For our prototype development, we utilized the top-down functional decomposition paradigm as well as the component level design. Since we already had a general idea of what our UI would look like and what our application would be capable of, it was simple to create an overview of the system. Simply put, the application would be a web page with a variety of applications(games) that could be selected and run based on the user's choice. The applications could therefore be designed individually, making each one a subsystem. Additionally, each of these subsystems would need to have their own smaller segment breakdown in order to be implemented completely. For a given application, we would divide the application into UI and game logic components. Since the design should be centered around the experience playing the game, we would start by developing the UI and the central game mechanic. For instance, torus tic tac toe began by implementing a means of interacting with the torus and clicking on it to mark x's or o's. Then, the game logic was implemented as a separate component with the necessary data structures. Finally, win conditions were set, and more UI features were added. In the case of the matching game, we started by developing a tile flipping animation and implementing the raycaster. Then we implemented the game logic detecting a match. Only after completing these components, was the design complete.

This design paradigm was effective because it broke down the applications we were building into smaller tasks that could be turned into functions. These functions could be reused when necessary and made changing code a lot easier. It also made combining the interface and game logic for our games much easier.