**DITZY**

**Software Quality Assurance Plan**

**Preliminary Version 1.0**

**April 3rd 2016**

**Document History and Distribution**

1. Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision # | Revision Date | **Description of Change** | **Author** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Distribution

|  |  |  |
| --- | --- | --- |
| Recipient Name | Recipient Organization | **Distribution Method** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

1. [Introduction 1](#_Toc215044574)

[2. Test Items 4](#_Toc215044575)

[3. Features To Be Tested 5](#_Toc215044576)

[4. Features Not To Be Tested 5](#_Toc215044577)

[5. Approach 5](#_Toc215044578)

[6. Pass / Fail Criteria 7](#_Toc215044579)

[7. Testing Process 7](#_Toc215044580)

[8. Environmental Requirements 8](#_Toc215044581)

[9. Change Management Procedures 9](#_Toc215044582)

[10. Plan Approvals 9](#_Toc215044583)

# Introduction

The DITZY game is an extended version of the standard 3X3 tic tac toe game. It is played on a 6X6 grid between two players (one may be a computer). The players will take turns to place a stone of their color and the winner will be the one who has the highest number of 4-stones in a row either vertically or horizontally or diagonally. While playing with the AI, the player can select the difficulty level either hard, medium or easy.

**1.1 Objectives**

The objective is to submit the project by April 28th 2016 within the time frame and to fulfill all the clients requirements and to provide a best working product, which is a DITZY game, played on a 6X6X4 grid where player has an option to play either with a AI or with a second player.

This document has the test plan for DITZY game which has following objectives:

* Identify existing project information and the proposed product to be tested
* List the recommended testing requirements
* Describe different testing strategies to be used
* List the deliverable elements of the test activities.

**1.2 Testing Strategy**

**Testing is the process of analyzing a software item to detect the differences between existing and required conditions and to evaluate the features of the software item.** (*This may appear as a specific document (such as a Test Specification), or it may be part of the organization's standard test approach. For each level of testing, there should be a test plan and an appropriate set of deliverables. The test strategy should be clearly defined and the Software Test Plan acts as the high-level test plan. Specific testing activities will have their own test plan. Refer to section 5 of this document for a detailed list of specific test plans.)*

We used Non-Execution-Based Testing here as follows:

1. Walkthroughs:

* Team members : Taylor, Daniel, Imran, Zach, Yiming.
* Team members are responsible for analyzing the specifications, finding all the faults and start working on the next work flow.
* To detect the faults in prototype by SQA representative and to follow up in the step of development.

1. Inspections:
   1. Overview : Inspection of all the documents including requirements, specifications, design and codes etc.
   2. Preparation: To analyze all the documents in detail and list all the faults if found.
   3. Inspection: By performing a walkthrough, in order to make sure if all the items are covered.
   4. Rework: Every issues to be resolved and re-tested to ensure no further faults are present.
   5. Follow-up: All the changes are checked to make sure if everything is correct.

Following features to be tested:

* Single Player
* Multiplayer
* Create User
* Login
* History
* Difficulty Level
* AI algorithm
* Exit Game

**1.3 Scope**

No further update are needed for this product. If needed it will notified to all the team members.

**1.4 Reference Material**

|  |  |
| --- | --- |
| **Material/Resources** | **Use** |
| Object Oriented and Classical Software Engineering, 8th edition | CS 3420 Spring 2016 |
| GitHub.com | Storing and uploading all the documents |
| Wikipedia | For definitions and information related to software engineering |
| Google drive | For uploading documents |
| Design Document | For planning out final execution of product |
| Webopedia | For Definitions and Acronyms |

**1.5 Definitions and Acronyms**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| 6X6X4 | A game is played between two players using stones of their choice of color. The player takes turns marking spaces on the 6x6 grid. Identification of the winner is based on the highest number of 4-stone in a row(either horizontally, or vertically, or diagonally) when the entire grid is occupied. |
| Platform | Software on which the Tic Tac Toe is expected to be designed. |
| Single player | Player plays against AI |
| Multi Player | Player plays against another player |
| Graphical User Interface | It is a type of interface that allows users to interact with electronic devices by the use of windows icons and menus that can be implemented at the user’s command. |
| Skill Level | Easy, Intermediate, Hard. They are the difficulty levels in single player game. |
| C# | C# is a simple, modern, general-purpose, object-oriented programming language developed by Microsoft within its .NET initiative led by Anders Hejlsberg. |
| DITZY | A name of game application. |
| Visual Studio | Integrated Development Environment for implementing the code |
| View History | View History of registered players scores and wins. |

|  |  |
| --- | --- |
| **Acronym** | **Meaning** |
| GUI | Graphical User Interface |
| AI | Artificial Intelligence |
| SQA | Software Quality Assurance |
| UML | Unified Modeling Language |
| SPMP | Software Project Management Plan |
| UC | Use Cases |
| SD | Sequence Diagram |
| DITZY | Overall Project Name |

# Test Items:

The items included in the preliminary test plan are:

**Integration testing:**  The phase in software testing in which individual software modules are combined and tested as a group.

* Run a program: The player will run the program and the main menu screen will appear.
* Create a username: The player will create a username in order to keep track of the player.
* View history: The player will be able to view his/her current history of past games played.
* Difficultly setting: If single player is selected, player can select the level of difficulty(easy, medium, hard).
* Game start: Once a difficulty setting has been chosen or player selects multiplayer mode, the game will display a 6x6 grid and the players can make moves.
* Exit: Player can exit the program at any time

**Regression testing:** In order to fix any bugs that can be detected during the integration testing. Following steps will be performed.

* Code fix will be done.
* Integration testing will be done after fixing all the bugs.

**System testing:** All the functions of the DITZY game is performed in order to check if it works according to the clients requirements.

**2.1 Program Modules**

Following are the modules in which the product is sub-divided for easier integration and testing.

* Log in/ create user name
* Display User interface
* Display History
* Difficulty Level
* Player Mode
* Play the Game

**2.2 Job Control Procedures**

*(Describe testing to be performed on job control language (JCL), production scheduling and control, calls, and job sequencing.)*

**2.3 User Procedures**

All members will test the final product for accuracy and to ensure the clients need have been met. Once the final testing is completed, the user documentation will be drafted.

**2.4 Operator Procedures**

The product has been written for windows operating system in C# and tested using the Visual Studio. Once the product is completed, running and debugging will take place in order to ensure the compatibility of the product with the client’s system.

# 3. Features To Be Tested

The following features to be tested:

* Login/ create a user name
* Single player mode
* Multiplayer mode
* Display history
* Start game
* Exit game

# 4. Features Not To Be Tested

All the features to be tested as they all are the part of clients requirements.

# 5. Approach

At this stage, the team leader and the team secretary will test all the pseudo code and the AI algorithm and look for all the errors and make sure to check the requirements of the product have been met according to our clients need. This process will consume around two weeks. Visual Studio will be used for compiling and debugging the product.

**5.1 Component Testing**

Each methods will be tested using Visual Studio. To test all the method we will use unit testing where each method will be tested to ensure that all methods are executed with no errors.

**5.2 Integration Testing**

* Top-down integration: If code artifact above sends a message to artifact below, then artifact above is implemented and integrated before below. Pros include fault isolation and major design flaws show up early. Cons include potentially reusable artifacts may not be adequately tested.
* Bottom-up integration: If code artifact above sends a message to artifact below, then artifact below is implemented and integrated before above. Pros include fault isolation and potentially reusable artifacts are adequately tested. Cons include major design faults are detected late.
* Sandwich integration: incorporates both top-down and bottom-up integration. Pros include fault isolation, major design flaws show up early, and potentially reusable artifacts are adequately tested. There are no cons.

**5.3 Conversion Testing**

Our product is not being converted from an old system format to a new one. Thus, we do not have historical data or data elements to show in this section.

**5.4 Job Stream Testing**

Job Stream testing will begin once we have a functional program or modules to be to test

**5.5 Interface Testing**

The interface testing mainly focused on the product’s GUI. Team member will be playing the actual games to monitor and compare the actual output.

**5.6 Security Testing**

The security testing is performing to check whether user can login with existed credential.

**5.7 Recovery Testing**

Recovery testing will begin once we have our data storage module up and running.

**5.8 Performance Testing**

The performance testing mainly focused on making sure the product runs accordingly with the client’s requirements.

**5.9 Regression Testing**

The regression testing will ensure each working update will be compatible with the previous version. In this case there will only be one version delivered to the client.

**5.10 Acceptance Testing**

Acceptance testing will be conducted when the product is completed. It will be performed by the client.

**5.11 Beta Testing**

Beta testing will conducted by the client at the end of the course.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type of Test | Specification | Scope | Verification and/or Validation | Category (White or Black) | Reponsibility (Programmer, Non-Programmer) |
| Unit Testing | Low Level | Small Code Units | Verification | White | Programmer |
| Integration | High Level | Multiple Classes | Verification Validation | Both | Both |
| Functional | High Level | Whole Product | Validation | Black | Non-Programmer |
| System | Requirements Analysis | Whole Product | Validation | Black | Non-Programmer |
| Acceptance | Requirements Analysis | Whole Product | Validation | Black | Customer |
| Beta |  | Whole Product | Validation | Black | Customer |
| Regression | Changed Docs | All | Both | Both | Both |

# 6. Pass / Fail Criteria

Every use case must work in order for the application to pass. If any of the use case fails to work then the whole application will fail.

**6.1 Suspension Criteria**

Testing will be suspended if:

* Defect in GUI responsiveness
* Unavailability of necessary hardware systems during execution
* Unavailability of human resources to conduct the testing.

**6.2 Resumption Criteria**

Resumption will take place if:

* The defect in the GUI is found, fixed and SQA team decide testing can resume
* The necessary hardware systems become available again
* When the human resources needed to conduct the testing are available

**6.3 Approval Criteria**

Once all the errors and faults resolved and after making sure all the requirement have been met according to the clients need, all members will approve.

# 7. Testing Process

*(Identify the methods and criteria used in performing test activities. Define the specific methods and procedures for each type of test. Define the detailed criteria for evaluating test results.)*

**7.1 Test Deliverables**

Testing deliverables are as follows:

* Successful testing reports
* Any faults encountered while testing.
* The input data used and the result of the input data.

**7.2 Testing Tasks**

Tasks necessary for testing include openly conversing about product and communicating with client on a regular basis. Once working modules are created, we will begin testing code and verifying that it is in accordance with client’s needs.

White box testing- each module once created will be tested by its creator, to be reviewed by SQA

Black box testing – once a working UI is developed along with internal code, testing will commence to ensure quality

**7.3 Responsibilities**

|  |  |  |
| --- | --- | --- |
| **Role** | **Responsibilities** | **Assigned to** |
| Team Leader | Lead the planning and execution of the project | Taylor Rowan |
| Development Team | Design and implement the software product | Taylor Rowan, Ming Zhu, Imran Hussain, Zach Myers, Daniel Owen |
| Project Manager | Coordinates and assigns project tasks to the team | Rowan Taylor |

**7.4 Resources**

Resources include requirements paperwork to ensure accuracy of client’s wishes Computers to test modules as they are developed.

**7.5 Schedule**

Each week testing will conducted on all paperwork to ensure the client’s requirements are being meet. Once programming begins and a module is completed, testing will be performed by the programmer who completed it and a SQA member of the team.

1st Phase – Beginning 4/3 - Ending 4/9

AI will be in a pre-Alpha stage, being feature complete with most features present. Testing will be done by Daniel and Zach.

UI will be in Alpha Stage. Testing by Taylor and Imran.

2nd Phase – Beginning 4/10 – Ending 4/16

Entering final testing of AI with Daniel leading and Zach and Ming testing as part of SQA.

UI will run through with a series of tests to ensure quality. Lead by Taylor with Imran and Ming as SQA.

3rd Phase – Beginning 4/17 – Ending 4/20

Modules of UI and AI will be merged lead by Ming to ensure compatibility and proper functioning.

4th Phase – Beginning 4/21 – Ending 4/25

UI and AI fully merged to be run through with a series of testing to ensure quality.

# 8. Environmental Requirements

The environmental requirements needed to design the software are:

* Personal Computer.
* Designing environment (Microsoft Visual Studio)
* Communication tools (Emails, Meetings, hangout app)

**8.1 Hardware**

Minimum hardware requirement for designing the software are:

* 2.0 GHz processor
* 1GB of RAM
* Either 32-bit or 64-bit operating system
* 20 GB of hard drive space

**8.2 Software**

Windows Operating System – Win XP /vista/7/8.0 or higher.

**8.3 Security**

No security will be needed for this environment.

**8.4 Tools**

Microsoft Visual Studio will be used for testing purpose.

**8.5 Publications**

The documents and publications that are required to support testing activities is the current document.

**8.6 Risks and Assumptions**

* Some members may drop from the course.
* Certain task may not be completed on time.

# 9. Change Management Procedures

Change Management Procedures are developed to ensure that changes are properly reviewed and approved by the team members with the required expertise prior to implementation.

# 10. Plan Approvals

This plan has been approved by all team members.

Taylor Rowan

Imran Hussain

Ming Zhu

Zach Myers

Daniel Owen