### What is Express.js?

**Express.js** is a **minimal and flexible Node.js web application framework** that provides robust features to build web and mobile applications.  
It makes it easier to handle **routes, requests, responses**, and **middleware**.

### Why Use Express?

* Easy to set up server and routes
* Handles HTTP requests efficiently
* Built-in middleware and 3rd party middleware support
* Used in building REST APIs
* Works well with MongoDB, PostgreSQL, etc.

## Step-by-Step Express.js

### ****Step 1: Initialize Project****

mkdir express-intro

cd express-intro

npm init -y

npm install express

### ****Step 2: Create**** index.js ****File****

const express = require('express');

const app = express();

// Middleware to parse JSON

app.use(express.json());

// Route

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

res.send('Welcome to Express.js!');

});

// Start server

const PORT = 3000;

app.listen(PORT, () => {

console.log(`Server is running on http://localhost:${PORT}`);

});

### ****Step 4: Create More Routes****

// GET route

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

res.send('About Page');

});

// POST route

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

const { name, email } = req.body;

res.send(`Hello ${name}, we received your email: ${email}`);

});

### ****Step 5: Understanding Middleware****

Middleware are functions that run before your route handler.

app.use((req, res, next) => {

console.log(`${req.method} ${req.url}`);

next(); // pass control to next middleware or route

});

### ****Step 6: Route Parameters and Query Strings****

// URL Parameter

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

res.send(`User ID is ${req.params.id}`);

});

// Query String

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

res.send(`You searched for ${req.query.q}`);

});