Angular Js

<!DOCTYPE html>

<html ng-app="QuizApp">

<head>

<meta charset="UTF-8">

<script src="app\lib\angular.min.js"></script>

<link href="style1.css" rel="stylesheet">

<script>

var MyApp=angular.module("QuizApp",[]);

MyApp.controller("QuizCt",["$scope","$http",function($scope,$http){

$scope.number=0;

$scope.Next=true;

$scope.b1=true;

$scope.Final=false;

$scope.Submit=false;

$scope.score=0;

$http({

method: 'GET',

url: 'data.json'

}).then(function successCallback(response) {

$scope.question = response.data;

}, function errorCallback(response) {

console.log(response);

});

$scope.increment=function(){

if($scope.ans.localeCompare($scope.question[$scope.number].answer)==0)

$scope.score+=1;

$scope.number+=1;

if($scope.number==$scope.question.length-1){

$scope.Next=false;

$scope.Submit=true;

}

if($scope.number>=$scope.question.length){

$scope.Final=true;

$scope.b1=false;

}

document.getElementById("myForm").reset();

}

}]);

</script>

</head>

<body ng-controller="QuizCt" >

<center><h1 style="color:brown">Quiz App</h1></center>

<div ng-show="b1" class="Qdiv">

<br><br>

<h2>Question: {{number+1}}</h2>

<br>

<h3>{{question[number].question}}</h3>

<br>

<form id="myForm" ng-submit="increment()">

<h3 ng-repeat="i in question[number].options">

<input type="radio" ng-model="$parent.ans" ng-value="i" name="ans" required> {{i}}

</h3>

<br>

<div ng-show="Next">

<input type="submit" value="next">

</div>

<div ng-show="Submit">

<input type="submit" value="Submit">

</div>

</form>

</div>

<div ng-show="Final" class="Score">

<h2>Your Score: {{score}} / 5</h2>

</div>

</body>

</html>

MyApp.component('calculatorList',{

template:

'<div>'+

'<input type="number" placeholder="Enter a number" ng-model="a">'

+'<input type="number" placeholder="Enter a number" ng-model="b">'+

'<select ng-model="operator">'+

'<option>+</option>'+

'<option>\*</option>'+

'<option>-</option>'+

'<option>/</option>'+

'</select>'+

'<p>Answer: {{result()}}</p>'+

'</div>',

controller: function($scope){

$scope.result=function(){

if ($scope.operator == '+') {

return $scope.a + $scope.b;

}

if ($scope.operator == '-') {

return $scope.a - $scope.b;

}

if ($scope.operator == '\*') {

return $scope.a \* $scope.b;

}

if ($scope.operator == '/') {

return $scope.a / $scope.b;

}

}

}

})

Form

<form method="post" onsubmit="validateForm()" action="http://127.0.0.1:5501/index.html#!/Home" name="myForm1" novalidate>

                    <div class="input-field">

                        <!-- <label id="Lgl" style="color:red; visibility: hidden;">Invalid</label> -->

                        <input type="email" placeholder="Email ID" name="Uem" ng-model="Uem" required>

                        <i class="uil uil-user"></i>

                    </div>

                    <div>

                        <span style="color:rgb(223, 15, 46)" ng-show="myForm1.Uem.$dirty && myForm1.Uem.$invalid">Invalid Email</span>

                        <span style="color:rgb(223, 15, 46)" ng-show="myForm1.Uem.$dirty && myForm1.Uem.$error.required">Email required</span>

                    </div>

                    <div class="input-field">

                        <!-- <label id="Lgl" style="color:red; visibility: hidden;">Invalid</label> -->

                        <input type="password" class="password" name="LgPassword" placeholder="Enter your password" ng-model="LgPassword">

                        <i class="uil uil-lock icon"></i>

                    </div>

                    <div>

                        <span style="color:rgb(223, 15, 46)" ng-show="myForm1.LgPassword.$dirty && myForm1.LgPassword.$error.required">Email required</span>

                    </div>

                    <div class="input-field button">

                        <input type="submit" value="Login Now" class="Btnn" ng-disabled = "(myForm1.UEmail.$dirty &&

                        myForm1.UEmail.$invalid) || (studentForm.lastname.$dirty && myForm1.LgPassword.$invalid)">

                    </div>

                </form>

Express Js:

//cookie check it in console document.cookie

//res.send we can place any url

//next is used for more than one callback functions

//param is a callback function

//for ejs use mostly or set we just directly call the file as file.js and then passs parameters.

//for router we need to export the file and here we need to use the app with routing path and name of the router acquired.

// / represents index file

const express=require('express')

const app=express()

//This use is for middleware in express everything is top to down parsing so we can put middleware wherever we want and all below will have access to it.

app.use(logger)

// For middleware to be accessed only to a spcific get we can do it like this

//app.get('/',logger,(req,res)=>{

//})

// Another method to render static html files is express.static it is a middleware that give access to all the files metioned inside the folder.

app.use(express.static("views"))

//the above method shows we can put multiple functions using ,

// app.set('view engine','ejs')

// app.param('id',(req,res,next,id)=>{

// console.log(id);

// next()

// })

//app.locals.tex="Hey"

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

// res.send('hey ')

// res.status(500).send('hey')

res.statusCode(500);// only code--message related to code

// res.download('server.js')

res.render('index.ejs',{text1:"new"})

// const rt=req.baseUrl/

// res.send('base url')

// res.json({name:'NameUser'})

// var urlc=url.parse('/hello?name=hey')

// var c1=urlc.query

// res.send(c1.name)

// res.redirect('/user')

// res.set({

// 'Content-Type': 'application/json'

// });

// res.cookie('hey','Hello',[]); []-options eg expiry,session,domain etc.

// res.send(res.get('Content-Type'))

// //Whateve we type afetr url ?category= will be displayed

// res.send(req.query.category)

// Regex write url /:id([0-9]{2})

// res.send(req.params.id); or res.send(req.param('id'));

// app.param -- this is called when a route is pointing to it

console.log("heiugef");

// res.send("Hey");

next()

},function(req,res){

res.send("Hello")

})

app.route('/').get((req,res)=>{

res.render('index.ejs',{text:"new",t1:"Hey"})

}).post((req,res)=>{

res.send("Heooloo")

}).put((req,res)=>{

res.send("Boook")

})

//middleware -- that runs betwenn when the request isent and before the response

function logger(req,res,next){

console.log(req.originalUrl)

next()

}

const UserRouter=require('./routes/users');

app.use('/users',UserRouter)

app.listen(3000)

Routes File.

const express=require("express")

const router=express.Router()

router.get('/',(req,res)=>{

res.send('User list')

})

router.get('/new',(req,res)=>{

res.send('User form')

})

module.exports=router

views folder:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

<body>

<form action='/' method="POST">

<label>Name</label>

<input type="text" name="Fvalue" value="<%locals.FirstName%>">

<input type="Submit">

</form>

</body>

</html>

MongoDb

const express=require('express');

var mongoose=require('mongoose');

var bodyparser=require('body-parser')

var cors=require('cors')

var path=require('node:path/win32')

const app=express();

const route=require('./routes/route')

const PORT=4000

mongoose.connect("mongodb://127.0.0.1:27017/contactlist",{useNewUrlParser:true}).then(()=>

console.log('MongoDb connection succeeded.')).catch((err)=>{

console.log('Error in DB connection: '+ JSON.stringify(err,undefined,2));

});

// mongoose.connect("mongodb://localhost:27017/contactlist");

// mongoose.connection.on('connected',()=>{

// console.log('Connected to database mongodb @')

// })

// mongoose.connection.on('error',(err)=>{

// console.log(err);

// })

app.get('/',(req,res)=>{

res.send("Hello world");

});

//Middleware

app.use(cors());

app.use(bodyparser.json());

app.use(express.static(path.join(\_\_dirname,'public')));

app.use('/api',route);

app.listen(PORT,()=>{

console.log(`App listening on port ${PORT}`);

});

// const mongoose=require('mongoose');

// mongoose.connect("mongodb://localhost:27017/CrudDbb",{useNewUrlParser:true}).then(()=>

// console.log('MongoDb connection succeeded.')).catch((err)=>{

// console.log('Error in DB connection: '+ JSON.stringify(err,undefined,2));

// });

// module.exports=mongoose;

mongoDb

var express=require("express");

var router=express.Router();

const { body, validationResult } = require('express-validator');

//get model of Page

var Page=require("../models/page");

/\*

GET pages index

\*/

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

// Page.find({}).sort({sorting:1}).exec(function(err,pages){

// res.render("admin/pages");

// pages:pages;

// });

res.send("admin area");

})

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

// Page.find({}).sort({sorting: 1 }).exec(function(err, pages) {

// if (err) {

// console.error(err);

// // Handle the error, e.g., send an error response or redirect

// } else {

// res.render("admin/pages", { pages: pages });

// }

// });

// });

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

// Page.find({}).sort({ sorting: 1 }).exec()

// .then(pages => {

// res.render("admin/pages", { pages: pages });

// })

// .catch(err => {

// console.error(err);

// // Handle the error, e.g., send an error response or redirect

// });

// });

/\*

GET add page

\*/

router.get('/add-page',function(req,res){

var title="";

var slug="";

var content="";

res.render('admin/add\_page',{

title:title,

slug:slug,

content:content

});

});

/\*

POST add page

\*/

// router.post('/add-page',function(req,res){

// req.check('title','Title must have a value').notEmpty();

// req.check('content','content must have a value').notEmpty();

// var title=req.body.title;

// var slug=req.body.slug.replace(/\s+/g,'-').toLowerCase();

// if(slug == ""){

// slug=title.replace.replace(/\s+/g,'-').toLowerCase();

// }

// var content=req.body.content;

// var errors=req.validationErrors();

// if(errors){

// res.render('admin/add\_page',{

// title:title,

// slug:slug,

// content:content

// });

// }

// else{

// console.log('sucess');

// }

// // res.render('admin/add\_page',{

// // title:title,

// // slug:slug,

// // content:content

// // });

// });

router.post('/add-page', [

body('title').notEmpty().withMessage('Title must have a value'),

body('content').notEmpty().withMessage('Content must have a value'),

], async function(req, res) {

var title = req.body.title;

var slug = req.body.slug.replace(/\s+/g, '-').toLowerCase();

if (slug === "") {

slug = title.replace(/\s+/g, '-').toLowerCase();

}

var content = req.body.content;

var errors = validationResult(req);

if (!errors.isEmpty()) {

console.log('error');

res.render('admin/add\_page', {

title: title,

slug: slug,

content: content,

errors: errors.array()

});

} else {

try {

console.log(slug);

const page = await Page.findOne({ slug: slug });

console.log(page);

if(page!=null) {

req.flash('danger', 'Choose another slug');

return res.render('admin/add\_page', {

title: title,

slug: slug,

content: content,

});

}

const newPage = new Page({

title: title,

slug: slug,

content: content,

sorting: 100

});

await newPage.save();

req.flash('success', 'Page added');

// Clear the input fields

title = '';

slug = '';

content = '';

// res.redirect('/admin/pages');

} catch (error) {

console.log(error);

// Handle the error

}

}

});

module.exports=router;

var express=require("express");

var path=require("path");

var mongoose=require("mongoose");

var config=require("./config/database")

var bodyParser = require('body-parser')

var session=require("express-session")

// var expressValidator=require("express-validator");

const { body, validationResult } = require('express-validator');

//db connection

mongoose.connect(config.database);

var db=mongoose.connection;

db.on('error',console.error.bind(console,'connection error:'));

db.once('open',function(){

console.log("connected!!");

})

//app initialization

var app=express();

app.set("views",path.join(\_\_dirname,'views'));

app.set("view engine",'ejs');

//public folder

app.use(express.static(path.join(\_\_dirname,'public')));

//gloval

app.locals.errors=null;

app.use(bodyParser.urlencoded({ extended: false }));

// parse application/json

app.use(bodyParser.json());

//express-session

app.use(session({

secret: 'keyboard cat',

resave: false,

saveUninitialized: true,

cookie: { secure: true }

}))

//express valid

// app.use(expressValidator({

// errorFormatter:function(parm,msg,value){

// var namespace = parm.split('.'),

// root=namespace.shift(),

// formParm=root;

// while(namespace.length){

// formParm+='[' + namespace.shift() + ']';

// }

// return {

// parm: formParm,

// msg: msg,

// value:value

// };

// }

// }));

//mesage

app.use(require('connect-flash')());

app.use(function (req, res, next) {

res.locals.messages = require('express-messages')(req, res);

next();

});

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

// res.render('index',{

// title: "Home"

// });

// })

//set routes

var pages=require("./routes/pages");

var adminPages=require("./routes/admin\_pages");

app.use('/admin/pages',adminPages);

app.use('/',pages);

var port =3000;

app.listen(port,function(){

console.log("server started on port"+port);

})