Testing Documentation

Prepared by:

Chat GPA Team

Under supervisor of DR:

Omar Nasr

Table of Contents

1. Introduction

The testing documentation for the Advanced Tic Tac Toe Game offers an in-depth review of the testing efforts aimed at ensuring the game's functionality, reliability, and performance. This section details the purpose of the testing documentation, the target audience, and recommendations for understanding the testing process more effectively.

1.1 Purpose

The objectives of this testing documentation are to:

- Present a comprehensive account of the testing strategies, test cases, and coverage reports conducted for the Advanced Tic Tac Toe Game.
- Confirm the accuracy, dependability, and performance of the game's features through thorough testing.
- Serve as a reference for developers, testers, and project stakeholders to evaluate the game's quality and readiness for release.

2. Testing Strategies

2.1 Unit Testing

Unit testing is a critical strategy used in the testing documentation for the Advanced Tic Tac Toe Game. It involves testing individual units or components of the software in isolation to ensure their correctness and functionality. The primary focus is on validating key components such as the Game class, Player class, AI algorithms, and other essential parts. All these tests are conducted within a single class called Test.

Approach

The Test class is designed to cover various aspects of the game, including:

1. **Game Logic Tests:**
    - Board Initialization: Verifies that the game board is correctly set up at the start.
    - Player Moves: Ensures that player moves are accurately processed and reflected on the board.
    - Win/Tie Detection: Confirms that the game correctly identifies win or tie conditions.
    - Game Outcomes: Validates that the game correctly reports the outcome (win, lose, tie) after each game**.**

2. **Player Action Tests:**
    - Move Validation: Tests that player moves are correctly validated.
    - Profile Management: Ensures that player profiles are properly managed, including creating and updating profiles.

- Interface Interaction: Verifies that players can interact with the game interface as expected.

3. **AI Behavior Tests:**
   - Algorithm Implementation: Assesses the AI opponent’s behavior, ensuring it follows the minimax algorithm with alpha-beta pruning.
   - Difficulty Levels: Ensures that the AI operates correctly across different difficulty levels.
4. **Session Management Tests:**
   - Game Addition: Tests that new games are correctly added to the session.
   - Score Updates: Verifies that scores are accurately updated throughout the session.
   - Data Retrieval: Ensures that session data can be retrieved correctly.
5. **Database Operation Tests:**
   - Data Saving: Confirms that game data, user profiles, and session information are saved correctly.
   - Data Retrieval: Ensures that saved data can be accurately retrieved from the database.
6. **Game History Tests:**
   - Record Accuracy: Verifies that past games are correctly recorded.
   - Storage Integrity: Ensures that game history is stored without errors.
   - Review Access: Confirms that users can access and review their game history correctly.

Tool

The Test class utilizes the Qt Test framework for writing and executing unit tests. This framework provides a structured approach to validate the functionality of the game's individual components, ensuring comprehensive coverage and reliability.

2.2 Integration Testing

Integration testing is essential to validate the interaction and integration of different modules within the Advanced Tic Tac Toe Game. It focuses on ensuring that the components work together seamlessly and that data flows correctly between them. This type of testing is crucial for identifying issues that may not surface during unit testing but can impact the overall functionality and user experience of the game.

Tool: Integration tests are performed using the Qt Test framework and GitHub action as **(.yaml)** file**.**

3.Coverage Report

Coverage reports are a critical component of the testing strategy for the Advanced Tic Tac Toe Game. They provide insights into the extent of code tested by the test cases, helping identify untested parts of the code and ensuring thorough testing.

Purpose

The primary goals of coverage reports are to:

- Measure the percentage of the codebase executed by the test suite.
- Identify untested code sections.
- Ensure comprehensive testing by highlighting areas that need additional tests.

Tools

- **Qt Test Framework**: Used for running unit and integration tests.
- **Coverage Tools**: Tools like gcov, lcov, or Qt Creator's built-in coverage analysis tools are used to generate coverage reports.

3.1Key Coverage Metrics

1. **Statement Coverage**:
    - **Definition**: Measures the percentage of executable statements executed.
    - **Goal**: Achieve high statement coverage to ensure most code statements are tested.
2. **Branch Coverage**:
    - **Definition**: Measures the percentage of branches (decision points) executed.
    - **Goal**: Achieve high branch coverage to ensure all possible paths are tested.
3. **Function Coverage**:
    - **Definition**: Measures the percentage of functions called during test execution.
    - **Goal**: Achieve high function coverage to ensure most functions are tested.
4. **Path Coverage**:
    - **Definition**: Measures the percentage of possible execution paths traversed.
    - **Goal**: Achieve high path coverage to ensure all potential paths are tested.

Coverage Analysis Process

1. **Run Tests**:
    - Execute the full test suite using the Qt Test framework.
2. **Generate Reports**:
    - Use coverage tools to generate coverage data and reports.
3. **Analyze Data**:
    - Review reports to identify covered and uncovered code sections.
    - Analyze metrics to evaluate the effectiveness of the test suite.
4. **Identify Gaps**:
    - Highlight areas with low or no coverage.

   - Identify critical untested components.

5. **Improve Coverage**:
   - Write additional tests for uncovered areas.
   - Enhance existing tests for better coverage.
   - Re-run tests and regenerate reports to ensure improvements are effective.

.

4.Conclusion

The testing strategy used in this project effectively identified and resolved potential issues, resulting in a highly reliable and functional Advanced Tic Tac Toe Game. By implementing comprehensive unit and integration testing, along with detailed coverage analysis, all critical aspects of the application were thoroughly examined and validated.