CS 4760 – Capstone – Fall 2022

Group: Cody Butterfield, Brielle Fernelius, Whitney Smith, Terra Tretter, Garrett Kuns

Project: Fitness App (Snake Den Fitness)

Snake Den Fitness

**Project Setup:**

**Setup project environment with virtualenv and pip:**

Within the root directory of the GitHub repository run the following (via command prompt or built in IDE terminal):

Clone the Repo:

https://github.com/bradleypeterson/snakedenfitness.git

Go to the application folder(the project is two layers deep):

cd snakedenfitness/snakedenfitness  
 (you should be able to run dir or ls and see the manage.py file)

Create a virtual environment:

WINDOWS:

py -m venv .venv

.venv\scripts\activate

macOS/Linux:

python3 -m venv .venv

source .venv/bin/activate

Then run:

pip install -r requirements.txt

python -m pip install channels\_redis

python -m pip install -U channels["daphne"]

Install Docker Locally: https://www.docker.com/products/docker-desktop/

Then run:

docker run -p 6379:6379 -d redis:5

The following steps will ensure your database is accurate to the code. Clearing out the migration folder in each of the different Django apps and running the following will ensure that:

python manage.py makemigrations community

python manage.py sqlmigrate community (migration number ex: 0001)

python manage.py makemigrations diet

python manage.py sqlmigrate diet (migration number ex: 0001)

python manage.py makemigrations fitness

python manage.py sqlmigrate fitness (migration number ex: 0001)

python manage.py makemigrations users

python manage.py sqlmigrate users (migration number ex: 0001)

Then run:

python manage.py migrate

Now, to start the server, run the following:

python manage.py runserver

**Set up Azure deployment:**

The site was hosted utilizing Azure Webapps. The following will walk you through hosting Snakeden Fitness on your own Azure profile.

First you will need to verify you have an azure account. Go to portal.azure.com to login and verify your account. This will be where you can check your Resource group and App service once you deploy the program. Once deployed, you will also see your Subscription. To this point I have not had to pay for the deployment.

If you have gotten to this point and can run python3 manage.py runserver and see the deployment locally hosted, then proceed. If not, go back and verify your setup. **You do not want to deploy an app that does not run locally.**

The setup of this deployment will be through VSCode.

To create Azure resources in VS Code, you must have the Azure Tools extension pack installed and be signed into Azure from VS Code.

[Download Azure Tools](https://marketplace.visualstudio.com/items?itemName=ms-vscode.vscode-node-azure-pack)

To be safe, in the application folder(where the manage.py file is), run:

code .

\*\*This will reopen VSCode in the directory and we will use this page from here on out. This is **crucial** to the correct deployment.

1. Locate the Azure icon in the left-hand toolbar. Select it to bring up the Azure Tools for VS Code extension.

If you do not see the Azure Tools icon, make sure you have the Azure Tools extension for VS Code installed.

1. In the Azure Tools extension for VS Code:
   1. Find the RESOURCES section and select your subscription.
   2. Select + (Create Resource...)
2. Choose the Create App Service Web App... option.
3. Enter the name snakedenfitness for this web app.

When deployed, this name is used as your app name in the form <https://snakedenfitness.azurewebsites.net>.

1. Select the runtime stack for the application. For this program, select Python 3.10.
2. Select the App Service plan (pricing tier) for this web app. The App Service plan controls how many resources (CPU/memory) are available to your app and how much you pay.

For this program, select the Basic (B1) pricing tier. This plan will incur a small charge against your Azure subscription but is recommended for better performance over the Free (F1) tier. (This instruction portion is from Microsoft’s site. I have yet to accrue a charge for using B1. You must also use the B1 for the F1 tier will not properly host the site and it will not display)

1. Select the Deploy button in the "Created new web app" notification.
2. Select the quickstart folder you are working in as the one to deploy.
3. Answer Yes to update your build configuration and improve deployment performance.
4. When the deployment is complete, a notification will appear in the lower right corner of VS Code. You can use this notification to browse to your web app.

This should properly deploy your application to your azure account. The deployment must happen like this because we are not the repo owner. As a result, whoever hosts the site must communicate with their team to run deployments as it is not automated and can only be done by the account owner.

To redeploy after a git pull, you will find the Azure tab on the left pane in VSCode. Once you’re in that tab, hover over the WORKSPACE directory. You should see a little cloud with an up arrow. When you click on the cloud there will be two options. Make sure to click ‘Deploy to Web App’. You will then be asked to override any previous deployment to which you will select yes. You should see a popup in the bottom corner of your VSCode showing deployment in progress. Once complete you should get a success message with an option to browse your site.

**Azure Docker Deployment:**

The community chat in the application runs through a docker container which, when run on your localhost, runs great. The deployment of the docker container to the hosted site was not successful so the community group chat hosted live does not work. This will need to be looked into if it is even possible to run a docker container on the side of the app deployment.

**Scope and Priorities:**

**Features and Priority Level:**

1 – Highest Priority

User Roles and Authentication

o Users of the system are able to register as a dietician, personal trainer, or a client and then login as such to use the system. Different roles have different actions they can perform once authenticated. These are documented as follows:

§ Personal trainers:

· able to view client data pertaining to the fitness section (if the client is associated with the trainer).

· able to accept/reject client requests to join them

§ Dieticians:

· able to view client data pertaining to the dieting section (if the client is associated with the dietician).

· able to accept/reject client requests to join them

§ Clients:

· able to enter data about their diet and workout (documented in-depth below).

· able to request to “join” a trainer or dietician (documented in-depth below).

· able to invite and accept/reject invitations to community groups and participate in them. (Documented in-depth below).

**Status: Completed.**

Data tracking of workout and diet profiles

o Client can enter data for both workout and diet pages.

o Dieticians and personal trainers can view the information relevant to them (if associated with the client).

o Workout data:

§ Subdivided by cardio, strength training, and endurance (more categories will be added if seen as necessary)

§ Within each subdivision, it is divided by the part of the body: arms, legs, core, back, etc.

§ Once in the category, the user can add exercise name, the number of reps, sets, weight (if applicable), etc.

§ Personal Records will be associated to the workout type itself (ex: deadlift, max bench, etc.). These are within the parts of the body subdivisions.

o Diet data:

§ Daily calorie counter easily visible

§ Subdivided into either snack, breakfast, lunch, or dinner

§ Within each of the subdivisions, there will be information such as calories, carbs, sugars, etc.

**Status: Completed.**

Allow user to add a dietician and personal trainer

o Client can request to be added by a dietician or personal trainer.

o Client can add a dietician/trainer to be able to view their data.

**Status: Completed.**

Clients will be able to add other clients to community group

o Client can add other clients to a community group by entering their username which sends a request to the other client if the username exists.

o Clients can accept or reject requests to join a community group.

o The groups will act as a social feed where clients can post about their workouts/dieting or anything else they want to share. All members in the community group will be able to view the posts.

**Status: Completed.**

Deployment on Azure

o Deploy site to Azure so it can be used without running the source code locally.

**Status: Majority Completed** - Complete aside from the messaging in the community tab due to an issue supporting a docker container on the hosted site. All other features are implemented on the hosted site. This is the top priority for future development.

2 – Desired Features

· File upload for workout guides/recipes – Somewhat Completed

o Clients can upload files to the community to share recipes/workouts/etc. with all users.

**Status: Somewhat Completed** - Functionality to upload a file exists but is not fully implemented for everyone to see and to handle multiple files being uploaded. This is the second highest priority for development.

Mobile friendly display (Not an app, just works for various screen sizes)

o Users will be able to access the site from a mobile device via an internet browser and the site will display cleanly and function the same as on a larger screen.

**Status: Not Completed.** Not a high priority, but still a priority after the higher priorities above have been completed

3 – Additional Features

· Community group enhancements.

o Notifications of community members posting in the community group

o Comments and reactions to posts

o More sleek design for posts, not like looking at a form.

**Status: Not Completed.** Not a high priority, but still a priority after the higher priorities above have been completed

Daily streaks/points/rewards

o Streaks of days in a row the client has worked out.

o Streaks of days in a row the client has been below some limit for calories or has eaten a meal with each food group.

o Possibility to view others’ points in client’s community group.

o Rewards can include point multipliers, a streak saver (would allow the user to skip a day without losing their streak), etc.

**Status: Not Completed.** Tertiary priority following above.

**End of Semester Goals**

User Roles and Authentication:

· Users of the system are able to register as a dietician, personal trainer, or a client and then login as such to use the system. Different roles will have different actions they can perform once authenticated. These are documented as follows:

o Personal trainers:

§ able to view client data pertaining to the fitness section (if the client is associated to the trainer).

§ able to accept/reject client requests to join them

o Dieticians:

§ able to view client data pertaining to the dieting section (if the client is associated to the dietician).

§ able to accept/reject client requests to join them

o Clients:

§ able to enter data about their diet and workout.

§ able to request to “join” a trainer or dietician.

§ able to invite and accept/reject invitations to community groups and participate in them.

**Status: Goal met**

Data tracking of workout and diet profiles

· Client can enter data for both workout and diet pages.

· Dieticians and personal trainers can view the information relevant to them (if associated to the client).

· Workout data:

o Subdivided by cardio, strength training, and endurance (more categories will be added if seen as necessary)

o Within each subdivision, it is divided by the part of the body: arms, legs, core, back, etc.

o Once in the category, you can add your exercise name, the number of reps, sets, weight (if applicable), etc.

o Personal Records will be associated to the workout type itself (ex: deadlift, max bench, etc.). These are within the parts of the body subdivisions.

· Diet data:

o Daily calorie counter easily visible

o Subdivided into either snack, breakfast, lunch, or dinner

o Within each of the subdivisions, there will be information such as calories, carbs, sugars, etc.

**Status: Goal met**

Allow user to add a dietician and personal trainer

· Client can request to be added by a dietician or personal trainer.

· Client can add one dietician and one trainer. This will give permission to the dietician/trainer to be able to view the client’s data.

Trainers/dieticians can remove/”reject” clients. Rejection does not and will not prevent the client from requesting that user as trainer/dietician in the future.

**Status: Goal met.**

Clients will be able to add other clients to community group.

· Client can add other clients to a community group by entering their friend code which sends a request to the other client.

· Clients can accept or reject requests to join a community group.

· The groups will act as a group chat where invited members can communicate with each other to share workouts, recipes, advice, motivations, etc.

**Status: Goal met.**

**Note: No promised goals were not met.**