Software Requirements Analysis Document

Document Number: 02

Date: Friday, July the 6th, 2020

Yahya, Odai, Heinrich, Alec - (group 32)

1. INTRODUCTION

The basic step of design is the requirement analysis part. In this part, the below objective has to be described. The report includes detailed definition of problems and further analysis on functional and technical requirements. Report also includes system models and management plans.

1.1 Purpose

Project purpose is developing a chess game that could be played against one another as well as against AI. The game is meant to be playable throughout a graphical user interface and on the Terminal. The development will be divided into three iterations, and a deliverable product shall exist at the end of each iteration.

1.2 Scope

• Terminal interface

• 2D GUI

• Undo and redo moves

• Cemetery

• Implementation of a basic AI and advanced AI

• Prevention of invalid moves

• Chess clock

• Support for a bilingual interface

• Resizable GUI

1.3 Goals Of The Team

• Learning how to manage project.

• Using documentation efficiently.

• Using time efficiently.

• Learning teamwork.

• Produce a satisfactory software.

1.4 Process Model

• The team is going to use an agile software development approach and namely scrum

• Roles are divided as follows:

    • Scrum Product Owner: Hannes Kallwies

    • Scrum Master/Tester: Yahya Alnajjar

    • Developer/Maintainer: Alec Winter, Heinrich Stremme, Odai Alsharif

1.5 Team Organization

Since every member has nearly the same experience about design patterns and concepts of this project, decisions about the project are going to be made by group consensus. This situation yields our team to have democratic decentralized. Communications within the group is going to be horizontal.

2 RESEARCH

2.1 Graphics

Aside from the terminal based capability, our game will also include a 2D modeling game view (possibly a future 3D implementation as well) and for that reason nearly the most important part of is graphics section. Since it has been requested by our customer not to use graphic library we are going to code it using JavaFx.

• JavaFx: is a software platform for creating and delivering desktop applications, as well as rich internet applications (RIAs) that can run across a wide variety of devices. Its also the standard GUI library for Java SE that has support for desktop computers and web browsers on Windows, Linux and macOS.

2.2 Sound

Because of importance of sound, we think to use a sound library which is free license from www.Freesound.org

2.3 AI

We are going to use a basic AI implementation for our game when playing against the computer.

In games, AI can be divided into two categories: deterministic and non deterministic AI. In deterministic AI there is no suspicion. Deterministic behavior is specified and predictable. Deterministic AI techniques are the easiest part of AI. They are fast, easy to implement, test, debug predictable. Deterministic AI does not make easy learning and evolving. Also it is predictable after a little game play. Nondeterministic AI is related about learning and unpredictability. Since we plan using a primitive AI, it will be deterministic.

3 DESCRIPTION

3.1 Game Modes

The properties of game modes are listed below:

3.1.1 Singleplayer

• Standard: Standard game mode is time based, in which every move should be done in a small period of time and the duration of the entire game is equal to 15 minutes.

• Unlimited: clocks are not used so basically both the increment and start time is zero. The opponents can consume as much time as they want to play the move.

• Puzzle: Usually the goal is to find the single best, ideally aesthetic move or a series of single best moves in a chess position, which was created from a real game.

3.1.2 Multiplayer

• Standard: Standard game modes is time based in which every move should be done in a small period of time and the duration of the entire game is equal to 15 minutes.

• Unlimited: clocks are not used so basically both the increment and start time is zero. The opponents can consume as much time as they want to play the move.

4 REQUIREMENTS

4.1 Functional Requirements

4.1.1 Menu Requirements

4.1.1.1 Start Menu

• New game: This menu item will let the player choose a game mode.

• Load game: This menu item will let the player load a saved game and continue where it's left off.

• Settings: This menu item will allow the player to change the language layout and resize the window.

• Exit: This menu item will close the game and send player to operating system window.

4.1.1.2 In Game Menu

•Pause: This choice will pause game play.

•Resume: This choice will return game play.

•Save: This choice will save the current game state.

•Exit: This choice will end the current game and exist to the previous menu.

4.1.2 Game Functions

4.1.2.1 Movement

•**Pawn:** Pawns shall move one space forward, optionally two spaces forward on their opening move.

•**Rook:** Rooks shall move vertically or horizontally any number of spaces unless impeded by another piec.

•**Knight:** Knights shall move two spaces either vertically or horizontally followed by one space perpendicularly.

•**Bishop:** Bishops shall move diagonally any number of spaces unless impeded by another piece.

•**Queen:** Queens shall move vertically, horizontally, or diagonally any number of spaces unless impeded by another piece.

•**King:** Kings shall move one space in any direction.

•**Castling:** When requirements are met for castling, kings may move two spaces towards a rook, with the rook moving onto the space crossed over by the king.

4.1.2.2 Capturing

•**General Capture:** If a piece other than a pawn, moving in its normal fashion, may move into a square occupied by an opposing piece, the friendly piece may capture the opposing piece.

•**Pawn:** Pawns shall capture by moving forward one space diagonally into an opposing piece.

•**En Passant:** When requirements are met for en passant capture, a pawn may capture as above into aspace crossed, but no longer occupied by an opposing piece.

4.1.2.2 Promotion

•**Promotion:** A pawn, having entered the rank opposite where it started, shall be promoted to a piece of its controller’s choosing.

4.2 Performance Requirements

• Artificial Intelligence has to make decisions quick enough for fluent game

playability.

4.3 Software Requirements

• Any operating system with Java support or a Java VM installed.

4.4 Non Functional Requirements

4.4.1 Code Quality

•Use of PMD code analysis plugin tool and JaCoCo plugin on the project's source code to generate a site report on coding style,metrics and tests coverage.

•Use of meaningful comments to ease on the understanding of the written code.

4.4.2 Testing

• JUnit Test coverage of 90% from the total written code (without the GUI classes)

4.4.3 Documentations

• Regular update of the requirements documentations via Story-Cards, Use Case Diagram.
• Regular update of the architecture documentations via UML diagrams.
• Regular update of the program usage documentations.

4.5 Project Constraints

4.5.1 Time Constraints

• Deadline of the first iteration is May 11th. 2020. which is a terminal based game model that allows a basic game play against another player with no invalid moves.

•Deadline of the second iteration is June 8th. 2020. which is a 2D game model that implements a basic AI to play against in single player mode.

•Deadline of the last iteration is July 6th. 2020. that includes implementation of further functionalities, improvements and adjustment to the overall game.

4.5.2 Language Constraints

•The language of the project will be Java, version 11 and JavaFx as requested by the customer.

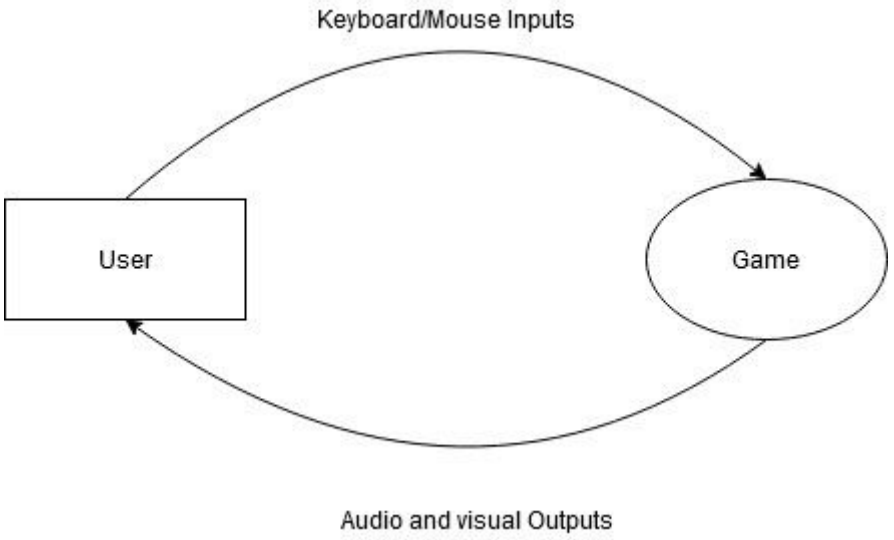
4.5.3 UI Constraints

•The UI has to be simple and easy to understand. The menus and GUI elements will be in the interface.

5 SYSTEM ANALYSIS AND MODELING

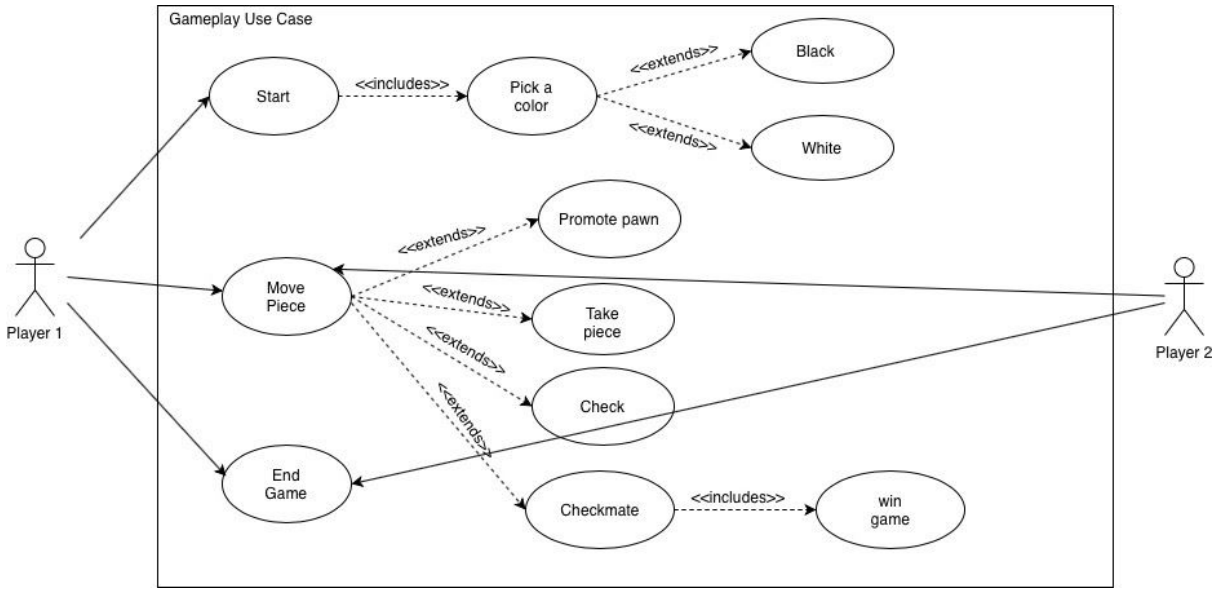
5.1 Functional Modeling

•Data Flow Diagram: shows the interactions between the user and the game. User initiates control inputs and the game responds with visual and audio outputs.

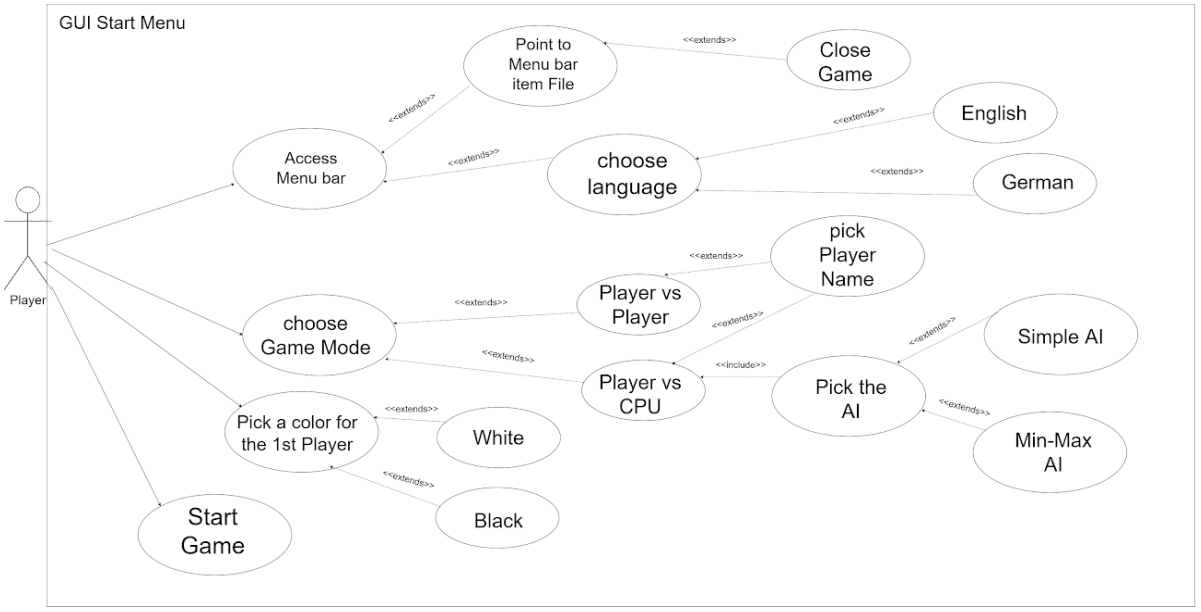


5.2 Use Case Analysis

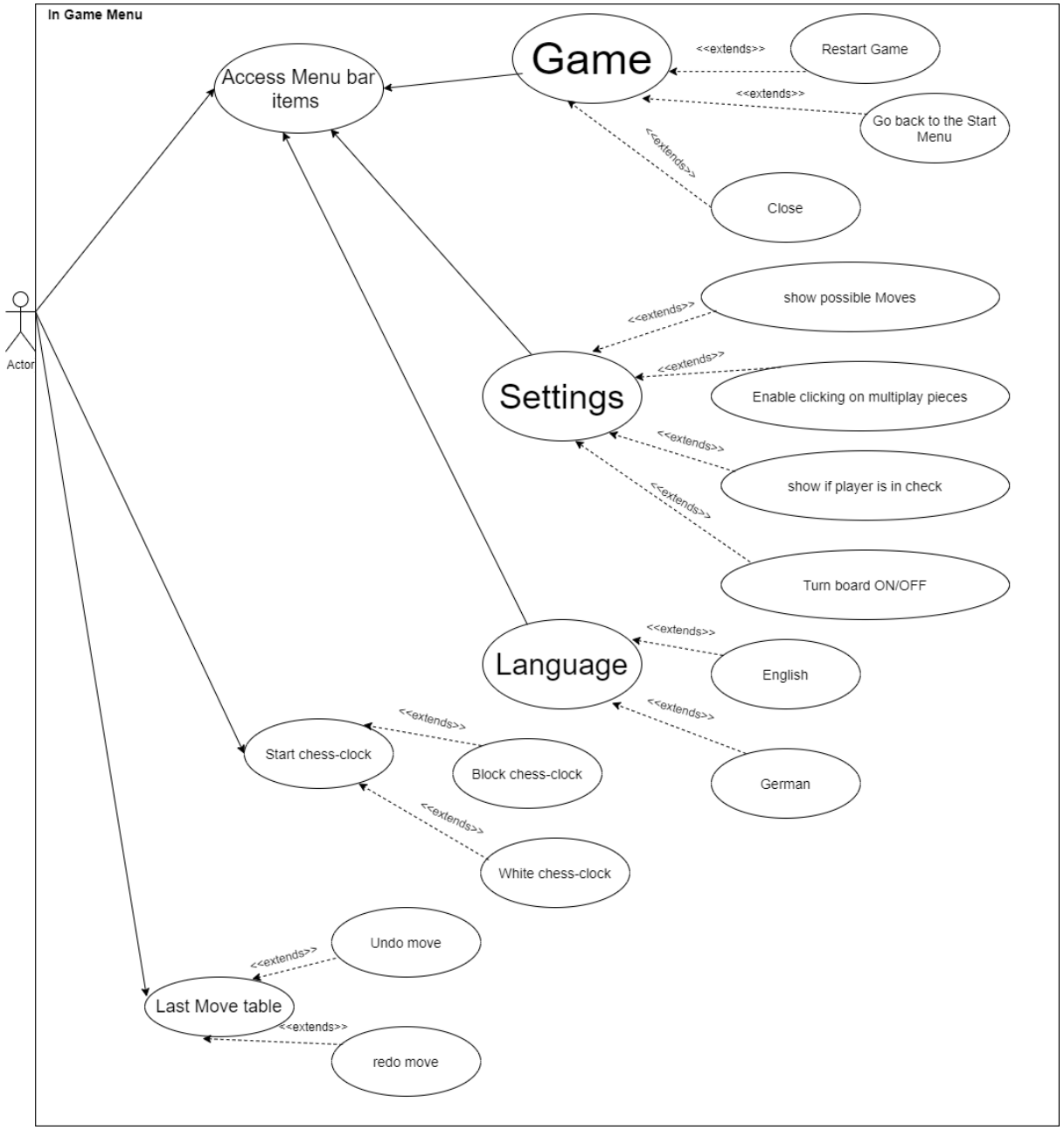
5.2.1 Gameplay Use Case



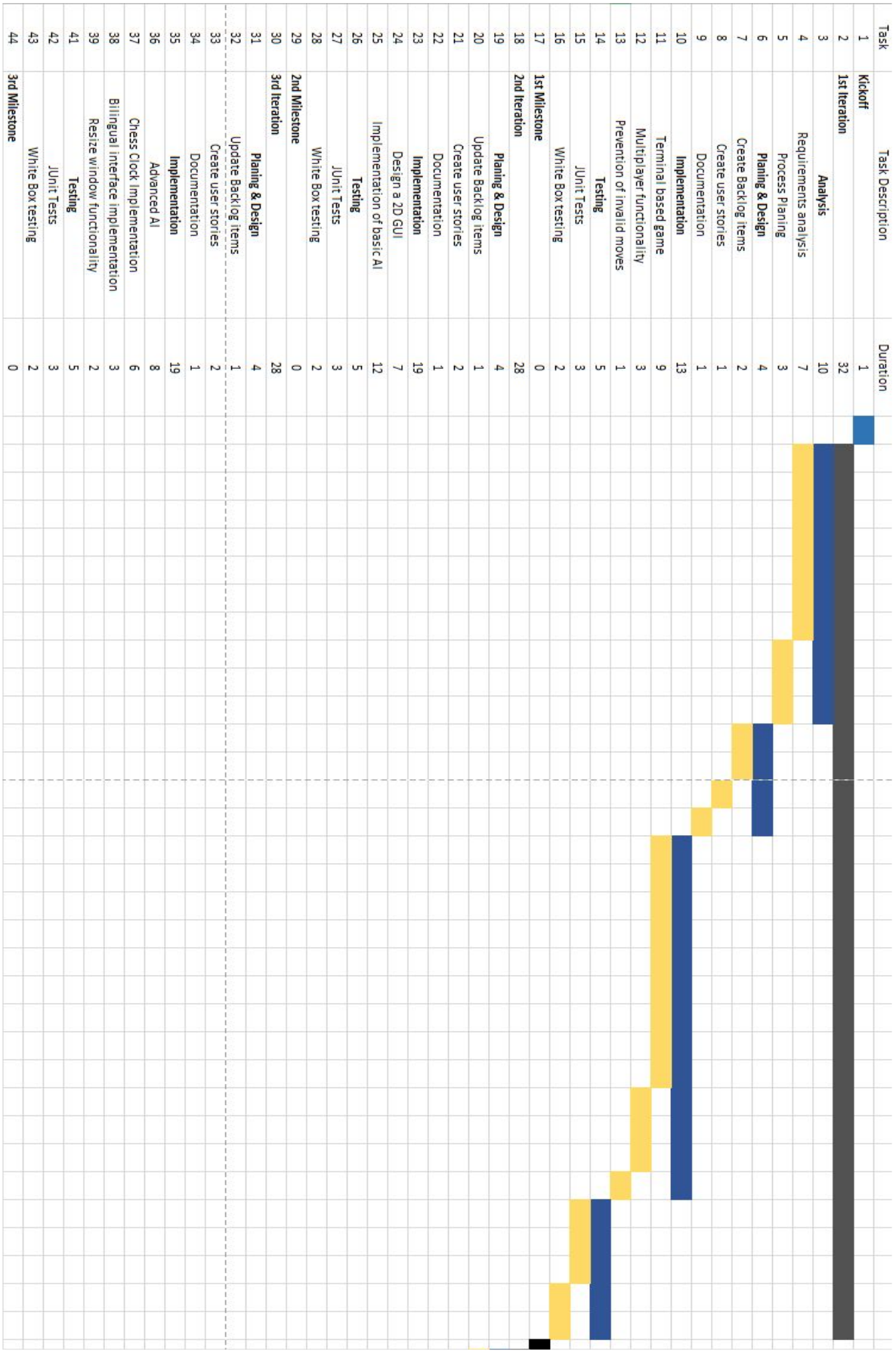
5.2.2 GUI Start Menu Use Case



5.2.3 GUI In Game Menu Use Case



6 GANTT CHART



Task	Task Description	Duration
1	**Kickoff**	1
2	**1st Iteration**	32
3	**Analysis**	10
4	Requirements analysis	7
5	Process Planing	3
6	**Planing & Design**	4
7	Create Backlog items	2
8	Create user stories	1
9	Documentation	1
10	**Implementation**	13
11	Terminal based game	9
12	Multiplayer functionality	3
13	Prevention of invalid moves	1
14	**Testing**	5
15	JUnit Tests	3
16	White Box testing	2
17	**1st Milestone**	0
18	**2nd Iteration**	28
19	**Planing & Design**	4
20	Update Backlog items	1
21	Create user stories	2
22	Documentation	1
23	**Implementation**	19
24	Design a 2D GUI	7
25	Implementation of basic AI	12
26	**Testing**	5
27	JUnit Tests	3
28	White Box testing	2
29	**2nd Milestone**	0
30	**3rd Iteration**	28
31	**Planing & Design**	4
32	Update Backlog items	1
33	Create user stories	2
34	Documentation	1
35	**Implementation**	19
36	Advanced AI	8
37	Chess Clock implementation	6
38	Bilingual interface implementation	3
39	Resize window functionality	2
41	**Testing**	5
42	JUnit Tests	3
43	White Box testing	2
44	**3rd Milestone**	0