Tech Info it   
Node Js express js tutorials  
[***https://www.youtube.com/watch?v=26QzZ2Zz99E&list=PLuHGmgpyHfRx4rP\_5VdE33LNKrYe4FvQw***](https://www.youtube.com/watch?v=26QzZ2Zz99E&list=PLuHGmgpyHfRx4rP_5VdE33LNKrYe4FvQw)  
  
1.   
[***https://expressjs.com/***](https://expressjs.com/)Rest Api kya hoti hai - protocol - set of rules – to integrate our application - for example backend ko frontend se connect krna   
client : frontend – server – node – backend : express ki help se server rest api create krenge – es rest api se client ko server se connect krenge  
or express – server ko database se bhi connect krenge, server(express, database – mongodb)  
rest api – client ko server se connect and request and response receive hoga.  
  
Express JS – creating Apis   
  
2. **Package.json**  
initialize node package manager - npm init – create package.json file  
licence – MIT – Open Source  
npm I express – node module package download  
  
**3. server.js**  
const express = require(‘express’);

//rest object  
const app = express();  
//route  
app.get(“/”, (req, resp) => {

Resp.send(“<h1>welcome to tech info it<h1>”);

});  
const PORT = 5000;  
app.listen(5000, () => {

Console.log(`server running on ${PORT}`);  
})  
  
run : node server.js  
  
**nodemon - to run**run – nodemon server.js

Localhost:5000**package.json***production me nodemon ki jarurat nhi hai   
we can add multiple script  
script : {*

*“start”:”node server.js”,  
“server”: “nodemon server.js“*

*}***4. Post Request with Postman Tool and Middlewares**

const express = require(‘express’);

//rest object  
const app = express();  
//route  
app.get(“/”, (req, resp) => {

Resp.send(“<h1>welcome to tech info it<h1>”);

});

app.post(“/contact-form”, () => {  
resp.send(“form submitted”);  
});  
const PORT = 5000;  
app.listen(5000, () => {

Console.log(`server running on ${PORT}`);  
});  
  
localhost:5000/contact-form  
can not get /contact-form

Post request ke liye hume koi na koi client pe depend rhna hota hai  
agar hume es request ko access krna hai to koi bhi client react , angular, vue ya phir koi bhi template engine ka use , and usko help se data ko send – post request

Rest api ko check krne ke liye insomnia , postman  
  
//middleware  
app.use(express.json());  
  
app.post(“/contact-form”, () => {  
resp.json({  
success : true,  
message:”welcome to techinfo it”  
});  
});

To parse json data use middleware  
  
we can also use body parser  
third paty package  
<https://www.npmjs.com/package/body-parser>  
const bodyparser = require(“body-parser”);  
app.use(bodyparser.urlencoded({extended : false}));  
  
Post Route - postman – testing – method – post   
localhost:5000/contact-form  
output   
({  
success : true,  
message:”welcome to techinfo it”  
})  
  
Post me form data rhta hai , User humko form data bhejega usko humko access bhi krna hota hai, kaise access krna hota hai   
  
Req Object : req object ki help se data ko get kr skte hai data ko   
user se data ko extract krenge   
  
app.post(“/contact-form”, (req, resp) => {  
 const {inputData} = req.body;

resp.json({  
success : true,   
message : `Welcome to ${inputData}`  
})

});  
  
Postman – method post - now we have to provide input data also  
localhost:5000/contact-form  
body : body- raw - textdata - json   
//json data , key value pair

Getting data from the user  
{

“inputData” : “xyz”

}  
  
**Middleware**//Morgan   
<https://www.npmjs.com/package/morgan>

Api ke bare me extra details chahiye ki urls sahi se hit ho rha hai ya nhi , kitna time le rha , kitna time lg rha hai ,   
  
install -npm install morgan   
Note : every third party package we need to import   
const express = require(“morgan”); //morgan dose not support ES6 concepts, we need to use ES5 concepts   
  
app.use(morgan(“dev”));

  
  
method – post

End point - /contact-form  
status- code – 200  
time – 4.708 ms – 48

Package : colours   
<https://www.npmjs.com/package/colors>

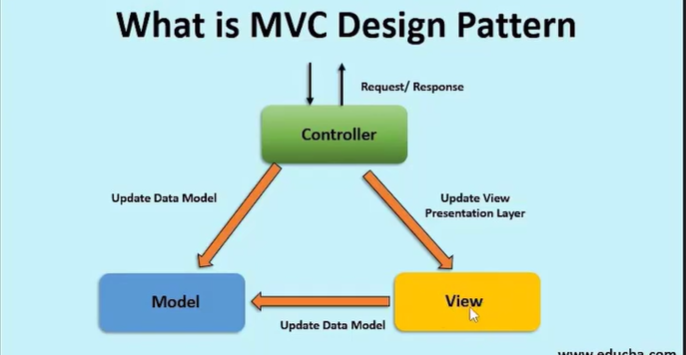
const colors = require(“colors ”)  
  
how to use   
app.listen(PORT, () => {

console.log(`$ {PORT}.red`);  
console.log(`$ {PORT}.bgCyan`); //to change background color

});

**5. MVC Pattern Converting Our App To MVC**

Design pattern



**Models  
views  
controller  
Routes:   
Server.js**

Controller/userController.js  
Routes/userRoute.js **Routes/userRoutes.js**const express = require(‘express’);  
//routes router object   
const router = express.Router();  
//routes  
router.get(‘/’, (req, resp) => {

resp.send(“<h1>welcome to tech info it”);

});  
//export router  
module.exports = router; **server.js**const express = require(“express”);

const userRoutes = require(‘./Routes/userRoutes’);

const app = express();

**//shorts cuts directly middleware add kr skte hai – another way to write this**app.use(‘/user’, require(./routes/userRoutes)); **run : npm run server**

localhost:5000/user  
welcome to techinfo it

we can use this callback function inside controller   
router.get(‘/’, (req, resp) => {

resp.send(“<h1>welcome to tech info it”);

});

**Controller/userController.js**const getUser = (req, resp) => {

resp.send({

success : true,  
message : “data coming from MVC pattern”

})

};

module.exports = {getUser};  
  
we need to import this controller file in userRoutes file  
**Routes/userRoutes.js**const {getUser} = require(../controller/userController.js)router.get(“/”, getUser);  
  
run   
localhost:5000/user  
success : true,  
message : “data coming from MVC pattern”

**server.js**whenever we are creating api recommended way to use below like this  
app.use(“/api/v1/user”, require(“./routes/userRoutes”));  
  
run   
localhost:5000/api/v1/user  
success : true,  
message : “data coming from MVC pattern”

same concepts applying for contact form  
**userRoutes.js**//contact form

const {addUser} = require(“../controllers/userControllers”);  
router.post(“/add-user ”, addUser);  
  
**controllers/userControllers.js**  
const addUser = (req, resp) => {

const {inputData} = req.body;  
resp.send({

success: true;

})

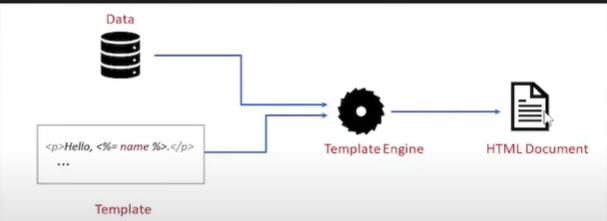
};  
module.exports = addUser;  
  
**Server.js**app.use(‘/api/v1/user’, require(‘./Routes.userRoutes.js’));  
note : hume multiple route create krne ki jarurat nhi padegi , kyuki ye route add ho chuka hai,

Jitne bhi user se related end point create krna hai hum routes/UserRoutes.js me create kr skte hai   
  
**Testing Postman -**   
localhost:5000/api/v1/user/add-user  
body- raw – json   
{

“inputData” : “tech info It ”

}  
doubt little bit about urls end point for post

**6. Template Engine example with PUG in node js   
views –** Jo bhi hamara UI wo es views folder ke undar rhega,

Yha pe jo views hai , html ka code server ki files me hai,   
to es server side ui ko   
run krne ke liye humko ek templateengine lgta haifile me jitna bhi ui rhega wo**,** wo templte engine ke through execute hoga  
****  
  
full stack application hume sirf node js and express ki help se bnana hai tabhi hum template engine ka use krenge  
  
node js , express js : mostly use to create rest apis   
client side ke liye : react js, angular, vue js, normal J query use krte haiii  
  
PUG

EJS

pug

npm install pug  
<https://www.npmjs.com/package/pug>  
views/firstfile.pug  
doctype html  
html   
 head  
 title = “PUG template engine example”  
 body   
 h1#greet.heading welcome to express PUG example   
now we will add above files in server.js

server.js  
//view engine  
app.set(‘view engine’, ‘pug’);  
app.set(‘’views’, ‘./views’);  
  
//view routes  
app.get(“/”, (req, resp) => {  
resp.render(“firstFiles”);  
});  
  
run : npm run server  
localhost:5000  
welcome to express pug example   
  
static websites create krna hai to eska estemal kr skte hai   
  
node js express js : rest apis

**7. mongodb database connection with node express js**[**https://www.mongodb.com/products/platform/atlas-database**](https://www.mongodb.com/products/platform/atlas-database)

Download and install it   
or   
we can use mongodb atlas  
mongodb string   
follow steps to create connection string  
create projects  
database – create cluster  
database access   
network access   
database – connect

We can use mongodb as well but wo kafi lenthy process hai , esliye mongoose use krna sahi hota hai   
  
**config/db.js**  
<https://www.npmjs.com/package/mongoose?azure-portal=true>

npm I mongoose  
const mongoose = require(‘mongoose’);  
//function connect mongo db database – es function ke undar promiss ko handle krna hai , promiss ko kyu handle krna hai, server pe jo hai , ek server se dusre server me connect kr rhe hai . to network request yha pe bhejh rhe hai , uske aane me means us response ko handle krne me hum promises ya async, await ka use krte hai   
1. Promises then chain block  
2. Async, await

const connectDb = () => {

try{

const conn = await mongoose.connect(‘ mongodb://localhost:27017/node-expressjs’);

console.log(` successfully connected to the data base : ${conn.connection.host}`);  
}

catch (error){  
console.log(‘ ${error.message}’);

process.exit(1);

}  
}  
module.exports = connectDb;  
  
**Server.js**const connectDb = require(‘./config/db’)

//mongodb function call   
connectDb();

Run : npm run server  
successfully connected to the data base : gkugiugymnbk

now we can perform crud operation in mongodb database (mongoose)

8. **nodejs Mongodb Crud Operation**mongo db database – no sql database –

Creating a documents   
we need to design a schema   
  
sql – table create

Application jo hai mongodb://localhost:27017/node-expressjs’ es port par jo hai connected hai, database es port pe connected hai ,   
  
hamare database ka name node-expressjs hai   
mangodb atlas me es database name (node-expressjs) se koi bhi database nhi milega, kyuki connection to stablish ho chuka hai , lekin ye tabhi show hoga jab hamare pass kuch na kuch document rhega, us database me   
kyuki nosql database kuch es tarike se kam krta hai ,  
  
whereus sql data base ki baat kare to , wha par table empty bhi rkh skte hai start me or   
database directly create krke rkh skte hai ,

Lekin nosql me humne agar koi bhi database create kar liya hai to uske undar kuch na kuch document hona chahiye tabhi wo mongodb server par database create hoga ,  
  
authentication , cookies, session etc  
**config/connectDb.js**

**controllers  
models**

**models/ userModel.js**  
schema design – schema ek json object hai – jiske undar hum bta skte hai ki , koun si chijhe hum receive krenge , for example for user – name, email, password, authentication, admin   
basic structure  
const userSchema = mongoose.schema({

});  
const User = mongoose.model(‘User’, userSchema);module.exports = User;

collection User name se create hoga by default wha par s lg jayega

const userSchema = mongoose.Schema({

name : {

type : String ,  
require : [true, ‘name is required’]

},

email : {

type : String,  
require: [true, ‘email is required’]

},  
password : {

type : String,  
require: [true, ‘password is required’ ]

},

isAdmin : {

type : Boolean,

default : false

}

}, {timestamps:true});//timestamps – kab koun sa user create hua hai – date and time   
const User = mongoose.model(‘User’, userSchema);

module.exports = User;

**routes/userRoutes.js**

const express = require(‘express’);  
const {createUser} = require(‘../controller/userController’);

const router = express.Router();  
//create new user  
router.post(‘ /create-user’, createUser);  
 // /create-user – yha pe kuch bhi likh skte hai   
  
**controller/userController.js**

const express = require(‘express’);  
constcreateUser = async(req, resp) => {

try {

const {name, email, password} = req.body;

}

catch (error) {

console.log(`${error}`)

}

}  
module.exports = createUser;

//proper error handling ke liye hum try catch block ka use krte hai   
postman – testing, method – post  
req – jo bhi chije hume user se chahiye , to hum request ki help se , access kr skte hai,  
agar jo bhi user ko response bhejhna hai , es response ki help se send kra skte hai

req.body  
req.header  
request me do hi chije hoti hai, req.body , req.header section  
header – headers ke undar meta data hota hai , man lijiye hamare pass kya hai , kuch tokens hai, to wo kya hoga , wo header section me chla jayega,   
header to header se destructure krna pdega   
req.headers

agar hamare pass visible data hai , mal lijiye hamare pass , user se  
 input lene wale hai, to ye body section me jayega , abhi hamare pas body se hai to ,  
body se destructure krenge , req.body  
  
kya kya destructure krenge ,aise time me schema ka use krenge   
here : name, email, password, isAdmin

const {name, email, password} = req.body;

yha se humne destructure kr liya , sara ka sara data get kr liya , ab eska krna kya hai,   
eski help se database me esko store krenge   
  
user model ka reference humko add krna pdega , eski help se user ko create krenge   
  
**controller/userController.js**

const express = require(‘express’);  
constcreateUser = async(req, resp) => {

try {

const {name, email, password} = req.body; //use schema here

}

catch (error) {

console.log(`${error}`)

}

}  
module.exports = createUser;

//for error handling hum try , catch block ka use krte hai   
  
request – jo chije hume user se chahiye wo request ki help se get kar skte hai  
  
response – response bhejhna hai to es response ki help se send kra skte hai

**//store in databases  
controller/userController.js**

const express = require(‘express’);  
const userModel = require(‘../models/userModel ’);  
constcreateUser = async(req, resp) => {

try {

const {name, email, password} = req.body; //use schema here

//store in database

const user = await UserModel.create({

name, email, password

});  
resp.status(201).json({  
message : ‘success’

user,

});

}

catch (error) {

console.log(`${error}`);  
resp.status(400).json({  
message : false,   
error  
});

}

}  
module.exports = createUser;  
  
run : npm run server  
  
Postman – Testing –

Create user, method – post

<http://localhost:5000/api/v1/user/create-user>

POST request : body-raw-json

Json formate   
{

“name”:”monika”,  
“email”:”monika@gmail.com”,  
“password”:”moni@12345”,  
isAdmin:false

}

Output – postman :  
we are getting response successfully along with the id

Create ka functionality successfully work kar rha hai

Operation – create operation done

9. **Nodejs Mongodb Crud Read Operation with Dynamic Routing**

**read operation**

previous video me – user collection ke undar document humne create ki thi

validation ki also important

routes/userRoutes.js

const {getAllUsers} = require(‘../controller/userController’);

//get all users

router.get(‘get-users’, getAllUsers);

controller/userController.js

const getAllUsers = async(req, resp) => {

try {

const users = await UserModel.find({}); //find all users empty object  
resp.status(200).json({

success : true,  
users

})

}

catch (error) {

resp.status(400).json({

success : false,  
msg : “get all user error”,

error : error.message

})

}

};

module.exports = getAllUsers;

Testing – Postman

Method – GET  
  
localhost:5000/api/v1/user/get-users

Success hume sare data mil rhe hai

Unlimited – n number of records , field – id, id ke base pe specific document ko search kr skte hai

**routes/userroutes.js**

//get single user – dynamic route id ke base pe routing perform krte hai

const { getSingleUser } = require(‘../controller/userController’);

router.get(‘/get-users/ : id ’, getSingleUser);  
  
controller/userController.js

//get single users

const getSingleUser = async (req, resp) => {

try {

const user = await UserModel.findById({req.params.id}); // id user will provide

if(user){  
resp.status(200).json({

success : true,  
user,

});

}

else {

resp.status(400).send(‘User Not Found’);

}

}

catch (error) {

resp.status(400).json({

success : false,

message : ‘single user error,

error : error.message

})

}

}  
module.exports = getSingleUser;

Postman – Testing : getSingleUser  
Method – Get  
localhost:5000/api/v1/user/get-users/useriduseriduserid

If userid match then success  
if not match then error

**10. Nodejs Mongodb Crud Operations - Update Delete Documents**update using id -   
  
**routes/userRoutes.js**//update user

Const updateUser = require(‘../conteroller/updateUser’);  
router.put(‘/update-user/:id’, updateUser);

controller/userController.js

const updateUser = async ( req, resp ) => {

const user = await UserModel.findById(req.params.id);

if(user){

user.name = req.body.name || user.name   
user.email = req.body.email || user.email

if(req.body.password){

user.password = req.body.password

}

const updatedUser = await user.save();  
resp.status(201).json({

success : true,  
\_id = updateUser.\_id,  
name : updatedUser.name  
email : updatedUser.email,

isAdmin: updatedUser.isAdmin

})

}  
else{

resp.status(400);

throw new Error(“user not found”);

}

}

module.exports = updateUser;

Testing – Postman – body – raw - jsonlocalhost:5000/api/v1/user/update-user/idididididmnbjkgi  
Method – Put

{

“name” : “gerger”,  
“email” : “ger@gmail.com”,

“password” : etebfhrty“”

}

**Delete request**

**routes/userRoutes.js**

//delete routes

router.delete(‘delete-user/: id’, deleteUser);

const deleteUser = async(req, resp) => {  
const user = await UserModel.findById(req.params.id);  
if(user){

await user.remove();  
resp.status(200).json({

success:true,  
message : ‘User Deleted’

});

}  
else{

resp.status(400);  
throw new Error(“User Not Found”);

}

};

**Postman – Testing :**

method – delete  
localhost:5000/api/v1/user/delete-user/ididididididididididididdididididid

***@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@***

***Code step by steps***[*Sheryians Coding School*](https://www.youtube.com/@sheryians)

***Express.js - Learn What Matters: Mastering the Framework | Backend (Node JS) Series***

***playlist***

[***https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3\_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3***](https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3)

[***https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL***](https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL)

[***https://youtu.be/T55Kb8rrH1g?si=lZjEEXDA9IHa5SDh***](https://youtu.be/T55Kb8rrH1g?si=lZjEEXDA9IHa5SDh)

***1.*** [***https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3\_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3***](https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3) ***install node js   
import export in node js  
install any package using npm  
  
  
2.*** [***https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL***](https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL)

***Node js , express js   
routing  
middleware  
req, resp  
route , parameter  
template  
static files   
http method – get and post  
error handling  
  
Routing -   
routing means urls ,   
localhost:5000/about***npm I express  
[***express - npm (npmjs.com)***](https://www.npmjs.com/package/express)