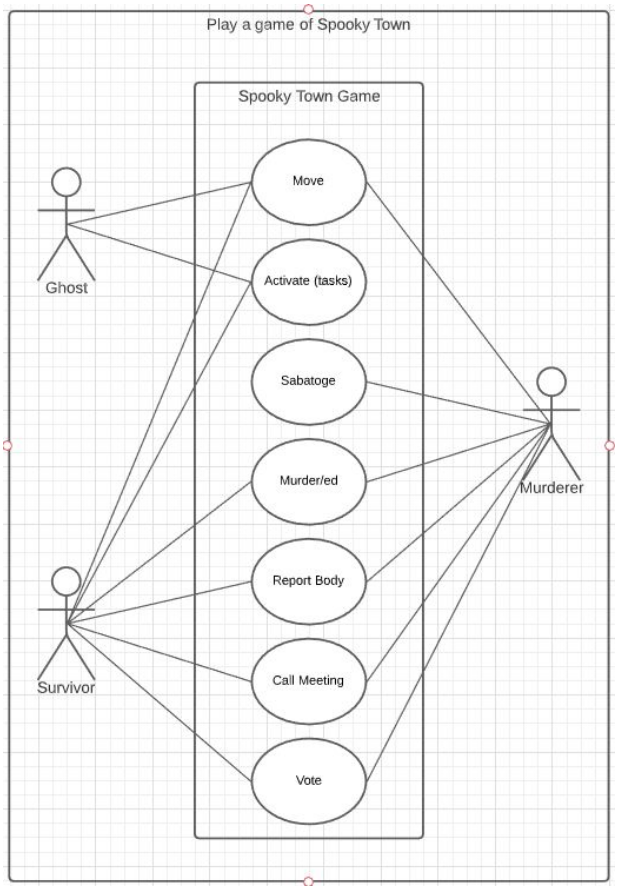
Project Summary:

- Name: Spooky Town
- Chandler Garthwaiter, Tanner Slemmer, Kolin Knewby
- We are building a text based game that combines strategy and deception. Similar to games like among us and trouble in terrorist town. This game will have two different roles. A majority of the players will have an innocent role(Can be renamed). And one or two will have the murderer role(Renamed). The goal of the game is for the murderers to kill all the innocents or survive until there are equal amounts of each role. The innocents can win by either completing tasks or by voting out the murderers. This game will require classes being able to interact with each other as well as possibly when an innocent/murderer dies they switch to another class(Ghost class)

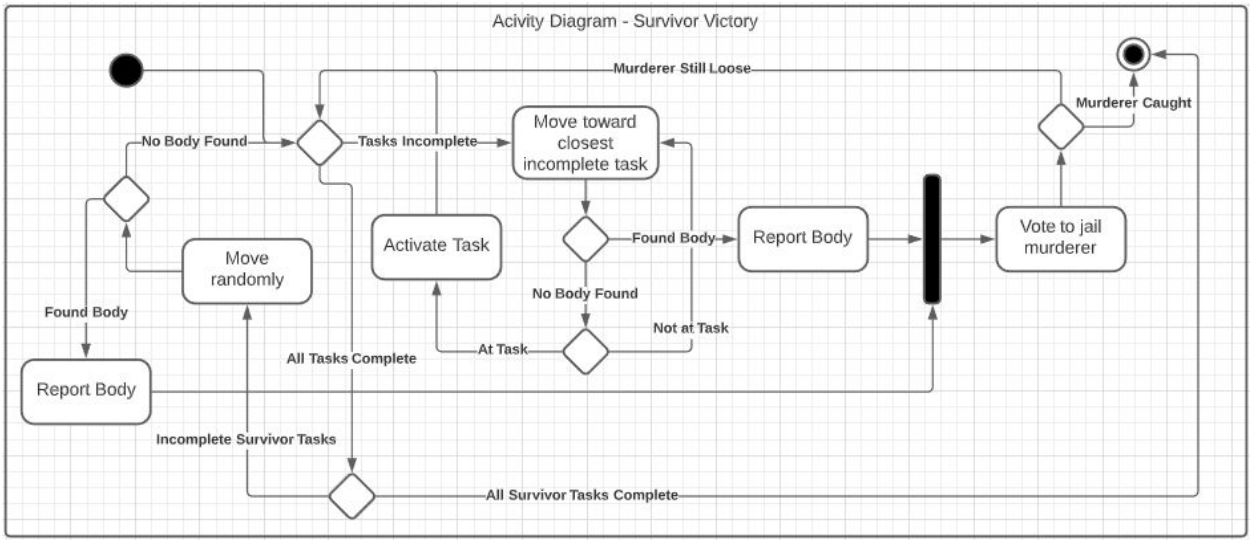
Project Requirements:

- Up to 10 players
- Connect to multiplayer(turn based/real time)
- Be able to handle turns and actions on each other(report, kill)
- Map is able to update on the server. Players map update based on what they can see.
- Handle different roles and their view of the map.
- If a character dies then their role is switched to ghost mode.

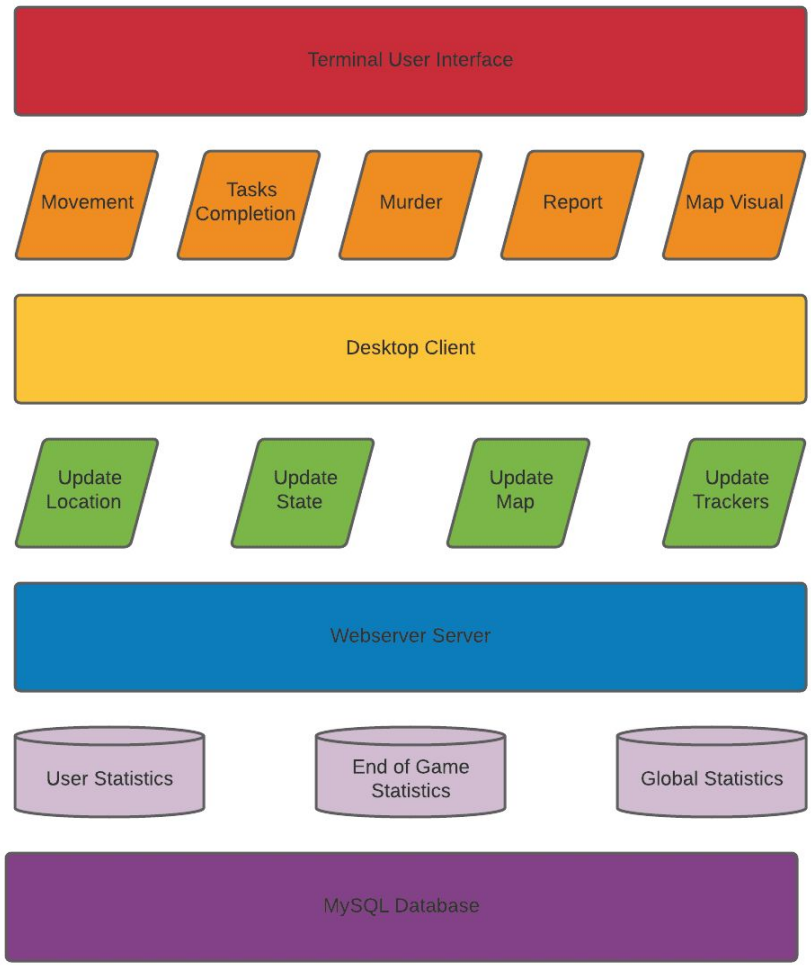
Use Cases:



UML Activity Diagram



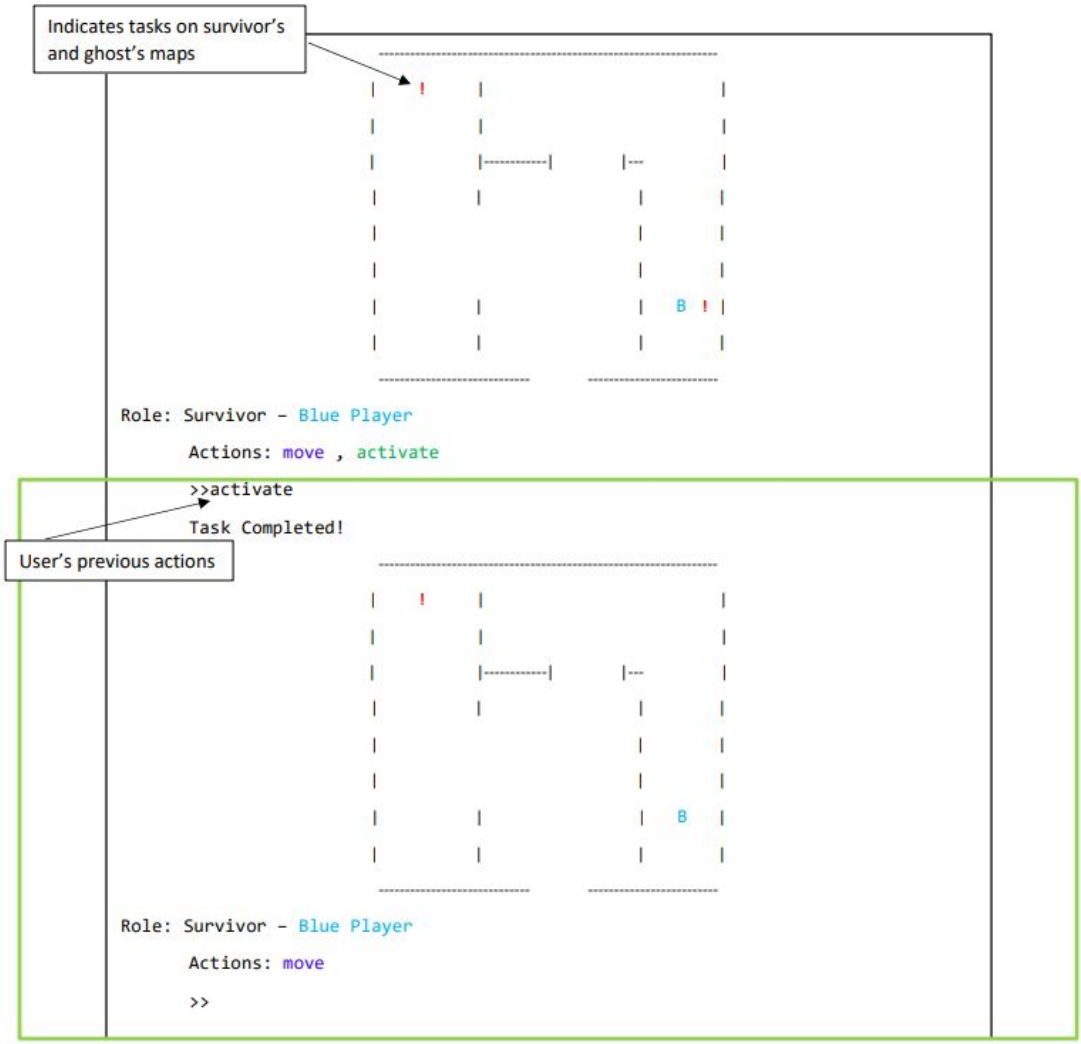
Architecture Diagram



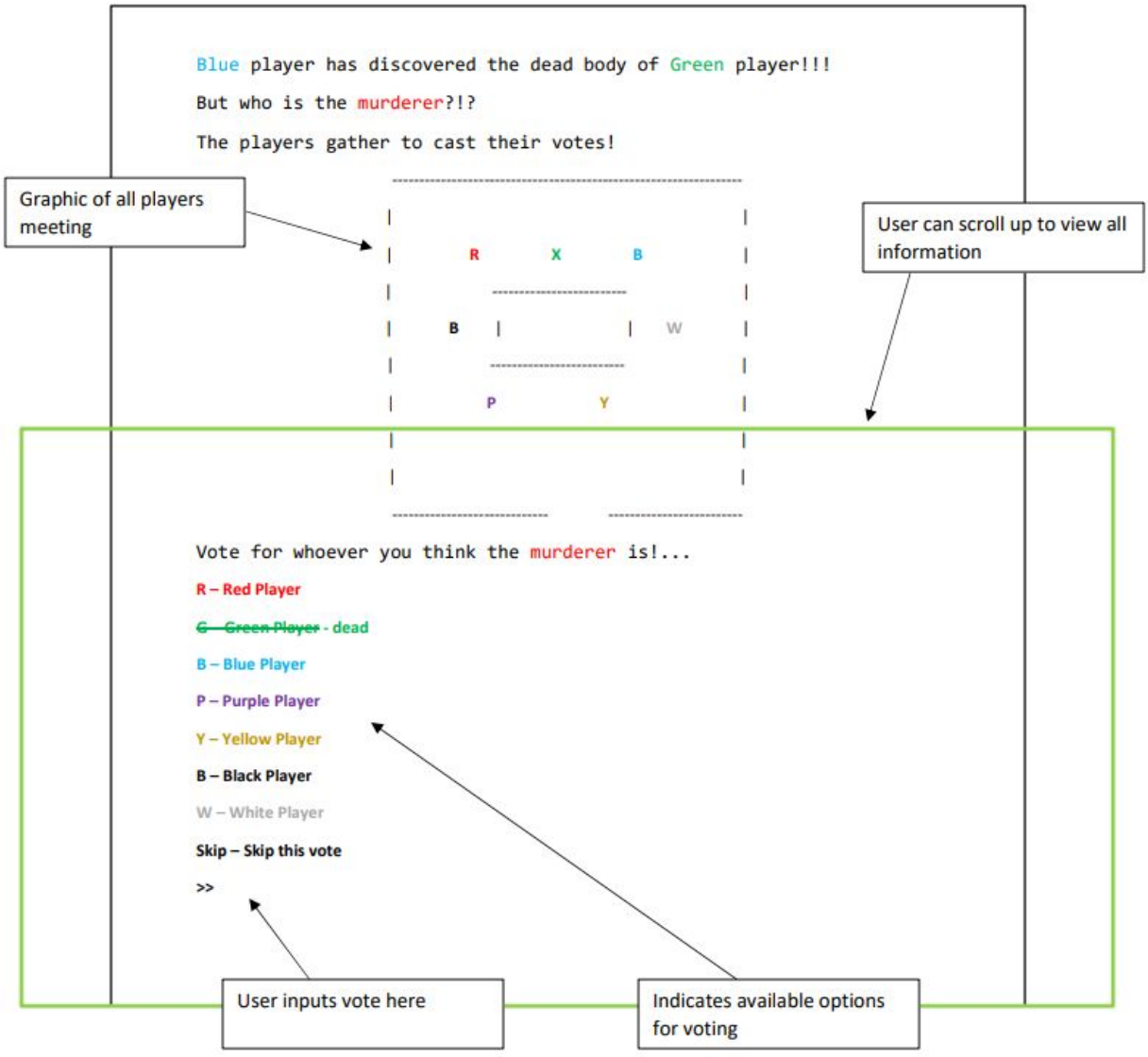
Data Storage:

- Using a MySQL database we will be able to globally store information for all clients with strings to load in the map (varchar can be incredibly large) and tables for player statistics and global statistics
    - For initial testing the map will most likely be a plain .txt file
    - For keeping track of unique players a basic ID system will be used
- The map will be loaded in through the map class on launch.
- Player and global statistics will be handled at the end of matches with the server class
    - These will be tracked by the server
    - Player statistics
        - Total matches played, innocent matches played, murderer matches played, games won, innocent games won, murderer games won
    - Global statistics
        - Total games played, total players killed, Innocent victories, Murderer victories, tasks completed

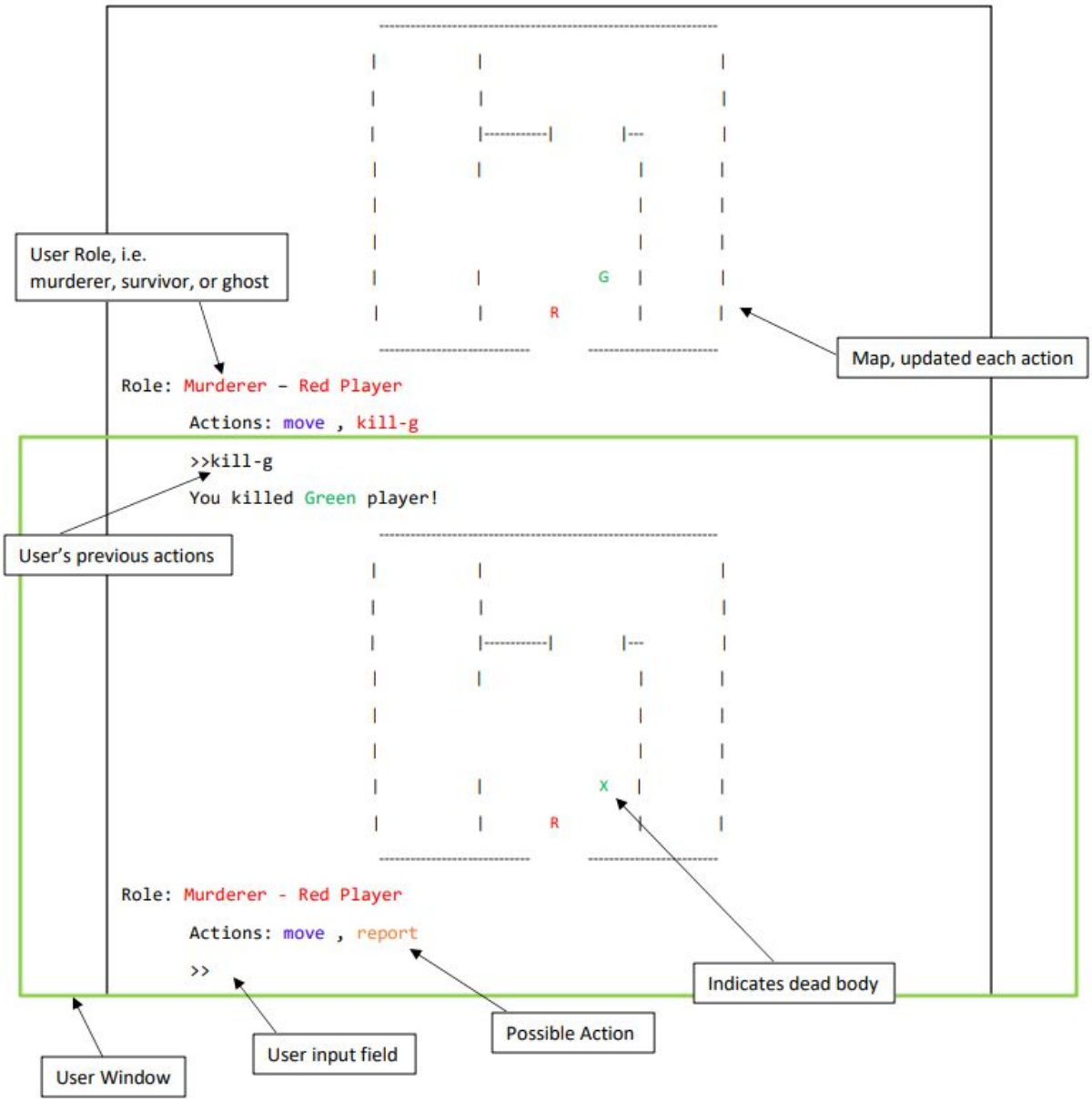
UI Mockups/Sketches:



^Survivor completing a task^

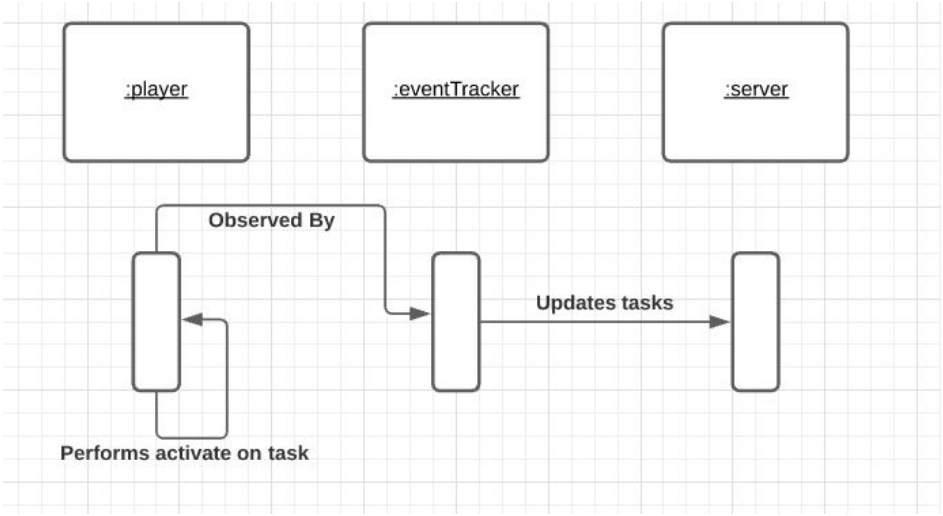


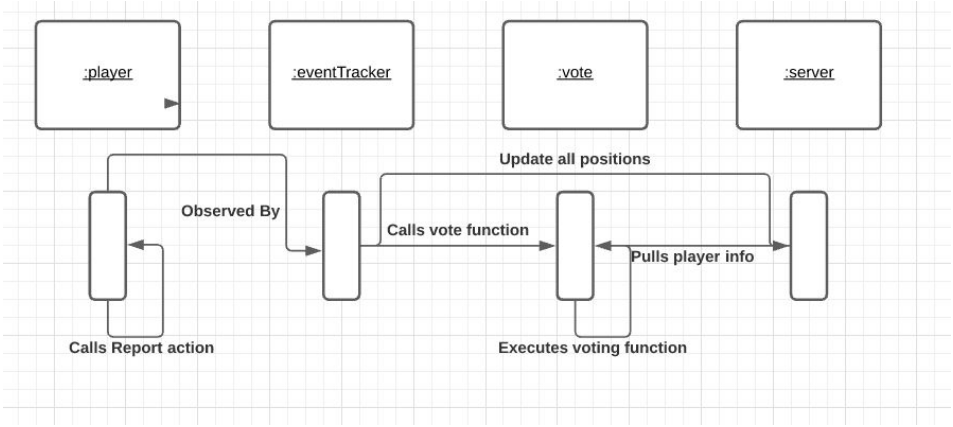
^Player voting for who to lock up^

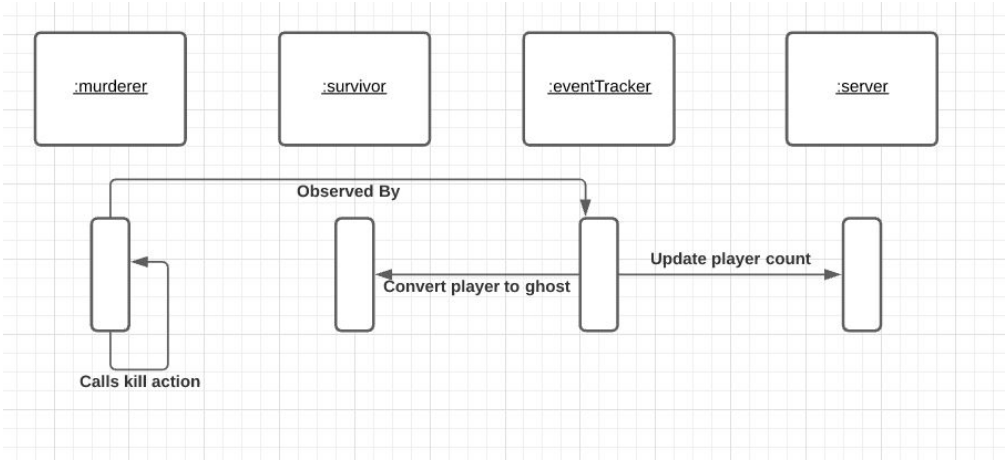


^Murderer killing a survivor^

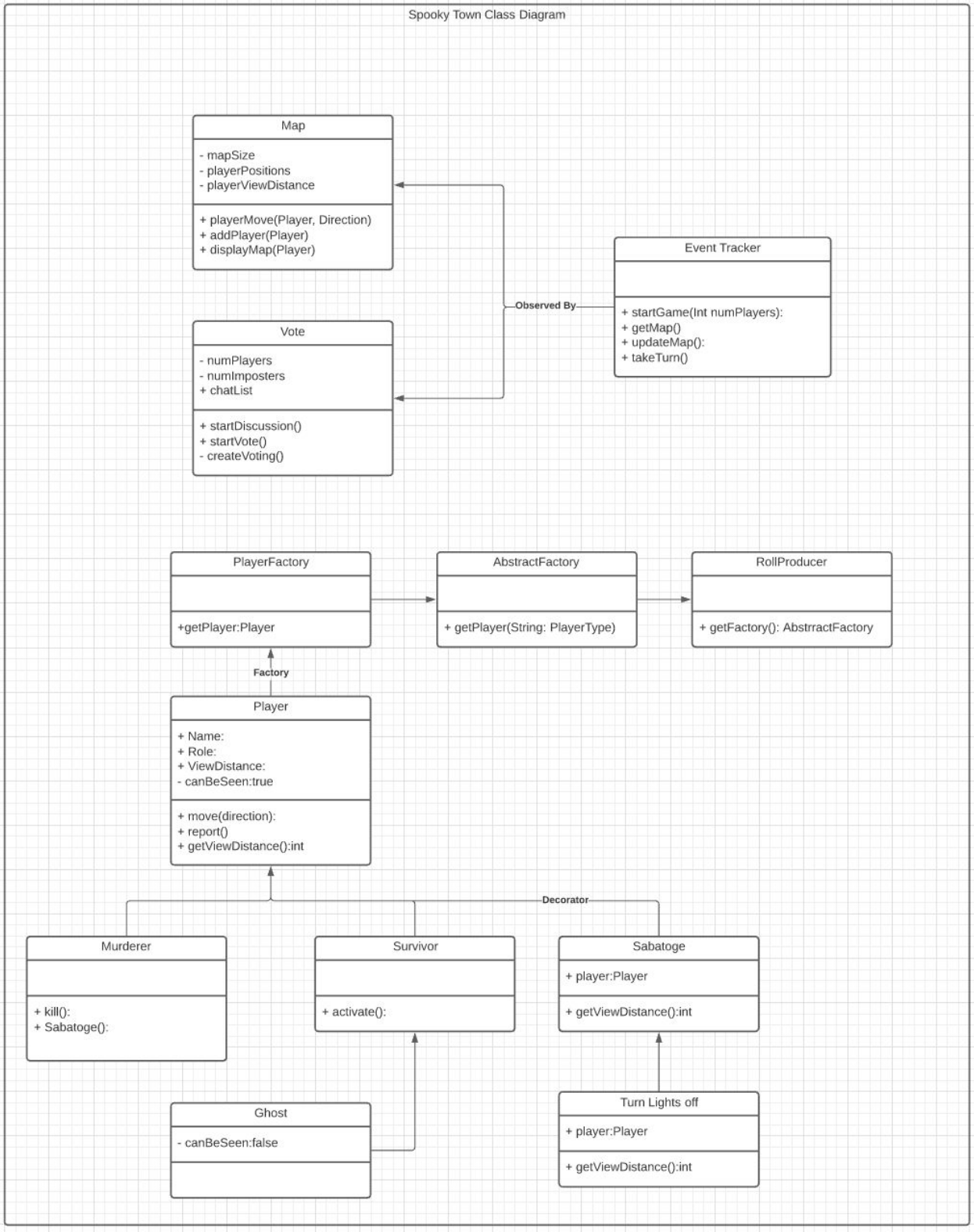
User Interactions/UML Sequence Diagrams:

- Survivor Completing Task
    - In the first UI Mockup you can see a blue player standing next to the “!” icon. Indicating they are next to a task. Upon activating it the task disappears from their set tasks and the overall task counter goes up. We will accomplish this by having the event tracker observe all events. When it observes the event of the player it will remove the task from their list, as well as incrementing the overall tasks completed
    - 

- Player Report Death
    - When a player enters a room and a dead body is located in that room. The option to report shows up as one of their actions. When entering “report”, everyone is returned to the main room and voting occurs. This will be done as when they enter the room a new map and actions will be updated based on who is in the room. When report action is called the observer will enact the voting function bringing everyone's position to main and start the voting process.
    - 

- Murderer Killing Survivor
    - Similar to above when entering a room options will display what the murderer can do. When there is a survivor in the room, they will have the option to kill them. Thus labeling them as dead, leaving a body there and creating a ghost character for the dead survivor. Then leaving the murderer with the option to report or move on. This will be done by the observing seeing that a kill action has been enacted
    - 

UML Class Diagram & Pattern Use:



- Outline
    - Text Base
    - Map
        - Starting grid of 15x15
        - Observes player movements and updates maps
        - Function that displays map per player. Take in player views into account
        - Sabotage - Uses the decorator pattern
        - UpdateMap(Player, x, y)
        - DisplayMap(Player, x, y)
    - Player (Abstract Class) (Name, Role, View Distance) (Functions: Move(Left,right,up,down))
        - Survivor (Override:View Distance) (Functions: Activate)
        - Murderer (Override:View Distance) (Functions: Kill, Sabotage
        - Ghost (When people die), Maybe observes the death of characters to create a ghost.
        - How do they talk to each other?
            - Adapter pattern? (Kinda overkill)
        - Factory to create players
    - Time or Event Tracker
        - Observable or something else?
        - Used to force meetings being triggered
    - How to make it multiplayer?
        - ?????
        - https://www.codingame.com/playgrounds/25775/codingame-sdk-documentation/create-a-multiplayer-game
        - https://www.youtube.com/watch?v=9vz-Dcdl8JA - alternative
    -
    - Patterns:
        - Observer - Player and Map observe each other
        - Factory - create player
        - Strategy for players - overriding the interaction
            - Like among us killers can't use tasks but can use doors
        - //Singleton - Map
            - Instead you have to use one of those pattern that turn off pass by reference in Java
        - Decorator - Sabotages
        - Command pattern for trigger sabotages???
        - Use adapter pattern w/ Multiplayer if server based
- Stretch Goals
    - Randomly generate map
    - Data storing statistics