Day 6: 14 Sep. 25

**Express JS :** Express JS is a third party module which internally use http module and provide extra features to develop web application using Node js. Even express js also known as Node js backend framework. Which internally follow MVC (model view controller), which support all http methods like get, post, put and delete. It will supporting routing features. Using express module we can create rest api very easily.

Node js provided lot of external web framework.

Creating simple web application using express js module

Create folder as **express module**

if you another sub folder you can create as express simple app

we can install third party module using

1. Locally : npm install modulename: if we install any third party module locally we need to install this module again and again in every project if you need.
2. Globally : npm install modulename -g

If we install any third party module globally no need to install that modules again and again.

Before install module we need to create the package.json file. This file contains your project configuration details as well as all modules which you use to develop the application with modules version.

Creating package.json file in node js project.

npm init (node package manager)

now we need to install express module

npm install express

install locally vs globally.

npm install : this command we need to run inside a project.

This command read all information from package.json file and whatever dependencies we use , it will install in the current project.

Creating the package.json file

npm init it will ask few option. You can provide or hit enter key then it consider default value.

npm install express

creating another project with Express JS with View as HTML pages

create folder as **express-with-html**

**npm init** it helps to create package.json file

**npm install express**

Get method

We can receive the value from client form using

request.query.formFieldName;

Post method

we can receive the value from client form using

request.body

by default express js doesn’t receive value from request body part.

We need to use another third party module ie bodyParser. Which help us to extract the value from request body part when request as post.

Old version express JS doesn’t provide bodyparser module with express. But new version whenever we install express js by default bodyparser module also install.

Old version express application we were installing bodyParser module separately.

**npm install body-parser**

first we need to load this module in our project.

Express js middleware modules. Middleware module add between client and server application.

Adding different middleware module base upon the requirement as

app.use(middlewareModule)

create new project as express-login-app help to do signIn and signUp with data stored in array variables.

npm init -y it create package.json file with default details.

npm install express installing express module

**app.js**

**index.html**

**signup.html**

**Web Service :** Giving the service for web application when both the application running using different technologies.

In Express Module or Http Module the view technologies is HTML. Html is static we can’t access any variable value which a part of Express Module or http module in html page.

Before web service

Express JS provided dynamic web engine or concept.

Creating express js project with **express view engine**

express-generator is a module which help to create express js project with jade as a dynamic view engine.

Express-generator is one of the third part module like create-react-app

npx express-generator **express-demo-app create project**

cd express-demo-app **move inside a project directory**

npm install **install all required dependency**

open the project in Vs Code

npm start

open the browser and write the URL as

<http://localhost:3000/>

Express JS provided lot of view engine. Every view engine follows its own syntax.

If we use normal HTML or any Express JS view engine as a View technology. That view technology tightly coupled with Express JS Application.

View technology backend technology

Angular Express JS

React JS Spring boot (java)

Asp.net

Php

Python

Spring boot or express

**Amazon**

Payment -🡪 Credit card

Debit card

Paypal asp.net

Paytm php

Gpay express js

PhonePay spring boot

JS object ------🡪 request

App1 XML/JSON App2

Express JS Java

🡨--------Java object –response

XML :eXtensible markup of language: data need to share in tag format.

JSON: Java Script Object Notation : data need to share in key/value format.

Web Service

1. SOAP base web service : Simple Object Access Protocol.

In SOAP Web Service we can consume and produce data only in the form of XML

SOAP Web Service follow its own message structure

In SOAP Web service we need to send and receive the data only in the form of XML with SOAP structure.

1. Rest full web service : Representation state transfer . In Rest full web service we can consume and produce data in any format base upon client requirement like JSON, XML, Plain text, html or any mediate type etc.

**Creating simple REST full Web service**

Create the folder as **Simple Rest API**

npm init -y

npm install express

Open the folder inside a VS Code

**Creating Rest API for Product with Array concept or file or database.**

**Array concept**

1. **Get method : get the resources** 
   1. Get all product in json format : select query
   2. Get particular product base upon some property like pid, pname, price
      1. Query param

URL?key=value

URL?key1=value1&key2=value2

* + 1. Path param

URL/value

URL/value1/value2

Select query with where clause

1. **Post method : create the resource** 
   1. store the product details in array or file system or database

insert query

1. **Put method : update the existing resource**
   1. Update the product price using pid

Update query

1. **Delete method : delete the resource** 
   1. Delete the product using pid

Delete query

**To test these end or rest api we need some rest api plugin.**

**Post Man plugin**

**Only Get method we can test through browser.**

**Creating REST API for product object with array concept**

Create the folder **product-rest-api**

**npm init -y**  create the package.json file

**npm install express**