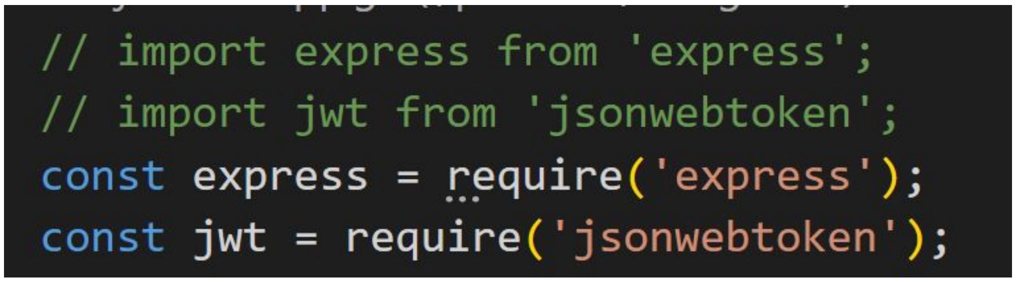
JavaScript, Node.js, MongoDB questions

Questions

1. If we try to run this application in a server with **Node.js 8** it will fail (we should change something in product.js).
    a. Why this code will not run?
    If we try to run the products.js on Node.js 8 it will fail because that version of node doesn't support the new ES6 syntax of imports.
    b. Make the necessary changes in *product.js* so that we can execute this in an environment with Node.js 8 installed.



```
// import express from 'express';
// import jwt from 'jsonwebtoken';
const express = require('express');
const jwt = require('jsonwebtoken');
```

2. Using MongoClient from https://www.npmjs.com/package/mongodb, write the code in *product.js* which creates a mongo client and saves it in the "mongodb" variable, that client will contact the database called: **behamics** in the mongo server with connection url: **mongodb://localhost:27017**.

You may find the code on the product.js file.

3. Suppose we have a collection in our mongo database called **product**, and a document in this collection will look like this:
Write a protected API (that is an API which uses authorization) in *product.js*, this API will respond to GET requests to the endpoint (path): **/product/:from/:to**. This API should get all products from **product** collection which are in the range [**from**, **to**], where **from** and **to** are parameters (string values of dates) sent from the client.

You may find the code on the product.js file.

4. Write the mongo db aggregation query which groups products by the **category** field and calculates the totalStock for each category from the sum of the **stock** field, and sorts the result by the **totalStock** field in descending order. Use mongodb JavaScript client.

You may find the code on the product.js file.

5. Suppose you have these two functions somewhere in your code:

- getUserById(id) - getProductById(id) both functions do a query in mongo and return a Promise object that resolves the result (resolves objects with user and product information respectively). Let's say you need to use these functions in some part of the application, how would you use **Promise.all** method in this case, to invoke both functions, considering they are independent of each other's result?

6. Let's say you're told you should create an index on **product** collection, which field or group of fields would you choose and why?

*I would choose to add an index on **name** and **category** as those two fields are probably the ones that will be used more frequently to perform queries over them and also the values on those fields have a higher cardinality.*