EECS 448 – Team 10 – Project 3:

Chat-Room Software Architecture Documentation

For this project, we went forward with a Client-Server architecture. We chose this architecture strategy because it almost exactly represented the requirements that would be set forth by a chat room. The Client will initiate a request to send a message to the Server, and the Server will then distribute that request to all other clients. A web-socket is used in our architecture so that the connection between Client and Servers are continuous and never interrupted, but they are still Client Server connections. This request-response messaging pattern is what allows for chatting between users to take place. Every time a single Client initiates a request to the server to send out a new chat, every single user connected to that chat room receives a response data set with that new chat outlined within it. Additionally, for project 4, this client server architecture will become more important as the Client will initiate login request, that will send separate Login credentials to the Server to be stored in a database. This request will then get a response to allow the Client to proceed to a chat room, and then the chat requests are made. This project 3 is further a client server architecture because in order for the chat room to work properly, you must run the servers that it communicates locally. The requests are then made to the localhost servers and communication is done through the HTTP protocol to send our PHP requests to the PHP server, which then delegates chats. No matter the number of local clients requesting the PHP server, it will send the response messages with the chat inside it to all of the clients.