Assignment:2

Aim:

Enhancing the Web Server with Express.js

Hardware Requirement:

- **Laptop: HP Omen 15**
  - **RAM: 16 GB**
  - **Storage: 1 TB SSD**
  - **GPU: RTX 3060**
  - **Processor: Ryzen 7 5000 series**
  - **Operating System: Windows 11**

Software Requirement:

- **Programming Language: Node JS**
- **Integrated Development Environment (IDE): VS Code**

Knowledge Requirement:

- **Understanding of Web Fundamentals**
- **Familiarity with JavaScript, Node JS**

Theory:

Express.js, commonly referred to as Express, is a powerful and widely-used web application framework for Node.js, offering a streamlined and minimalist approach to building robust and scalable web applications. As an unopinionated framework, Express provides developers with the flexibility to structure their applications while offering essential features and middleware for routing, handling HTTP requests and responses, and managing application states.

Importing Express:

- const express = require("express");
- const app = express();

Defining Routes:

- The app.get() method is used to define routes based on the HTTP GET method.
- Three routes are defined: '/' for the home page, '/About' for the about page, and '/Contact' for the contact us page.
- Each route has a callback function with req (request) and res (response) parameters. The callback function sends a simple text response using res.send().

Listening on a Port:

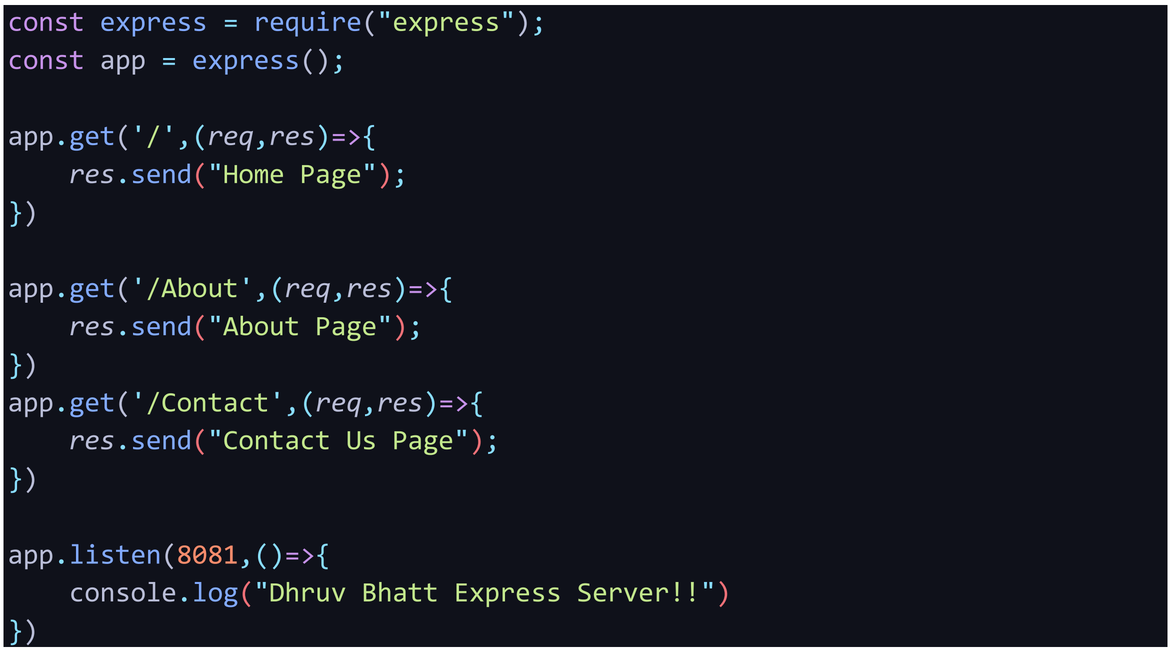
- The app.listen() method binds and listens for connections on the specified port (8081 in this case).

Making the server listen:

- **app.listen(port, [hostname], [callback])**: Makes the server listen on a specified port and hostname.
- port: The port number to listen on.
- hostname: The hostname or IP address to bind the server to. (Optional, defaults to localhost).
- callback: An optional function to run when the server starts listening.

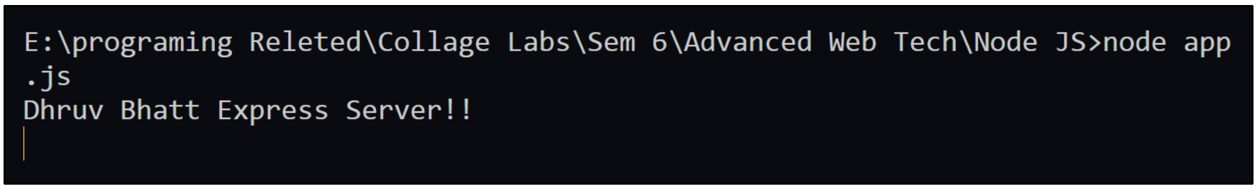
Code and Output:

- **Code:**

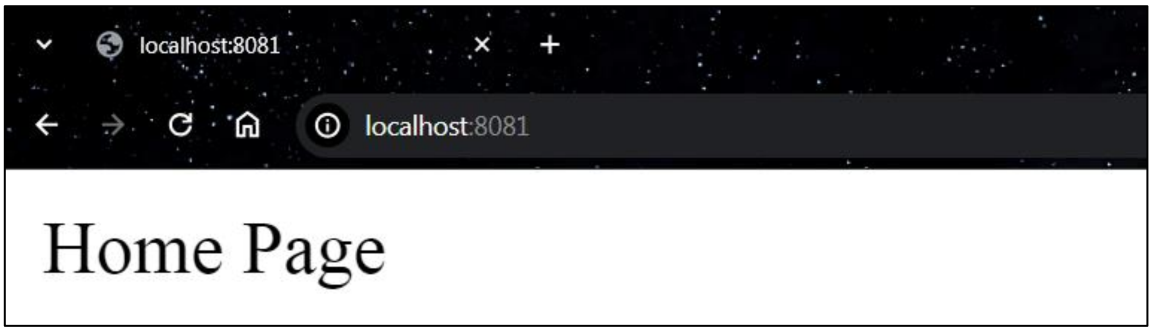


```
const express = require("express");
const app = express();

app.get('/',(req,res)=>{
    res.send("Home Page");
})

app.get('/About',(req,res)=>{
    res.send("About Page");
})
app.get('/Contact',(req,res)=>{
    res.send("Contact Us Page");
})

app.listen(8081,()=>{
    console.log("Dhruv Bhatt Express Server!!")
})
```

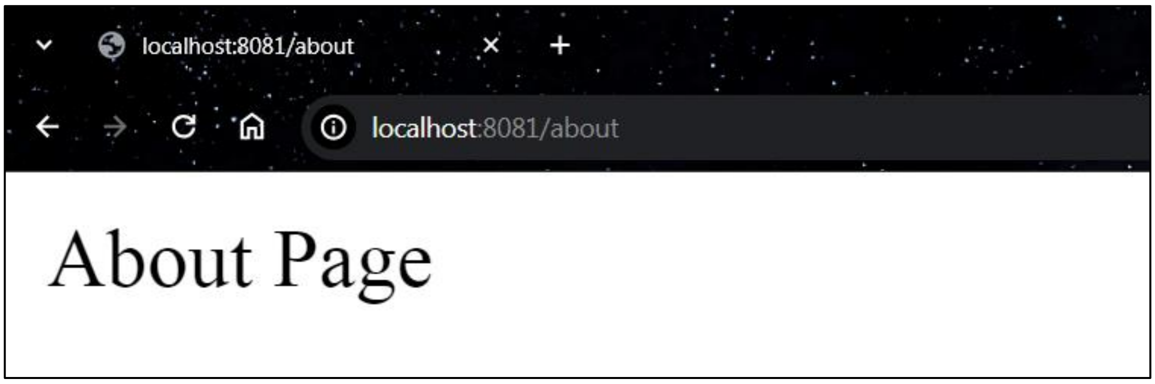
- **Output:**



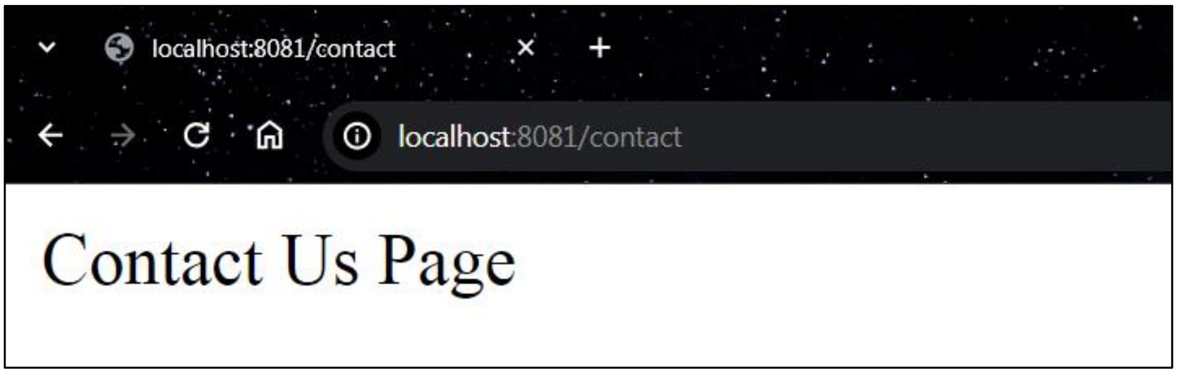
```
E:\programing Releted\Collage Labs\Sem 6\Advanced Web Tech\Node JS>node app
.js
Dhruv Bhatt Express Server!!
```



Home Page



About Page



Contact Us Page

Conclusion:

Express.js framework for creating web servers in Node.js. The simplicity of Express.js is evident as it enables the definition of routes with ease, specifying distinct responses for various HTTP GET requests. The code showcases three routes, namely the home page ('/'), the about page ('/About'), and the contact us page ('/Contact'), each responding with straightforward text messages. The usage of the app.listen() method allows the server to listen on port 8081.

References:

https://www.javatpoint.com/expressjs-tutorial

https://www.geeksforgeeks.org/express-js/

https://www.tutorialspoint.com/expressjs/index.htm

https://expressjs.com/

https://medium.com/geekculture/create-a-basic-server-with-express-js-really-basic-but-delicious-c5cceaca1c60