**Node Assignment 3**

**Module: - 3 Node- web Development with node**

1. What is Express.js?

ANS: - Express.js is a web application framework for Node.js, a runtime environment for executing JavaScript code server-side. It is designed to simplify the process of building web applications and APIs by providing a minimal and flexible set of tools and features. Express.js is often referred to as a "micro" framework because it provides only the essential functionalities needed for web development, allowing developers to add additional features through third-party middleware and plugins as needed.

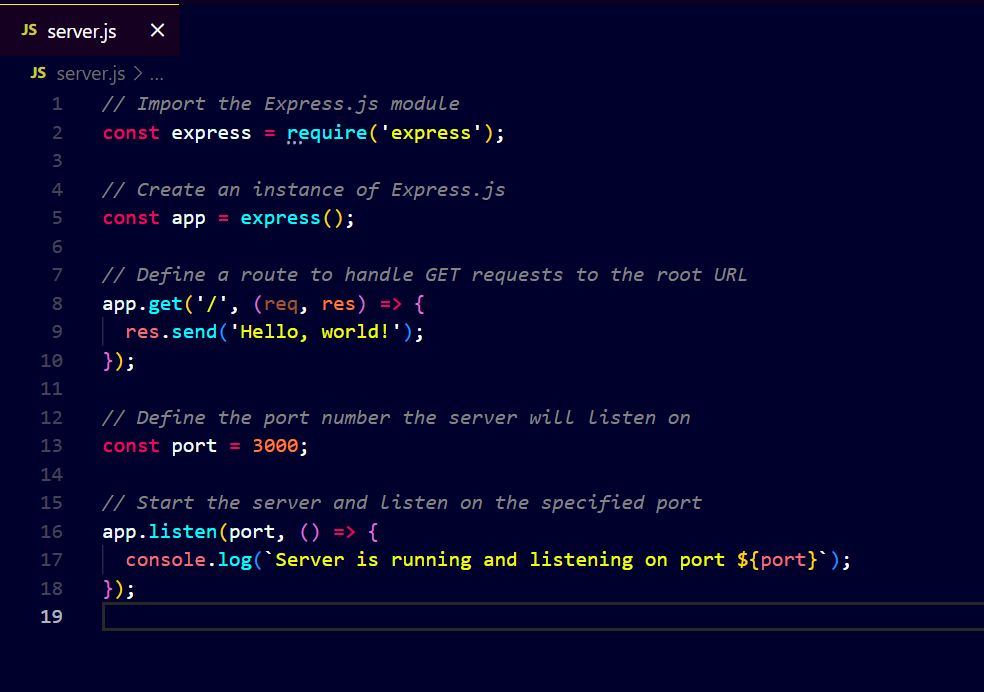
Key features of Express.js include:

1. **Routing:** Express.js provides a simple and expressive way to define routes for handling HTTP requests. Developers can define routes for different HTTP methods (GET, POST, PUT, DELETE, etc.) and specify the corresponding functions to execute when a request matches a particular route.
2. **Middleware:** Middleware functions are functions that have access to the request and response objects in an Express application's request-response cycle. They can perform tasks such as logging, authentication, error handling, and more. Express.js allows developers to use middleware to modify request and response objects, execute code asynchronously, and terminate the request-response cycle.
3. **Template Engines:** While Express.js itself does not include a built-in template engine, it allows developers to integrate popular template engines like EJS, Handlebars, Pug, etc., to dynamically generate HTML content based on data provided by the server.
4. **Error Handling:** Express.js provides built-in error handling mechanisms to handle errors that occur during the execution of route handlers or middleware functions. Developers can define custom error-handling middleware to centralize error handling logic and improve code maintainability.
5. **Static File Serving:** Express.js allows developers to serve static files such as images, CSS files, and client-side JavaScript files from a directory on the server. This is useful for serving client-side assets required by web pages or web applications.
6. **Integration with Node.js:** Express.js is built on top of Node.js, which means it leverages the asynchronous, event-driven nature of Node.js to handle concurrent requests efficiently. Express.js applications can take advantage of the vast ecosystem of Node.js libraries and modules available via npm (Node Package Manager).

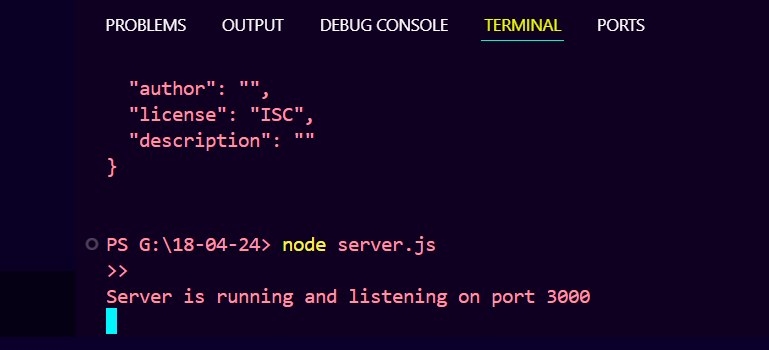
* Overall, Express.js is widely used in the Node.js community for building web applications and APIs due to its simplicity, flexibility, and performance. It is often chosen for its lightweight nature and ease of use for both beginners and experienced developers alike.

1. Create localhost server using express.

ANS:- Sure, here's a simple example of how to create a localhost server using Express.js:



To run this code, you'll need to have Node.js and Express.js installed on your system. Then, save the code to a file (e.g., server.js) and run it using Node.js:



This will start the Express.js server, and you'll be able to access it by navigating to http://localhost:3000 in your web browser. You should see the message "Hello, world!" displayed on the page.

1. Create basic “Hello world” app using node and express.

ANS: - Certainly! Here's a basic "Hello world" app using Node.js and Express.js:

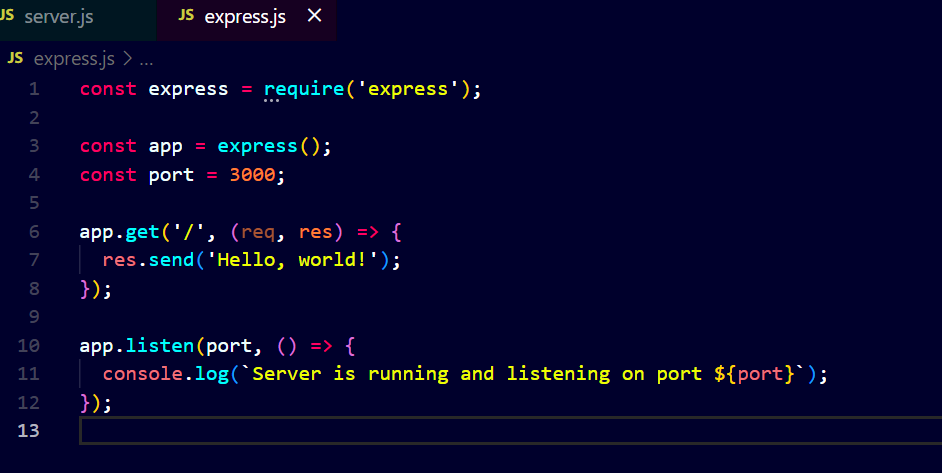
1. First, make sure you have Node.js installed on your system.
2. Create a new directory for your project and navigate into it using your terminal or command prompt.
3. Initialize a new Node.js project by running the following command and follow the prompts:

* npm init -y

1. Install Express.js as a dependency for your project:

* npm i express

1. Create a new JavaScript file (e.g., app.js) and add the following code to it:



1. Save the file and run your server:

* Node app.js

1. Open your web browser and navigate to http://localhost:3000. You should see the message "Hello, world!" displayed on the page.

* That's it! You've created a basic "Hello world" app using Node.js and Express.js. This is a simple starting point that you can build upon to create more complex web applications and APIs.