WEEK 3 Module 10

Express Framework: Web Serving Made Elegant

Creating the server

The first step to creating a server with express is to install the express framework itself. This can be achieved using

```
npm install express
```

Now, we can require express in our NodeJS app and instantiate the NodeJS Express server using

```
const app = express()
```

and we can listen at our desired port

```
app.listen(5000, ()=>{
console.log("server is listening at port 5000")
})
```

Routes, Methods, and Errors

The way routes are handles in Express is different from how we handle it in pure NodeJS server. It's simple and fast. Both routes and methods are handled together in Express. An example is

```
app.get('/', ()=>{
console.log('trying to access the homepage')
})
```

This is a basic route setup for the homepage using a GET method. Remember we talked about methods in module 3. You can checkout the module for more information on methods.

The common methods using in express are

app.get

app.post

app.put

app.delete

app.patch

app.all

Middleware and Serving NavyBary

In express, middleware allows us to perform certain function between the request and response layer of the server.

Express middleware are functions that execute during the lifecycle of a request to the Express server. Each middleware has access to the HTTP request and response for each route (or path) it is attached to.

To use a middleware in express, we use the method

app.use()

Keep in mind that **app** is simply the variable we used when we instantiated express, if you use a different variable, then everything else will use that variable. However, on a standard, the **app** variable is used.

Now, let's proceed to serving **NavyBary** with Express.

https://expressjs.com/en/starter/hello-world.html