First Iteration Demo

Chengqi Dai (cd3046), Yiqian Wang (yw3225), Wenbo Song (ws2505), Zhongkai Sun (zs2341)
Fall 2018 - COMS W4156 Advance Software Engineering - Prof. Gail Kaiser

1. The date and time at which you already completed this demo, and briefly describe any challenges that arose during the demo.

Date: Nov 8

Time: 11:25 A.M.

First challenge is that we need to input a server IP address to take place to local machine's IPv4 address.

Second challenge is we should make sure information is transmitted in the same format within each part of our programs.

2. The specific user stories and conditions of satisfaction that were demonstrated, with an explanation of any changes since your revised proposal.

As we only write user stories for the second and third iterations, we need to add new user stories for the first iteration demo:

(1) As a user, I want to create an account, and record my game history, so that my conditions of satisfaction is determined by if I could login my account the next time and if the game history is correct.

(2) As a user, I want to see who wins the game, so the algorithm to determine who wins the game should be precise, and show the winner's name at the end.

(3) As a user, I want to choose whether to exit or restart the game after one game, so the game can provide exit and restart buttons, and I don't need to log in if I restart the game.

Test Cases for user stories:

(1) Users satisfaction depend on:
   - (a) How to login
   - (b) How to sign up
   - (c) If the server can remember me at the next time
   - (d) If the game history is correctly stored in database

    (e) Precisely determine who wins the game
    (f) Provide the choice of restart or exit after a game

(2) In the test cases, test what our program provide:
    (a) Test if user can type username, password, and server IP address
    (b) For new user, test if they can select sign up when username is not exist before
    (c) For return user, test if they can select login when the username is already in the database
    (d) In the game, check the output of every step, which contains location, and the color of the piece
    (e) When the game ends, check if every step is encoded and if winner name is stored in the mlab database
    (f) By storing each step of the game in a matrix, we can test if the algorithm can correctly determine the winner who has 5 pieces in a line
    (g) Test if game will close by pressing close button, and if chessboard will clean up and restart the game by pressing retry button.

3. A brief discussion of your CI mechanisms, including which technology you used.

In Travis CI, we need to pre-commit file from client to server, and return values back to client side by post-commit.

Pre-commit:

The detail for pre-commit test is inside the file "pre-commit".

Post-commit CI:

The detail for post-commit test is inside the file .travis.yml including language, version, install and requirements.

The post-commit run both python testcase for client and java test cases for server.

client/test.py -- the post-commit file includes multiple boundary conditions and potential faults test.

Travis CI could build our post-commit test and report the job log result.

4. A link to the github repository where your entire codebase resides. Tag the revisions that were shown in the demo.

https://github.com/OliviaWYQ/Gomoku-Desktop

The revisions that were shown in the demo has been tagged as v1.1 in Github repository. We changed the server from the LAN Server to the Amazon Cloud EC2 Server so that we do not have to change server IPv4 address each time.