**User Acceptance Testing (UAT) Template**

|  |  |
| --- | --- |
| Date |  |
| Team ID | LTVIP2025TMID32102 |
| Project Name | SmartzSDLC – AI-Enhanced Software Development Lifecycle |
| Maximum Marks |  |

**Project Overview:**

Project Name: SmartSDLC – AI-Enhanced Software Development Lifecycle.

Project Description: SmartSDLC is a full-stack, AI-powered platform that redefines the Software Development Lifecycle (SDLC) by automating key stages using advanced NLP and Generative AI (Watsonx Granite-20B). It converts unstructured requirements into actionable code, test cases, and documentation, streamlining the SDLC pipeline.

Project Version: v1.0

Testing Period: 21-06-2025 to 23-06-2025

**Testing Scope:**

Features to be Tested:

1. Requirement Upload and Classification
2. AI Code Generator
3. Bug Fixer
4. Test Case Generator
5. Code Summarizer
6. Floating AI Chatbot Assistant

User Stories / Requirements to be Tested:

* Uploading PDFs and classifying sentences into SDLC phases.
* Generating code from natural language prompts.
* Fixing syntax/logical errors in uploaded code.
* Creating test cases from code or requirement input.
* Summarizing code logic and structure.
* Providing contextual help via an AI chatbot.

**Testing Environment:**

URL/Location: https:\Localhost:5000

Credentials (if required):N/A

**Test Cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| TC-001 | Requirement Classification | 1. Upload PDF  2. Click “Classify”  3. View results grouped by SDLC phase | Requirements sorted as per SDLC stages |  | Pass |
| TC-002 | Code Generation | 1. Enter prompt “Create login function”  2. Click “Generate”  3. View code | AI-generated code matching prompt |  | Pass |
| TC-003 | Bug Fixer | 1. Paste buggy code  2. Click “Fix”  3. Compare before and after | Logical and syntax errors fixed |  | Pass |
| TC-004 | Test Case Generator | 1. Paste working function 2. Click “Generate Test Case” 3. View output | Valid unittest/pytest test cases generated |  | Pass |
| TC-005 | Code Summarizer | 1. Paste code 2. Click “Summarize” 3. Read explanation | Accurate and human-readable summary |  | Pass |
| TC-006 | Floating AI Chatbot Assistant | 1. Ask “How to write a unit test?” 2. View response 3. Ask “What is requirement analysis?” | Intelligent and helpful contextual answers |  | Pass |

**Bug Tracking:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bug ID** | **Bug Description** | **Steps to reproduce** | **Severity** | **Status** | **Additional feedback** |
| BG-001 | Code Generator outputs syntax errors | Enter complex prompt, click generate | Medium | In Progress | Needs better parsing for edge cases |
| BG-002 | Classifier misses some non-English text | Upload multilingual PDF | Low | Open | Add multilingual support if possible |
| BG-003 | Chatbot repeats answers | Ask similar questions repeatedly | Low | Open | Could benefit from session memory |

**Sign-off:**

Tester Name: Lakshmi Divya Thota

Date: 25-06-2025

Signature:

