The previous diagram shows a user node communicating with a switch, then communicating to a modem, which then connects with a server that is for processing application logic, which then communicates with the database management system on the final tier of the system.

For the physical architecture of the system, we needed to start with something small, easy to use, and affordable but also a design with scalability in mind. Karoline’s system, we decided, should use a three-tier client-server architecture. The cost is not as high as say a client-based or server-based architecture while also allowing for scalability in the future, since all Karoline needs to do to expand is add another server with the third party that is hosting it.

We chose three-tier in order to keep from having any issues with different operating systems communicating and using the system, the second-tier server will process the application logic – again, being hosted by a third party.

As of right now, Karoline doesn’t need much, so the physical deployment should only include the user node, a server to process application logic, and a server hosting the database management system to hold all of Karoline’s databases/tables e.g. inventory tables, cake tables, menu, cookie tables, etc.