**Roadmap**:

      - building a MERN stack app

                 -MERN: Mongo + Express +React +Nodejs

| nodejs |
| --- |
| React (is front end) |
| Express (web server and database is backend) |
| mongo database |

 Steps:

1. Backend intro  → Node + Express
2. React intro  →  front end
3. Build an API
4. build REACT front utilizing API
5. Authentication - passport

           Node.js:- we’ve seen js. Running in a browser.

                                 We’d like to create “real” apps that run on a computer.

           Solution: Node

                                     -node is a runtime environment for javascript

Setup:

            download + install Node;

           This installed node + npm

Node

          is the runtime

Npm

         node package manager.

         keeps track of packages used by app

         imports packages on demand

          creates initial app

Packages:-

                 Components used by app that perform certain functions

                   Examples:   Express       →   allow to build web server

                                      Mongoose    →   connect to MongoDB

                                       React           →   build front end

                                       Others VVE,ANGULAR

We develop in a dev environment and will eventually need to deploy(in production environment).

Npm can manage this b/c it keeps track of packages

* --package.json

For backend Dev.

1. Node
2. IDE-visual studio code or webstorm
3. Postman—-- for backend testing without having frontend

Create an app

1. made a root dir
2. in dir, npm init
3. create main .js file

      4.  Add packages ,etc.

      5. Npm install\_\_\_\_\_\_\_\_\_\_\_ - - save

<https://www.npmjs.com/package/cat-me>

npm install cat-me --save

npm install express --save

**Express:**

Create first express express

***Express is a web server package***

app.get(‘/’,function(req,res){                      | ←—--------------------- route

});

Req:- is request what is coming from browser

Res:- response ..going to send back or talk to browser

res.send();

REST:-

           Represtentational stage

Connect to DB.               Mongo/Express/

const express = require('express'); // same as create an instance

const app = express();              // same as new in Java

//set up app to listen for connection

app.listen(3010,function(){           // pick a port and set function to it

    console.log("Server started....")

});

node server.js

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

app.get('/', function(req,res){

    res.send("This is the home page")

});

node server.js

Graphical user interface, text

Description automatically generated

**//Other ROUTE**

app.get('/content', function(req,res){

    res.send("This is the content page")

});

* We have to run  node server.js in terminal

Graphical user interface, text, application

Description automatically generated

//another ROUTE

app.get('/content/:myparam', function(req,res){

    var myvar= req.params.myparam;

    res.send("You send " + myvar + " to the server")

});

* We have to run  node server.js in terminal

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

POSTMAN

Graphical user interface, application

Description automatically generated

**Today's Code:**

const express = require('express');

const app = express();

//root ROUTE

app.get('/', function(req,res){

    res.send("This is the home page")

});

//Other ROUTE

app.get('/content', function(req,res){

    res.send("This is the content page")

});

//another ROUTE

app.get('/content/:myparam', function(req,res){

    var myvar= req.params.myparam;

    res.send("You send " + myvar + " to the server")

});

//set up app to listen for connection

app.listen(3010,function(){

    console.log("Server started....")

});

**ClassmateNote**

Roadmap:- building a MERN stack app

                 -MERN: Mongo + Express +React +Nodejs

                  -mongo database

                  Express               web server and database is backend

                  React  —------is front end

                  nodejs

 Steps:-  1. Backend intro-Node + Express

               2. React intro:-       front end

               3. Build an API

               4.build REACT front utilizing API

              5.Authentication :passport

           Node.js:- we’ve seen js. Running in a browser.

                                 We’d like to create “real” apps that run on a computer.

           Solution:-Node

                                     -node is a runtime environment for javascript

Setup: -

            download + install Node;

           This installed node + npm

Node

          is the runtime

Npm-

         node package manager.

         keeps track of packages used by app

         imports packages on demand

          creates initial app

Packages:-

                 Components used by app that perform certain functions

                   Examples:- Express —------------> allow to build web server

                                      Mongoose—-----------> connect to MongoDB

                                       React—---------------->  build front end

                                       Others VVE,ANGULAR

We develop in a dev environment and will eventually need to deploy(in production environment).

Npm can manage this b/c it keeps track of packages

* --package.json

For backend Dev.

1. Node
2. IDE-visual studio code or webstorm
3. Postman—-- for backend testing without having frontend

Create an app

1.made a root dir

2.in dir, npm init

3.create main .js file

4. Add package ,etc.

5.Npm install\_\_\_\_\_\_\_\_\_\_\_ - - save

Express:-

Create first express express

    Express is a web server package

app.get(‘/’,function(req,res){                                                | ←—--------------------- route

});

Req:- is request what is coming from browser

Res:- response ..going to send back or talk to browser

res.send();

REST:-

           Represtentational stage

Connect to DB.               Mongo/Express/

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PS C:\Users\SDC101> cd G:

PS G:\> cd CWEB603

    Directory: G:\CWEB603

Mode                 LastWriteTime         Length Name

----                 -------------         ------ ----

d-----          3/8/2022   2:11 PM                intro

PS G:\CWEB603> cd .\intro\

PS G:\CWEB603\intro> npm init

This utility will walk you through creating a package.json file.

It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields

and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and

save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (intro)

version: (1.0.0)

description:

entry point: (index.js)

const { resolveSoa } = require('dns');

const express = require('express');

const app=express();

app.get('/',function(req,res){

    res.send("This is a home page");

});

//set up app to listen for connections

app.listen(3011,function(){

    console.log("Server started..");

});

const { resolveSoa } = require('dns');

const express = require('express');

const app=express();

//root route

app.get('/',function(req,res){

    res.send("This is a home page");

});

app.get('/content',function(req,res){

 res.send("This is the CONTENT page");

});

app.get('/content/:myParam',function(req,res){

    var myVar=req.params.myParam;

    res.send("You sent " + myVar + " to the server ");

});

//set up app to listen for connections

app.listen(3011,function(){

    console.log("Server started..");

});

Text, letter

Description automatically generated

Text, letter

Description automatically generated

Text, letter

Description automatically generated

A white paper with writing on it

Description automatically generated with low confidence