What is express Js

Express Js is a back end web application framework used for building single page, multipage and hybrid web applications.

The first step to create an express Js web application is download express Js from the official Node Js website.

node -v is a command used in command line to execute JavaScript commands.

npm -v(node package manager) is usually used check the npm version that has been installed and ensure everything has been installed correctly.

mkdir my-express-app is a command which creates a new directory named my-express-app where the express Js files will be stored.

cd my-express-app changes the directory to my-express-app allowing for file creation and running commands specifically in the given directory.

npm init -y creates a package. json file with default settings.

What is a package.json file---

npm install express --save

npm install installs a package

express specifies the package you want to install

--save adds the package to the dependencies section of the package. json file.

From this point we create the index.js file and open it to give our app body.

const express declares the variable named express. Then require("express") imports an express library allowing for web server creation and http requests.

const app declares a constant named app and express() calls the express function(For handling incoming requests and responses.

const port declares a constant variable named port. 3000 assigns a value 3000 which is the port number your server will listen on.

app.get defines a route handler for GET requests to the specified path('/') which indcates the root URL of the web appication.

(req, res) => { an arrow function that takes two parameters req(request object) adn res (response object) which executes when the get request is made to the root URL.

res.send Sends a response back bak to the client which is hello world. It is seen when the user visits the root URL.

}); CLOSES THE ROUTE HANDLER FUNCTION

app.listen starts the express server and makes it listen for incoming request. The port coommand makes it so that it will listen on specified port in this case 3000.The arrow function executes once the server is up and running.

The console.log message ois output to the console and then the app.listen function is closed.

This code sets up a simple Express server on port 3000. When a user accesses the root URL, a "hello world" message is displayed and a log message indicates that the server is running.

**Creating the Express Application**

The first step involves importing the Express library, which is useful for building web applications in Node.js.

Calling epress() creates an instance of an Express web application which is stored in the **app** variable which will manage routes and handle requests and responses.

**Configuring the server**

A constant port is defined to specify the port number in this case 3000 on which the server will listen. It determines where clients can connect to the server.

**Defining a route**

A code is set up for handling GET requests at the root URL. When a user accesses this URL, the server triggers a specified callback function. In this function, the response object is used to send back a "Hello World" to the user.

**Side note:** What is GET

**Starting the server**

A code known as **app.listen** is called to start the server, which tell Node.js to listen for any incoming connections to the defined port. The server starting succesfully executes the provided callback function, logging a message to the console which confirms that the server is running. Once it is running users can interact with the server by navigating to <http://localhost:3000> in their web browser.

**Conclusion**

This code sets up a simple Express server on port 3000. When a user accesses the root URL, a "hello world" message is displayed and a log message indicates that the server is running.