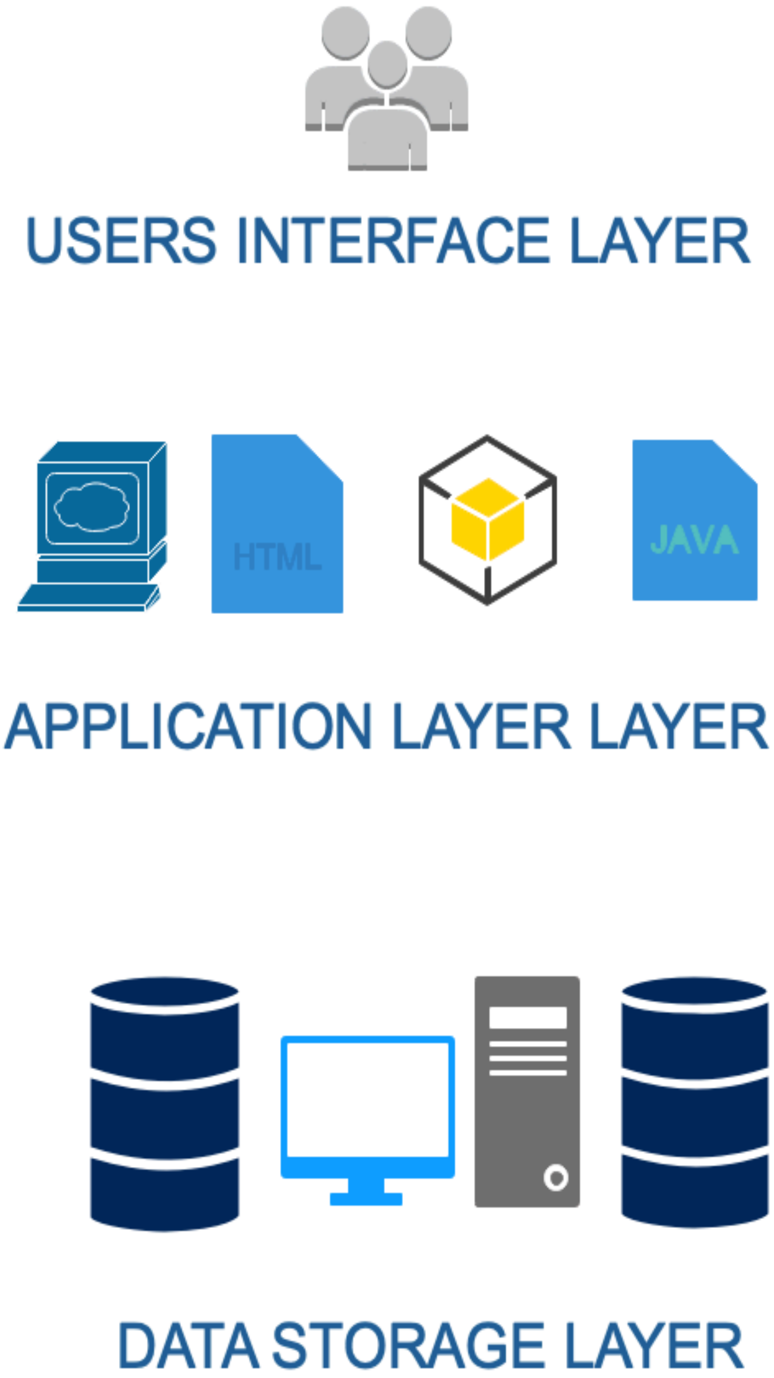
Danny Parks
TimeSaver App – Week2 Service Design

User input operation
POST → /user (create user)
POST → /user/createWithArray (create user with the given input array)
GET → /user/login (logs the user into the system)
GET → /user/{username} (get user by the user name)
PUT → /user/{username} (update user information)
DELETE → /user/{username} (delete user)

User output operation
GET → /user/{username} (get user by the user name)
GET → /room/{id} (get room name)
GET → /supplies (returns list of supplies)
GET → /student (get student information for room)
GET → /foodLevel (get amount of food levels)

Data output operation
GET → /user (returns the user information)
GET → /user/{username} (get user by the user name)
PUT → /user/{username} (update user information)
DELETE → /user/{username} (delete user)
POST → /output (post information to end users device)



USERS INTERFACE LAYER

APPLICATION LAYER LAYER

DATA STORAGE LAYER

The Interactions

User Interface layer. Responsibility for presentation and user interaction resides with the first-layer components. These client components enable the user to interact with the second-layer processes in a secure and intuitive manner. WebSphere Application Server will support several client types. Clients do not access the third-layer services directly.

Application Layer. The second-layer processes are commonly referred to as the application logic layer. These processes will manage the business logic of the application and are permitted access to the third-layer services. The application logic layer is where most of the processing work occurs. Multiple client components can access the second-layer processes simultaneously, so this application layer must manage its own transactions.

Data Storage Layer. The third-layer services are protected from direct access by the client components residing within a secure network. Interaction must occur through the second-layer processes.

Communication among layers. All three layers must communicate with each other. Open, standard protocols and exposed APIs simplify this communication. I will write client components most likely in languages as Java or C++. This will also allow the clients/end users to run on any operating system, by speaking with the application layer. Databases in the third layer will be of the design in my database design proposal. The most important key to this architecture is the application layer.