**Note: For error output, should I change message to error?**

**Updates: removed salt from user model, changed some required inputs**

**user model:**

\_id: type: ObjectId; notes: automatically created, don’t need to keep track of this

username: type: String; notes: required, unique

email: type: String; notes: required, unique

fullName: type: String

password: type: String; notes: required

admin: type: Boolean

createdAt

updatedAt

**API calls for user model:**

**url/registerUser:**

HTTP request: POST

required inputs: username, email, and password

optional inputs: fullName

successful output: username and message

error output: message

**url/loginUser:**

HTTP request: POST

required inputs: username and password

optional inputs: none

successful output: token, username, email, fullName, createdAt, updatedAt, and message

error output: message

**url/editUser:**

HTTP request: PUT

required inputs: token

optional inputs: email, fullName, and password

successful output: token, username, email, fullName, and message

error output: message

Note: only pass in the optional inputs that need to be edited

**url/deleteUser:**

HTTP request: DELETE

required inputs: token, username, and password

optional inputs: none

successful output: username and message

error output: message

**url/getAllUsers:**

HTTP request: GET

required inputs: token from admin account

optional inputs: none

successful output: users array of all users

error output: message

**routine model:**

\_id: type: ObjectId; notes: automatically created, **need to keep track of this**

userId: type: ObjectId; notes: required

routineName: type: String; notes: required

coffeeNap: type: Boolean; notes: required

napTimer: type: Number

pomTimer: type: Number

breakTimer: type: Number

pomCount: type: Number

breakCount: type: Number

largeBreakCount: type: Number

goalHit: type: Boolean

createdAt

updatedAt

**How routine model works:** If the user is in coffee nap timer, when they press start timer, you will call createRoutine and pass in the token, “routineName”: “Coffee Nap”, and “coffeeNap”: true, and napTimer. Nothing else needs to be passed for coffee nap.

If the user is in pomodoro timer, when passing the JSON, coffeeNap should always be false and you don’t need to pass napTimer.

**API calls for routine model:**

**url/createRoutine:**

HTTP request: POST

required inputs: token, routineName, and coffeeNap

recommended inputs for **coffee timer**: napTimer

recommended inputs for **pomodoro timer**: pomTimer, breakTimer, pomCount, breakCount, largeBreakCount, and goalHit

successful output: token, routine (will contain what you passed in), and message

error output: message

**url/userRoutines:**

HTTP request: POST

required inputs: token

optional inputs: none

successful output: token, routines array, and message

error output: message

**url/editRoutine:**

HTTP request: PUT

required inputs: token and \_id (\_id of the routine to be edited)

optional inputs for **pomodoro timer**: pomTimer, breakTimer, pomCount, breakCount, largeBreakCount, and goalHit

successful output: token, routine, and message

error output: message

**Note: This api call is NOT for coffee naps**

**url/deleteRoutine:**

HTTP request: DELETE

required inputs: token and \_id (\_id of the routine to be deleted)

optional inputs: none

successful output: token, routineName, and message

error output: message

**url/deleteAllUserRoutines: ONLY USE THIS AFTER USER DELETES ACCOUNT**

HTTP request: DELETE

required inputs: token

optional inputs: none

successful output: deletedCount and message

error output: message

**url/getAllRoutines:**

HTTP request: GET

required inputs: token from admin account

optional inputs: none

successful output: routines array of all routines

error output: message

**How to test the API:**

**NOTE: You will need to install node.js**

Unzip focuspocus

Go to focuspocus on your console

C:\Users\Phillip\Desktop\Big Project\example 01.jpg

Type and enter: node index

You should see:

C:\Users\Phillip\Desktop\Big Project\example 02.jpg

Note: To exit: ctrl+c

If you want to test loginUser, use:

http://localhost:3000/loginUser

use the correct HTTP request and pass the required json

For whatever api call you want to use, the format is:

url/apiCall

example 1, if you want to login a user: HTTP request: POST: url/loginUser

example 2, if you want to delete a user: HTTP request: DELETE: url/deleteUser

**ADMIN USE ONLY API CALLS:**

url/getAllUsers and url/getAllRoutines

These two api calls can only be used with a token from logging into an admin account

The admin account’s details are in admin\_account.txt, located in the config folder on github