**Tic-Tac-Toe**

**Team ByTe Me**

**Software Quality Assurance Plan**

**Version: (1) Date: (11/06/2016)**

**Document History and Distribution**

1. **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision #** | **Revision Date** | **Description of Change** | **Author** |
| Version 1.0 | 11/6/2016 | Initial Preliminary Documentation | Liejo Fernandus & Benjamin Monge |
|  |  |  |  |
|  |  |  |  |

1. **Distribution**

|  |  |  |
| --- | --- | --- |
| **Recipient Name** | **Recipient Organization** | **Distribution Method** |
| Shenli Yuan | Client | UHD Blackboard |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**TABLE OF CONTENTS**

1. [INTRODUCTION](#_30j0zll)

2. [TEST ITEMS](#_2s8eyo1)

[3. FEATURES TO BE TESTED](#_1ksv4uv)

[4. FEATURES NOT TO BE TESTED](#_2jxsxqh)

[5. APPROACH](#_z337ya)

[6. PASS / FAIL CRITERIA](#_qsh70q)

[7. TESTING PROCESS](#_1pxezwc)

[8. ENVIRONMENTAL REQUIREMENTS](#_2p2csry)

[9. CHANGE MANAGEMENT PROCEDURES](#_23ckvvd)

[10. PLAN APPROVALS](#_32hioqz)

# **INTRODUCTION**

(NOTE 1: THE SOFTWARE TEST PLAN GUIDELINES WERE DERIVED AND DEVELOPED FROM IEEE STANDARD FOR SOFTWARE TEST DOCUMENTATION (829-1998)).

*(Note 2: The ordering of Software Test Plan (STP) elements is not meant to imply that the sections or subsections must be developed or presented in that order. The order of presentation is intended for ease of use, not as a guide to preparing the various elements of the Software Test Plan. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content.)*

*The Introduction section of the Software Test Plan (STP) provides an overview of the project and the product test strategy, a list of testing deliverables, the plan for development and evolution of the STP, reference material, and agency definitions and acronyms used in the STP.*

**The Software Test Plan (STP) is designed to prescribe the scope, approach, resources, and schedule of all testing activities. The plan must identify the items to be tested, the features to be tested, the types of testing to be performed, the personnel responsible for testing, the resources and schedule required to complete testing, and the risks associated with the plan**.

**1.1 Objectives**

The objective is to test the tic tac toe game for any faults that it may arise as it been play. We will be testing the game for utility, reliability, performance and correctness. Each member of the team will play the game at different levels of the game to find any bugs. We will go over the use cases and perform testing on each one of them as a group. All the current features of the game will be tested.

**1.2 Testing Strategy**

The strategy is to test each use case to see if they perform correctly and to see if it satisfies the specification plan. Test all the features of the game like viewing the high score, login, new player registration etc.

*Specific test plan components include:*

* *Purpose for this level of test,*
* *Items to be tested,*
* *Features to be tested,*
* *Features not to be tested,*
* *Management and technical approach,*
* *Pass / Fail criteria,*
* *Individual roles and responsibilities,*
* *Milestones,*
* *Schedules, and*
* *Risk assumptions and constraints.*

**1.3 Scope**

**Testing will be performed at several points in the life cycle as the product is constructed. Testing is a very 'dependent' activity. As a result, test planning is a continuing activity performed throughout the system development life cycle. Test plans must be developed for each level of product testing.**

**1.4 Reference Material**

* Requirement Plan
* Software Management Plan
* Product Specification Plan
* Detail Design

**1.5 Definitions and Acronyms**

|  |  |
| --- | --- |
| **Acronym** | **Meaning** |
| AI | Artificial Intelligence |
| GUI | Graphical User Interface |
| FOV | Field of view |
| PvP | Player versus Player |
| PvE | Player versus Environment/Computer |
| TTT | Tic Tac Toe game |

# **TEST ITEMS**

The Test Items included in the Software Quality Assurance Plan are listed below:

* Requirements specification
* Design specification
* Users guide
* Operations guide
* Installation guide
* Features (availability, response time)
* Defect removal procedures
* Verification and validation plans.

**2.1 Program Modules**

Program modules are tested simultaneously after completion of each module to check for proper functionality of items and that they are properly implemented into the main application of the software product.

**2.2 Job Control Procedures**

The team was encouraged to follow a specific schedule where the team met routinely every week through group calls and meetings for the discussion and distribution of responsibilities and arrangements of tasks.

**2.3 User Procedures**

Testing of the user procedures will be followed step by step as documented to proof read and detect any errors. Any deviations from the step by step procedure will be documented. The user procedure will be repeated continually until the step by step procedure is completed with no deviations.

**2.4 Operator Procedures**

The software product will be tested for the bare minimum functionality needed to run the product. Proper troubleshooting and documentation will be made if errors are discovered.

# **3. FEATURES TO BE TESTED**

* The game starts correctly
* The game quits correctly
* New player registration
* Login to the game
* Single player mode
* Multiplayer mode
* View the game rules
* View the game high scores

# **4. FEATURES NOT TO BE TESTED**

* Audio functionality for the software product will not be tested currently as it is not implemented and is not within the requirements plan of the software product.
* Selecting which player, in a two-player game, gets the first move will not be tested because the program will randomly choose which player will gets the first move.

# **5. APPROACH**

Several methods are used to approach testing the different aspects of the product. These include: Component Testing, Integration Testing, Conversion Testing, Job Stream Testing, Interface Testing, Security Testing, Recovery Testing, Performance Testing, Regression Testing, Acceptance Testing, and Beta Testing

**5.1 Component Testing**

Components of the product are tested one by one to clarify the accuracy of the code used to program the product. This form of testing is important for finding bugs quickly. Single player and two-player modes are tested separately by trying to start a game in each mode. Registration components are tested to ensure proper records are archived in a local server.

**5.2 Integration Testing**

There are two levels of integration testing. One is the process of testing the software capability where the program functions without freezing/crashing or not performing the tasks assigned specifically. The other level is the algorithm of the AI for the difficulty settings of the game.This algorithm will be implemented into the code of the product so that the AI can use the respective algorithm once the user decides what mode of difficulty they would like to play against when playing single-player mode.

**5.3 Conversion Testing**

The product will only be implemented to work on the platform in which it is designed on, thus no conversion testing will take place.

**5.4 Job Stream Testing**

No job stream testing will be done as the product will only be able to work on the platform it is designed on.

**5.5 Interface Testing**

Integration regression testing is performed at any time an interface attribute has been changed, e.g. the value of a parameter that is passed. Every feature has a possibility of changing after changes are applied thus it is important to test if the implemented changes only affects what was intended to be changed and nothing else.

**5.6 Security Testing**

Test is performed on the Account login feature to ensure that a user can log into an existing account and denied access without the proper credentials required for logging into the system, at which point the user is notified to enter the correct credentials for the account. Registration feature will be tested to ensure that a user cannot create an account with information that has been used to register for an existing account. The product will also be tested to make sure that the registered accounts are recorded and archived in the database.

**5.7 Recovery Testing**

Testing is performed to ensure that the archived information in the database is backed up and restored to fully operational in the case of software failure.

**5.8 Performance Testing**

Every feature noted on the Requirement Plan is tested to ensure that the product responds instantly. Testing is done to ensure that the software product will run equally in all forms of the assigned Operating System.

**5.9 Regression Testing**

Testing will be conducted after applying changes to the program to check that every feature has a possibility of changing with only the intended changes as successful while not affecting other parts of the code.

**5.10 Acceptance Testing**

Acceptance testing is required to check the client’s requirements objectives and criteria can be checked by the client and accepted. All the features under ‘Features to be tested’ will be tested with the client to ensure the product meets its objectives.

**5.11 Beta Testing**

Once the product has passed Acceptance testing, the client will be able to beta test the product having met the assigned functional requirements. The client can then confirm with the team if there are any faults or failures with the product.

# **6. PASS / FAIL CRITERIA**

Every component of the product has specific requirements that are expected to surpass for it to be approved and be satisfactory for team members and the client.

**6.1 Suspension Criteria**

If the product does not give expected results or fails a test, the testing will be suspended until the problem has been resolved.

**6.2 Resumption Criteria**

Once a feature is in the suspension criteria, it will be tested again until the problem is fixed and gives the expected results. Only then will the testing resume.

**6.3 Approval Criteria**

The product features and items tested have specific and clear criteria requirements to meet the approval of the team and client

* The game starts immediately when the user selects the options they want to play with and hit start button
* The game quits immediately at any screen as soon as the user hits the exit button
* Account login feature functions work properly by registering new accounts into the database and denying users from creating accounts with information used in previous account registrations. Notifications pop up when the wrong credentials are entered for logging into an existing account asking the user to re-enter their account credentials.
* Single player mode functions properly allowing for a user to play a game session with the software AI
* Multiplayer mode functions properly allowing two users to play a game session against each other.
* The score history functions properly where a registered user is able to view their scores from previous game sessions. This list should show up instantly when the user clicks on the Score History button and is accessible to the user from the main screen.

# 7. **TESTING PROCESS**

The testing process for the software product includes development and management of the execution and non-execution strategies, test deliverables documented for the use of the team as well as the client, testing tasks performed by the team and the responsibilities of each team member in accordance with the testing procedure.

Team members will review the product together and share their efforts with other team members while running tests together with non-execution based strategies. Each team member will explain the purpose of the test they are responsible and walk through the rest of the team of the procedure of the test to be performed. The team will then review the product based on the tests aran and record faults/errors, if they are any. Preliminary documentations, preliminary analyses, and preliminary designs of the product are shared and revised by multiple team members to quickly find faults.

Black-box testing is conducted by the team to test the execution of the programmed items on the product. Multiple combinations of buttons are tested which can pertain to abnormal scenarios to detect faults and prevent any issues in the future by the QA members. The test will also give a better understanding to the team of how the AI responds to certain moves made on the game board to understand the AI’s decision making process. Registration and login information is tested by inputting various combinations of inputs and credentials to check the security of the product. The information is also tested to see the limits of the product to verify that it does not crash on its own.

The source code of the software product is shared and reviewed by each member of the Quality Assurance by using White-box testing methods. Any change or addition to the code is to be tested separately and recorded so that the rest of the team is made aware of the changes. Team members will then run test cases on objects individually and then again with the current source code. Management sets a predetermined number of faults detected for each part of the code for which, if the limit is reached, the part of code is discarded, redesigned and recorded.

**7.1 Test Deliverables**

The test deliverables will include:

* summary reports of testing done on the software product such as programming errors, insufficient data and/or incidents during testing.
* data taken from the AI which follows the set algorithm, that gives the success rate of the AI as well as a history of the game session results from testing.

**7.2 Testing Tasks**

Testing tasks are arranged and team members are assigned specific activities to perform for testing. Testing strategies need to be clarified among team members of what types of testing and the amount of testing will be necessary to find defects and faults with the software product. Test cases prepared in advance are used to check that the software product will actually meet all its requirements. Test data will consist of input data as well as database test data to use while executing test cases in the environment to carry out testing.

**7.3 Responsibilities**

Each team member is assigned roles and responsibilities on the software product to maximize working efficiency and productivity. The team leader is responsible in bridging communication between the client and team members to analyze testing and approve of actions taken. The co-leader is responsible of summarizing testing reports and plan documents, ensuring that the work productivity does not deviate from the assigned plans. The team secretary is responsible for documenting information received from quality assurance testing, scheduling, group meetings and communication with the team. The QA members are responsible for designing and implementing plans into the product, testing the program with each addition or change to the product.

**7.4 Resources**

|  |  |  |
| --- | --- | --- |
| Role | Minimum Resources Recommended | Specific Responsibilities |
| Test Manager | Travis, Liejo, Ben | Provide management oversight, technical direction, appropriate resources, management reporting. |
| Test Designer | James, Shawn | Identify, prioritize, and implement test cases. Generate test plan, and evaluate test effort. |
| System Tester | Travis, Liejo Ben | Execute tests, log results, recover from errors, and document defects. |
| Test System Administrator | Travis, James | Ensures test environment and assets are managed and maintained. Administer test management system. Install/manage work access to test systems. |
| Database Administrator | Shawn, Ben | Ensures test data environment and assets are maintained. |
| Designer | Travis | Identifies and defines the operations, attributes, and associations of the test classes. |
| Implementer | James, Liejo, Ben | Implements and unit tests the test classes and test packages. Creates the test classes and packages implemented in the plan. |

**7.5 Schedule**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Stage* | *In-Stage Assess. Date* | *Stage Exit Date* | *Deliverable (date delivered)* | *Work Product (date completed)* | *QA Activity* |
| *Planning* | *Sept.* | *Nov.* | *Project Plan* | *Project Schedule* | *Define requirements, Review plan procedures, roles and responsibilities assigned* |
| *Preparation* | *Sept.* | *Nov.* | *Functional Requirements Document Acquisition Plan Installation Plan* | *Revised Management Project Plan* | *Review plan procedures. Clarify roles and responsibilities.* |
| *Software Design* | *Oct.* | *Dec.* | *Functional Design Document Security Plans Training Plan Sys. Test Plan Acceptance Plan Conf. Mgmt. Pln. Conversion Plan* | *Design Plan. Software Test Plan. Revised Project Management Plan.* | *Trace design components to requirements. Trace requirements to design components* |
| *Programming and Integration* | *Nov.* | *Dec.* | *Implementation and System documentation* | *Revised Project Management Plan and Software Test Plan.* | *Implement design components into software product. Test product continuously.* |
| *System Testing and Acceptance* | *Nov.* | *Dec.* | *Test results System documentation (final) Operational system* | *Revised Project Management Plan and Software Test Plan.* | *Test product continuously.* |

# **8. ENVIRONMENTAL REQUIREMENTS**

The game is need it to be tested on a computer running on a windows base system, preferably on the latest operating system. It is not dependent if the tester is on the internet since it is not necessary. It is not necessary to have special test tools. It is only necessary to have a computer with a working monitor, keyboard and mouse/trackpad.

**8.1 Hardware**

The game is able to run in any personal computer as long as it’s running on windows operating system. No network requirement is necessary to complete the test activities. Since the game does not have an online feature.

**8.2 Software**

To be able to test the game the computer need to be running windows 7 or higher. Since it is being built using C#. A computer running apple operating system will not work.

**8.3 Security**

We will test the login feature to make sure that it is not easily that someone will tampered someone else scores and information. The game have a login feature to make sure to know who is playing the game. To be able to store the information to the rightful person.

**8.4 Tools**

No special tools are necessary. It is only necessary to have a working keyboard and a mouse/trackpad to be able to utilise the game. As every team member will take turns to test the game as everyone else on the team will be analysing how the each feature reacts to the command that the team member does.

**8.5 Publications**

The Requirement Plan, Software Management Plan, Product Specification Plan, Detail Design documents have more in depth information about how the game works. This documents should support our testing.

**8.6 Risks and Assumptions**

Assuming that the person will use the right operating system for the game. One of the constraints on testing is that it can not be test using Apple operating system. A Risk could be that we will miss an use case or that a bug will not be able to be detected using the use cases as tests. For this risk to not happen every team member will go thru the game and test it out to make sure everything is tested. Not enough time to test every single line of code. The team will take a look at the code and test it out to make sure that there is no bug hidden. For that we will use white box testing.For the the contingency plan for the risks are warn the users that the game is only playable on the specify operating systems.

# **9. CHANGE MANAGEMENT PROCEDURES**

Any changes made to the testing process should be discussed with team members during the team meetings. Team members should be notified when any additions or changes are made to the documents through email with the specific update. The Leader, Co-Leader and Secretary are responsible of appraising the finalization of the amended test plan change management.

# **10. PLAN APPROVALS**

**Leader: James Castillo**

**Signature: . Date:**

**Co-Leader: Travis Cook**

**Signature: . Date: \_**

**Secretary: Shawn Manalel**

**Signature: . Date:**