Seven Nation Army

Diplomacy

Test Plan

Version: 1.0 **Date: 03/07/2019**

Document History and Distribution

1. Revision History

Revision #	Revision Date	Description of Change	Author
1.0	03/07/2019	First Draft	

TABLE OF CONTENTS

1. Introduction

In order to ensure a smooth experience for the end user we will have to rigorously test our product; this document will lay out our plan to accomplish this. Our strategies, methods, and test features will be described in detail. References to our other documents may be made and will be noted in the references section at the end of the document.

The testing in this plan is limited in scope to what is defined in our Vision Document as well as tests for basic functionality; as that document is updated, so will our Test Plan.

2. Test Items

2.1 Account Management

These items pertain to how user accounts are managed. They will be tested to ensure user accounts function as intended and are handled appropriately.

2.2 Game Management

These items pertain to how games are created and stored for each player. Players should be able to create, join, and delete games they have created.

2.3 Game Mechanics

These items pertain to how the game is played. Almost every feature within this section will relate to a specific game rule and must be handled according to the rules of Diplomacy. Orders must function correctly for the game to proceed as intended.

2.4 Game Rendering

These items pertain to how the game is displayed on screen. Items must display correctly for the game to be played. We must test the rendering to ensure there won't be bugs that prevent the game from being seen and played.

2.5 Game Functionality

These items pertain to how the user interacts with the game. The primary mode of input will be with the mouse and keyboard. We must ensure that each click and keypress initiates the correct action.

3. Features To Be Tested and Their Priority

3.1 Account Management

- Creation - H
- Login - H
- Deletion - L

3.2 Game Management

- Creation - H
- Joining - H
- Leaving - M
- Deleting - L

3.3 Game Mechanics - H

- Hold
- Attack
- Move
- Support
- Convoy
- Retreat
- Disband
- Create Units
- Submit Orders
- Resolve Orders
- Changing of Seasons
- Winning
- Surrendering

3.4 Game Rendering

- Placement of Items on Screen - M

3.5 Game Functionality

- Clicking initiates actions - H

4. Features Not To Be Tested

- N/A

5. Approach

5.1 Tools Used in Testing

We will use the basic tools available to us. The game will be made in Electron so all our testing will be done within the IDE we choose to use. Unit testing will be of utmost importance to ensure each piece is functional before we move onto the next part.

5.2 Metrics To Be Collected

The only metrics we will collect are pass/fail results for our tests. We will keep track of what and where items fails.

5.3 Configuration Management

We will be using a combination of GitHub and Trello to manage this project. GitHub will help with version control and our documentation while Trello will help with task management. We hope that both of these will help establish a consistent product throughout this project's lifespan.

5.4 Configurations To Be Tested

We will only be testing this project on a desktop environment.We will test this on Windows and macOS and there are no guarantees beyond those operating systems.

5.5 Regression Testing

Each feature will be tested as its being implemented. Once a feature has been determined to be functional we will move onto the next. We hope that the Object Oriented nature of this project will prevent future features from affecting completed items, but we can never be sure that each item will be completely isolated from one another. So we will test each feature and its dependencies as we go and if a defect is detected we will trace it back to its root to determine the severity and what needs to be done to correct the issue.

5.6 Processing Untestable Elements

Seeing as we are responsible for the entire project, from documentation to testing to completion, there won't be many requirements that don't make sense. If anything in the requirements is too ambiguous and needs clarification a discussion must be had to clarify the item and rewrite it.

If we do encounter an item that is untestable we will check anything that is dependent on that item and make sure nothing has gone wrong. As it is untestable we can only follow the logic of the item to ensure the logic is correct and proceed from there.

5.7 Testing Constraints

The only constraints in testing are time and experience. We have a deadline to meet and cannot spend too much time testing every possible scenario. We can test the obvious scenarios and make sure the logic of the code follows, but we are unable to spend months testing our code.

We are also limited by our experience with making a game and working with Electron. This is a new venture for everyone on our team so we will be learning as we go, thus our methods for testing may not be the best, but we hope they will be adequate.

6. Pass/Fail Criteria

6.1 Unit Testing

A unit test is successful if all test cases have been completed without errors. Each test must be repeated multiple times to ensure the validity of the results. If there is an unexpected result it must be evaluated to determine if the error lies with the code or if there was an incorrect assumption made during testing.

Minor defects will be defined as defects that do not affect the outcome of the test or the operation of the code. We have decided that minor defects will be dealt with on a case-by-case basis and have not yet determined a minimum number of acceptable minor defects.

6.2 Master Test Plan Level

At the Master Test Plan level, success is defined as a full run through of the game without any errors that impact functionality or experience.

7. Suspension Criteria and Resumption Requirements

A test is defined as terminated if major changes must be made to it before continuing.

A test is defined as suspended if only minor changes must be made to it; such as an incorrect number or variable name. The logic of the test has not changed, only a minor value has been adjusted.

Any tests that return useful data and do not contain errors will be allowed to run until completion.

7.1 Suspension and Termination

If a series of tests on a single item continually returns failing results we may terminate the tests to determine if the error lies with the test itself or the item being tested. A 100% fail rate must mean there is an inherent issue somewhere along the line that must be dealt with before wasting time with more tests that will also fail.

If a series of tests is returning data that does not make sense it will also be terminated until the issue can be found. Tests that do not return useful data are not useful and must be corrected before more time is spent on them.

If a series of tests has been found to include incorrect data it will be suspended to remove the error and will continue once it has been corrected.

7.2 Resumption

A series of tests will always resume once any errors in the test or test item have been corrected. These tests cannot be ignored if we are to ensure the functionality of our product. Testing and retesting is vital to creating an operational game.

8. Test Deliverables

This test plan is to be delivered on 02/28/2019 along with other documentation that was created prior to active development. Because there hasn't been active development, only this plan and the test cases without test data will be delivered.

9. Test Tasks

N/A

10. Environmental Needs

- Most recent version of Electron
- Predefined test cases

11. Responsibilities

This project is largely a group effort and because parts of the code will be written by different people, each item will be tested by the person most familiar with it. Anyone will be free to test any items, but those most familiar with the code's function must be the primary tester.

12. Staffing and Training Needs

The only training required is to ensure we are all using the same versions of the same software. We must also ensure that we are all using the same test plan and use cases. No other special training is required.

13. Schedule

13.1 Scheduled Dates and Descriptions

Description	Planned Date
Begin Prototype Development.	3/8/19

Finalize testing of prototype.	3/19/19
Functional prototype of game.	3/21/19
Ensure prototype functions properly.	3/21/19
Begin Development of Final Product.	3/21/19
Finalize Testing of Final Product.	5/02/19
Ensure final product functions and last minute fixes	5/02/19
Complete Development of Final Product.	5/09/19

13.2 Slippage and Deviations from Dates

The dates presented above are rough estimates based on the date the prototype and final product are due. Once we begin there may be more milestones added in between the currently specified dates, but as we progress we will update this plan with the most current milestone projections.

Realistically we hope to meet the dates presented, but if we fall behind we will try to adjust the next dates to accommodate our slippage while keeping the final delivery date the same. If a deadline is approaching and we know we will not make it, we may request an extension of no more than a week.

14. Risks and Contingencies

14.1 Risks

- Running out of time
- Lack of experience with Electron causes numerous defects
- Lack of experience in game development causes a stall in development
- Changes being made to the design causes delays in testing

14.2 Contingencies

- Running out of time
    - Request an extension
    - Modify the design to accommodate the new timeline
- Lack of experience with Electron
    - Track down defects to specific bits of code

    - Research Electron examples to fill the gap in our knowledge
- Lack of experience with Game Development
    - See above point
- Changes in design
    - Reevaluate the timeline
    - Determine what changes are causing the issue
    - Determine whether changes are necessary
    - Remove design elements that cannot be completed

15. Approvals

Because this is a group project for a class the only parties involved are the professor and our group. The professor sets the deadlines for the prototype and final project, but it is up to our group to determine any deadlines in between and to decide when we are satisfied with testing and ready to move onto the next step.

As a group we will decide on the acceptable number of defects, the pass rate, and the number of functional features that will still designate a successful testing phase.