#30DaysOfCode | Backend Track

Day 18

mongo god

Using ExpressJs or the native Http module, write a server that takes in username, email and password from a SIGNUP route to create a user, and has a LOGIN route to authenticate the user using the password and the ***email*** *or* ***username****. As a step farther, reimplement the signup route to be able save the time and date the user signed up and have that data ready on a protected* ***''getuser"*** *route. The* ***"getuser"*** *route will take in the user's email. On a user's authenticated login through the* ***"login"****, return a session token (tokenize the email) as the response to make sure only an authenticated user with that token can access the* ***"getuser"*** *route; this token will be sent with the* ***"getuser"*** *request in the request header as the value of the "****Authorization****" key; it will have a "Bearer" prefix. The token will be verified on each* ***"getuser"*** *request to be sure the session token is valid and has not expired.*

As a follow up from previous work, create a local MongoDB instance and a user collection to store each user's details in the database and this time the **"getuser"** route will return a generatedObjectID along with the data previously stored in the database (the user should be retrieved with either email or username). Remember to use the Mongoose ORM (Object Relational Mapper). Enjoy :-)

>> All requests and responses in JSON.

>> Refer to Google, Youtube, StackOverflow and MongoDB and Mongoose Docs for help.

Request Format

Path: [GET]https://anyserver/getuser

Example:

Body:

> {

“email”: “david@aol.com”

}

Response Format

Path: [GET]https://anyserver/getuser

Example:

> {

“_id”: *sampleobjectId,*

“email”: “david@aol.com”,

“username”: “david”,

“date”: “09-04-2020”

“time”: “12:22pm”

}