The Battleship game we were given for Project 2 can be classified as being built and organized in a  3-tier architecture. The Project 1 team used object-oriented programming in C++ to divide responsibilities among three categories of classes.

The “Executive” class was the **Presentation** tier. It was responsible for the top-level user interface, and gathered input data about where ships were placed, where players would like to target, etc. This presentation tier was presented in the terminal command-line, where the game boards were displayed and user input collected.There is however some overlap in the presentation tier, specifically in the “Executive” and “Game” class. This overlap can be seen in other classes as well as some class functions are also responsible for outputting text to the terminal.

The “Game” class was the **Logic** tier. It was responsible for the backend logic behind the game such as switching turns, updating boards, keeping track of player’s hits, and tracking the current progress of the game. This “Game” class is also responsible for the construction and implementation of the various array pointers for the player’s ships and boards.

Finally, the team included several other classes such as “Board” and “Ship.” These were the **Data** tier of the architecture. These classes created objects that operated like databases, storing information about the particular objects. For example, the “Ship” class stored information about a ship’s size, location, and orientation, while the “Board” class stored information about where player’s had already guessed, the size of each board, etc.