## horizontal line

Test Plan for WEare Social Network

**─**

[**Git: WEareProject\_Testing**](https://github.com/Alpha61-QA-Team3-Final-Project/WEareProject_Testing.git)

Gergana Georgieva

Daniel Ivanov

Velislav Petev

Telerik Academy Alpha QA

Sofia, Bulgaria

**1. Introduction**

This document describes the test plan for the Social Network web application - WEare. Testing will focus on both functional and non-functional aspects of the application. WEare Social Networking Platform enables users to connect, share, and engage in social activities online. The Testing Strategy defines the scope, methods, resources and timeline. The goal is to ensure the platform meets functionality requirements and provides a seamless user experience.

### **2. Objectives and Tasks**

**2.1 Objectives:**

* Implement a structured testing process that addresses functionality ultimately enhancing the overall quality of the user experience. This will be fulfilled with execution of different testing types and results analysis.

**2.2 Tasks:**

* **Test Documentation** - create test plan, detailed test cases, test scripts, and test data to ensure thorough test coverage.
* **Testing Activities** - conduct comprehensive testing of WEare social networking platform functionalities, including user registration, profile management, post creation, liking, commenting and connectivity features.
* **Test Execution -** execute the defined test cases on the WEare platform, monitoring for any deviations from expected behavior.
* **Post-Testing Activities** - summarize and analyze test results, producing comprehensive test reports.
* **Documentation Maintenance** - keep test documentation up to date to reflect any changes in the testing process.
* **Test Closure** - prepare for the formal closure of testing activities, including final reports.

### **3. Scope of Testing**

**General Scope:**

* User Registration and Login Functionalities:
  + Validating Happy path and Unhappy path for User Registration.
  + Validating Happy path for User Login.
* Functionality Requirements for Posts:
  + Post Content Functionalities.
  + Like/Unlike Button Functionalities.
  + Commenting on Post Functionalities.
* User Actions:
  + User Authentication and Logout.
  + User Profile Update.
  + User Connection Requests.
  + Post Creation and Visibility.
  + Interactions With Posts.
* Administrative Actions for Admin Users.
  + Admin Login and Logout.
  + Admin Profile Management.

### **4. Hardware and Environment Requirements**

* **Hardware**: Standard desktop/laptop with at least 8GB RAM for running Selenium and JMeter.
* **MacOS**, **Windows** **10, 11**.
* **Software Tools**:
  + **Browsers**: Chrome, Firefox, and Edge for cross-browser testing.
  + **Project Management Tool - JIRA**: for tracking of tasks and bugs.
  + **GitHub:** for source code management and collaboration**.**
  + **IntelliJ IDEA:** integrated development environment for coding and scripting.
  + **Docker:** for creating and managing containers for application deployment and testing.
  + **Selenium WebDriver**: for automated UI testing.
  + **Postman and REST-Assured**: for API testing.
  + **JMeter**: for load testing.
  + **MySQL:** to interact with the database for data-related testing.
  + **Microsoft Teams:** for team communication and collaboration.

### **5. Test Schedule**

The test schedule will follow this timeline:

1. **Week 1 (21.10 - 27.10)**: Complete the test plan, test scenarios and test case preparation.
2. **Week 2 (28.10 - 03.11):** Perform functional and non-functional manual testing based on test scenarios.
3. **Week 3 (04.11 - 10.11)**: Develop and perform automation functional testing of the core features (users, posts, comments).
4. **Week 4 (11.11 - 17.11)**: Develop API testing and start preparation for test reports.
5. **Week 5 (18.11 - 26.11)**: Finalizing test reports and evaluation of exit criteria.

### **6. Resources/Roles & Responsibilities**

* **QA Experts (Gergana Georgieva, Daniel Ivanov, Velislav Petev)**: Test Planning, Schedule & Task Management, Manual Testing, Automation Testing, Functional and Non-Functional Testing.

### **7. Risks and Assumptions**

* **Risks**: Incomplete requirements or unclear specifications may lead to delays in testing.
* **Risks**: Insufficient time for execution of all tests.
* **Assumptions**: The testing environment will be stable, and test data will be available during testing.

### **8. Entry criteria**

* All functional and non-functional requirements are clearly defined, documented, and approved.
* The test plan is completed, reviewed, and approved.
* Test cases or test scripts are prepared.
* Test data is prepared, reviewed, and loaded into the test environment.
* Test environment is configured and ready for testing.

### **9. Exit criteria**

* **Successful Execution of Test Cases (UI, API)**:
  + All test cases for the core functionalities such as **user registration**, **login**, **logout**, **post (CRUD)**, **likes and dislikes**, **comment (CRUD)**, **administrative actions** have been successfully executed with status “pass”.
  + Test cases with medium or low priority may be left unexecuted if there are blockers.
* **Defects**:
  + No major or critical defects remain that could block or significantly disrupt the user experience.
  + Some minor defects may remain unresolved, as long as they do not affect the core functionality or hinder the user experience. These can be fixed in future versions.
* **Test Execution Time Limit**:
  + The time limit for testing is reached.
  + End of testing time period - 26.11.2024.

### **10. Severity and Priority definitions**

| **Severity Value** | **Definition** |
| --- | --- |
| Critical  (Blocker) | The functionality is substantially inoperable or unusable, causing a significant impact on multiple end users. No workaround exists, and the issue must be resolved immediately. |
| High | A major functionality is not working, or there is a substantial performance issue affecting multiple end users. A workaround is available and can be implemented. |
| Medium | Defects that cause the website to malfunction but do not significantly impact end-user operations. Also applies to low-impact inquiries, such as "How-to" questions. |
| Low | Minor errors or documentation issues that do not affect the end-user experience. Suitable for product enhancement requests. |

|  |  |
| --- | --- |
| **Priority Value** | **Definition** |
| Highest | A condition whereby a critical issue severely impairs the core functionality of the system, rendering it inoperable or significantly impacting multiple end users. No viable workaround is available, and immediate attention is required to restore normal system operation. |
| High | A condition whereby a major issue disrupts system functionality or performance, affecting multiple end users. While a workaround exists, addressing the issue promptly is essential to minimize disruption and ensure a satisfactory user experience. |
| Medium | A condition where system malfunctions or non-critical issues occur, causing some inconvenience to end users. Workaround options are available, and the problem does not significantly hinder day-to-day operations. |
| Low | A condition involving non-critical system errors, minor documentation issues, or low-impact product inquiries. End-user functionality remains largely unaffected, and the problem may not be urgent. |
| Lowest | A condition primarily related to suggestions for product improvements, enhancements, or documentation enhancements. These items do not represent critical issues and can be scheduled for future development or documentation updates. |