There are several components that will comprise the Fish game system as a whole: presentation (what the human hackers see), data (every bit of information), and logic (how to make everything run smoothly). Each of these components belong to their eponymous layers in the standard three-layer architecture. First, let us consider the "presentation layer". This basically covers the HCI aspect, which is crucial–though the game is being played by AIs, a human hacker still needs a user interface to interact with. Essentially, the hackers need to "see" the instance of the server, perhaps in the form of a website, in order to be able to have a physical way to sign up their AI players, pay an entry fee, watch the game take place in real time, and be notified of whether they won, lost, or were terminated from the platform. When watching the game, the hackers need to see the game board, the hexagons, and the avatars and names of the AI players. The presentation layer also needs to be aesthetically pleasing; it is what will essentially sell the business to human hacker clients. Next, we can consider the "data layer", as we need to keep track of critical information related to both the game server and the Fish game. In terms of information related to the game system, we need to keep track of when the signup period starts and ends and who the signed-up hackers are, along with information about their AI players/the software that defines these players. One key piece of data we need to store is the players' ages, as the youngest player goes first in the Fish game. Perhaps we also need to store a blacklist of who the terminated hackers are. In terms of the Fish game itself, we need to store the game state. Specifically, we need to have information about whose turn it is, how many fish each player has, how many hexagons are on the board, and how many fish are on each hexagon, all in real time. We also need to store the complete set of game rules. Finally, we can consider the "logic layer". This layer coordinates all aspects of the server and game and makes sure that everything is running smoothly. If a hacker tries to sign up, this layer will work with the payment subsystem to get their entry fee processed, and will also make sure that the hacker has not been banned from the platform. Then, it will take the information about the hacker and their AI player and work with the data layer to get this data stored. The logic layer also controls the game itself and can be thought of as the "referee". It determines the board layout and uses the data layer's set of rules to make sure that valid moves are being made. It also uses the data layer's information about whose turn it is in order to make sure that no other players make a move at the same time or the wrong time. It will also help connect the AI player's software to the Fish game and will begin the termination process based on whether an AI player's software is malfunctioning. Once the winner is determined it will work with the payment subsystem to pay the prize money. Throughout all of this, it will tell the data layer to update certain bits of information. The logic layer interacts with the presentation layer by "telling" it what to display, such as whether it is time to display the website homepage, the sign up screen, or the game board itself. The logic layer also will understand whether the server is malfunctioning or overloaded, and perhaps will tell the presentation layer to display a "Temporarily down" page. The logic layer is what facilitates communication among the three layers and connects them together. Therefore, each component is related to each other by this: the presentation layer displays what is in the data layer, and the logic layer tells each layer what to do and when to do it.