Advanced Tic Tac Toe

Testing Summary Report

By

The X Factor

Table of Contents

1. Scope

The scope of this testing summary report focuses on the specific tests and results obtained for the Database and Game components of the project. The testing efforts targeted key functionalities within these components to ensure their correctness and reliability.

2. Test Environment Details

2.1 Hardware Details

- **Device**: MacBook Pro M1 Pro
- **Hardware Constraints**: None

2.2 Software Details

- **Operating System**: macOS Sonoma 14.5

2.3 Development and Testing Tools

- **Testing Framework**: Google Test version 1.14
- **IDE**: XCode
- **Compiler**: Clang

2.4 Dependencies and Libraries

- **Database Library**: SQLite3
- **Standard Libraries**: iostream, vector

2.5 Database Details

- **Database Management System**: SQLite3

2.6 Network Configuration

- **Network Environment**: The device was in a networked environment with no specific network constraints

This environment was used to conduct all the unit tests for the Database and Game components of the project, ensuring consistent and reliable testing conditions.

3. Summary of Testing Types and Results

3.1 Functional Testing

Purpose: To ensure that each function in the Database and Game components performs as expected.

- **InitializationTest**:
  - **Objective**: Verify the successful initialization of the database.
  - **Result**: Passed.
- **SignUpNewUserTest**:
  - **Objective**: Ensure new user registration is successful.
  - **Result**: Passed.
- **SignUpExistingUserTest**:
  - **Objective**: Handle existing user registration attempts appropriately.
  - **Result**: Passed.

- **LoginSuccessTest**:
  - **Objective**: Verify successful login with correct credentials.
  - **Initial Result**: Failed due to an incorrect password issue.
  - **Action Taken**: Reimplemented the login function to fix the bug.
  - **Final Result**: Passed.
- **LoginFailureTest**:
  - **Objective**: Ensure login fails with incorrect credentials.
  - **Result**: Passed.
- **Game AI Tests (Easy, Medium, Hard Levels)**:
  - **Objective**: Verify AI move logic and decision-making.
  - **Result**: Passed. All AI move logic tests (corner case, blocking moves, picking center and corner cells, making random moves, winning moves) were successful.

3.2 Integration Testing

Purpose: To ensure that different components interact correctly when integrated.

- **Database and User Authentication**:
  - **Objective**: Verify the integration of database operations with user signup and login processes.
  - **Result**: Passed. The integration was seamless, with the database correctly handling user data during signup and login.
- **Game Mechanics and AI Logic**:
  - **Objective**: Ensure the game mechanics and AI logic work together smoothly.
  - **Result**: Passed. The AI made appropriate moves based on the game state and difficulty level.

3.3 Regression Testing

Purpose: To ensure that recent changes or bug fixes do not negatively impact existing functionalities.

- **Login Function Reimplementation**:
  - **Objective**: Verify that the reimplemented login function works correctly and does not affect other functionalities.
  - **Result**: Passed. The login function worked correctly, and no other functionalities were impacted.

4. Recommended Improvements

4.1 Expand Test Coverage

To ensure even greater reliability and robustness of the Database and Game components, it is recommended to expand the test coverage to include additional edge cases and scenarios. Some areas to focus on include:

- **Special Characters and Edge Cases**: Test with special characters in usernames, extremely long passwords, and other atypical inputs to ensure the system can handle them gracefully.
- **Different Network Conditions**: Simulate various network conditions to test the resilience and performance of the system under different scenarios.

4.2 Incorporate Performance Testing

In addition to functional testing, it is crucial to understand how the system performs under various loads. Performance testing can provide insights into the system's behavior under stress and help identify any bottlenecks or performance issues. Recommendations include:

- **Load Testing**: Simulate a high number of concurrent users to test the system's scalability and performance.
- **Stress Testing**: Push the system beyond its limits to see how it handles extreme conditions and to identify any potential points of failure.

4.3 Enhance Usability Testing

Ensuring that the system is user-friendly and intuitive is as important as its functional correctness. Usability testing can help identify areas for improvement in the user interface and overall user experience. Steps to consider:

- **User Feedback**: Gather feedback from actual users to understand their pain points and areas where the user experience can be improved.
- **Usability Studies**: Conduct formal usability studies to observe how users interact with the system and identify any usability issues.

4.4 Continuous Monitoring and Logging

Implementing continuous monitoring and detailed logging can provide real-time insights into the system's health and performance. This will help in quickly diagnosing and resolving issues as they arise. Recommendations include:

- **Monitoring Tools**: Use monitoring tools to track key performance metrics and alert on any anomalies.
- **Detailed Logging**: Ensure that the system logs detailed information about its operations, which can be invaluable for troubleshooting and debugging.