Q. What is Node?

1. Node is an open source server environment and it is used for develop the server side application as well as dynamic web application by using JavaScript library or frameworks
2. Node js runs on various platform (Window, linux ,mac etc)

Note: using node js and its framework we can develop dynamic web applications using JavaScript.

Q. What is a server side?

Server side application means application must be execute on server called as server side environment

Q. What is a server?

Server is application/software or program which is responsible to accept request from remote application i.e client and provide response to remote application or client as well as server is application which communicate with database called as server
Example of server software

a. Apache tomcat b. FlashFish c. node etc

Q. What is a client?

Client is also software which is used for send request to server and get response from a server
Example: Web browser is client in web application

Note: means when we install node js software in your system then node js software create server environment in your computer

Q. How do servers and clients communicate with each other on the network?

Client and server communicate with each other by using protocols

Q. What is protocol?

Protocol is some standard rules and regulations maintained for communication purposes between two entities called protocol.
If we think about computer system we have n number of protocols

Types of protocol

1. **http :** http stands for hyper text transfer protocol basically this protocol is used in web application development where we can send request in the form text/html format and get response using html format
2. **FTP :** FTP stand for file transfer protocol which is used for upload the data to server by using file format

3. **TCP/IP :** Transmission control protocol it is used for to work with socket programming or using lan based application
4. **POP3 :** POP3 means Post Office protocol this is used for read the email from a server or download the email from server

5. **IMAP :** IMAP stands for Internet Message Access Protocol. It's a protocol used by email clients to retrieve emails from a mail server over a TCP/IP connection.

6. **Https :** hypertext transfer protocol security layer this is http protocol with some security feature

Q. What is a dynamic web application?

Dynamic web application means those applications connected with databases called dynamic web applications or those applications can dynamically change data on a web page using a database called as dynamic web application.

Q. What is a static web application?

Static web application means those application cannot modify its content or web page with fixed text content called as static web page or application
Note: if we want to work with dynamic web applications we require the server environment and server side technologies or framework.

JAVA Technologies for dynamic web application
1. Using a servlet and JSP(technologies)
2. Spring MVC or Spring boot(framework)

PHP - using php we can develop the dynamic web application or laravel or

.NET: [ASP.NET](ASP.NET) for dynamic web application or MVC

Python: Django or flask for dynamic web application

JavaScript: Node js technology or server environment or express js framework

How to work with node JS

---

1. **Download and install node js**
   https://nodejs.org/en/download
2. **Create application using .js file**
   Error! Filename not specified.

3. Run the application : if we want to run JavaScript code using a server environment we have some steps.

> **a. Go where js file save**
> **b. Type the command node filename.js**
> Error! Filename not specified.

Error! Filename not specified.

Core Module in Node JS

Q. What is the core module in Node JS?

Core Module is the same as JavaScript libraries. A set of functions you want to include in your application.
Node.js has a set of built in modules which you can use without any further installation like as http , fs , assert etc

Some of the node.js module given below

---

1. **Http** : http module is used for create the server by using node js it includes some classes ,methods and events for creating a server side API
2. **FS: this module is used for to work with file system like as create file, delete file etc**
3. **Assert :** this module provides some assertion functions which help us in **testing.**
4. **QueryString:** this module is used for work with query
5. **Path:** this method provides methods to deal with file paths.
6. **URL: module provide utilities for URL resolution and parsing**
7. **OS :** provide information about operating system information.
8. **Process:** this module helps us provide information about the current process of nodes.

Core Modules in Node js

In every programming language there are some basic features like database connection feature, file creating feature , API calling feature as well as to process the code there are some inbuilt API called as core module.
Example: FS, Buffer, http etc

There are two types of Core Module

---

1. **Global Module :** global module means a module not needed to import in an application called as global module.
   Error! Filename not specified.

2. Non Global module : non global module means those modules need to be imported in an application called as non global module.
Example: We want to create a file and store some data in it.

test.js

```
let fs=require("fs");   //importing module in node
fs.writeFileSync("demo.txt","Welcome in non global module concept");
console.log("File Write Successfully...........");
```

Http module

Http module help us to create server means http module has one function name as createServer() which is used for create server and this function contain one call back function which two parameters request and response and when we create server we required to call one method name as listen() this method is used for provide port number to server

Example: test.js

```
let http=require("http");
let server=http.createServer((req,res)=>{
            res.write("welcome in node environment");
            res.end();
});
server.listen(3000,()=>{
          console.log("Server Started");
});
```

Q. What is Port number and Why use it?

Port number is the unique identity number of the server means if we want to access any server we require to uniquely identify the number to the server known as port number.
If we think about computer then single computer can have more than one server application
So when we want to access any server before that we required to machine where server install so we can access server machine by using ip address but if we want to access particular server in that machine then we required to provide port number to server so this is major we need to provide unique identity to server application called as port number
And we cannot same port number to more than one server and if we try to give same port number to more than one server then we only one server can execute

Q. What is the request and response?

Request means when we visit the any web page using browser or submit form or click on hyperlink known as request
If we think about response if we think about web page and if anything display on web page send by server called as response means if we think about our code we have statement res.write("welcome in node environment") here this message displayed on the web page so this is called as a response sent by the server on the web page.
If we want to createServer() method callback function it contain two parameters req, res
Here req parameter accept the request send by client like as form data or request using hyperlink and res parameter send response to client

Note: When we want to work on node project we need to know the one important file known as package.json

Q. What is the package.json file?

Package.json file hold the all detail about project like as project name, project version , repository name , command used in project , installed packages like as mongodb, logging etc

How to create package.json file in node project

If we want to create package.json file in node project we have following command

Syntax: npm init or npm init -y

Error! Filename not specified.
Error! Filename not specified.
Or
We can create package.json file by using npm init -y
Error! Filename not specified.
When we work with node we required one more folder in project name as node_module
Node_module folder where all libraries present or when we installed any external library then library installed under the node_module folder
If we want to create node_module folder we have command
Syntax : npm install

Q. What is npm?

npm stands node package manager basically it is build tool which is used for download the required libraries for node application and we not need to install it externally it is present in node environment means when we installed node application then with node by default we get npm
If we want to check version of npm we have command
Error! Filename not specified.
So now we want to create node_module folder in application
Error! Filename not specified.
Error! Filename not specified.

Example: we want to design node by project for display good morning message

1. **Steps to create standard node application**

a. Create application and generate package.json file

Error! Filename not specified.
2. Create package-lock.json

Error! Filename not specified.

3. Create a program and execute it.
demo.js

---

console.log("good morning");
Output
Error! Filename not specified.
If we think about above code we get output text color in black color but we want to generate the text color in different color format like as red, blue, green etc
If we want to change output text color we have color dependency provided by node so we required to download it in application as library
Error! Filename not specified.
Note: when we install any dependency or library in node the entry of library is added in package.json file
Error! Filename not specified.

Example: demo.js

```
let colors=require("colors");
console.log("good morning".red);
console.log("good afternoon".green);
```

Output
Error! Filename not specified.

ExpressJS

Q. What is express JS?

Express JS is a framework of node which is used for web application development purposes and mobile applications.
Means using expressJs we can develop the server side application by offering to use API for routing middleware,http utilities etc

Q. What kind of task can you perform by using expressJS?

---

1. Building RESTFUL APIS
2. Creating Single Page Application,multi page application and hybrid applications
3. Developing server side logics for web applications and mobile applications
4. Handling routing and middlewares
5. Integrating views and rendering engines
6. Accept request and processing on it and sending to database and getting back response from database and send as response to client.

etc

How to use expressJS Practically in node environment

Steps to use expressJS Practically in node environment

---

1. **Install express library**

Error! Filename not specified.

2. **Import express library in application**

If we want to import express library in application we have to use following
Example: let express=require("express");

Once we import express library in application we can call express() function

Syntax: let variablename=express();
Error! Filename not specified.

3. **Use standard http methods to work with web applications to handle the request and response**

If we want to work with web application in any technologies we required to send request and get response from a server and for that we have some inbuilt methods or some standard http methods

get(): this method is used for send request to server and get some response from a server according to request send by client page

Example: normally we use this method for fetching data from a server or searching data on server etc
Example: suppose consider we want to fetch detail employee using its id then from a server then we can send id of employee to server using get method using query string or path variable and get response server

post() : post method is used for create resource at server side normally this method is used for form submission purpose or send request to server and create new resource at server side

Example: suppose consider we are working on admission processing module and we want to take new student data and store in database as admission means we need to submit form to server then we can use post method
put() : this method is used for update resource or data on server
Example: if we want to update the employee record on server then we can use put method
delete() : this method is used for delete resource from a server or record from a server
patch(): path method is used for updating the partial resource at server side.
Etc
Example: we want to one API using get method name as /welcome and this API generate response hey we are working with web using express

Syntax:
Error! Filename not specified.
Example:
Error! Filename not specified.

Once we create API and if we want to test or run it we required a server so we can create server by using express and for that we have one method name as listen()
Syntax of listen

Error! Filename not specified.
Error! Filename not specified.
Error! Filename not specified.
Once we start your server we can test your API

If we want to TEST API we have two ways

1. **Using browser : browser can test only get API not POST**

   Steps to test API using browser

   ---

a. **Run program and start server**
b. **Open the browser and check API like as**
   Syntax: http://localhost:portnumber/APINAME
   Or
   http://ipaddress:portnumber/APINAME

Error! Filename not specified.

2. **Using POSTMAN**

   Q. What is POSTMAN?

   POSTMAN is a Platform or tool used for debugging ,building and documenting API. using POSTMAN we can mange the request send by API as well as Generate response from API and manage the life cycle of API
   Steps to use POSTMAN

   ---

   1. **Download and install the POSTMAN**

      ---

      https://www.postman.com/downloads/
      Note: you can download the POSTMAN using above URL

2. **Open the postman**
   Error! Filename not specified.
   Note: you can generate HTML content as response via send method

   Example: we want to design HTML login form using send() method

```
let express=require("express");
let app=express();

app.get("/welcome",(req,res)=>{
        res.send("<input type='text' name='name' value=''
style='width:400px;height:40px;'/><br><br><input type='text' name='pass' value=''
style='width:400px;height:40px;'/><br><br><input type='submit' name='s' value='Login'
style='width:400px;height:40px;'/>");
        res.end();//response end
});
```

```
app.listen(3000,()=>{
        console.log("server started successfully.....");
});
```
Output
Error! Filename not specified.

Example: if we think about above code we generate response using HTML format but it is not standard approach to generate the UI as response using express
Better way you can call directly html page as response from express

Steps to call HTML page as response to server

---

1. **Create project using node**
2. **Create public folder in project**
   Error! Filename not specified.

3. **Create html page under the public folder**

   ---

   first.html
   Error! Filename not specified.
   Note: save this file under public folder
4. **Create .js file and write a following code in it**

a. Import express library , path library
b. Use express.static() middleware at application level
c. Create API using get() method
d. Send HTML file as response

Error! Filename not specified.

If we think about above code we use path module basically it is core non global module of node which is used for access path of system folders and files and we use the __dirname name this module can access the current working folder path means as per our example D:\\oct2024\nodeprojects\fourth and we have one statement +"\" public means we combine inner folder so our path is D:\oct2024\nodeprojects\fourth\public and if we think get api i.e app.get("/",(req,res)=>{ }); here res.sendFile() is method send html page as response to browser and we use p+"/first.html" means path with first.html and send as response on browser.

Example: we want to design registration page using html file and call it from express js

Steps.

---

1. **Create project**
2. **Create public folder under the project and create html page and .css file under public folder**
3. **Create API using express and call html page**

   public/register.html

```
<html>
 <head>
   <title>registration page</title>
   <link rel="stylesheet" href="style.css" />
 </head>
 <body>
  <input type='text' name='name' value='' class='control'/><br><br>
  <input type='text' name='email' value='' class='control' /><br><br>
  <input type='text' name='contact' value='' class='control' /><br><br>
  <input type='submit' name='s' value='Register'  class='control' />
 </body>
</html>
```

public/style.css

```
.control{
  width:400px;
  height:40px;
}
```

index.js

```
let express=require("express");
let path=require("path");
let app=express();
app.use(express.static("public"));
let p=path.join(__dirname+"/public");
app.get("/",(req,res)=>{
     res.sendFile(p+"/register.html");
});
app.listen(3000,(req,res)=>{
   console.log("server started");
});
```

How to submit HTML page to express server

Steps

1. Create HTML page and call it from using get() method as response to end user
2. **Use form tag in html page for submit html form to server page**

Note: <form> is tag in html which help us submit HTML form control as request to server
Syntax of form tag
Error! Filename not specified.
<form>: tag for send html form request to server page as request
name: name indicate form name it is identity of form in java script or server side also
action : action indicate server page URL where user send requested data
method: Method indicate Form submission Technique

GET : when we submit form using GET method then all form data append in browser address bar as query string in the form name and value pair and submit to server and access at server side by using query string

POST: when we submit form using post method then form data send as request from browser header means not append on browser address bar as query string
Means request send via body of page to server
enctype: we will discuss enctype parameters in file uploading examples.

Complete example with source code

---

index.js

```
let express=require("express");
let path=require("path");
let app=express();
app.use(express.static("public"));
let p=path.join(__dirname+"/public");
app.get("/",(req,res)=>{
    res.sendFile(p+"/register.html");
});
app.get("/save",(req,res)=>{
              let name=req.query.name;
              let email=req.query.email;
              let contact=req.query.contact;
             res.send(`Name is ${name} <br> Email is ${email} <br>Contact is ${contact}`);
});
app.listen(3000,(req,res)=>{
   console.log("server started");
});
```

Register.html

```
<html>
 <head>
   <title>registration page</title>
   <link rel="stylesheet" href="style.css" />
 </head>
 <body>
 <form name='frm' action='/save' method='GET'>
  <input type='text' name='name' value='' class='control'/><br><br>
  <input type='text' name='email' value='' class='control' /><br><br>
  <input type='text' name='contact' value='' class='control' /><br><br>
  <input type='submit' name='s' value='Register'  class='control' />
 </form>
 </body>
</html>
```

Style.css

```
.control{
  width:400px;
  height:40px;
```

}
Flow
Error! Filename not specified.
Now we want to submit form by using post method

Note: when we want to submit form data using post method we required to install body-parser dependency as middleware

Error! Filename not specified.

Note: you have to import body-parser in application and use as middle ware like as given below

Example with complete source code

```
let express=require("express");
let path=require("path");
let bodyParser=require("body-parser");

let app=express();
app.use(express.static("public"));
app.use(bodyParser.urlencoded({extended:true}));


let p=path.join(__dirname+"/public");
app.get("/",(req,res)=>{
    res.sendFile(p+"/register.html");
});
app.post("/save",(req,res)=>{
        let{name,email,contact}=req.body;
        res.send(`Name is ${name}<br>Email is ${email}<br>Contact is ${contact}`);
});
app.listen(3000,(req,res)=>{
  console.log("server started");
});
```

Register.html

```
<html>
 <head>
   <title>registration page</title>
   <link rel="stylesheet" href="style.css" />
 </head>
 <body>
 <form name='frm' action='/save' method='POST'>
  <input type='text' name='name' value='' class='control'/><br><br>
  <input type='text' name='email' value='' class='control' /><br><br>
  <input type='text' name='contact' value='' class='control' /><br><br>
  <input type='submit' name='s' value='Register'  class='control' />
 </form>
 </body>
```

</html>
Style.css

```
.control{
  width:400px;
  height:40px;
}
```

How to connect node application with mysql database

If we want to connect node application with mysql database we have to use mysql/ mysql2 dependencies

Steps to connect node application with mysql database

1. **Install mysql2 dependencies**
   Error! Filename not specified.
2. **Import mysql2 module in application**
   Error! Filename not specified.
3. **Write database configurations**
   If we want to connect any technology application with database we have to specify the following things like as database name, host name,username and password etc
   If we think about node we have createConnection() function of mysql library and it contain one JavaScript object where we required provide database credential like as host,username,password etc
   Error! Filename not specified.

4. **Checking connection success or not :** if we want check your database is connected or not for that we have function name as connect() and this function contain err as parameter and if connection failed err object generate otherwise not.
   Error! Filename not specified.
5. **Work with database :** once we connect you application with database we can perform different operation on database like as insert/delete/update/select etc
   If we want to perform DDL and DML operation on database using node we have query() function from mysql

   Syntax:
   Error! Filename not specified.

   Example: we want to create employee table in database and insert record in table

```
mysql> create table employee(eid int(5) primary key auto_increment,name varchar(200),email varchar(200),contact varchar(10),sal int(5));
Query OK, 0 rows affected, 2 warnings (0.14 sec)
```

   Example with source code

```
let mysql=require("mysql2");
let conn=mysql.createConnection({
 host:"localhost",
 user:"root",
 password:"root",
 database:"augsepoct"
});

conn.connect((err)=>{
  if(err)
  { console.log("database is not connected");
  }
  else
  {  console.log("database is connected");
  }
});

conn.query("insert into employee
values('0',?,?,?,?)",["ram","ram@gmail.com","123456",10000],
(err,result)=>{
    if(err)
            { console.log("Record not save");
      console.log(err);
            }
            else{    //1>0
                        if(result.affectedRows>0)
                        {  console.log("Record Save Success");
                           console.log(result);
                        }
                        else{
                                    console.log("Some problem is there");
                        }
            }
});
```

Output
Error! Filename not specified.

Example: we want to delete employee record by using its id

```
let mysql=require("mysql2");
let conn=mysql.createConnection({
 host:"localhost",
 user:"root",
 password:"root",
 database:"augsepoct"
});

conn.connect((err)=>{
  if(err)
  { console.log("database is not connected");
  }
  else
```

```
          {  console.log("database is connected");
          }
         });

         conn.query("delete from employee where eid=?",[1],
         (err,result)=>{
            if(err)
                 { console.log("Record not save");
              console.log(err);
                 }
                 else{    //1>0
                         if(result.affectedRows>0)
                         {  console.log("Record deleted success");
                            console.log(result);
                         }
                         else{
                                  console.log("Some problem is there");
                         }
                 }
         });
```

Example: WAP to update employee name,salary using its id

```
let mysql=require("mysql2");

let conn=mysql.createConnection({
 host:"localhost",
 user:"root",
 password:"root",
 database:"augsepoct"
});

conn.connect((err)=>{
  if(err)
  { console.log("database is not connected");
  }
  else
  {  console.log("database is connected");
  }
});

conn.query("update employee set name=?,sal=? where eid=?",["Shyam",20000,2],
(err,result)=>{
    if(err)
         { console.log("Record not save");
      console.log(err);
         }
         else{    //1>0
                 if(result.affectedRows>0)
                 {  console.log("Record updated success");
                    console.log(result);
                 }
                 else{
                         console.log("Some problem is there");
                 }
         }
});
```

Example: we want to fetch all employee details from mysql table

---

```
let mysql=require("mysql2");

let conn=mysql.createConnection({
 host:"localhost",
 user:"root",
 password:"root",
 database:"augsepoct"
});

conn.connect((err)=>{
  if(err)
  { console.log("database is not connected");
  }
  else
  {  console.log("database is connected");
  }
});
conn.query("select *from employee",(err,result)=>{
            if(err)
            { console.log("error is  "+err);
            }
            else{
                    result.forEach((row)=>{

            console.log(row.eid+"\t"+row.name+"\t"+row.email+"\t"+row.contact+"\t"+row.sal);
                    });
            }
});
```

Example: we want to design an employee page with field name,email,contact and salary and store it in a database table.

Step to develop the project

---

1. **Create project and install express mysql2 and body-parser library**

Error! Filename not specified.

2. Create public folder in project and store html page in public folder

public/emp.html

```
<html>
 <head>
  <title>emp reg </title>
 </head>
 <body>
    <input type='text' name='name' value=''/><br><br>
            <input type='text' name='email' value=''/><br><br>
            <input type='text' name='contact' value=''/><br><br>
            <input type='text' name='sal' value=''/><br><br>
            <input type='submit' name='s' value='Add New Employee'/><br><br>

 </body>
</html>
```

3. **Create node server and create api using get() method and call html page**

Example with source code

Style.css

```
input{
 width:400px;
 height:40px;
}
```

Emp.html

```
<html>
 <head>
  <title>emp reg </title>
  <link rel="stylesheet" href="style.css" />
 </head>
 <body>
 <form name='frm' action='/save' method='POST'>
    <input type='text' name='name' value=''/><br><br>
            <input type='text' name='email' value=''/><br><br>
            <input type='text' name='contact' value=''/><br><br>
            <input type='text' name='sal' value=''/><br><br>
            <input type='submit' name='s' value='Add New Employee'/><br><br>
 </form>
 </body>
</html>
```

index.js

```
let express=require("express");
let bodyParser=require("body-parser");
let path=require("path");
let mysql=require("mysql2");
let conn=mysql.createConnection({
            host:"localhost",
            user:"root",
            password:"root",
            database:"augsepoct"
});
conn.connect((err)=>{
           if(err){
                       console.log("connection failed");
           }
           else{
                      console.log("Database is connected");
           }
});
let app=express();

app.use(express.static("public"));

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

let p=path.join(__dirname+"/public");
app.get("/",(req,res)=>{
            res.sendFile(p+"/emp.html");
});

app.post("/save",(req,res)=>{
            let {name,email,contact,sal}=req.body;
            conn.query("insert into employee
values('0',?,?,?,?)",[name,email,contact,sal],(err,result)=>{

                              if(err)
                              { console.log("Not inserted"+err);
                              }
                              else{
                                 if(result.affectedRows>0)
                                 { res.send("Employee Added");
                                 }
                                 else{
                                           res.send("Employee Not Added");
                                 }
                              }
            });


});
app.listen(3000,()=>{
  console.log("server started");
});
```

EJS

EJS stands for Embedded JavaScript Templates basically it is scripting language we can use with express for generate the dynamic UI or Dynamic content on web page
EJS use all HTML code or tags for generate UI + EJS contain some additional tags which help us to generate the dynamic content and access dynamic from express server API and show on view page or html page

Steps to work with EJS

1. **Create node project**
   Error! Filename not specified.

2. **Install express and ejs dependencies**
   Error! Filename not specified.

3. **Create view folder under the node project**
   View is folder where we can store the all .ejs file means it contain all UI parts
   Error! Filename not specified.
4. **Create html pages and save it using .ejs extension under the view folder**
   Error! Filename not specified.
5. **Write JavaScript code at server side and set ejs as view engine**

   When we work with ejs we have to set ejs as view engine and for that we can use following middleware
   Error! Filename not specified.

6. Create API using express and render the ejs file from express API
   If we want to call ejs files from express we have render() function and it is member of response object so using this we can ejs file
   Error! Filename not specified.
   Note: when we use ejs file then we can send data from express to ejs file and we can generate dynamic UI using ejs which is not possible by using plain HTML

   If we want to work with dynamic data using EJS we have some tags given by ejs file

   <%=%> : this is EJS express tag which is used for display the output on web page send by express server and express server can send data in the form of key and value pair so in ejs file just we required to access key for display data

Error! Filename not specified.

Complete source code

index.js

```
let express=require("express");
let app=express();
app.set("view engine","ejs");
app.use(express.static("public"));
app.get("/",(req,res)=>{
    res.render("demo.ejs",{
        name:"Ram",
        id:1,
        salary:10000
    });
});
app.listen(3000,()=>{
  console.log("Server Started");
});
```

Demo.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>NAME is <%=name%> </h1>
    <h1>Id is <%=id%></h1>
    <h1>Salary is <%=salary%></h1>
</body>
</html>
```

<% %> : this tag is used for write logics or business logic on UI part using EJS file like as loop, if, etc

Error! Filename not specified.

<%-   include("filename") %> : this is used for include the another page content in ejs file

Example: including another page content in ejs file

Navbar.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <a href="Home">Home</a> &nbsp;&nbsp;<a href="About">About</a>

</body>
</html>
```

Demo.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <%-include("navbar.ejs")%>
    <%for(var i=1; i<=10; i++)
      {
    %> <h1><%=i%></h1>
    <%
      }
      %>
</body>
</html>
```

Example: we want create one web page using ejs file with one textbox and one button and when user submit form then form data should be access by express server and generate table of number

Error! Filename not specified.

Project

1. Create project and install express ejs and body-parser dependencies

Style.css

```
input{
    width:400px;
    height:40px;
    border-radius: 5px;
    padding:10px;

}
```

index.js

```
const bodyParser = require("body-parser");
let express=require("express");
let app=express();
app.set("view engine","ejs");
app.use(express.static("public"));
app.use(bodyParser.urlencoded({extended:true}));

app.get("/table",(req,res)=>{
    res.render("table.ejs",{no:0});
});
app.post("/printtab",(req,res)=>{
    let {num}=req.body;
    res.render("table.ejs",{no:num});
})
app.listen(3000,()=>{
 console.log("server Started");
});
```

Table.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="style.css" />
</head>
<body>
    <form name="frm" action="/printtab" method="POST">
    <input type="text" name="num" value="" /> &nbsp;&nbsp;
    <input type="submit" name="s" value="Print Table" />
    </form>
    <br><br>
  <table width="80%"  border="5"
><tr><th>Iteration</th><th>X</th><th>Numbr</th><th>=</th><th>Result<
/th></tr>
     <%
    if(no!=0){
      for(var i=1;i<=10; i++)
      {
   %>
     <tr
align="center"><td><%=i%></td><td>X</td><td><%=no%></td><td>=</td><t
d><%=i*no%></td></tr>
   <%
      }
    }
     %>

  </table>
</body>
</html>
```

CRUD Application using node js

Objective of this Application

Employee Management Application

Task

1. Add New Dept/view Dept/Search/Delete Dept/Update dept
2. Add New Employee under dept
3. View Dept wise employee /search employee/delete employee/update employee
4. Upload Bulk employee using CSV file under specific dept

Screens

Home.ejs
Error! Filename not specified.
New Dept Page: When a user clicks on the New Dept menu then we want to open the new page name as adddept.ejs with the following screen.
Error! Filename not specified.

View Dept pages : when user click on View dept then we want to show the all dept in grid and perform delete /update and search operation on dept

Error! Filename not specified.
When user click on new employee we have following screen
Error! Filename not specified.

When user click on view employee menu bar then your UI look like as

Error! Filename not specified.

Technologies

Front End : HTML/CSS/Bootstrap /JS/ejs
Backend: Node /MYSQL
Technologies: AJAX
API - REST API - data exchange format - JSON
API TESTING TOOL : POSTMAN
Deployment Architecture : Monolithic
Web Development Design Pattern : MVC

Q. What is MVC?

MVC stands for Model view Controller basically it is a design pattern in web development where we separate the UI logics and business logics

V : V stands view means place where the user can provide input and get results means all end user interface designing part known as view.

Example: registration form , login form etc
Normally we can design view page using React/Angular/HTML/ejs etc

C : C stands for controller if we think about node controller is handle function where we can submit request via view and send to service layer or model layer
And get result from service or model and send appropriate response to end user

Q. What is the service layer?

Service layer are the some functions or classes written in JavaScript and they communicate with controller function and normally we write the business logics in service layer

M : models are the some functions or classes written in JavaScript and they can accept data from service layer or controller and communicate with database
Error! Filename not specified.

Q. What is standard folder structure of node project

If we want to work with a node project we have some standard folder structure provided by node to us.
Error! Filename not specified.

Now we want to start project development
Note: before start project development we want to understand the concept of middleware

Q. What is middleware?

Middleware in express refers to functions that process requests before reaching the route handlers. These functions can modify the request and response objects ,end the request and response cycle or call the next middleware function.
Middleware functions are executed in the order they are defined
Using middleware we can perform task like as authentication , logging , error handling etc
Error! Filename not specified.
Error! Filename not specified.

(req,res,next) : this is the middleware function where you can perform actions on the request and response objects before the final handler is executed.
next(): this function is called to pass control to the next middleware in the stack if the current does not end the request and response cycle.

What middleware does in express.js

---

1. **Execute the code:** middleware can run any code when a request is received

2. **Modify the request and response :** Middleware can modify the request and response object at run time
3. **End the request and response cycle:** middleware can send responses to clients and end the cycle of response.
4. **Call the next middleware:** Middleware can call next() to pass control to the next function in the middleware stack

How Middleware works

In Express Js middleware functions are executed sequentially in the order they are added to the application.

1. **Request arrives at the server**
2. **Middle function: are applied to the request one by one**
3. **Each middleware can either**

   Send a response and end the request-response cycle

   Call next() to pass control to the next middleware

   4. If no middleware ends the cycle the router handler is reached and a final response sent

Types of middleware

1. **Application level middleware :** application level middleware is bound to the entire express application using app.use() or app.METHOD(). It executes for all routes in the application ,regardless of the specific path or HTTP Method.

   This type of middleware normally perform logging ,body-parsing , authentication-check or setting headers for every incoming request.

   Error! Filename not specified.

2. **Router level middleware:** Router level middleware is applied to specific router instances using router.use() or router.METHOD(). It only applies to routes defined within that particular router , making it perfect for module application where middleware is only relevant to specific routers

   This type of middleware is used e.g authentication or user management) etc

   Error! Filename not specified.

3. **Error handling middleware :** error handling middleware is a special type of middleware used to catch and respond to error during the request-response cycle. It is defined with four parameter ,err,req,res,next

   This middleware is essential for sending a consistent error response and avoiding unhandled exceptions that might crash the server.

   Error! Filename not specified.

4. **Built in Middleware :** express JS provides some built in middleware to us to perform some common task like serving static files or parsing data.

   Example: express.static(), express.json() etc

   Error! Filename not specified.

5. **Third-Party Middleware :** this party middleware is developed by external developers and packed as a npm module.

   This middleware packages add additional functionality to your application such as request logging,security feature or data validation

Example: the morgan middleware logs HTTP request and body-parser help parse incoming request bodies for easier handling of form data
Etc

Error! Filename not specified.

Now we want to implement the middleware practically

Error! Filename not specified.

Login.ejs

```
<html>
 <head>
  <title>login page</title>
 </head>
 <body>
   <form name='frm' action='/save' method='POST'>
   <input type='text' name='user' value=''/> <br><br>
   <input type='text' name='pass' value=''/> <br><br>
   <input type='submit' name='s' value='Login'/> <br><br>
   </form>
   <%=msg%>
 </body>
</html>
```

index.js

```
let express=require("express");
let bodyParser=require("body-parser");
let app=express();
app.set("views engine","ejs");
app.use(express.static("public"));
app.use(bodyParser.urlencoded({extended:true}));
app.get("/",(req,res)=>{
     res.render("login.ejs",{msg:""});});
app.use((req,res,next)=>{
            let {user,pass}=req.body;
             console.log(user);
            if(user=="admin")
            { next();
            }
            else{
                        res.render("login.ejs",{msg:"login failed"});
            }
});
app.post("/save",(req,res)=>{
              res.send("Request receiced");
});
app.listen(3000,(req,res)=>{
              console.log("server started");
```

});

Q. What is a .env file and why use it?

Here .env stands for environment variables which are used for storing data in key and value pairs.
Normally we store configuration data which may be changed in future in this file.

Example of .env file

Error! Filename not specified.

Error! Filename not specified.
Install env dependencies using express

Note: create .env file in root folder of project and configure data in key and value pair which we want
Error! Filename not specified.

Start Project Development

1. Create database
   mysql> create table dept(deptid int(5) primary key auto_increment,deptname varchar(200) not null unique);

Query OK, 0 rows affected, 1 warning (0.16 sec)

mysql> create table employee(eid int(5) primary key auto_increment,name varchar(200) not null, email varchar(200) not null unique,contact varchar(10) not null unique,sal int(5) default 0 ,photo varchar(200),deptid int(5),foreign key(deptid) references dept(deptid) on delete cascade on update cascade);
Query OK, 0 rows affected, 3 warnings (0.11 sec)

2. **Install following libraries**
   npm install express dotenv ejs mysql2
3. **Create standard folder structure**

URL Rewriting

URL rewriting means we send requested data to API via browser address in query string in the form of name and value pair

Syntax: http://localhost:portnumber/urlname?key=value&key=value

How to upload file to server

If we want to upload the file to the server using express we have the following steps.

1. **Install multer dependencies in application**
   Multer dependency provide API to us for file uploading purpose

Error! Filename not specified.

2. **Write a form tag in following fashion**

   If we want to upload file to server we have to use following syntax of form tag

   <form name='frm' action='/serverapi' method='POST' enctype='multipart/form-data'>

   </form>

Q. Why is it recommended post method to upload the file?

If we get method then there is some limitation

a. Get method append requested data in key and value in browser address so data may visible in browser so data security concern is raise

b. Get method send data using browser address bar and browser address bar cannot store more than 256 kb data and file size may me more than that

So avoiding these two problem when we send file request then we required send via of body of page or header so we use post method is recommended because post method send request from body of page

Q. What is the enctype?

Enctype property is the MIME type of content that is used to submit the form to server.

There are two types of enctype?

a. **application/x-www-urlencoded :** this is the default enctype in format means we not need to mention in form and this type enctype send data to server with different request name every time in the form of key and value pair means key is html control name and value is html form control data.

b. **multipart/form-data :** when we upload the file to server then your enctype must multipart/form-data means using we can send file request and with form detail like as filesize,content type etc

Example with source code

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

    <form name="frm" action="/upload" method="post"
enctype="multipart/form-data">
    <input type="file" name="file"
value=""  style="width:400px;height:40px"/><br><br>
    <input type="submit" name="s" value="Upload File to server "/>
    </form>
</body>
</html>
```

3. **Create new folder under public folder e.g images**

Error! Filename not specified.

4. **Configure the middleware in express for set the destination of file**

```
let storage=multer.diskStorage({
    destination:(req,file,cb)=>{
        cb(null,"public/images/");
    },
    filename:(req,file,cb)=>{
        console.log(file.originalname);
        cb(null,file.originalname);
    }
});
```

If we think about above code it contain three parameter
req: The HTTP request object

You can access user data ,header ,session info etc
Userful if the destination folder depends on who uploads the file.

file : the file object being uploaded contain metadata about file like as originalname, mimetye ,size etc

cb : A callback function that you must call to indicate the result
Error: pass null if there is no error
Destination: where we want to save the file

3. : Write handler for accept file request and call dependency

Authentication and Authorization in Node

Authentication: authentication is process of verify the identity of a user or system

It ensures that the person or system is who they claim to be.

Common methods of authentication

---

1. Passwords
2. Biometric scan(finger prints,face ID)
3. Two Factor Authentication (2FA)
4. OAuth (used for social logins like as Login with google etc)
5. Certificates or token authentication (like JWT-JSON web token)

Authorization: Authorization means what an authenticated user is allowed to do. It defines user permission or access level within a system.

Common authorization technique

---

1. Role Based Access Control (RBAC) - e.g admin, editor, viewer etc
2. Policy Based Access

Etc

Q. What is the difference between Authentication and Authorization?

Authentication	**Authorization**
Who are you?	What are you allowed to do?
Credential (password,token)	Permission(roles,rules)
Before authorization	After successful authentication
Example: login with email and password	Admin can delete user or update user etc

Can you give a real time scenario of authentication and authorization?

Suppose if we consider banking application
Authentication: user login with email and password
Authorization:
a. User can check own balance
b. User can transfer money to other account from own account
c. User can see the own profile and update it

Q. What are the benefits of authentication and authorization in web applications?

---

1. **Security:** protects user data and prevents unauthorized access to sensitive information
2. **Access control:** ensure users only access function or data they permitted to reduce the risk of data leak or misuse

3. **User Experience:** allow personalized experience based on the user role or preferences
4. **Audit monitoring :** logs can track user access, helping with compliance and forensic investigation
5. **User privacy :** means we can provide data security to user according to his account login
   E.g if we think about gmail user can view own receive mail or send mail

Etc

How many ways to implement authentication in web application

---

1. Session based authentication
2. Token based authentication
3. OAuth2.0
4. OIDC authentication
5. Multi-factor authentication (password+OTP)

How many ways to implement authorization practically

---

1. Role Based Access Control
2. Attribute based authentication
3. Policy control access
   etc

Now we want to work with session based authentication

Q. What is a session?

Session is a communication period between client and server over the network called as session or session is process between login to logout.

Q. Why is the session used?

Http is by default stateless protocol means every request of client is consider as new client by the server called as stateless so when we use session then server create one session object at server side for every client and assign one session of that object and same session id used in client header and so when client visite first then client has no session and if server found there no session id in client then create new session and assign session id to client and when client revisit to server then verify his session if session id found then consider it is existing client called as stateful protocol.

Q. Can you give a real time scenario where a session is beneficial in a project?

User privacy : if we think about session object then for every client separate session object is created so using that we can maintain user data privacy

Example: Suppose consider we are working online examination portal and we want to show student result in student account but we want to his result in his account not other students

Example: suppose consider we are working on online shopping portal and we want to show the order to custom after login but show only own order not others

Session provide some security to portal : means session help us every must have login to system and server should verify user and after that provide login

Example: Suppose we access any middle url of the application without login and try to visit then you can verify user session on that if we do not find the user session then redirect on login page.

Count online login user: means when user login to system then user session get started and when user session started then counter increase by 1
etc

How to work with session practically in express JS

Or
How to use session in express JS

---

1 create node project
2. Install express-session library
Error! Filename not specified.

3. Configure session in express application

You can configure the session as middleware in the express application.

Error! Filename not specified.
Description about above code

---

secret: used to sign the session id.
resave:false : don't save session if unmodified
saveUninitialized:true : save new session that are not modified
4. Use session

If we want to use session means you can store data in session , retrieve data from session or update data in session or access key session etc
Normally session can access by request object

How to store data in session or How to retrieve data from session

---

Syntax for storing a session

Error! Filename not specified.

How to retrieve data from session using key

Error! Filename not specified.
Working example of session
Error! Filename not specified.
Example with source code

```
let express=require("express");
let bodyParser=require("body-parser");
let session=require("express-session");
let app=express();

app.use(express.static("public"));
app.set("view engine","ejs");

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

app.use(session({
    secret:"ABC##$$123$$",
    resave:false,
    saveUninitialized:true
}))
app.get("/",(req,res)=>{
    res.render("login.ejs");
});
app.post("/validate",(req,res)=>{
    let {name,pass}=req.body;
    req.session.uname=name;
    req.session.upass=pass;
    res.send(`Username ${req.session.uname}
    <br><br>Password ${req.session.upass}`);

})
app.listen(3000,()=>{
  console.log("server started");
});
```

Example using a session handling

Error! Filename not specified.

Start Project

1. **Install following dependencies**
   D:\OCT2024\nodeprojects\profileapp>npm install express ejs body-parser mysql2 express-session dotenv multer

   DB table

```
mysql> create table register(rid int(5) primary key auto_increment,name
varchar(200) ,email varchar(200),contact varchar(200),username
varchar(200),password varchar(200),photo varchar(200));
Query OK, 0 rows affected, 1 warning (0.38 sec)
```

Cookie

Cookie is part of session management or session handling means we can handle the session by using cookies also.

Q. What is the difference between cookie and session?

---

1. Cookie can store its data client side i.e in browser and session can store its data at serverside
2. Session can vanish its data when we close the browser or client and cookie can store data after client close or for specified time period.
   Session data has no life and cookie has life
3. Cookie data stored in the browser window and session can store its data at server side.

There are two types of cookie

---

1. **Persistent cookie :** persistent cookie means cookie with life span called persistent cookie.
2. **Temporary cookie :** temporary cookie means when cookie vanishes its data when we close the browser means we can say without age cookie called as temporary cookie so we can say cookie work as normal session object.

How to use cookie in nodejs

Steps.

1. **Installed the cookie-parser dependency**
   Error! Filename not specified.
2. **Register cookie as application middleware**
   Error! Filename not specified.
3. **Store data in cookie and send as response :** if we want to send cookie as response to server then we have to use following syntax

   Error! Filename not specified.

JWT token

Q. What is JWT?

JWT (JSON web token) is a compact and URL safe token format used to represent claims between two parties. It is used for authentication and authorization purposes in web applications.

Structure of JWT

Error! Filename not specified.

1. **Header :** if we think about header in JTW it contain two things first is algorithm and second is token type

   Header Example
   Error! Filename not specified.

2. **Payload (claims):** this contains the data or claims means if we think about claims we store user data like as id , role , iat , exp etc
   Error! Filename not specified.
   Description above code

   Id,role: custom claims (user data)
   iat: issued at time
   exp: expiration time.

3. **Signature:** signature is created using the header,payload and secret key
   Signature help us to provide security to token means signature ensure the token is not tempered with
   Error! Filename not specified.

Q. What are the benefits of JWT token?

---

1. **Stateless : (No server memory usage)**

a. The server does not need to store any session information
b. All user data is embedded in the JWT and stored on the client
c. Make it easier to scale horizontally (e.g across multiple servers or containers)

2. **Better for API and SPAs(single page applications)**

---

a. JWT is ideal for single page application (SPAs) like as React , Vue,Angular

Commonly used with RESTFUL APIs and Mobile Apps.

3. **Easy for Cross-Domain or Mobile Authentication**
   JWT can be used in cross domain scenarios and its language /framework agnostic
   Work well for mobile application where session cookies are less practical
4. **Built in expiry :** JWTs have built in exp claim so tokens expire automatically certain time
   You can also implement refresh token for long lived session

5. **Transportable and compact :** JWTs are compact (base64 encoded) and can be easily passed in
   Http headers(authorization: Bearer token
   Cookies
   Query string (less secure)

Q. What is the difference between JWT token and session?

Feature	JWT	Sessions
Storage	Client side(localStorage or cookie)	Server side (in memory /redis /database)
Scalability	High (stateless)	Limited(stateful, needs centralized session store)
Cross domain	Support to Cross domain easily work with single application or distribute app	More suitable for traditional web applications
Decentralized auth	Great for microservices,no shared session store	Require centralized session management

Q. What are the limitations of JWT token?

Limitation	Details
Token Revocation	JWTs are hard to invalidate unless you track them manually(e.g denylist) but session can invalidate or destroy immediately
Security risk	If the token is stolen and have not expire then attacker can use it
Token size	JWT are larger than session ids which can affect performance if overloaded
Store management	Client must securely store the token(means must be stored in cookie not in local storage)

Q. When to use JWT token and when to use Session?

Use case	Use JWT	Use session
REST API,Mobile app ,SPA	Yes	Not ideal
Traditional web application with server side rendering	Less ideal	Yes
Need for horizontal scaling/microservices	yes	Complex
Need to easy logout and revocation	Harder with JWT	Simple

How to create JWT token by using node

Steps to use JWT token using Node

---

1. **Install the jsonwebtoken dependency**
   Error! Filename not specified.
2. **Create JWT token**
   Error! Filename not specified.
   Example with source code and output

```
let jwt=require("jsonwebtoken");
const secretkey="abcd12345";
const token=jwt.sign({
   id:1,
   username:"TECHHUB"
},secretkey,{expiresIn:'1h'});
console.log(token);
```

Output

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwidXNlcm5hb
WUiOiJURUNISFVCIiwiaWF0IjoxNzUwNDIyMjU3LCJleHAiOjE3NTA
0MjU4NTd9.9S7rB25FnA6-
u1cGMZ1_P6PYh7WKZbiXxCoxSEPsxcw
```

How to verify JWT Token

If we want to verify token we have verify() method of JWT

Example with source code

```
let jwt=require("jsonwebtoken");
const secretkey="abcd12345";
const token=jwt.sign({
   id:1,
   username:"TECHHUB"
},secretkey,{expiresIn:'1h'});
console.log("Token Generate "+token);

jwt.verify(token,secretkey,(err,decoded)=>{

           if(err){
                    console.log("invalid token");
           }
           else{
                   console.log("\nToken verified "+decoded.id);
                   console.log("\nToken verified "+decoded.username);

           }
});
```

Output

```
D:\OCT2024\nodeprojects\tokenapp>node demotoken.js
Token Generate
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwidXNlcm5hbWUiOi
JURUNISFVCIiwiaWF0IjoxNzUwNDIyNjQ2LCJleHAiOjE3NTA0MjYyNDZ9.
M31lKdGs90Bbz3EORFCHea-qyufW5VYxsZg5HOoskfw

Token verified 1
Token verified TECHHUB
```

Example: Create application using node with JWT token where user first register in web applications and store its info in mysql database table and after that user should login to system and verify user in database and generate JWT token for that specified user and if user want to check its profile then first user need to verify token and after that can see own profile

OAUTH 2.0 Implementation

OAauth is open authentication technique means we can authenticate user without registering in your database by using social media platform or third party portal like as google, facebook,linked in etc

Steps to work with oauth implementation by using node

---

1. **Create project using google console platform and generate client id for application**

Note: you have to visit this URL
https://console.cloud.google.com/

2. **Create node project and installed following dependencies**
   Error! Filename not specified.
   Error! Filename not specified.