5	102217240	Navdeep S. Sidhu	MAEL	Experiential Learning	Learning from multi-agent experience	[9]
			Context Engineering	Prompt Structuring	Better reasoning using engineered context	[17]
			LLM Code Gen Survey	Comparative Survey	Overview of LLM agent capabilities	[19]

TABLE 2: Research Findings

2.1.4 Problem Identified

Despite the rapid advancement of Large Language Models (LLMs) in software development assistance, a major limitation lies in their ability to autonomously and efficiently handle complex, multi-step programming tasks. Most existing systems operate in a monolithic fashion, relying on a single model to understand, plan, and generate code for an entire application. This often results in hallucinations, logical inconsistencies, and bloated code outputs that are difficult to debug or scale.

Moreover, the lack of structured task decomposition makes it challenging to align the generated code with software engineering principles like modularity, abstraction, and clarity. While frameworks such as Chain-of-Thought (CoT) and Tree-of-Thought (ToT) reasoning have shown promise in enhancing LLM decision-making, their application in real-world software generation is still limited and largely experimental.

A further issue is the underutilization of **multi-agent collaboration** strategies where multiple specialized LLMs could cooperatively handle subtasks like UI generation, API integration, documentation, testing, and DevOps setup. Current single-agent systems fail to leverage the advantages of role-based parallelism, leading to slower development cycles and reduced reliability of outputs.

Therefore, there is a pressing need for a **hierarchical, multi-agent, and tree/DAG-based orchestration framework** that can accurately interpret user intent, decompose tasks, assign them to appropriate AI agents, and iteratively refine the code through verification, testing, and optimization phases. This will not only improve the accuracy and quality of code generation but also make the process more explainable, scalable, and developer-friendly.