CS 1530 – SPRINT 1 DELIVERABLE

— https://github.com/drb56/CS1530 —

David Bickford (drb56@pitt.edu)

Craig Kodman (cmk126@pitt.edu)

Joe Meszar (jwm54@pitt.edu)

David Tsui (dat83@pitt.edu)

DUE DATE

9/29/2016

SCRUM MASTER

Craig Kodman

USER STORIES

1) AS A **USER**, I WANT **A PROGRAM TO PLAY CHESS**, SO THAT I CAN **PLAY CHESS ON A COMPUTER**.

2) AS A **USER**, I WANT **THE PROGRAM NAME TO BE "*LABOON CHESS*"**, SO THAT **THE CHESS PROGRAM NAME IS UNIQUE**.

3) AS A **USER**, I WANT A **CHESSBOARD**, SO THAT I CAN **PLAY CHESS**.

4) AS A **USER**, I WANT **CHESS PIECES**, SO THAT I CAN **PLAY CHESS**.

5) AS A **USER**, I WANT **THE CHESS PIECES TO BE REPRESENTED AS THE FIRST LETTER OF THEIR NAME**, SO THAT I CAN **TELL WHICH PIECE IT IS**.

6) AS A **USER**, I WANT TO MOVE MY **CHESS PIECES**, SO THAT I CAN **PLAY CHESS**.

7) AS A **USER**, I WANT TO **ABIDE BY THE UNITED STATES CHESS FEDERATION (USCF) RULES**, SO THAT I CAN **PLAY A REGULATION GAME OF CHESS**.

8) AS A **USER**, I WANT TO **BE WARNED OF ILLEGAL MOVES**, SO THAT I CAN **MAKE ONLY LEGAL USCF MOVES**.

9) AS A **USER**, I WANT TO **HAVE A GRAPHICAL INTERFACE**, SO THAT I CAN **PLAY WITH A MOUSE**.

10) AS A **USER**, I WANT TO **SEE A 2-D OVERHEAD VIEW OF THE CHESS BOARD**, SO THAT I CAN **SEE WHERE ALL OF THE CHESS PIECES ARE PLACED**.

11) AS A USER, I WANT THE **USCF LETTERS AND NUMBERS NEXT TO THE 2-D CHESS BOARD**, SO THAT I CAN **REPRESENT EACH CHESSBOARD POSITION AS A NUMBER-LETTER COMBINATION**.

12) AS A **USER**, I WANT TO **HAVE BLACK AND WHITE CHESS PIECES**, SO THAT I CAN **PLAY CHESS**.

13) AS A **USER**, I WANT TO **HAVE THE BLACK CHESS PIECES ON TOP AND WHITE PIECES ON BOTTOM**, SO THAT **THE SIDES ARE ALWAYS THE SAME**.

14) AS A **USER**, I WANT TO **BE ABLE TO CHOOSE MY TEAM COLOR**, SO THAT I CAN **PLAY AS EITHER BLACK OR WHITE**.

15) AS A **USER**, I WANT TO **MOVE CHESS PIECES USING A MOUSE CLICK**, SO THAT I CAN **PLAY CHESS INTERACTIVELY**.

16) AS A **USER**, I WANT TO **SAVE MY CHESS GAME IN ITS CURRENT STATE**, SO THAT I CAN **RESUME THE GAME AT A LATER TIME**.

17) AS A **USER**, I WANT TO **LOAD A SAVED CHESS GAME**, SO THAT I CAN **RESUME A PREVIOUSLY-PLAYED CHESS GAME**.

18) AS A **USER**, I WANT THE **SAVE-GAME ACTION TO BE A BUTTON**, SO THAT I CAN **CLICK THE BUTTON AND SAVE THE CURRENT GAME**.

19) AS A **USER**, I WANT THE **LOAD-GAME ACTION TO BE A BUTTON**, SO THAT I CAN **CLICK THE BUTTON AND LOAD A SAVED GAME**.

20) AS A **USER**, I WANT TO **BE ABLE TO UNDO A CHESS MOVE**, SO THAT I CAN **TRY A DIFFERENT CHESS MOVE**.

21) AS A **USER**, I WANT TO **HAVE A TIMER COUNTING HOW LONG IT HAS TAKEN FOR ME TO MAKE A MOVE**, SO THAT I CAN **KEEP COUNT OF THE GAME LENGTH**.

22) AS A **USER**, I WANT THE **OPTION TO SET A PER-TURN TIMER**, SO THAT **A CHESS TURN WILL BE FORFEITED AFTER THE TIME IS UP**.

23) AS A **USER**, I WANT TO **BE ABLE TO PLAY AGAINST A COMPUTER**, SO THAT I **DO NOT NEED TO HAVE ANOTHER PERSON TO PLAY WITH**.

24) AS A **USER**, I WANT TO **SET THE COMPUTER'S *ELO* DIFFICULTY LEVEL**, SO THAT I CAN **PLAY AN EASY, MEDIUM, OR HARD GAME**.

25) AS A **USER**, I WANT **SIMPLE INSTRUCTIONS ON HOW TO USE THE PROGRAM**, SO THAT **A GAME OF CHESS CAN SUCCESSFULLY BE PLAYED**.

26) AS A **USER**, I WANT **A NOTIFICATION TO LET ME KNOW IT IS MY TURN**, SO THAT I **KNOW WHEN TO MAKE A CHESS MOVE**.

27) AS A **USER**, I WANT A **POP-UP SAYING "GAME OVER" WHEN THE CHESS GAME ENDS**, SO THAT I **KNOW WHEN THE CHESS GAME HAS ENDED**.

Test Plan:

1. Test that each chess piece can move.

2. Test that the system can recognize when a move is legal according to USCF rules.

3. Test that the system can recognize when a move is illegal according to USCF rules.

4. Test that when the save button is pressed, it saves the current game in progress.

5. Test that when the load button is pressed, it loads a game from a PGN save-game list.

6. Test that when the exit button is pressed, it exits the program.

7. Test that when the undo button is pressed (from Edit -> undo), it correctly undoes a move.

8. Test that the user color is black, when the user chooses black.

9. Test that the user color is white, when the user chooses white.

10. Test that the "game timer" correctly counts down the time since the game was started.

11. Test that when the "game timer" has elapsed, the current game is forfeited.

12. Test that the “per turn timer” is correctly set at the beginning of each turn.

13. Test that when the "per turn timer" has elapsed, the current move is forfeited.

14. Test that the "player turn" notification correctly notifies the player of their turn.

15. Test that a player cannot make more than one move at a time.

16. Test that the title of the game is "Laboon Chess".

For our project we are going to use a method called TDD, or Test Driven Development. For TDD we will write our tests before we write the actual code for the project. This will allow us to ensure that all tests will pass, and that the code is testable code.

Branching/Integration Plan:

During each sprint our group will work in two teams. Teams will change from sprint to sprint, and equally divide up the work between the two groups. Once a team has finished their work and is ready to push to Git master, the other group will first review their work to check for errors. Finalizing the work, the reviewing team will "+1" their Git work to show that it's progress is to be considered complete. This ensures we have multiple sets of eyes on the code so we have a better opportunity to catch any errors.

Branching names: feature/"name of feature"

Who can integrate branches into master: Anyone, as long as the other group accepts the branch.

Decision descriptions

We interviewed our customer to learn what kind of product they wanted us to build. We asked various questions about what could possibly be done to help the customer to give us a full picture of the end product. During the interview, the customer prioritized some of the requirements. From that list of requirements, we wrote user stories and ordered them according to the importance given to us by the customer. We had a high priority list of items, followed by items the customer wanted but didn't consider high priority, and then followed with the rest of the requirements that the customer would like but would not be absolutely necessary to make them happy.

Based on our user stories (derived from our interactions with our customer), we created our paper prototype. We thought through how the start screen would look, and what functionality it would have. That led to some buttons that would lead to other screens to allow us to meet the customer's requirements. After reviewing our first design, we started over with a better design.

We did research and found two options for building java GUI's - JavaFX and Swing. We ended up choosing JavaFX because it was a newer technology, and it appears that JavaFX is becoming more popular in the development community.

We had discussions on how we were going to develop our software. There was a disagreement between writing the code first or writing the tests first. We ended up with a healthy agreement on using TDD (test driven development). Our standups were productive. Our communication throughout via slack, text, google hangouts and in person has been without issue. We will have

far more to say once we start writing and pushing code regarding our process. So far, our process has been without issue.