# **Notes App Test Plan**

## **1. Introduction**

This document outlines the testing strategy for the Notes App, a web-based application designed for creating, editing, and managing personal notes. The purpose of this test plan is to define the scope, approach, and resources required to ensure the application's quality, functionality, and usability before its release. Comprehensive testing is crucial to identify and fix defects early, ensuring a stable and reliable user experience.

## **2. Scope**

### In Scope:

The testing efforts will focus on the core functionalities of the Notes App, including:

#### User Management:

* + **Registration:** Verifying new user sign-up with valid and invalid data.
  + **Login:** Testing successful and unsuccessful user authentication.
  + **Logout:** Ensuring a user can securely log out of their account.

#### Notes Management:

* + **Create Note:** Validating the creation of new notes with various content types.
  + **Edit Note:** Confirming the ability to modify existing notes.
  + **Delete Note:** Verifying the successful removal of notes.
  + **View Notes:** Ensuring all notes are displayed correctly for the logged-in user.

#### User Interface (UI):

* + Testing the look and feel, layout, and responsiveness of the application on different devices.
  + Verifying that all buttons, links, and forms are interactive and functional.

### Out of Scope:

The following areas are considered out of scope for this testing phase:

#### **Performance and Load Testing:** Evaluating the app's behavior under heavy load or stress.

#### **Security Penetration Testing:** Advanced security testing to identify vulnerabilities beyond basic access control.

#### **Automated Accessibility Testing:** Ensuring compliance with WCAG standards for users with disabilities.

#### **Third-party Integrations:** Testing any external services that may be integrated in the future.

## **3. Testing Approach**

Our testing approach combines manual and API testing to ensure comprehensive coverage.

### **Manual Testing:** This will be the primary method for functional, UI, and usability testing. A dedicated QA tester will execute the test cases manually to simulate real user interactions.

### **API Testing:** Using a tool like Postman, we will test the backend endpoints directly to validate data integrity and business logic independently of the UI. This helps in early detection of issues and provides a solid foundation for the application's core logic.

### **Automation:** If time permits, we will begin developing a suite of automated tests for critical, high-priority user flows (e.g., login, create note) to ensure regression coverage for future releases.

## **4. Test Levels and Types**

### Test Levels:

* **Unit Testing:** (Developer responsibility) Individual code components are tested to ensure they function as expected.
* **Integration Testing:** Testing the interaction between different modules (e.g., the registration form and the user database).
* **System Testing:** (QA responsibility) Testing the complete, integrated application to verify that it meets the specified requirements.
* **User Acceptance Testing (UAT):** A limited group of end-users will test the app in a realistic environment to ensure it meets their needs and expectations.

### Test Types:

* **Functional Testing:** Validating all specified features and functionalities as per the requirements.
* **UI/Usability Testing:** Checking the application’s user interface for visual correctness and ease of use.
* **Negative Testing:** Performing tests with invalid or unexpected input (e.g., incorrect passwords, blank fields) to ensure the system handles errors gracefully.

## **5. Test Environment and Data**

### **Browsers:** Chrome, Firefox, Safari, and Microsoft Edge (latest stable versions).

### **Operating Systems:** Windows 10/11, macOS, and popular mobile OS versions (Android and iOS).

### **Test Data:** We will use a predefined set of test data for user accounts (e.g., testuser@example.com with password password123) and sample notes. This ensures test reproducibility and consistency.

## **6. Entry and Exit Criteria**

### Entry Criteria

* The application build is stable and deployed to the testing environment.
* All major features (as per the scope) are implemented.
* The test plan and test cases are reviewed and approved.

### Exit Criteria

* All high-priority and medium-priority test cases have been executed.
* No high-severity defects remain open.
* All critical bugs identified are resolved and verified.
* A final test summary report is prepared and approved.

## **7. Risks & Mitigation**

### **Risk:** The login feature fails, blocking access to other functionalities.

* + **Mitigation:** Prioritize testing the login feature first. If it fails, all other functional testing will be blocked, and we will report the issue immediately.

### **Risk:** A critical bug is found late in the cycle.

* + **Mitigation:** Implement daily stand-ups to review progress and reported bugs. Focus on testing critical paths early in the cycle.

### **Risk:** The test environment is unstable.

* + **Mitigation:** Coordinate with the development and DevOps teams to ensure the test environment is stable and available before testing begins.

## **8. Deliverables**

Upon completion of the testing phase, the following deliverables will be provided:

### **Test Cases:** The detailed test cases spreadsheet.

### **Bug Reports:** A list of all reported defects with their status.

### **Test Summary Report:** A final report summarizing the test results, including metrics on executed tests and defect status.

### **Postman Collection:** If API testing is performed, a collection of API requests will be provided.