**Context Aware testing system for financial ecosystem**

Team Name: AI Problem Solvers

Solution Name: AI-Powered Context-Aware Testing System

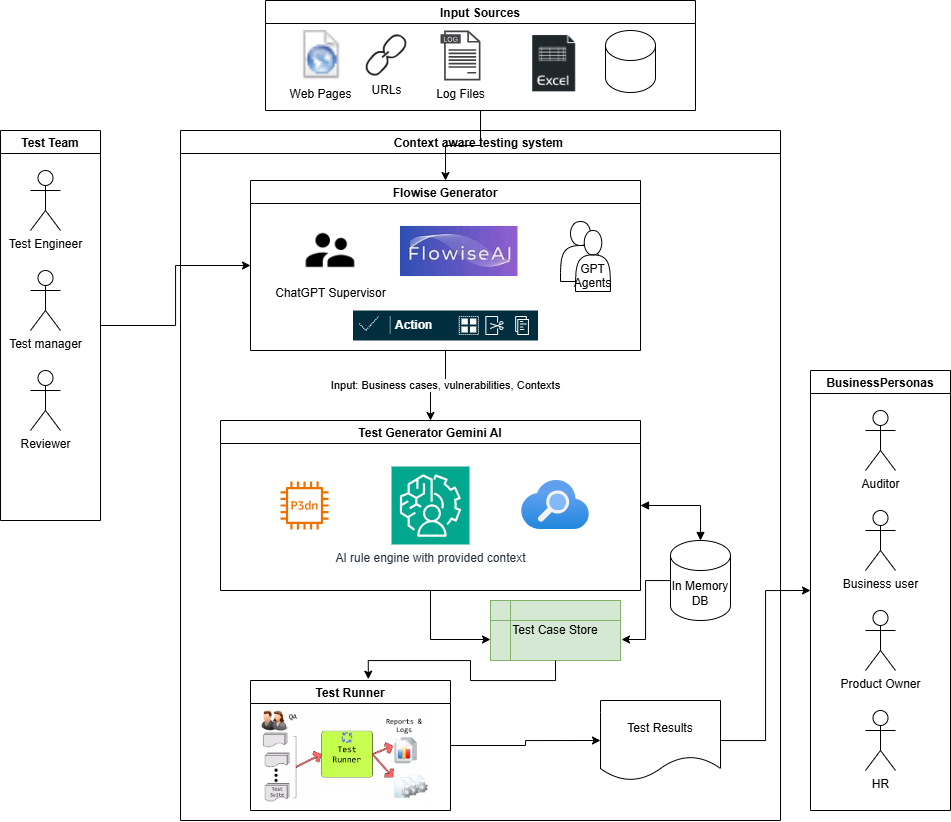
**1. Introduction**

Financial institutions operate within intricate ecosystems composed of numerous interdependent applications, APIs, and transactional systems. Traditional testing methods rely heavily on static mocks and manually crafted test scenarios, leading to inefficiencies, high maintenance costs, and limited adaptability to dynamic system changes.

To address these challenges, our solution introduces an AI-driven Context-Aware Testing System that leverages Generative AI to intelligently create, modify, and adapt test scenarios based on real-time system contexts. This approach significantly reduces manual effort while enhancing test coverage and effectiveness.

**2. Solution Overview**

context-aware testing system integrating Flowise AI and Gemini AI for automated test generation. It processes inputs (web pages, logs, Excel) to create business-context-driven test cases, which are stored in-memory and executed by a test runner. Results are shared with test teams and business personas (auditors, product owners, etc.).



Our system consists of three core components:

1.1. Context Analyzer (The Flow)

Financial institutions operate in **highly dynamic and interconnected environments**, where testing various systems, APIs, and transactions presents challenges such as **high maintenance costs, static mock tools, and inefficient test coverage**. Traditional test automation methods fail to **adapt to changing contexts**, leading to gaps in validation.

To solve this, we propose a **context-aware AI-driven testing system** using **Flowise 2.3.2**. Our approach dynamically **generates, adapts, and executes test cases based on real-time data**, ensuring **higher accuracy, efficiency, and risk-based test coverage in plain language.**

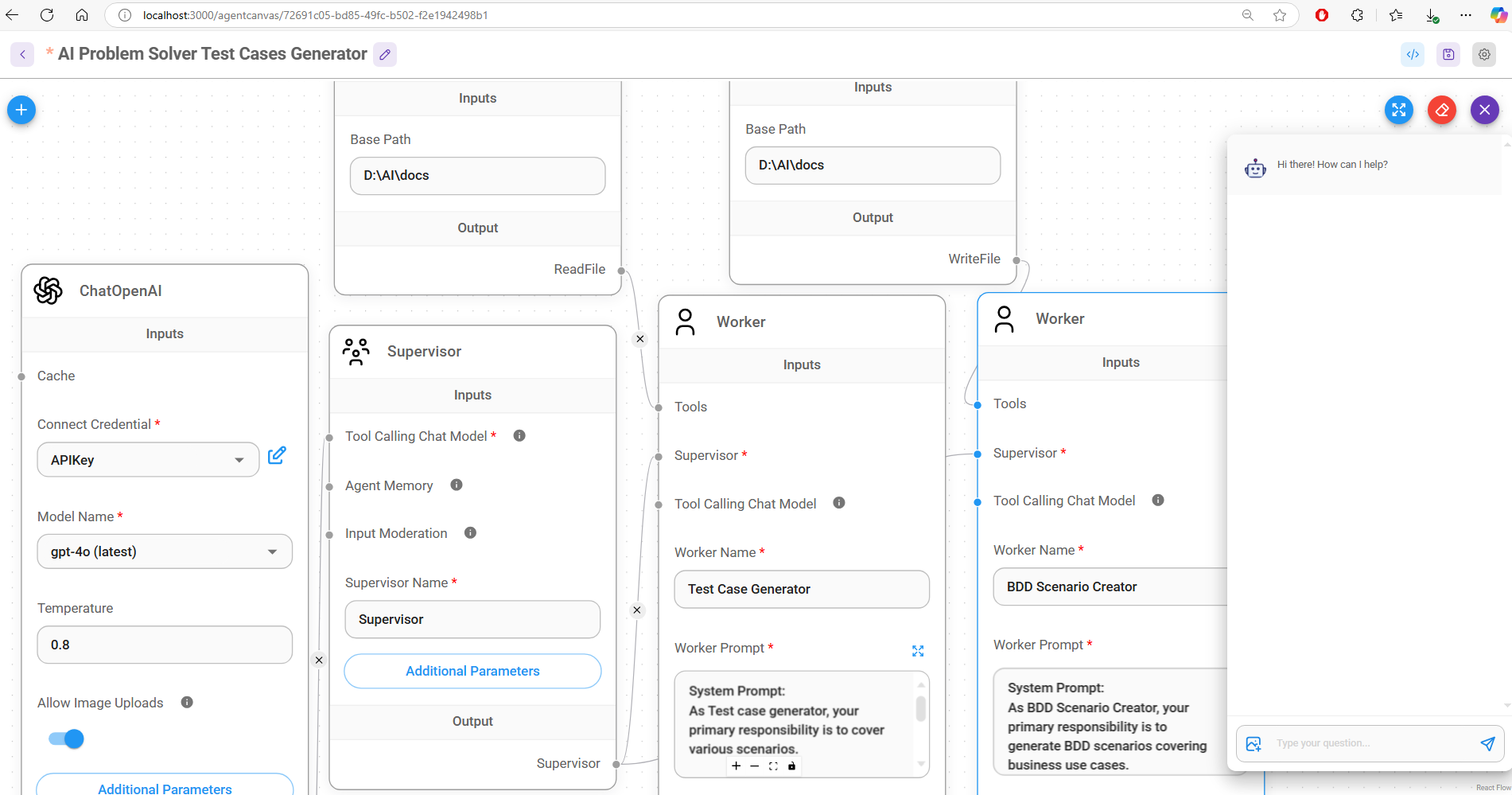


Figure 1FLow to generate Automated business test case

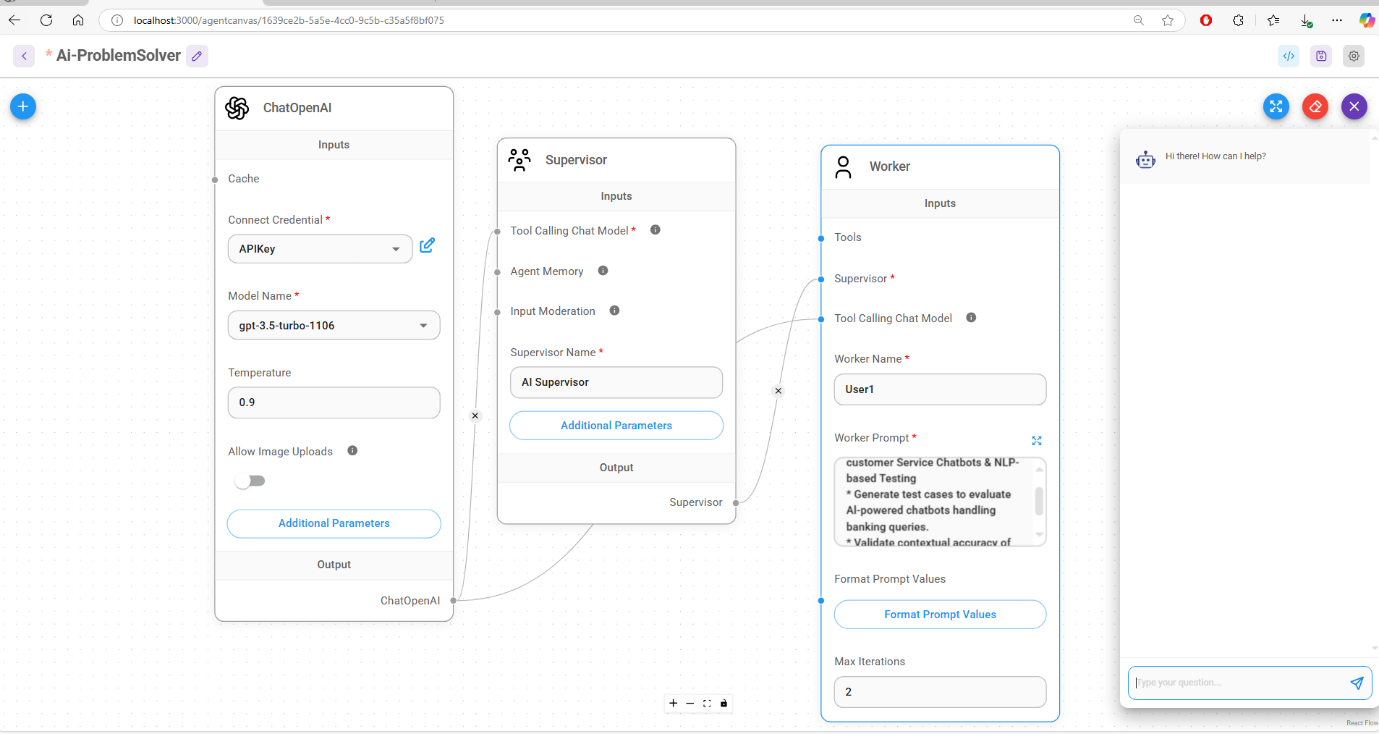


Figure 2Flow to Find test cases based on the user quires

2.2. AI Test Generator

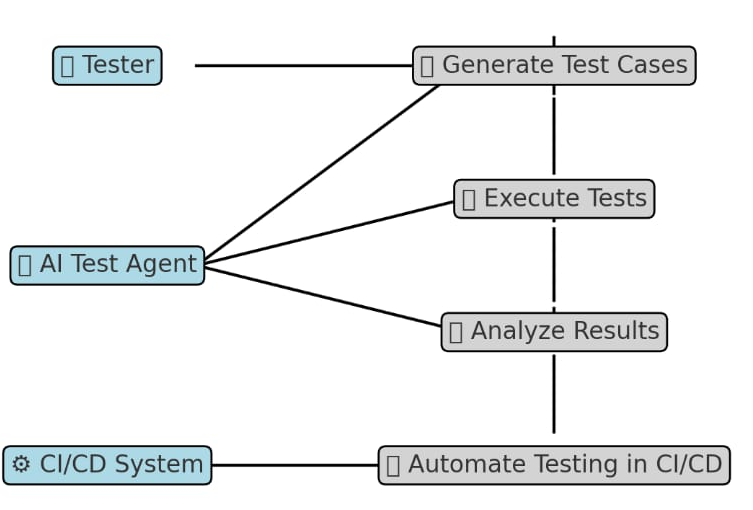
Uses Generative AI to dynamically create comprehensive test cases based on URLs and DOM.

Figure 3 The generator flow

Ensures coverage for various test types, including functional, regression, edge cases, and security tests.

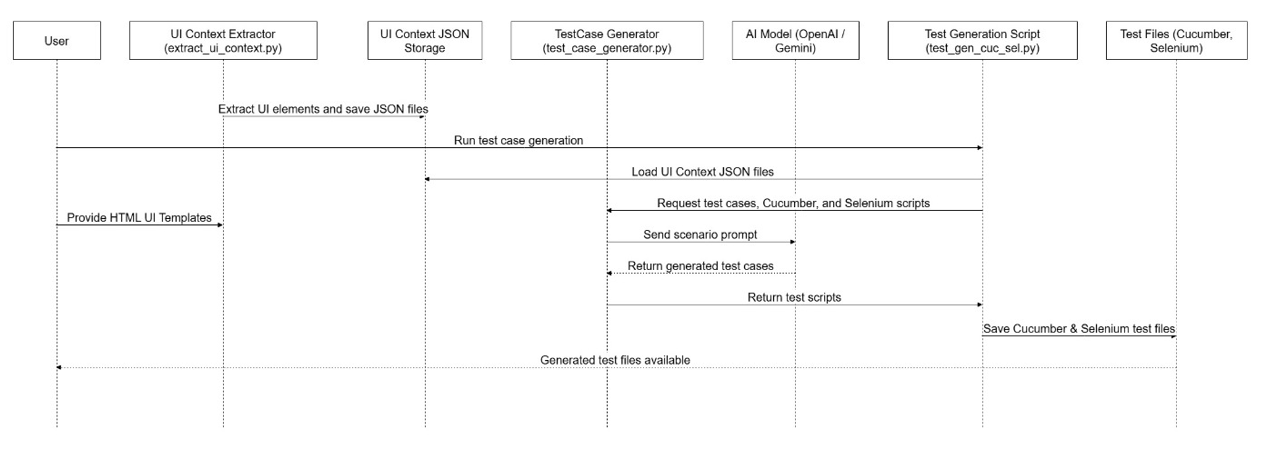


Figure 4 Sequence diagram of code generator

2.2.1 Smart Execution

- Executes generated test cases in real-world financial environments.

- Uses AI-driven analytics to prioritize high-risk scenarios.

- Generates detailed reports with insights on test coverage, defects, and system vulnerabilities.

**3. Key Features & Benefits**

AI-driven test case generation - Reduces manual effort and speeds up test creation.

Dynamic context adaptation Enhances accuracy by adapting to evolving financial ecosystems.

Automated API & transaction validation -Increases efficiency and reduces dependency on static mocks. |

Comprehensive risk-based test coverage - Ensures critical business workflows are fully tested.

Intelligent execution & reporting- Prioritizes high-risk test cases for optimized testing.

**4. Technical Architecture**

1. Data Sources: APIs, transaction logs, and system configurations, User Stories, URLs

2. Context Analyzer: Extracts key data points and evolving system changes.

3. AI Test Generator: Uses Generative AI models (e.g., GPT-4, LLMs) to create adaptive test cases.

4. Execution Engine: Runs tests in real-time and captures results.

5. Reporting Dashboard: Displays insights, trends, and actionable recommendations.

**5. Implementation Approach**

1. Data Collection: Ingest transactional data, system logs, and API interactions.

2. AI Model Training: Fine-tune AI models for financial transaction validation.

3. Prototype Development: Build an MVP to demonstrate real-world applicability.

4. Integration & Testing: Connect with real financial systems and assess performance.

5. Optimization & Scalability: Improve accuracy, reduce test execution time, and scale across multiple ecosystems.

**6. Expected Impact**

- 80% reduction in manual test case creation effort.

- 50% faster test execution and validation cycle.

- Enhanced test coverage, ensuring robust financial system reliability based on changes in document object model.

- Reduced maintenance costs, eliminating static test case dependency.

**7. Future Enhancements**

- Self-healing test automation to auto-fix failing tests.

- Integration with CI/CD pipelines for seamless DevOps adoption.

- Enhanced fraud detection testing by simulating complex fraud patterns.

**8. Conclusion**

Our AI-powered Context-Aware Testing System revolutionizes financial ecosystem testing by leveraging Generative AI to create dynamic, adaptive, and intelligent test cases. By reducing manual efforts and increasing efficiency, our solution ensures higher test coverage, better risk management, and seamless compliance validation in evolving financial environments.

9. Team Members & Roles

1. **Ayush Raj (Captain)** - AI & Machine Learning Engineer

2. **Lakshmi Priya Bandi** - AI & Machine Learning Engineer

3. **Madan Mohan Yaram** - AI & Machine Learning Engineer

4. **Rajesh Takkella** – UI Expert & Business Analyst

5. **Ramakrushna Mishra** – AI & Machine Learning Engineer

**10. Technology Stack**

- AI & ML: OpenAI GPT-4, Google-Generative-AI (gemini-1.5-flash) & Flowise AI(>2.3.2)

- Backend: Python, Node.js, Angular, Java

- Frontend: Python, beautiful-soup4 selenium,

- Database: H2B In memory database

**11. References**

<https://docs.flowiseai.com/>

<https://gemini.google.com/app?hl=en-IN>

<https://angular.dev/>

<https://www.python.org/downloads/>

<https://nodejs.org/en/download>

<https://app.diagrams.net/>