**1. Test Strategy**

|  |  |
| --- | --- |
| Introduction | The test strategy outlines our approach to testing the API of the fake post management application. The goal is to ensure the API is functioning correctly, find as many bugs as possible, and provide meaningful reports on the testing results. |
| Objectives | * Validate the functionality of all API endpoints. * Ensure data integrity in the SQLite database. * Verify user authentication using JWT. * Test edge cases and error handling. |
| Scope | API Endpoints:   * User creation * Post creation * List all posts (including public and user-specific) * Update and delete user posts   Authentication via JWT |
| Test Types | * Functional Testing: Verify that each endpoint works as expected. * Integration Testing: Ensure the API integrates correctly with the SQLite database. * Security Testing: Check authentication mechanisms using JWT. * Negative Testing: Test with invalid inputs to ensure the API handles errors gracefully. |
| Test Environment | * Docker container with the SQLite database located at /app/sql\_app.db. |
| Tools | * Programming Language: Python, using libraries such as requests and unittest/pytest for test automation. * Reports: Using pytest-html to generate detailed reports. |
| Testes case | * Detailed test cases and scenarios for each feature will be documented and executed. |