Lab3 - Express server

1- Provide REST API implementing CRUD operations on Inventory Items

- Use a filesystem, same as Lab01.
- You must use middleware to validate inputs.
- You must return descriptive status codes in case of errors.
- Routes:
    - GET /products
    - GET /products/1
    - POST /products
    - DELETE /products/1
    - PATCH /products/1
    - Add a functionality to restock and destock, design the route(s).

2- Provide a Static files server

- Download this image and serve it using express static.

3- Serve SSR html page on route GET /

- Render the current products list in styled components

Bonus:

1- Can filter products using queryparam
/products?status=””&category=””

Notes:

- **Make sure to try the debugging environment**
- **Make sure that you enable restart on change using nodemon**
- **Make sure to split your logic into modules**
- Please make sure that you separate your logic in functions
- Make the names of your functions and variables are expressive of what they are

- Start by defining each functionality and the steps you should make to achieve it
- Please make sure not to pollute the global scope[refrain from using global variables]
- Before writing a function, ask yourself, is there a function in javascript that can do that, if not, write your own.

Tips:

- If you can't do something, search, then ask for help, this is not an exam.
- Please please please, don't copy someone else's code.

Useful Links:

- npm | build amazing things
- Nodejs Docs
- Fs module docs fs (you will need to use the sync version of the functions)[ends with the word Sync]
- HTTP module to make your http server
- Nodemon for restarting node
- Headers - Web APIs
- Introduction to Node.js

Useful Readings:

- Build an HTTP Server
- An introduction to the npm package manager
- The V8 JavaScript Engine
- How the Web works - Learn web development

Must read before Day4:

- **Introduction to Mongoose for MongoDB**
- **REST API Best Practices – REST Endpoint Design Examples**
- **Authorization vs Authentication**
- **Hashing vs. Encryption vs. Encoding**

- JWT
- ES6