Structured Walkthrough

Classes and use cases will be in **bold.** Use Case will be shortened to **UC**

1. The System will start the **Game** and will constantly be displaying new information on the screens for **Players (UC9)**. **Mainview** will show the Menu to the **Player (UC9)**. **Players** must then connect to the website, enter the **Game** code and select “Play **Game**” to join a **Game**. The **Player** must then wait for all other **Player**s to join before starting the **Game**. Once the Lobby is full, no other **Player** may join.

**Branches to:** **UC**2 (Choose Playing Order), **UC**9 (Update Display)

1. When the **Game** is started, **Player** must decide the playing order. Each **Player** will take one letter from the letter bag. **Server** will check the letter that each **Player** has. **Player** with the closest letter to A will be chosen to make the first move, a blank tile is the most valuable tile. All the letters will now be returned to the Letter bag (Letter bag is an attribute of the **Board**). If two or more **Player**s have the same letter, they must choose a new letter each from the Letter bag. The Letter bag will now be shuffled.

**Branches to**: **UC**3 (Setup **Game**), **UC**9 (Update Display)

1. At this point, the **Game** must be set up (use case 3). **Board** will be empty and all the **Player**s will be in a ready state to receive 7 random letters from the letter bag, after playing order is chosen (**UC2)**. Each **Player** will now receive and examine their letters before making a turn (**UC**4) .

**Branches to**: **UC**4 (make turn), **UC**9 (Update Display)

1. The **Server** works out the playing order (**UC**2), and then the **Player** that has been chosen will be the first to go. Make Turn allows a **Player** to move **Piece**(s) on **Board**. Once a **Player** makes a turn, the **Board** returns to a ready state. So, a **Player** plays **Piece**s and the **Game** validates the combination of **Piece**s (words). The **Player** is then awarded points for the word(s). The **Board** then distributes new **Piece**(s) to the **Player**, so that he has a total of 7. If **Player** cannot form a word, he Skips Turn (Use Case 6). If **Server** cannot validate the word (**UC**10), it means a Wrong Turn has been made (**UC**5). If the **Player** has no **Piece** left, the **Game** is over (**UC**8).

**Branches to**: **UC**5 (Make wrong turn), **UC**6 (skip Turn), **UC**9 (Update Display), **UC**10 (Verify move)

1. The system will be validating every move that a **Player** makes (**UC**10). When **Player** decides to place a **Piece** on **Board**, if **Game** cannot validate the word(s) (**UC**10), then **Player** will be notified and will take back tiles to have a chance for another move(**UC**4).

**Branches to**: **UC**6 (skip Turn), **UC**4 (Make Turn), **UC**9 (Update Display), **UC**10 (Verify move)

1. If **Player** doesn't have any **Piece** that he could use, he has the option to skip his current turn. There are two options for skipping a turn. **Player** may choose to skip a turn to swap out all their **Piece**s for a new set of 7 random **Piece**s from the Letter bag. Option two is that **Player** may also skip a turn without swapping out their **Piece**s, if they choose to do so.

**Branch to**: **UC**5 (Make wrong turn), **UC**6 (skip Turn), **UC**4 (Make Turn), **UC**9 (Update Display), **UC**10 (Verify move)

1. **Player** has the option to leave at any point during a **Game**. Doing so will prevent him from having any interaction with the **Game** he left, and his **Piece**(s) that weren't on the **Board** will be returned to the **Piece** list of **Board**.

**Branch to**: **UC**8 (End **Game**), **UC**9 (Update Display)

1. The **Game** is over and a winner is announced. The **Board** compares the scores of every **Player** and displays the results (**UC**9). If more than one **Player** has equal highest score, the **Game** is a tie. The **Player** has no interaction with the **Board** by this point, he can only observe.

**Branch to**: **UC**9 (Update Display)

1. The **server** will be sending information to all **Client**s so that each **Client**s screens are updated to show new and old information. Information s**UC**h as scores, names, **Piece**s are stored in a cache.

**Branch to: UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8, UC10**

1. When a **Player** places a **Piece** on the **Board** (**UC**4), the **Board**’s state will be changed. When this happens, these new **Piece**s on the **Board** will form a word(s) which will be validated to ensure it’s a real word(s). The main trigger will be **Player** making a turn (**UC**4), when this happens, the **Server** will validate the word and all other words on **Board**. Update display will now be called and displays the new word on **Board** (**UC**9). If word is incorrect (**UC**5), a failure message is displayed (**UC**9) to the **Player**.

**Branch to**: **UC**4, **UC**5, **UC**6, **UC**9