Adding routes to your web server

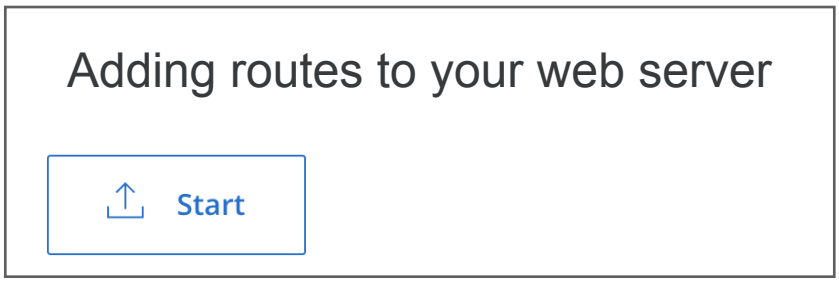
Welcome to this Lab activity

In this lab activity you will be exploring how to add new routes to your Express web application. So far all the web servers that you created using either the **http** module or the **Express** framework only had one route; the root route "/". Although nothing will stop you from having single route applications, the majority of web applications utilise multiple routes to organise their content.
If you take a website as an example, there are indeed single route websites on the web that organise their content with a single web page but there are also websites that organise their content using multiple routes; "https://website", "https://website/about", "https://website/search" and so on.
Let's dive into this lab activity and add some routes to your web server application!

Start the Lab environment application

It is simple to launch a lab exercise. You only need to click on the button "Start" below the activity title to enter a lab environment.

Let's explore this lab activity. Go ahead and click on the "Start" button!



Task 1: Adding new routes to your express server

Although you have now reached *topic3* of this course, in this lab activity you will be working inside the *topic2/myFirstExpress* folder as you will only need to add a new route to your existing Express web server.

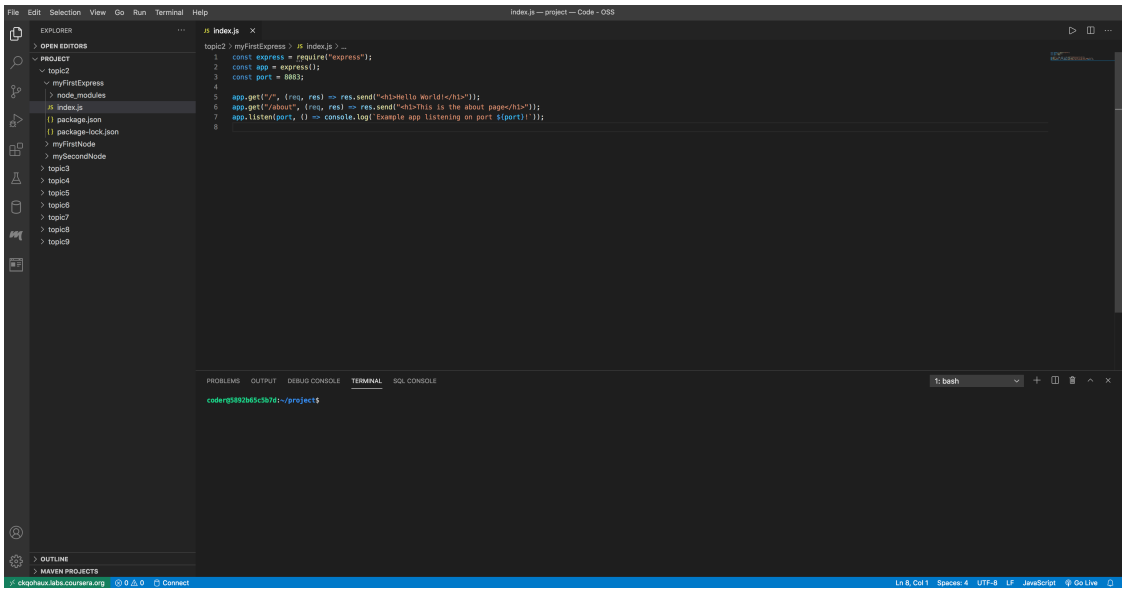
Let's add some code to the *index.js* file located inside the *topic2/myFirstExpress* folder in order to add your new route.

Add the following code to the *index.js* file and remember to save it:

app.get("/about", (req, res) => res.send ("<h1>This is the about page</h1>"));

The above line of code will add the "/about" route to your web server application.

If you have correctly followed all the above steps, your environment should be similar to the one below:



Running the index.js file

Run the *index.js* file with the following Terminal command:

- **node index.js**: type this command and press Enter. The node command, followed by the file name, tells Node.js to execute the content of the file.

The above command will start a web server running on port 8083.

Task 2: Access your server via HTTP

Now that your code is running, serving your application on port 8083, you can access your web page from a browser!

Use the “Browser Preview” plugin to visualise your web application.

If you do not remember how to use the “Browser Preview” plugin, please refer to the “Creating my first Node.js web server” lab instructions.

Type *localhost:8083* on the “Browser Preview” tab and press *Enter* on your keyboard to visualise your web application “root” page:

Type *localhost:8083/about* on the “Browser Preview” tab and press *Enter* on your keyboard to visualise your web application “about” page:

End of Section

Congratulations for completing this section. As long as you have saved your work, your files will remain when you close this lab activity so do not worry about losing your data. What you have written so far are all static pages; in the next lab activities we will discuss how you can make dynamic applications.