2.3 Risk Analysis

Risk	Description	Impact	Mitigation Strategy
LLM Hallucination	Large Language Models may generate code that is syntactically correct but functionally incorrect.	High	Implement multiple verification layers including test case generation, correction loops, and validation.
Integration Failures	Code generated by multiple LLMs may not integrate seamlessly due to differing styles, assumptions, or versions.	Medium	Use prompt chaining with context preservation and assign role-specific LLMs for consistent outputs.
Dependency Mismatches	Conflicts may arise from package versions or library incompatibilities during project setup.	Medium	Auto-check environment dependencies and freeze versions using requirement files.
Testing Bottlenecks	Auto-generated test cases might not cover edge cases or integration issues.	Medium	Include both unit and integration test templates with feedback loops from execution logs.
Infrastructure Limitations	Limited access to GPU or server downtime could affect LLM response time or fail generation during large-scale tasks.	Medium	Utilize hybrid local-cloud infrastructure and include retry and failover mechanisms.

TABLE 3: Technical Risks

Risk	Description	Impact	Mitigation Strategy
Data Leakage via Prompts	Sensitive information might get exposed via prompt history or logging.	High	Mask sensitive inputs and ensure secure prompt transmission without logging.
Bias Propagation	Pretrained LLMs might reflect biases in generated outputs or documentation.	Medium	Apply ethical filters and conduct regular audits of generated code and comments.

TABLE 4: Security and Ethical Risks