ASSESSMENT 2: BUILDING IT SYSTEMS

DietPepsiMaxUltraCreamingSoda

The Team

Caleb Tagliaferri	S3521944
Vinh D Tran	S3500659
Joshua Coppen	S3775648
Manh Cuong Tu	S3743565
Paul Stubbs	S3609575
Liam Chan-Wicht	S3803217

Contents

Project Description

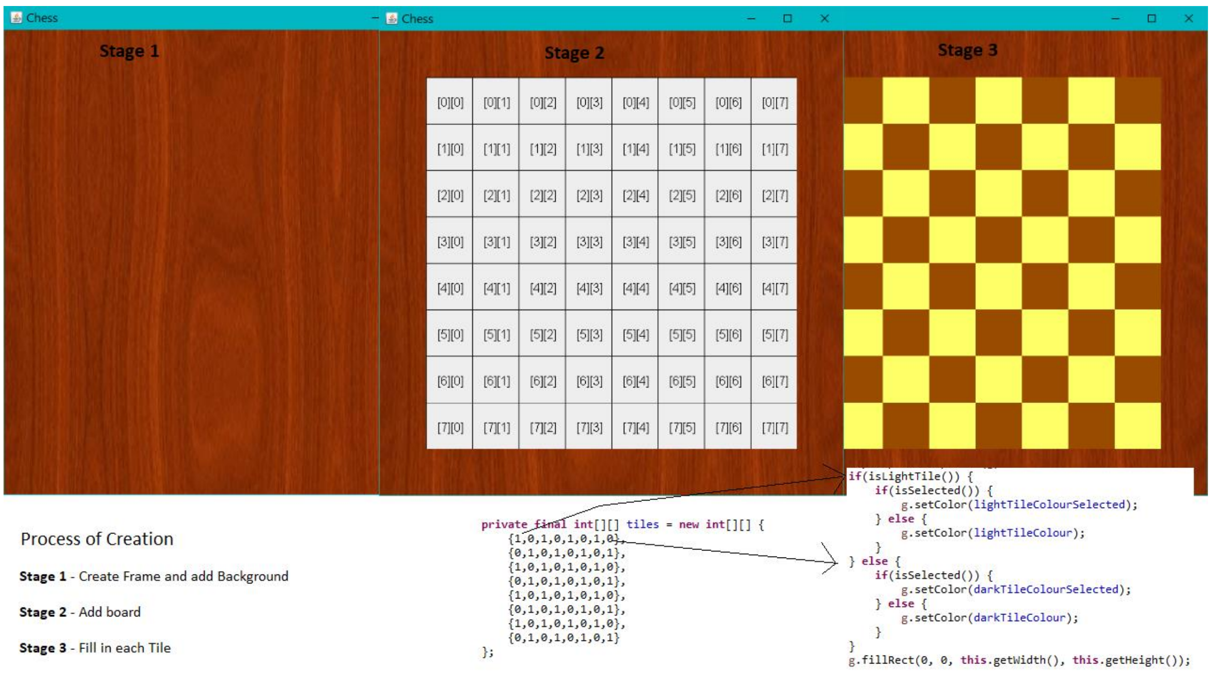
Our project is to create a standard 2D chess game with the Western standards which uses Staunton chess set on a 64-tiled board (8 x 8).

After completion of the standard chess game, we will add a move timer to increase the competitive experience of the game. We will also increase the customizability of the game by allowing players to decide on a change board formation and allow them to spend an allocated amount of coins to create their unique chess set and format them on the board.

Core Features

Core Feature 1 - Board Creation

Author: Caleb Tagliaferri **Create Date:** 17/10/2019



Process of Creation

Stage 1 - Create Frame and add Background

Stage 2 - Add board

Stage 3 - Fill in each Tile

```java
private final int[][] tiles = new int[][] {
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1},
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1},
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1},
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1}
};
```

```java
if(isLightTile()) {
    if(isSelected()) {
        g.setColor(lightTileColourSelected);
    } else {
        g.setColor(lightTileColour);
    }
} else {
    if(isSelected()) {
        g.setColor(darkTileColourSelected);
    } else {
        g.setColor(darkTileColour);
    }
}
g.fillRect(0, 0, this.getWidth(), this.getHeight());
```

Core Feature 1 Validation Testing:

- Background is displayed
- 8x8 Grid of tiles is displayed on Background
- Each tile is painted correctly on the Grid with correct dimensions
- Clicking a tile will highlight it

Core Feature 2 – Piece Modelling

Author: Joshua Coppen **Create Date:** 29/09/2019

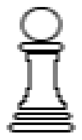


Figure 1

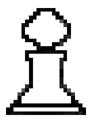


Figure 2

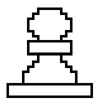


Figure 3

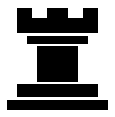


Figure 4

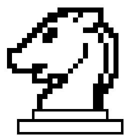


Figure 5

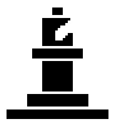


Figure 6

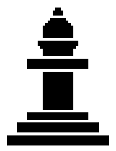


Figure 7

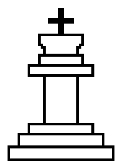


Figure 8

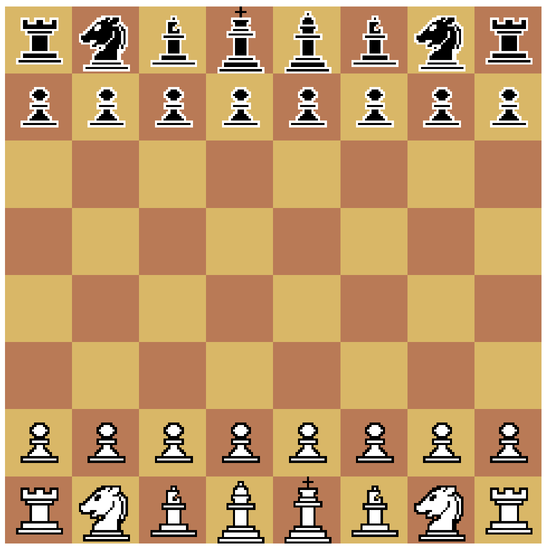


Figure 9

The piece modelling minimum viable feature involved creating all the art assets which represents all the different chess pieces. Figures 1 and 2 are the first two attempts made to create a pawn chess piece. Figures 3 - 8 show the finished piece models for the chess pieces and figure 9 shows a visual representation of what the pieces on the chessboard will look like.

Core Feature 2 Validation Testing:

- All pieces have the correct dimensions in proportion to the chessboard.
- Each piece represents the correct piece of code (king image is added to King class).
- All pieces can be shown on screen through running the code.
- All pieces are shown in correct locations on the chess board according to the standard chess game.

Core Feature 3 - Piece Movements

Author: Vinh D Tran **Create Date:** 16/10/2019

8
7
6
5
4
3
2
1
a
b
c
d
e
f
g
h

(Illustration only - not a true representation of the final product)

This core feature allows the player to move their chess pieces across the board in accordance with the rules of chess. Only legal moves will be allowed, and all possible movements may be highlighted when a piece is selected (The tile will change colour if the move is legal - this feature may be implemented in one of the extended features).

For a more complete guide on the rules of chess and piece movement, please refer to the below guide.

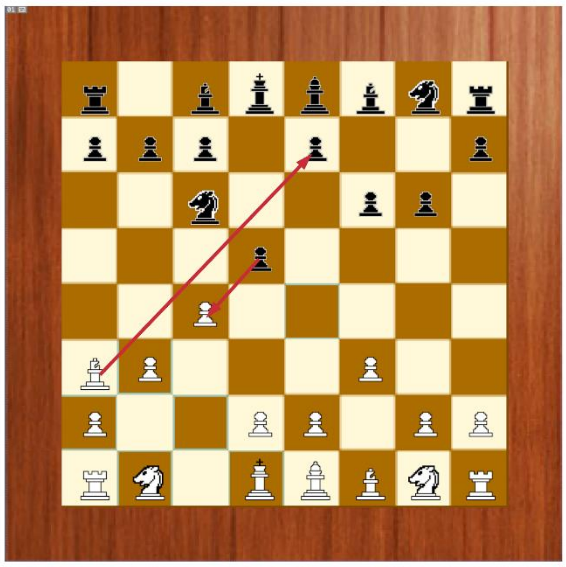
Rules of Chess.

Core Feature 3 Validation Testing:

- Piece movements will be tested thoroughly once coding of the project is completed or near completion.
- Piece movements and collision with allied and non-allied piece will also be checked.
- Special movements such as Rook castling and en passant may also be implemented as one of the extra features if possible and will also be tested.
- Only legal moves according to standard chess rules will be allowed.

Core Feature 4 – Attack Move

Author: Manh Cuong Tu **Create Date:** 30/09/2019



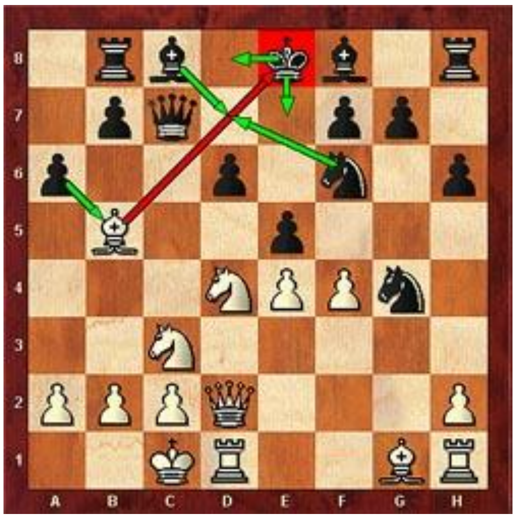
This core feature allows the player to eliminate enemy pieces. Overall structure of attacking builds upon piece movement. Once enemy piece has taken out of play after a successful attack, it will remain gone for the rest of the game.

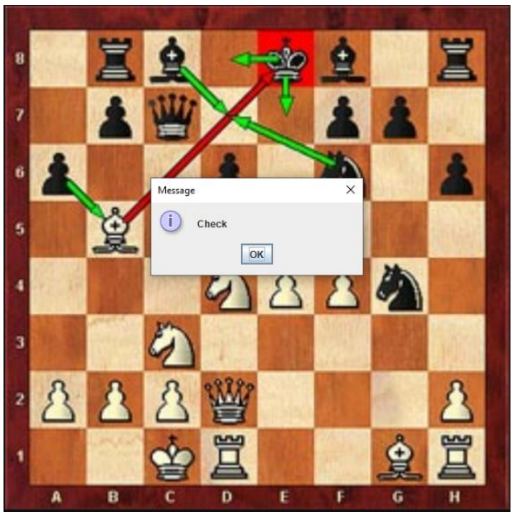
Core Feature 4 Validation Testing:

- Attack Move will overlap with Piece Movement. When there is a piece blocking the final destination of move; If it's an enemy piece, it is destroyed and replaced. If Ally piece, Piece movement validation will be in effect and move will be invalid.
- With the exception of 'Pawn' Attack Move. Can only move forward and attack diagonally, all else considered illegal.

Core Feature 5 - Check/Checkmate

Author: Paul Stubbs **Create Date:** 27/10/2019





Check

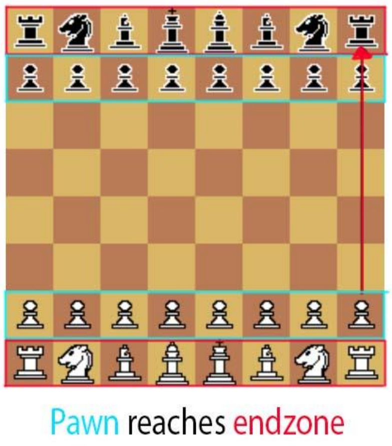
This core feature is critical to the correct function of the game, this feature will check to see if the last move places the “king” at risk and will call the check or check-mate feature.

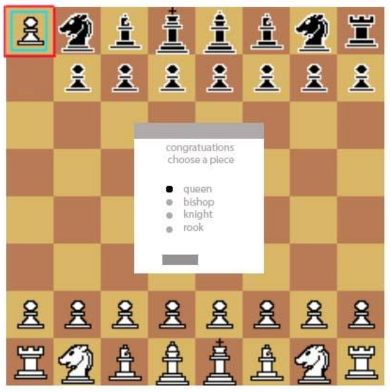
Core Feature 5 Validation Testing:

- King needs a check run on it at end of each turn to ensure that it is not threatened (check/checkmate)
- When check is initialized a message will be displayed saying check or checkmate.
- When player is in check, they can only do moves that take them out of check.
- When player is in checkmate the game will auto defeat the player.

Core Feature 6 – Pawn Evolution

Author: Liam Chan-Wicht **Create Date:** 22/10/2019





(Illustration only - not a true representation of the final product)

This core feature allows players to 'evolve' their Pawn piece (teal) to another piece by reaching the opponents' end of the board (red).

Upon reaching the end of the board, a dialog box will appear asking the player to choose another piece to 'evolve' into. These choices will be limited to Queen, Bishop, Knight and Rook. You will not be allowed to change your piece into a King or another Pawn.

Core Feature 6 Validation Testing:

- Pawn evolution will be tested thoroughly once coding of the project is completed or near completion.
- Tile, board, piece, piece movement and pawn classes will be cross referenced to ensure correct validations are conducted throughout.
- User dialog box will be simple and impossible to misinterpret - allowing players to know immediately what and how actions are happening.
- Only legal moves according to standard chess rules will be allowed.

Extended Features

Extended Feature 1 - Additional Pieces

Author: Josh Coppen **Create Date:** 18/10/2019

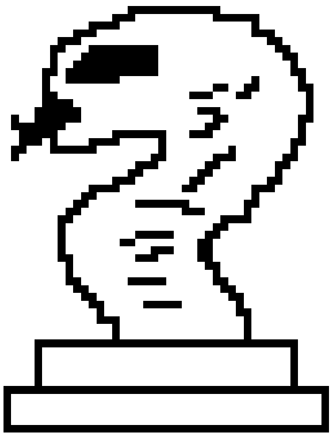


Figure 1

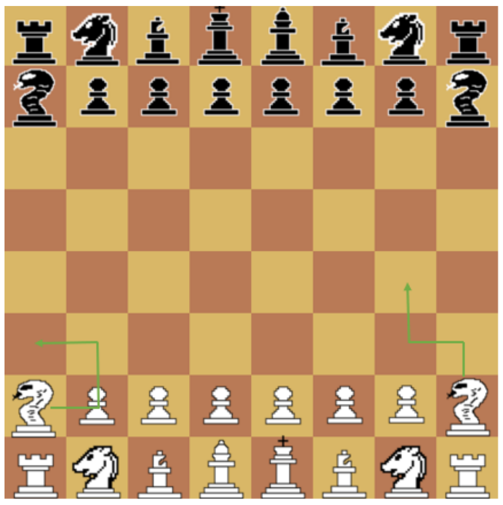


Figure 2

In addition to the regular chess game we want to modify the game in a few different ways, and one of those ways is to experiment with additional pieces. These pieces will look and behave differently to any of the chess pieces in the standard game.

One of the ideas brought up thus far is a snake like piece which will slither around the board in many different directions. As the project develops, we will have a greater idea of what these pieces look like.

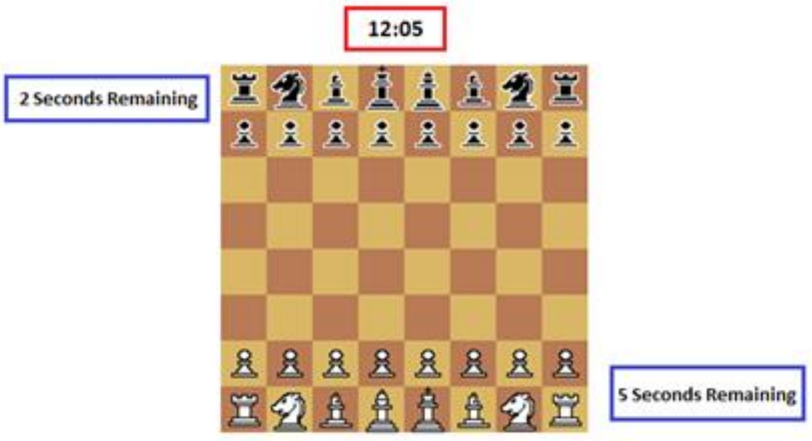
Figure 1 shows a very rough sketch of what this snake piece might look like. Figure 2 shows a visual representation of what the chess board might look like with these additional pieces added to it, as well as showing the snake pieces potential move set.

Extended Feature 1 Validation Testing:

- Additional pieces to have the correct dimensions in proportion to the chessboard.
- Additional pieces can be shown on screen through running the code.
- Additional pieces fit in with the standard pieces of the game. We do not want these pieces to be overpowered but to feel like a natural addition to the game of chess

Extended Feature 2 – Move Timer

Author: Manh Cuong Tu **Create Date:** 28/10/2019



(Illustration only – possible implementation of the final product)

This core feature allows players to keep track of how long the game has been in progress since the first move. Timer (Red) will not stop until the game is over and is only reset on the beginning of a new game.

Timer highlighted in blue is an extra feature that can represent how much time left the player must make a move or count ascendingly to keep track how much time the player used to make a move.

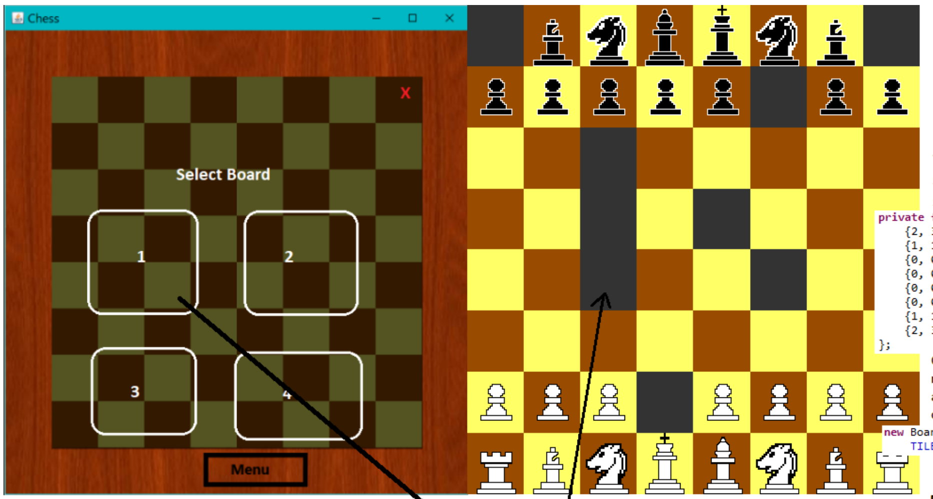
Extended Feature 2 Validation Testing:

Game Timer (RedBox) will always initialise once the first move has been made, marking game commencement, and will continue until one side is defeated or the player concedes.

- Verified within the format (Minutes: Seconds)
- Player Timer (Blue Box) may be implemented as an extra game feature. Forcing players to act within a limited amount of time, creating a different experience. Must be timed in seconds.

Extended Feature 3 – Additional Board

Author: Caleb Tagliaferri **Create Date:** 20/10/2019



Board menu appears after selected in main menu

```
private final int[][] tiles = new int[][] {
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1},
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1},
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1},
    {1,0,1,0,1,0,1,0},
    {0,1,0,1,0,1,0,1}
};
```

Each board is a configured version of this array, each number defines the type of each tile

Tile Key

0 - Dark tile
1 - Light tile
2 - Invalid tile

```
private final int [][] board2 = new int[][] {
    {2,0,1,0,1,0,1,2},
    {0,1,0,1,0,2,0,1},
    {1,0,2,0,1,0,1,0},
    {0,1,2,1,2,1,0,1},
    {1,0,2,0,1,2,1,0},
    {0,1,0,1,0,1,0,1},
    {1,0,1,2,1,0,1,0},
    {0,1,0,1,0,1,0,1}
};
```

This is an additional board, notice chess pieces don't spawn on invalid tiles

Different Setups

In these examples, the setups have been standard with the pieces that would be on the **invalid** tiles not spawning. Pieces are configured similar to Tiles

```
private final int[][] pieceNumbers = new int[][] {
    {2, 3, 4, 5, 6, 4, 3, 2},
    {1, 1, 1, 1, 1, 1, 1, 1},
    {0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0},
    {1, 1, 1, 1, 1, 1, 1, 1},
    {2, 3, 4, 6, 5, 4, 3, 2}
};
```

Creating a new board & piece setup just requires you to adjust or create a new array and pass it through the game constructor as a parameter

```
new Board(TILES_X, TILES_Y, TILE_WIDTH,
    TILE_HEIGHT, board2, pieceNumbers, this);
```

Piece Key

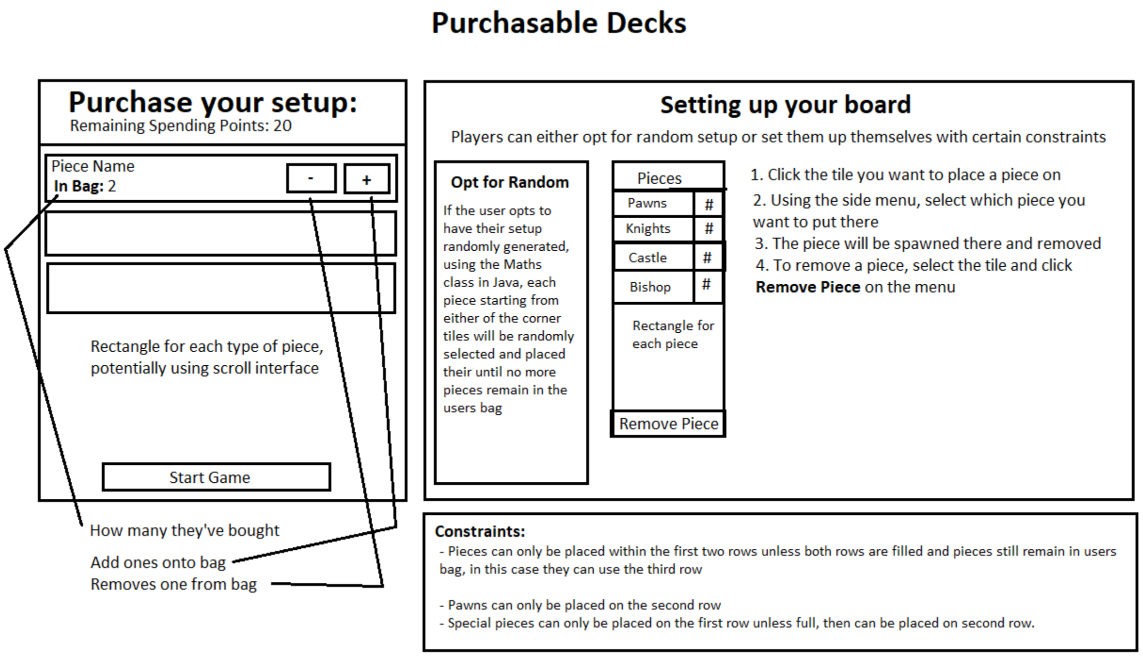
0 - No piece
1 - Pawn
2 - Castle
3 - Bishop
4 - Knight
5 - Queen
6 - King
7 - Snake (Additional)

Extended Feature 3 Validation Testing:

- **Select Board** interface appears correctly
- Selecting a board restarts the game and loads in correct configurations
- **Pieces** cannot be moved onto Invalid tiles
- Each tile is either Brown, Creamy or Grey
- **Pieces** don't spawn on invalid pieces
- One **King** must be on each team
- Pawns can move diagonal if blocked by invalid tile

Extended Feature 4 – Purchase Decks

Author: Caleb Tagliaferri **Create Date:** 20/10/2019



Rectangle for each type of piece, potentially using scroll interface

Players can either opt for random setup or set them up themselves with certain constraints

If the user opts to have their setup randomly generated, using the Maths class in Java, each piece starting from either of the corner tiles will be randomly selected and placed their until no more pieces remain in the users bag

1. Click the tile you want to place a piece on
2. Using the side menu, select which piece you want to put there
3. The piece will be spawned there and removed
4. To remove a piece, select the tile and click **Remove Piece** on the menu

Constraints:

- Pieces can only be placed within the first two rows unless both rows are filled and pieces still remain in users bag, in this case they can use the third row

- Pawns can only be placed on the second row
- Special pieces can only be placed on the first row unless full, then can be placed on second row.

Extended Feature 4 Validation Testing:

- Purchase menu & side menu appear correctly
- All buttons in menus work
- Players cannot spend more than allowed points
- Constraints are implemented during setup of board
- Two kings aren’t buyable, one king required on each team to be bought
- Random generated setup follows constraints
- No pieces can be placed on invalid tiles

Project Estimation

The spreadsheet with all estimated timelines for all minimum viable features as well as extended features with justifications is included below.

Assessment 2 Estimation Spreadsheet

Listing Technologies

Collaborative workspaces

Throughout this project, we will be collaborating via a variety of workspaces. For example:

- Trello
- GitHub
- Office 365

Trello is an online collaboration tool that assists with project management. Through the use of cards and columns, each issue is clearly defined and easily sorted into specific groups. These cards can then be further detailed with a variety of modifiers such as due dates, colour coding, and pictures. In addition, we have also implemented the use of 'individual swim lanes' that have each team member's tasks and when they are due. Alongside these individual swim lanes, each group of cards are clearly labelled to ensure all group members are able to find collaborative workspaces within our board.

GitHub is a cloud-based version-control system that allows developers to collaborate and edit projects through specific actions. These actions such as pushing, pulling and merging, allow each team member to modify features that influence the whole. With this project, we are using GitHub to emulate a professional environment where version-control is an industry standard. With GitHub, it is also simple to assign features to members and have a review process of work done.

Office Suite 365 (Word) is a collaborative word processing platform that allows team members to access and edit the same file without having to transfer and collate individual files from each member. This particular workspace is easy to use as word processing is familiar to almost anyone that has touched a computer. As such, many employers look for experience in the Microsoft Office Suite as it has become such a standard across industries.

Software

- Eclipse IDE
    - Eclipse if an integrated development environment used for developing applications mainly in Java but also supports other languages such as C/C++, Python etc.
    - Eclipse main purpose is to provide the developer with the tools and environment necessary to create great software. Eclipse if customisable via different available plug-ins which will come in handy for us.
    - Eclipse is open-source and free and we'll be using the latest stable version 4.12.
    - Some team members prefer to use IntelliJ as an alternative, see below for more info.

- IntelliJ IDE
    - Like Eclipse, IntelliJ is also an IDE for developing computer software. Developed by JetBrains, IntelliJ comes in both a free and paid edition. o IntelliJ provides the tool and environment which enable the developer to create many software's.
    - IntelliJ has a free and a paid edition. The paid edition is free for students if they sign up using their student email.
    - We will be using the latest stable version .

- Discord
    - A free VoIP communication software designed for and used mainly by gamers. As this software gained popularity, its use is extended to other contexts that requires teamwork and collaboration.
    - We will be using discord as our main communication tool as it enables our team to communicate with each other via text and voice chat room for free.

- Office Suite
    - The Office Suite encompass several software useful for all type of work. For this project, we will mainly be using Word Document for the writing and documentation of our project and OneDrive for sharing and collaboration of the word document.

Tools

- GitHub
    - GitHub is an open-source version control system and collaboration tool which enable developers to work together on the same project without overlapping each other. Developers can add or modify codes collaboratively which is then stored in a central repository.
    - We have chosen to use GitHub for our project as it's a hugely popular and proven collaboration tool. As there are six team members working on the same project, it would be difficult to coordinate our code implementation without some sort of collaboration tool, GitHub meets our need.
    - GitHub is open-source and is free to use, we will be using the latest version available

Resources

- Java Documentation
  - Java doc provides help and documentation of the many API available to Java developers.

- YouTube
  - YouTube is an American video-sharing website headquartered in San Bruno, California. Three former PayPal employees—Chad Hurley, Steve Chen, and Jawed Karim—created the service in February 2005. Google bought the site in November 2006 for US$1.65 billion; YouTube now operates as one of Google's subsidiaries.
  - We will be using YouTube as a resource and guide to help get us started on building our project. Some team members are relatively new to programming in Java so YouTube will be a great resource for everyone.

- Lynda.com
  - Lynda.com also known as LinkedIn Learning is an American website offering video courses taught by industry experts in software, creative, and business skills. It is a subsidiary of LinkedIn. It was founded in 1995 by Lynda Weinman as Lynda.com before being acquired by LinkedIn in 2015. Microsoft acquired LinkedIn in December 2016.
  - Unlike YouTube, Lynda provides short courses created by industry professionals on many subjects including Java programming. Lynda will be a great resource for us to turn to when required.
  - Lynda is usually a paid subscription service but fortunately it’s free for RMIT students