Software Test Plan for Pocket App

Prepared by: [Doron Haim]

---

1. Purpose

This Software Test Plan (STP) is designed to outline the testing strategy for the Pocket app, focusing on ensuring its quality, functionality, and performance. It aims to provide a structured approach to testing and highlight key aspects that contribute to a successful testing phase.

---

2. Objectives

- **Verify Functionality:** Ensure all core features work as intended.
- **Assess Usability:** Evaluate basic user interactions and navigation within the app
- **Test Compatibility:** Confirm that the app performs correctly across the tested Android versions and BlueStacks.
- **Check Integration:** Validate that various app features and systems work together seamlessly.

---

3. Scope

Included:

- **Functional Testing:** Verification of article saving, retrieval, and sharing functionalities.
- **Usability Testing:** Assessment of user interface and navigation ease.
- **Accessibility**: Testing across various devices and operating systems.
- **GUI:** Checking for spelling errors, numerical mistakes.
- **Integration:** Testing the integration between at least two or more systems
- **Exploratory:** Combines learning, designing, and executing tests simultaneously to uncover defects through creative and intuitive approaches.
- **Security Testing:** Evaluation of authentication processes and data protection.
- **Accessibility:** Testing the app's support for accessibility features to ensure usability for users with disabilities.

Excluded:

- **Performance Testing:** Not included due to scope limitations.
- **Recovery Testing:** Not applicable with current resources.

---

4. Testing Approach

1. Functional Testing

- **Objective:** Validate core functionalities against requirements.
- **Method:** Execute test cases covering all functional aspects.
- **Focus Areas:** Article management, synchronization, sharing features.

2. Usability Testing

- **Objective:** Evaluate the app’s user experience and interface.
- **Method:** Conduct usability studies and gather user feedback.
- **Focus Areas:** Onboarding process, navigation ease, and overall user satisfaction.

3. Accessibility Testing

- **Objective:** Ensure the app is accessible to users with disabilities.
- **Method:** Test using accessibility tools and evaluate compliance with standards.
- **Focus Areas:** Screen reader functionality, keyboard navigation, color contrast.

4. Integration

- **Objective:** Verify app integration across different devices and platforms.
- **Method:** Perform tests on various devices.
- **Focus Areas:** Cross-platform functionality, browser compatibility.

5. Security Testing

- **Objective:** Ensure login errors aren't specific and block unauthorized access.
- **Method:** Test authentication by verifying error handling during login attempts.
- **Focus Areas:** Secure login, data protection.

6. Exploratory Testing

- **Objective:** Discover potential issues through creative testing.
- **Method:** Perform ad-hoc tests and explore unusual use cases.
- **Focus Areas:** Unexpected user behaviours and edge cases.

---

6. Resources

- **Testing Tools:** Google Drive, BlueStacks X, Payment’s methods.
- **Devices:** iOS and Android smartphones, Windows 11 laptop
- **Browsers:** Brave, Chrome.

7. Criteria for Success

Entry Criteria:

- **Feature Completion:** Core features are implemented and ready for testing.
- **Test Environment:** Complete setup of testing devices and browsers.
- **Documentation:** Test Plan, Test Cases, and bug tracking systems are in place.
- **Resource Availability:** Testing tools and android devices along with payment methods.

Exit Criteria:

- **Test Execution:** At least 80% of test cases are executed.
- **Defect Documentation:** Major bugs are documented with resolution plans.
- **Deliverables:** Software Test Plan (STP), Software Test Description (STD), Bug Reports and Software Test Report (STR) are finalized.

8. Feature list:

1. User Registration and Login
2. Saving Content
3. Sharing Saved Content
4. Offline Access
5. Reader function
6. Settings and Premium
7. Saves Management
8. Tagging and Organizing
9. Search Functionality

9. Bug Reporting Guidelines

This section will provide detailed instructions on how bugs should be documented to ensure consistency and clarity in reporting.

1. **Bug Title:** Provide a clear title for the bug that summarizes the issue.
2. **Bug Description:** Include a detailed description of the bug, explaining what the problem is and how it affects the system.
3. **Steps to Reproduce:** List all the steps required to reproduce the bug.
4. **Expected Result:** Describe what the expected behaviour of the system should be when following the steps above.
5. **Actual Result:** Describe what happens when following the steps.
6. **Bug ID:** Assigned ID for tracking purposes.
7. **Severity:**
    - **Low:** Minor issue easily resolved.
    - **Medium:** Affects functionality but has workarounds.
    - **High:** Significant issue needing urgent attention.
    - **Critical:** Core functionality unusable, needs immediate resolution.

Notes, Including Environment Details: Include any additional relevant information.