Hector Alvarez  
FallB Oct 27th 2021  
Week2 Capstone

Database Design:

For the Database, 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 a library called Mongoose, I will make user schemas that can validate data types, and general information.

The user schema will require 4 fields, which are first name, last name , email and password. Using a validator library, we can check email and password fields as the user is creating their account to let them know if the fields are up to standard. Validator is also a good library to use in the schema creation, since we can send back error messages according to what information is incorrect in the input (for example the type, length of the strings, if they contain certain words ).

Additionally, users will be able to add new data points for their weight to their account. This will require another schema but can be related to another schema using Mongoose references. This collection will contain all users weight data points, but will be found by the users ID being the reference point. This collection will contain values for weight , units (lbs or kg , lbs by default) and will automatically generate a time stamp (required for the application for data analysis).

Users will also have access to add their new workouts and progress stats to the database. This collection will require the muscle group that the user worked out, the weight lifted , the units, and will again create a time stamp for the data analysis portion of the application.

Finally, the database will be able to save video / virtual trainer videos for the application. This will just contain video files and images.

I have started working on the backend, and created some preliminary design of the User schema (in Mongo dB document form) will look like:

{

fname: {

type: String,

required: true,

trim: true,

},

lname: {

type: String,

required: true,

trim: true,

},

age: {

required: false,

type: Number,

},

email: {

type: String,

required: true,

trim: true,

lowercase: true,

validate(value) {

if (!validator.isEmail(value)) {

throw new Error("Email is invalid");

}

},

},

password: {

type: String,

required: true,

minLength: 8,

trim: true,

validate(value) {

if (value.toLowerCase().includes("password")) {

throw new Error('Password cannot contain word "password" ');

}

},

},

});

Preliminary design for aggregating new user weight data point will look like :

weight: {

type: Number,

required: true,

},

units : {

required :true,

type : String

},

user: {

type: mongoose.Schema.Types.ObjectId,

required: true,

ref: 'User'

},

datecreated : Date

}

And finally, user adding a new workout:

{

musclegroup: {

type: String,

required: true,

},

exercise: {

type: String,

required: true,

trim: true

},

force: {

type: Number,

required: true

},

units: {

required: true,

type: String

},

user: {

type: mongoose.Schema.Types.ObjectId,

required: true,

ref: 'User'

},

datecreated: Date

}, {

timestamps: true

}