Test Plan for LoanPro Calculator CLI

1. Introduction

1.1 Background

The LoanPro Calculator CLI is a command-line tool designed to perform basic arithmetic operations, including addition, subtraction, multiplication, and division. While initial unit testing has been performed, this test plan aims to thoroughly validate the application's functionality, reliability, and robustness. The objective is to uncover any bugs, performance issues, or inconsistencies before the product is released to end-users.

1.2 Purpose

The purpose of this document is to outline the testing approach, define the scope of testing, set objectives, and detail the resources and schedule for the testing phase. This plan ensures that all critical features of the LoanPro Calculator CLI are tested and that the application meets the required quality standards.

2. Scope

2.1 In-Scope

- Functional testing of the four arithmetic operations: addition, subtraction, multiplication, and division.
- Negative testing, including scenarios with invalid inputs and operand errors.
- Edge case testing, focusing on the handling of extreme values and boundary conditions.
- Regression testing to ensure that previous bugs remain fixed and do not reoccur.
- Compatibility testing across supported environments: macOS (Apple silicon and Intel), Linux (x86_64 and ARM64).
- Automation of test cases using Python scripts for continuous testing and integration.

2.2 Out-of-Scope

- Performance and load testing are not covered in this test plan.
- Usability testing and user experience evaluation are outside the scope.
- Testing on unsupported platforms, including Windows.

3. Objectives

3.1 Functional Validation

- Ensure that all arithmetic operations return correct results.
- Validate that error messages are user-friendly and accurately describe the issue.
- Confirm that the application handles invalid input gracefully without crashing.

3.2 Edge Case Handling

- Test the application’s ability to manage large numbers, precision limits, and division by zero.
- Verify the correctness of results when inputs approach the boundaries of expected values.

3.3 Automation and Efficiency

- Develop a suite of automated tests using Python to minimize manual testing effort and ensure consistency.
- Integrate automated tests into a continuous integration pipeline for ongoing quality assurance.

4. Test Strategy

4.1 Test Methodology

- **Black-box Testing:** Focus on input-output validation without knowledge of internal code structure.
- **Exploratory Testing:** Identify potential edge cases and undocumented behaviors through ad-hoc testing.

4.2 Test Phases

- **Unit Testing:** Conducted by developers to validate individual functions.
- **Integration Testing:** Ensure that all components work together as expected.
- **System Testing:** Validate the full application in a production-like environment.
- **Regression Testing:** Re-test after bug fixes or updates to ensure no new issues are introduced.

4.3 Test Execution

- **Manual Testing:** For exploratory and edge case testing that may not be easily automated.
- **Automated Testing:** For repetitive, predictable test cases, and regression scenarios.

5. Test Environment

5.1 Hardware and Software Requirements

- **Operating Systems:**
    - macOS (Apple Silicon, Intel)
    - Linux (x86_64, ARM64)
- **Docker:** Required for running the CLI tool in isolated containers.
- **Python:** For scripting automated test cases.

5.2 Environment Setup

- Ensure Docker is installed and properly configured.
- Pull the latest version of the LoanPro Calculator CLI Docker image.
- Set up Python environment with necessary libraries (e.g., `subprocess` for running CLI commands).

6. Test Tools

6.1 Test Automation Tools

- **Python:** The primary scripting language for test automation.
- **PyTest:** For structuring and running automated test cases.
- **Docker:** To simulate the production environment for the CLI tool.

6.2 Defect Tracking Tools

- **JIRA or Bugzilla:** For logging, tracking, and managing defects found during testing.

7. Test Schedule

7.1 Milestones and Timeline

- **Test Planning:** 2 days
- **Test Case Design:** 3 days
- **Environment Setup:** 1 day
- **Test Execution:**
    - Manual Testing: 2 days
    - Automated Testing: 2 days
- **Bug Reporting:** Ongoing during test execution
- **Regression Testing:** 2 days
- **Final Reporting and Test Closure:** 1 day

7.2 Dependencies

- Access to the latest version of the LoanPro Calculator CLI Docker image.
- Availability of the test environment and necessary tools.
- Timely access to development resources for bug fixes and clarifications.

8. Resources

8.1 Human Resources

- **Test Engineer:** Responsible for designing, executing, and automating tests.
- **Developer:** Available for addressing issues and fixing bugs identified during testing.
- **DevOps Engineer:** To assist with CI/CD pipeline setup and maintenance.

8.2 Tools and Software

- **Test Automation:** Python, PyTest
- **Defect Tracking:** JIRA or equivalent
- **Docker:** For running the CLI tool in isolated environments