Hector Alvarez
Fall B, Capstone
Oct 23 2021

Motivatr : A fitness application

We have all witnessed the New Year's phenomenon; after a winter season of delicious food, everyone wants to shed off their newfound weight. January arrives, and everyone makes it their new year's resolution to join a gym. Everything is going good for the first few weeks, but by the end of January, less than half of those people are still going to the gym. In fact, according to CCD statistics, nearly one half of adults in the United States have tried to lose weight in the last 12 months. Yet many claim that they find it difficult to find a routine or a schedule that pushes them consistently to reach their fitness goal.

This is the purpose of Motivatr, a fitness application focused on providing positive feedback to the user as often as necessary to keep them on their fitness journey. The application will ask users what their fitness goals are, for example, losing weight, getting toned, or building muscle, and will design a program to help the user achieve their goal. The application will supply the user with videos of the exercises that they should do, along with the number of sets, repetition, and time required for each exercise. In this way, the application will serve as a digital trainer, in which the user will not need to pay additional costs for an in-person trainer. Additionally, at the end of every training session, the user will input their workout statistics, such as how many pounds they were able to perform on each machine. This data will be stored in the user's profile and will be displayed in the user's dashboard every time they log into the application. By seeing progress over time, the application will further motivate the user to maintain their progress. Additionally, the program will also keep a record of the user's weight and display this information as well. By using push notification, the application will maintain a line of communication with the user notifying them about upcoming workout sessions, as well as positive comments regarding their recent achievements.

Who is the target audience for Motivatr? Well, anyone! As previously stated, about half of Americans have tried to lose weight in the past year alone. Motivatr is for them. Motivatr is also for current avid gym goers that would like to track their progress and see live statistics of their physical improvement. Additionally, Motivatr doesn't have to stop at the gym, and can be used to track stats for runners and bikers. This application can be used by anyone who wants to keep a more consistent schedule doing the physical activities that they love.

Motivatr will function the most optimally as a phone application since the user can take Mortivatr on the go. The users can access the video workouts at the gym, and then input their statistics as soon as they finish with the workout. Additionally, the application needs to send the user push notifications regarding upcoming workouts and positive and re-enforcing messages regarding their recent progress. In time, a more developed web application can be made to check on their workout charts/statistics more easily.

Minimal Value Product

Features:

User must be able to create a profile, and after thus be able to sign into that profile. This will require the use of a Database to store user information. User information must remain secure. This will be done by using a library to hash the password. As the password is stored in hashed form, it will make stealing user sensitive data more difficult. Additionally, the user needs to interact with the application to input the required fitness data into the database. This therefore means that the database should have some relational component that can connect user contact information (such as their name email password) to their gym metrics (such as their physical information, age, height and weight) as well as their gym performance. The data will be stored, and time stamped. This is important since the application will also need to display graphs of metrics over time. The graphs will have an option to display information from the current week, the current month, 3 months, 6 months, and a year. Data will not have a limit and can be stored indefinitely.

Additionally, the application will have a selection of trainer workout videos to which the user can select a video and learn more about a workout. The user will have a library available to them and will not be limited by what fitness program they chose. The fitness program will allow the user to select what their ideal fitness program will be, and the user will have 3 primary options: To lose weight, get toned, or body build. Additionally, the user will be able to also select how many times a week they would like to work out. The program will outline a suggested schedule, but the user will ultimately have a final decision on their fitness schedule.

Additionally, the program will provide push notifications to keep the user updated on upcoming workouts, as well as positive feedback regarding the user's recent progress. This is essential since it's the motivation aspect of the application. By consistently showing the user positive results, we will rely on positive reinforcement to keep users more engaged using the application, as well as reaching their fitness goals.

Finally, the program will provide feedback regarding user progress. The program will run diagnostics from the user data to provide them with more information about their recent progress.

Minimal features required:

The application will minimally require a database to store user information and gym statistics, such as their weight, their target goal, their current gym workouts. Additionally, the application will need to output that data with charts regarding the user's weight goal, and gym workouts progress. The application can stand to lose the video lessons/ virtual trainer. But the application is to essentially record data and show

user progress over time. This application will at least need to create a UI to initially prototype, and in time can be optimized to meet user experience demands.

Architecture:

This application will utilize a React-Native front end. I have specifically chosen React native, as it is a hybrid application in which both Android and Apple phones can use the application. This will open the application to more users, and potentially expand its userbase, profits and popularity. Additionally, it lies within my technical limitations as a programmer since I have the most familiarity with the React Framework. Additionally, the application can easily expand into a web app refactoring most of the same code.

If possible, I will also use a CSS framework called Tailwind to speed up the development process.

For the back end, I will utilize NodeJS and create the server-side program using the express library to create the RESTApis required for this application. I have selected to use NodeJS and Express strategically, since the front end is also written in the JavaScript language. This reduces the complexity of the overall application by sticking to only one language for both client and server sides of the application.

For the Database, I will use either opensource database such as Mongodb or PostgreSQL. Both databases are optimal options with both pros and cons. By using a relational data Database such as PostgreSQL, I can create a schema that connects user account information and ID to their personal progress, such as their weight over time, and their workout improvements as well. Mongodb is also an optimal choice, considering that mongo cloud (ATLAS) makes connecting to a database extremely easy. Using MongoDB also allows for more validation in the server side. Using NoSQL databases also makes for very quick prototyping and data creation.

Hosting:

For the service layer, I am considering using AWS to host my application on a remote server. By creating a Linux instance, I can quickly download all the dependencies required to run the program, and then run the server. Using Jenkins, I can create a CI/CD pipeline to update the project as I create the changes, while relying on Jenkins to catch build errors. This will be an optimal route, since I have full control of the remote server, and how to configure the Linux instance, as well as the settings for Jenkins. If this turns out not to be a viable option, I will use an easier configurable hosting service such as Heroku.

Data overview:

The Data for this application can be divided into application provided information, and the user generated data. Application provided data will be the workout/virtual trainer videos. It is important to keep these in the Database to reduce the size of the application that the user must download to their physical memory, especially since video files can quickly take up a lot of space. This will also improve the performance of the application by reducing loading times. Secondly, Data generated by the user. This data will be managed by a schema creating dependency called Mongoose. This library is perfect, as it allows for data and data types to be checked before they are added to the database. This will reduce the number of errors to be added into the database. Using MongoDB also allows you to reference other schemas as a data point, which will connect user information to their workout and weight data.