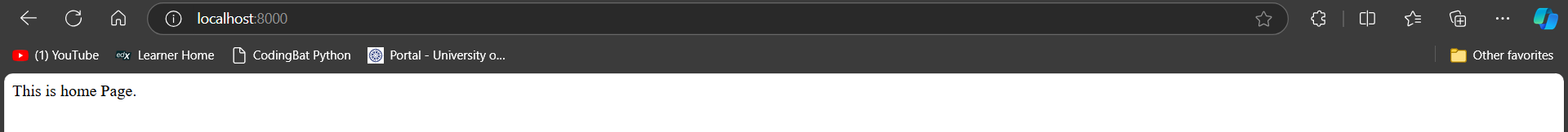
## Screen Shot 1

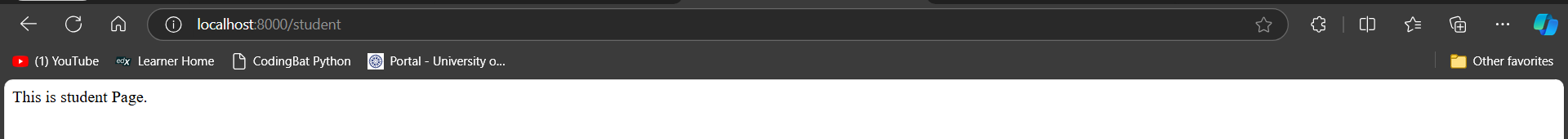
A black screen with a black background

Description automatically generated

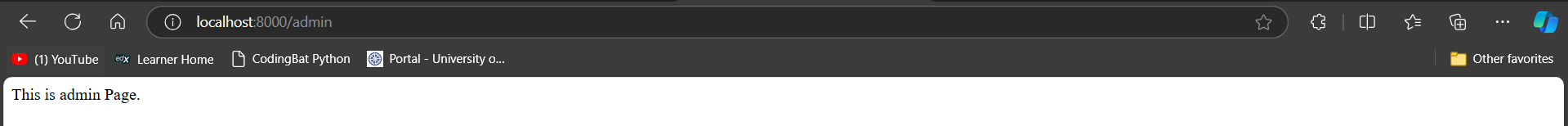
## Screen Shot 2



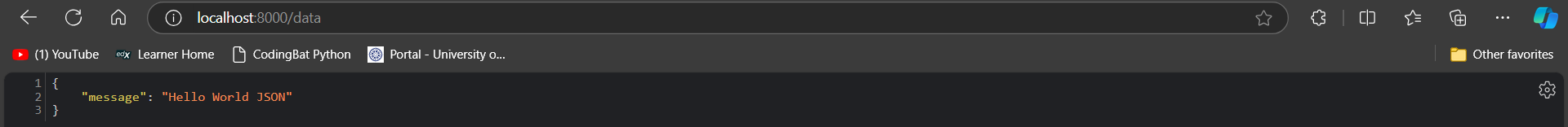
## Screen Shot 3



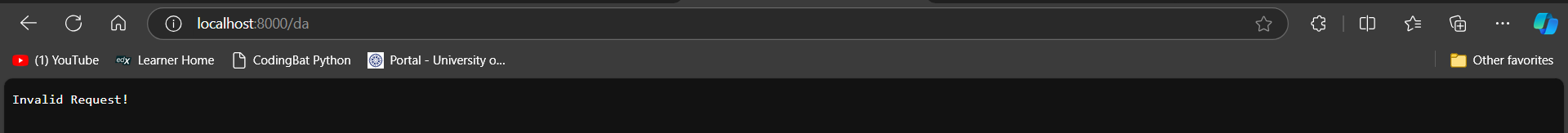
## Screen Shot 4



## Screen Shot 5



## Screen Shot 6



## Task 4:

Node.js enables the creation of efficient web servers to handle HTTP requests for web applications. Typically, web applications rely on web servers to manage these requests. For example, ASP.NET web applications commonly use IIS (Internet Information Services), while PHP and Java applications often use Apache as their web server. However, Node.js introduces its own capabilities to create a web server, handling HTTP requests asynchronously. Although Node.js applications can be run on traditional web servers like IIS or Apache, using Node.js’s built-in web server is generally recommended due to its efficiency and compatibility.

Node.js includes a built-in HTTP module that allows developers to create their own web servers without needing external software. This module enables the transfer of data over the Hyper-Text Transfer Protocol (HTTP). By using the require() method, the HTTP module can be included in a Node.js script.

With the HTTP module included, the createServer() method can be used to establish a web server that listens to incoming requests on a specific port and responds to the client accordingly. This asynchronous handling of HTTP requests makes Node.js well-suited for scalable and real-time applications, as it ensures non-blocking operations, enabling the server to handle multiple concurrent requests efficiently.