Connect4

Test and Coordination Plans

SIC

Sam S.

Ian M.

Connor J.

Course: CPSC 224 - Software Development

Instructor: Aaron S. Crandall

I. Introduction

I.1. Project Overview

The software that is to be tested is the backend of our Connect4 code. We will also be testing to see if the code that we have written works and “meshes” well with the rest of the code in order to accomplish what the software was designed to do. Some key requirements of the program:

- Customization screens for up to 2 players
- Difficulty selection for the single-player game mode
  - Working bots that will play at the selected algorithm corresponding to difficulty.
- Grid works and is capable of showing players' moves and piece positions.

I.2. Scope

The purpose of this document is to confirm and set in place a testing strategy and plan for our team to follow while developing the software/project. Our group will utilize a variety of tests to ensure that our front and backend all work as intended, as well as, work when put together to produce a finished product that is free of bugs.

II. Testing Strategy

.We will use the following flow/strategy for testing our software:

- Plan and define the scope of testing
- Identify the requirements for testing
- Create a test plan and test cases
- Execute tests according to the test plan
- Analyze the results and report any issues found
- Fix the issues and retest
- Repeat until all the issues are fixed and the system is ready for deployment.

III. Test Plans

We need to test our backend and front end using unit testing, integration testing, and testing our front end of our software using integration testing and system testing to check if our software works together cohesively and without issue. We will be combining our functional and user acceptance tests; in order for our group to perform our user acceptance tests we will only be able to give them fully working software.

III.1. Unit Testing

Unit testing will be up to the discretion of the team to decide who will be writing them. Ideally, we would have everyone unit test the code they wrote, however, it may be more efficient to have one person go along and work by branching off of code branches and writing unit tests whenever a behavior/method is finished. Overall, the person assigned to unit testing will be up to the team's discretion.

III.2. Integration Testing

The overall structure of our integration testing may be structured at the discretion of the team. This testing will occur once we begin trying to put both front and backends together. All members involved with code that is attempting to be integrated together will need to work together and meet to decide if the code has been integrated and works as all members involved intended.

III.3. System Testing

The system test will be the quickest to know if it has passed and all members will be required to perform this test at least once to work together to make sure that the overall software works as intended.

III.3.1. Functional testing:

Each member of the team will go through and check each of our project's initial goals and check to see if they were met and are fully functioning. We will also check to find any bugs that may be obvious and have a high probability of being uncovered by relevant users.

III.3.2. User Acceptance Testing:

Each member of the team will discuss where to source users to test our software in order to find possible bugs or to see if the software works as intended.