Monday, July 25, 2022 10:10 AM

We are doing server side NODE !!!
--- create a folder myexpresswebserver
--- npm init
You get a package.json file in the folder
Package.json is your **project descriptor** - it is a JSON object that contains the info about your project

--- add the express library to it
npm install express (npm install command is used to install the node libraries from the net)
It will create a node_modules folder in the current folder
The node_modules folder contains express library and all its dependent libraries
The entry is added in the package.json as dependencies
The lock file is created --- so it keeps track of the exact version used in the project

- We started express web server
- We provided mappings for get requests
- We will set up Router and route modules on the server side

Express uses a **route module** to define SUB PATHS
For ex direct paths
http:localhost:5000/

http:localhost:5000/data

A sub path would be all paths under /book

http://localhost:5000/book

http://localhost:5000/book	/
http://localhost:5000/book/add	/add
http://localhost:5000/book/change	/change
http://localhost:5000/book/change	/delete

A sub path would be all paths under /math
http://localhost:5000/math

http://localhost:5000/math	/
http://localhost:5000/math/calc	/calc
http://localhost:5000/math/factorial	/factorial

To **return dynamic html content through express server** --- **TEMPLATE ENGINES** are used for dynamic html generation
Template Engines (expressjs.com)

Lets have a demo of hbs Template Engine !!!

1. Install the library for hbs ------
   npm install hbs (this will add the library in the node_modules and add it in package.json dependencies)
2. Create htmls using the syntax of hbs
   view folder >> first.hbs , greet.hbs
3. in the express server add the views and template engine so that the server knows which template engine is used and where to find its pages

```
app.set('views', './views');
app.set('view engine', 'hbs');
```

4. in the request mapping render the respective pages

```
app.get('/greet/:uname',(req,res)=>{
  let user1 = req.params.uname
  res.render('greet',{user:user1})  //MODEL AND VIEW
})
```

Node -----

1. Async Await in node ----------
   async = keyword in node
   this is used to define async functions

   --- **async functions are always returning a promise!!**

   await = keyword in node
   This is used to **block** the execution of the aync function till the **Promise resolves**
   We cannot use await in simple function, it can be used only in async function