The software architecture we chose for this project while keeping the design paradigm in mind is three tier architecture. There is a slight difference of how our architecture is implemented compared to the description provided in the lecture slides. Our code is completely local, it is not done through three physically independent machines.

Our application’s presentation tier is executed through an html webpage that presents the list of games we have made in python. There is another component of our presentation tier, the games themselves have their own presentation method. Both Tetris and Pong use the pygame window to provide visual feedback to the user. It is a window that refreshes at 60 frames per second and allows the player to manipulate the objects using the controls on the keyboard.

The second layer of logic is held locally just like the presentation layer. Our logic layer contains all of the code from Tetris, Pong, and our webpage. It serves as the middleman between the presentation layer and the data layer, allowing the user to see and play the game or use the webpage.

The code will utilize the local machine’s storage to store and retrieve information from the files which then goes back into the logic tier to be processed. Types of information stored inside files would be arrays/lists, and various variables made to save the data of the game, this would serve as the data tier. While this may not be the same, we still benefit from the advantages of three-tier architecture. We can edit the logic of the code whilst keeping the data tier and presentation tier intact.