Re-Design: Service Layer(s)

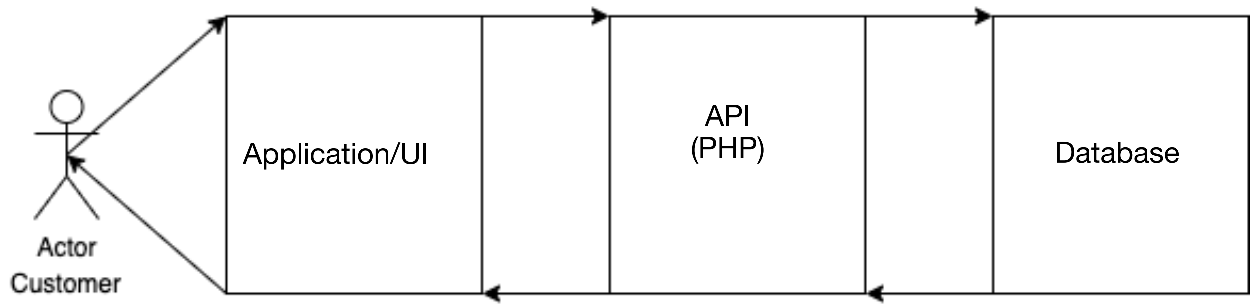
By: Tyler Pranger

Change-Log

Expanded on the high-level overview. Added several different error handling and API responses. Finally, there are now several different endpoints that the user will be interacting with.

High-Level Overview

There will be three basic service layers to the application, this includes: the application its self, the server, and the database. As the user interacts with the application it will communicate with the server which will do a database hit if needed. For example, when the user logs into their account it will send the necessary data all the way to the database and then return either a success or fail. The application it self will be using HTTP while the database will be running SQL.



Essentially when the user clicks on a button in the UI it will hit the an API which will query the database and retrieve the correct results. For example, when the user signs up for the service, the API will hit the database and check if the user already exists. If it comes back as a former user it will bring a reply back to the user, “User already exists in the system. Forgot username and password?”. If the user does not exist it will INSERT a new row in the database and return a result back to the user in the form of logging the user in.

Error handling/API Responses

There are several areas where error handling is needed in order to have a good experience. The API will check if the server is not responsive there will be a response code of 500 on the webpage. If it is successful it will return a code of 200. These are standard messages to a user using any web application. Below are examples of error handling and API responses for both success and failures.

1. User authentication
    1. When a user logs into the system the application will need to query the database and compare the result of the query to what the user had entered into the system. Since this will be most likely a POST statement it will not change the URL at all. If the user fails to enter the correct information it will bring back a message to the user “Incorrect username or password. Forgot username and password?”.
    2. This will be a POST operation and will affect the URL at all.The reason for that is any user information will not be posted in the URL.
2. Sign-up
    1. If the user tries to sign up again the API will query the database and check for the ‘email’ column of the database. If the user does already exist it will bring back a message to the user “User already exists in the system. Forgot username and password?”. Otherwise it will log that person in.
    2. This will be a POST operation and will affect the URL at all. The reason for that is any user information will not be posted in the URL.
3. Exercises
    1. Adding or removing an exercise will be a POST command and it will affect the URL. The response that will be given in the url will be something like this
4. GET - Still deciding if doing a GET within the program is necessary, but if it is this would be an example: “http://localhost:8888/Training_Program/home.php?exercise=bench_press&reps=10&sets=3".

Endpoints

Below are where the endpoints for the user will be located at and what will happen.

1. Home Screen
    1. User authentication/registration
2. Menu
    1. Login/Logout
    2. Go to account settings
    3. Go to exercise results
    4. Go to Generate workouts

3. Account Settings
    1. Deactivate Account
    2. Modify any user information
        1. First Name
        2. Last Name
        3. Email
        4. Password
4. Exercise Results
    1. Retrieve past exercise results
5. Generate Workouts
    1. Auto generate workouts
    2. Add exercises
    3. Remove exercises
    4. Modify exercises (reps and/or sets)