper to an active co-developer, capable of producing robust software components tailored to user requirements.

Strategic Decomposition of Tasks

At the heart of the system lies a smart task decomposition mechanism. When a user inputs a broad objective such as "Build a fast API server for fibonacci series" CodeCodez begins by breaking it into subtasks using a **tree-based hierarchy**. This hierarchical view helps separate concerns such as user interface design, API logic, data processing, and model inference. These subtasks are then translated into a **Directed Acyclic Graph (DAG)** to reflect interdependencies, allowing the system to process components in an order that mirrors real development workflows.

This intelligent decomposition ensures that modules are generated in the right sequence, with each component having the context and prerequisites it needs. It also enables easy debugging and