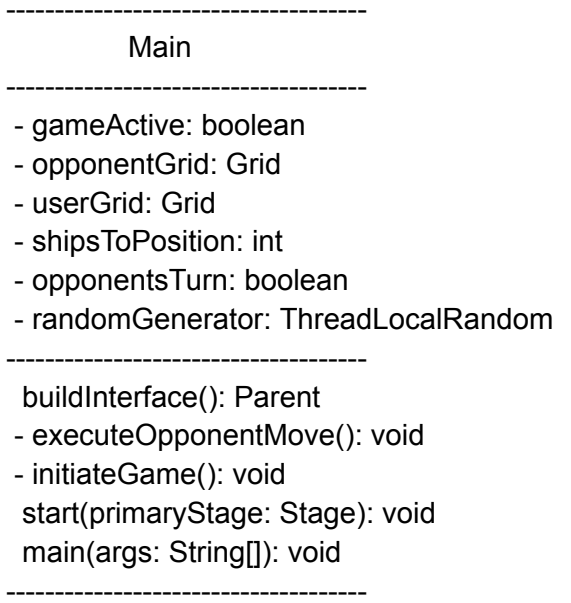
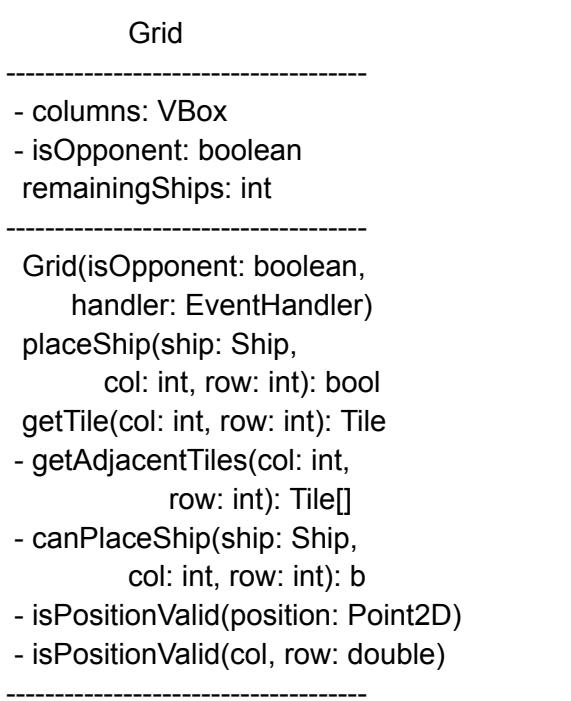
UML's:

Main.java



Grid.java



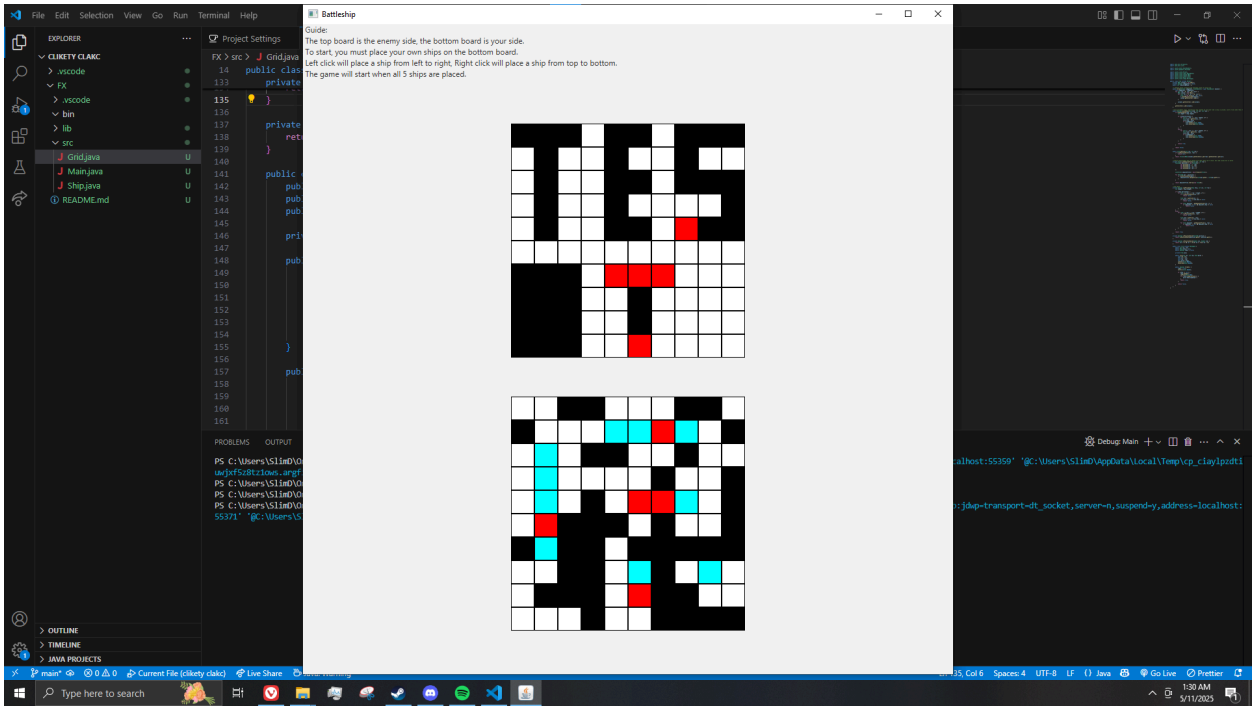
contains

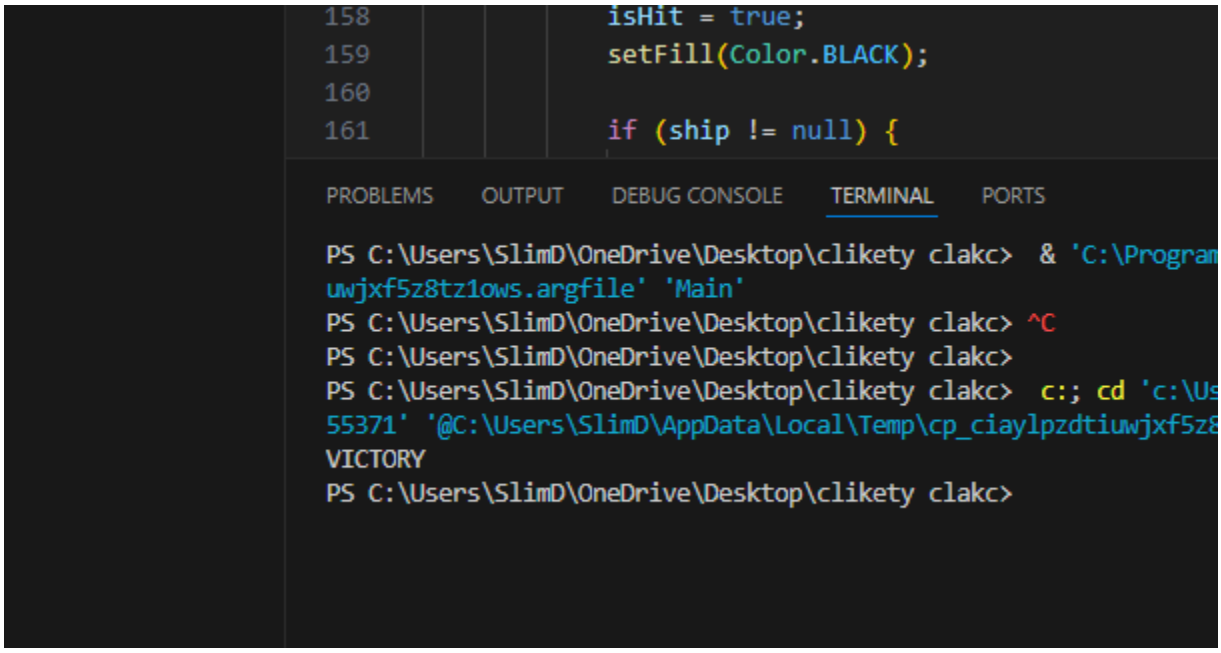
```
-------------------------------------
          Grid.Tile
-------------------------------------
 col: int
 row: int
 ship: Ship
 isHit: boolean
- grid: Grid
-------------------------------------
 Tile(col: int, row: int,
       grid: Grid)
 fireAt(): boolean
-------------------------------------
```

Ship.java

```
---------------------
      Ship
---------------------
 length: int
 horizontal: boolean
- durability: int
---------------------
 Ship(length: int,
    horizontal: boolean)
 damage(): void
 isOperational(): boolean
---------------------
```

Documentation:





Notes:

It is easier to test the program by running the program yourself. I felt that extra screenshots would be kind of redundant.

This definitely has a lot of room for improvement. I will mention some that I wish I had time for.

Better AI (Opponent AI is literally random, does not recognize that it hit something)

Option for human opponent: This was kind of rough. I was considering basically doing 4 grids instead of 2, left for player one and right for player 2, just to mess with things. I then considered trying to set up a local host/client thing, but for some reason my clients would never connect to the host (I think my port forwarding on other projects messed with something) and so I scrapped that.

The Victory/Defeat message is just being printed in the console, after the game is done, this should be easy to implement, but I ran out of time.

The AI moves immediately after the player moves, which is kind of awkward. I contemplated adding like a rest/wait thing, but decided against it as it felt forced. The computer instantly making moves was kinda cool.

Regarding the ship placement logic. I needed to make sure the ships never overlap during placing, but my implementation is a band-aid at best. The result is that you can't have 2 ships right beside each other, and I decided I would rather have this bug than the alternative.
Also relating to ships, I decided with the easy and lazy method to deal with ship sizes. I want to say that the original Battleship was like one 5-length, two 4-lengths, 1 3-length, and 1 2-length. I just took the easy way out by decreasing the amount of ships left to place and length at the same time, which made it a lot simpler. There could have been 5 separate ships to work with, but didn't get around to it.

I think that is about it. Please let me know if there are any issues with compiling or something, VSCode is really rough with JavaFX for some reason, and I had to manually set libraries. Otherwise much of the jank can be attributed to poor time management/laziness.