EXPRESS JS

What is Express JS ?

Why we use Express.js ?

What is Express JS?

A web application framework built on top of Node.js .

Why we use Express.js ?

✓ Handles HTTP requests efficiently (GET, POST, PUT, DELETE).
✓ Supports middleware for authentication, logging, and error handling.
✓ Simplifies server-side development compared to pure Node.js.
✓ Works well with databases like MongoDB, MySQL, and PostgreSQL.

REQUEST AND RESPONSE IN EXPRESS.JS

req (request) : object in Express JS it used to access data, headers and parameters that came from HTTP request.

res (Response) : object is used to send the HTTP responses to the client. Used to send data back in various formats (HTML, JSON, etc.).

```
// app.js
const express = require('express');
const app = express();
const PORT = 8000;

// GET route
app.get('/', (req, res) => {
  // Log the request URL
  console.log('Request URL:', req.url);

  // Log the request method
  console.log('Request Method:', req.method);

  // Sending a response
  res.send('Hello ghofran!');
});

// Starting the server
app.listen(PORT, () => {
  console.log(`Server is running on http://localhost:${PORT}`);
});

```

GET AND POST IN EXPRESS.JS

● What is a GET request?

✅ Used to fetch data from the server.

✅ Does not modify the server data.

● What is a POST request?

✅ Used to send data to the server.

✅ Requires middleware (express.json()) to parse JSON request bodies.

```
const express = require('express')
const bodyParser = require('body-parser')
const app = express()
const PORT = 8080;

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.get('/', function(req, res){
    res.sendFile(__dirname+'/index.html');
 });
 app.post('/', function(req, res){
    console.log(req.body);
    res.send("Received your data!!");
 });

app.listen(PORT,
  () => {
      console.log( `Server is listening at http:localhost:${PORT}`);
  });
```

● What is Routing in Express.js?

✅ Routing defines how server handles different URLs and HTTP methods.

✅ Routes map requests to specific functions.

Basic Routing Example :

```
app.get('/home', (req, res) => res.send('Welcome to Home Page'));
app.post('/submit', (req, res) => res.send('Form Submitted!'));
app.put('/update', (req, res) => res.send('Data Updated!'));
app.delete('/delete', (req, res) => res.send('Data Deleted!'));
```

```
// server.js
const express = require('express');
const app = express();
const PORT = 4000;

// define route
app.get('/home',
    (req, res) => {
        res.send(
            '<h1>welcome to GeeksforGeeks!</h1>'
        );
    });

app.listen(PORT,
    () => {
        console.log(
            `Server is listening at http://localhost:${PORT}`
        );
    });
```

MIDDLEWARES IN EXPRESS.JS

How a Request is Processed in Express.js?

How a Request is Processed in Express.js?

(Client Makes Request)

⬇

- Middleware Runs (app.use())

⬇

- Route Handler Executes (app.get())

⬇

- Response Sent to Client (res.send())

SUMMURY

Express.js: → A web framework for building applications with Node.js.

Request: → (req) Contains client request data like parameters & body.

Response: → (res) Used to send data from server to the client.

GET Request: → Fetches data from the server.

POST Request: → Sends data to the server.

Routing: → Defines URL paths and their handlers.

Middleware: → Functions that run before sending a response.