Week 4

How to move data from view to the controller

1) Explain in detail about express project creation.

The most well-known minimalist framework is Express. It is based on NodeJS's built-in HTTP module and allows for quick and easy communication between front-end and back-end logic via APIs, as well as a beautiful organisation of our business logic. We can use it for both the web and Android because it is quite adaptable. Additionally, it offers a very straightforward error handling process.

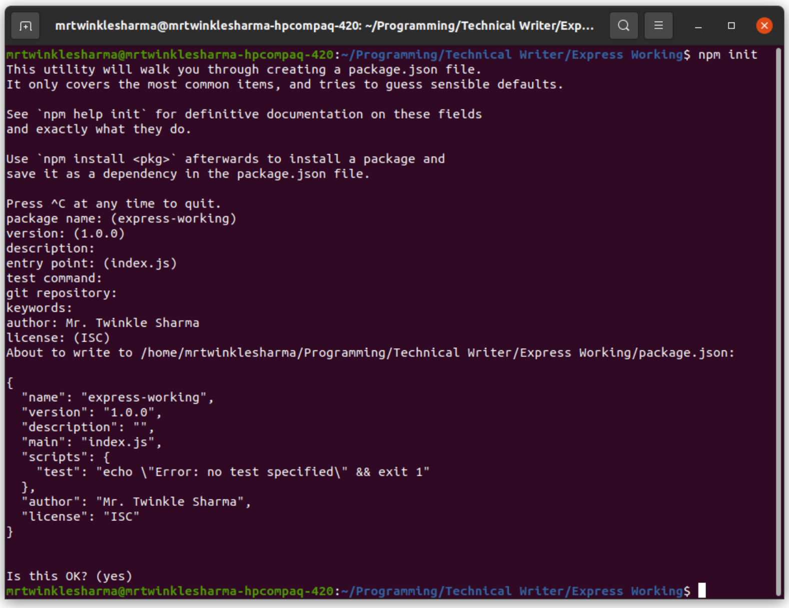
Method: The basic steps to building an express app are listed below. Here, we'll go over setting up the environment through the installation of modules, developing applications, operating web servers, and engaging in fundamental server connection. Must have a working understanding of how to install dependencies and modules via the terminal, use the node package manager for basic tasks, and have a solid understanding of ES6.

Step 1: Because our express server will function inside the node application, type this command into your terminal to build a nodejs application.

Syntax:

npm init

You can fill out the configurations requested here as needed, and you can make changes later using the package.json file.

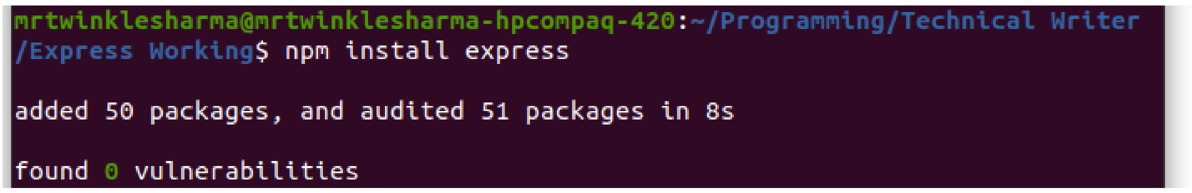

mrtwinklesharma@mrtwinklesharma-hpcompaq-420: ~/Programming/Technical Writer/Exp...

```
mrtwinklesharma@mrtwinklesharma-hpcompaq-420:~/Programming/Technical Writer/Express Working$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (express-working)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author: Mr. Twinkle Sharma
license: (ISC)
About to write to /home/mrtwinklesharma/Programming/Technical Writer/Express Working/package.json:

{
  "name": "express-working",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Mr. Twinkle Sharma",
  "license": "ISC"
}


Is this OK? (yes)
mrtwinklesharma@mrtwinklesharma-hpcompaq-420:~/Programming/Technical Writer/Express Working$
```

Step 2: Install the dependencies our programme needs.

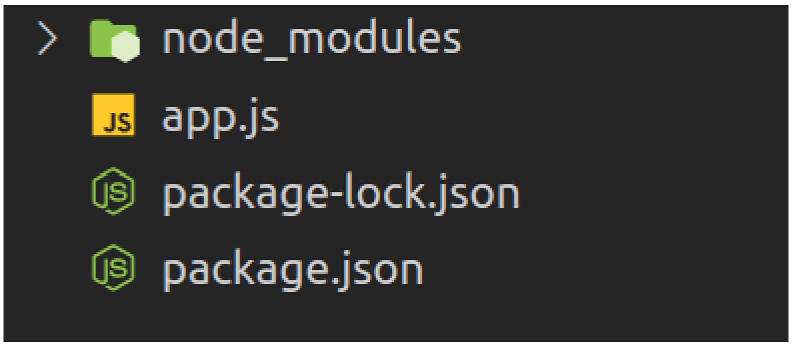
npm install express

A display similar to this will appear after a successful installation.



```
mrtwinklesharma@mrtwinklesharma-hpcompaq-420:~/Programming/Technical Writer
/Express Working$ npm install express

added 50 packages, and audited 51 packages in 8s

found 0 vulnerabilities
```

Step 3: The following project structure will be used.



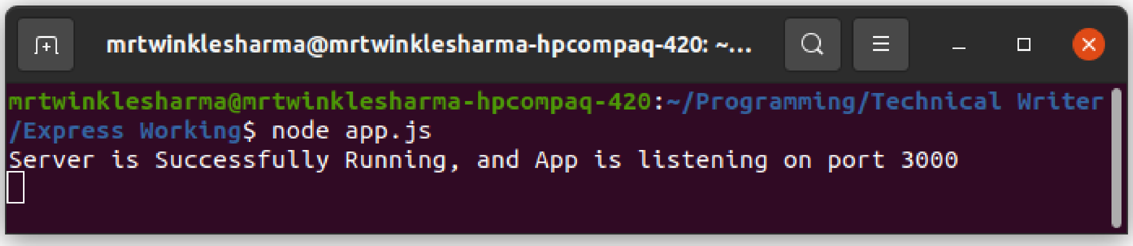
```
> node_modules
  app.js
  package-lock.json
  package.json
```

Make a file called app.js; in this article, we'll write all of the express code inside. This is how our folders will be organised. Currently in app.js, Create an app by invoking the express() function provided by the express framework after importing express with the require keyword. Set the port for our local application; the default is 3000, but you can choose any one based on the port availability. Call the listen() function; it takes two arguments: path and callback. The default host is localhost, and the default path for the local machine is localhost:3000, where 3000 is the port that we previously established. It begins listening to connections on the provided path.

```
const express = require('express');


const app = express();

const PORT = 3000;


app.listen(PORT, (error) =>


{
    if(!error)
        console.log("Server is Successfully Running,
                      and App is listening on port "+ PORT)
    else
        console.log("Error occurred, server can't start", error);
    }
);
```

Steps to execute the application: Enter the following command in your terminal to launch the express server now that we have successfully constructed a serve.

Output: You will see something like this on the terminal.



```
mrtwinklesharma@mrtwinklesharma-hpcompaq-420:~/Programming/Technical Writer
/Express Working$ node app.js
Server is Successfully Running, and App is listening on port 3000
```

Now that we have completed all of this, the server has been formed and successfully operated. If your server is not starting, an error may be there; try to investigate and decipher that problem and take the necessary corrective action.

Finally, if you try to open the URL (localhost:3000) in a browser after a successful run, it will indicate that you cannot GET / because we have not yet specified any routes for this application.

Step 4: At this point, we'll set up all of our application's routes.

Routes are the server's endpoints, which are set up on our backend server. When someone attempts to visit a route, the route responds in accordance with how it was defined in the backend. If you're just getting started, you can think of a route as a function that is invoked whenever someone requests the specific path connected with it and that responds with the value that was anticipated. For HTTP methods like get, post, put, and others, we can define routes.

Syntax:

The specified function will be executed when the path and the request method resemble. This is the basic syntax for these kinds of routes.

2)List out the core features of express framework.

Express is a Node.js web framework that is quick, forceful, necessary, and moderate. You might imagine express as a layer added to Node.js that assists in managing a server and routes. It offers a complete collection of tools for creating online and mobile applications.

Here are a few of the key components of the Express framework:

- Single-page, multi-page, and hybrid web apps can all be created with it.
- It enables middleware configuration for HTTP request responses.
- According to the HTTP method and URL, it defines a routing table that is utilised to carry out various operations.
- It enables the dynamic rendering of HTML pages by using template arguments.

Express.js tutorial explains the language's fundamental and advanced principles. Both pros and amateurs can benefit from our Express.js lesson. A web framework for Node.js is called Express.js. Its nature is fast, reliable, and asynchronous. Our Express.js tutorial covers all Express.js-related subjects, including how to install Express.js on Windows and Linux, how to use the request object and response object, how to handle cookies, how to use scaffolding, how to upload files, how to use templates, etc.

- The basic and advanced concepts of the language are covered in the Express.js tutorial. Our Express.js tutorial is useful for both experts and beginners.
- Express.js is a web framework for Node.js. Its nature is asynchronous, dependable, and quick.
- Our Express.js tutorial covers every Express.js-related topic, such as how to set up Express.js on Windows and Linux, how to work with cookies, how to utilise scaffolding, how to upload files, how to use templates, etc.
- The Express.js lesson covers both the language's fundamental and more complex ideas. Both beginners and experts can benefit from our Express.js lesson.
- A web framework for Node.js is called Express.js. Its nature is trustworthy, fast, and asynchronous.
- Every Express.js-related topic is covered in our guide, including how to install Express.js on Windows and Linux, how to use cookies, how to utilise scaffolding, how to upload files, how to use templates, etc.

3)What is npm?

The Node Package Manager is known as Npm. It serves as the Node JavaScript platform's package manager.

The largest software registry in the world is referred to as Npm. Npm is a publishing and sharing tool used by open-source developers worldwide.

Npm is made up of three parts:

- You can search for third-party packages on the website, create profiles, and manage your packages.
- You can communicate with npm through the command-line interface, sometimes known as the npm CLI, which is run from a terminal.
- The registry is a sizable online repository for JavaScript source code.

You can install a new package from the registry using npm. With npm, you will mostly use it for this purpose. Additionally, npm enables you to find and share your fresh node packages.

Packages are a common tool for building and scaling websites, software, and other digital products in many programming languages. With the use of these packages, engineers can increase a project's capabilities without creating and maintaining new code.

You've undoubtedly overheard engineers discussing NPM, Node.js packages, or a package management if your firm employs Node.js. To help you understand how these phrases function and why engineers use them, this article will define them from the standpoint of a designer.

Utilize UXP in Merge, a ground-breaking technology that enables designers to create user interfaces with fully functional code components, to bridge the gap between design and development.

The open-source Node Package Manager (NPM) is a collection of tools used by programmers to create websites and applications.

Two items comprise NPM:

- A location where open-source projects can be published. A digital storage and retrieval system, to put it simply
- An interactive command-line interface (CLI) for the repository A tool to communicate with the storage facility, to put it simply

4)Discuss about Node.js Frameworks?

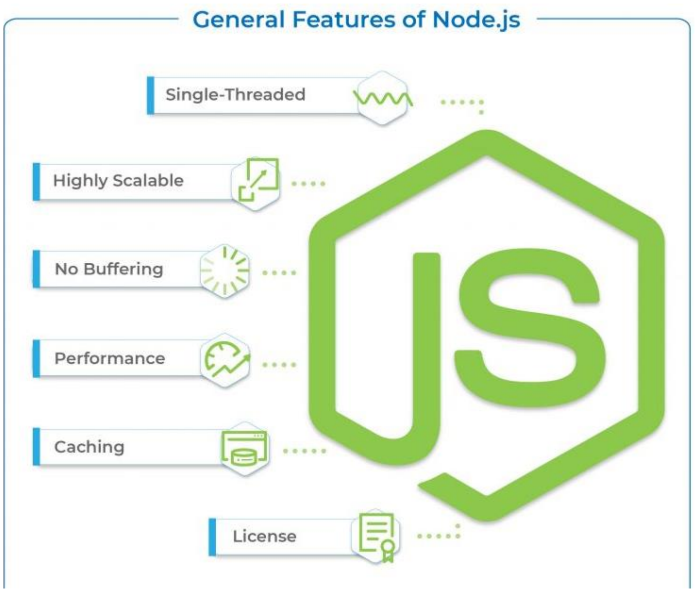
A framework is a collection of different libraries and tools that are needed during the software application development process. It serves as a foundation for the creation of various software programmes. A Node.js workspace platform called a node framework enables developers to create both the front end and the back end of an application using JavaScript. Node frameworks are a diverse group of frameworks that are based on Node and further its features and functions.

Node.js is an open-source server environment that works with Windows, Linux, Unix, Mac OS X, and other operating systems. It utilises Chrome's V8 JavaScript engine, is free, and is developed in JS.

On its website, Nodejs is described as follows:

"Nodejs is a framework for creating quick and scalable network applications that is based on Chrome's JavaScript runtime. Nodejs is a JavaScript runtime that is event-driven and asynchronous, making it possible to create robust network applications. Nodejs users don't have to worry about dead-locking the process because there are no locks. Almost no Node.js function does I/O directly, so the process never blocks until synchronous Node.js standard library functions are used to conduct the I/O. Scalable systems may be created in Nodejs extremely easily because nothing blocks.

This cross-platform runtime tool was created by Ryan Dahl to help programmers create networking and server-side applications. Nodejs streamlines development by providing a large selection of JS modules that allow programmers to build web apps with greater precision and less stress.


General Features of Node.js
Single-Threaded
Highly Scalable
No Buffering
Performance
Caching
License
JS