Guided Lab: Creating an AWS Lambda Function to Return an HTML Page

Description

AWS Lambda is a serverless compute service that allows you to run code without provisioning or managing servers. With Lambda, you can execute your code in response to events, such as HTTP requests, changes to data in S3, or updates in DynamoDB, without worrying about the underlying infrastructure.

Function URL is a simple and scalable way to expose a Lambda function to HTTP(s) requests without the need to set up an API Gateway. This is especially useful for simple use cases where you just need a public URL to invoke a Lambda function directly.

In this lab, we will create a Lambda function that returns an HTML page about the Philippines, showcasing how to serve static content from a Lambda function using a Function URL.

Prerequisites

This lab assume you have basic understanding of Basic understanding of JavaScript (Node.js) and Familiarity with HTML and CSS., Lambda functions and Function URLs.

If you find any gaps in your knowledge, consider taking the following lab:

- Creating a NodeJS Function in AWS Lambda
- Invoking Lambda functions through Function URL

Objectives

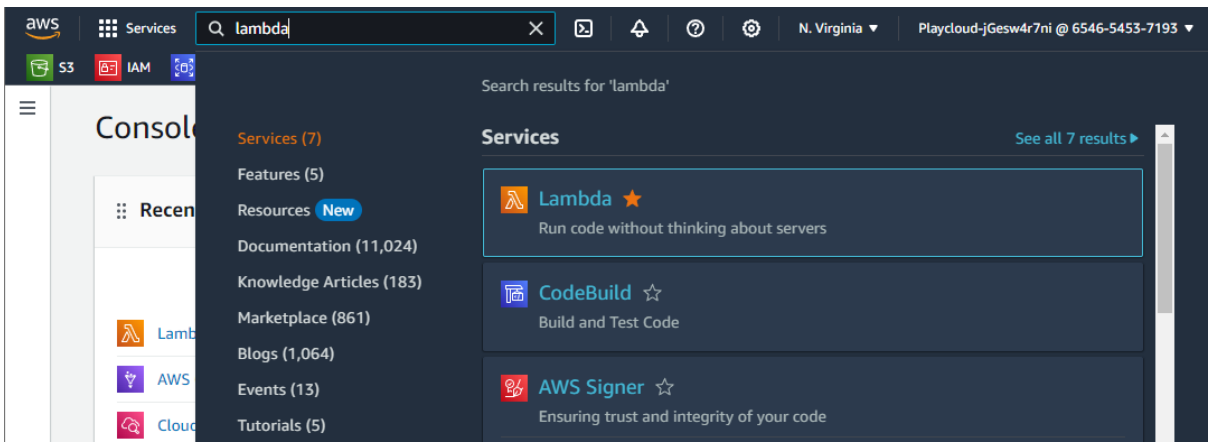
By the end of this lab, you will:

- Create an AWS Lambda function that returns a static HTML page.
- Serve the HTML content through a Function URL.
- Test the Lambda function using the provided Function URL.

Lab Steps

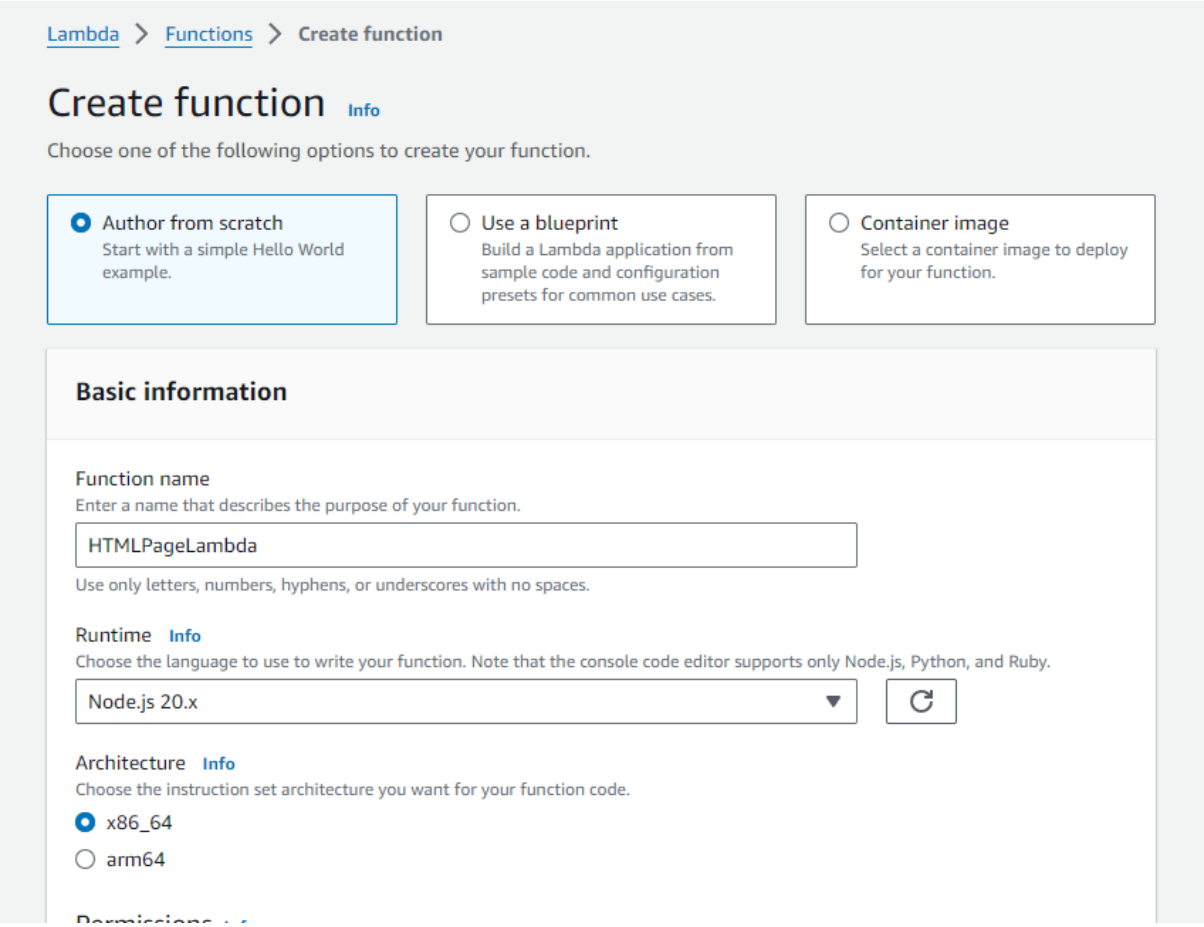
Creating the AWS Lambda Function

1. Navigate to AWS Lambda Console

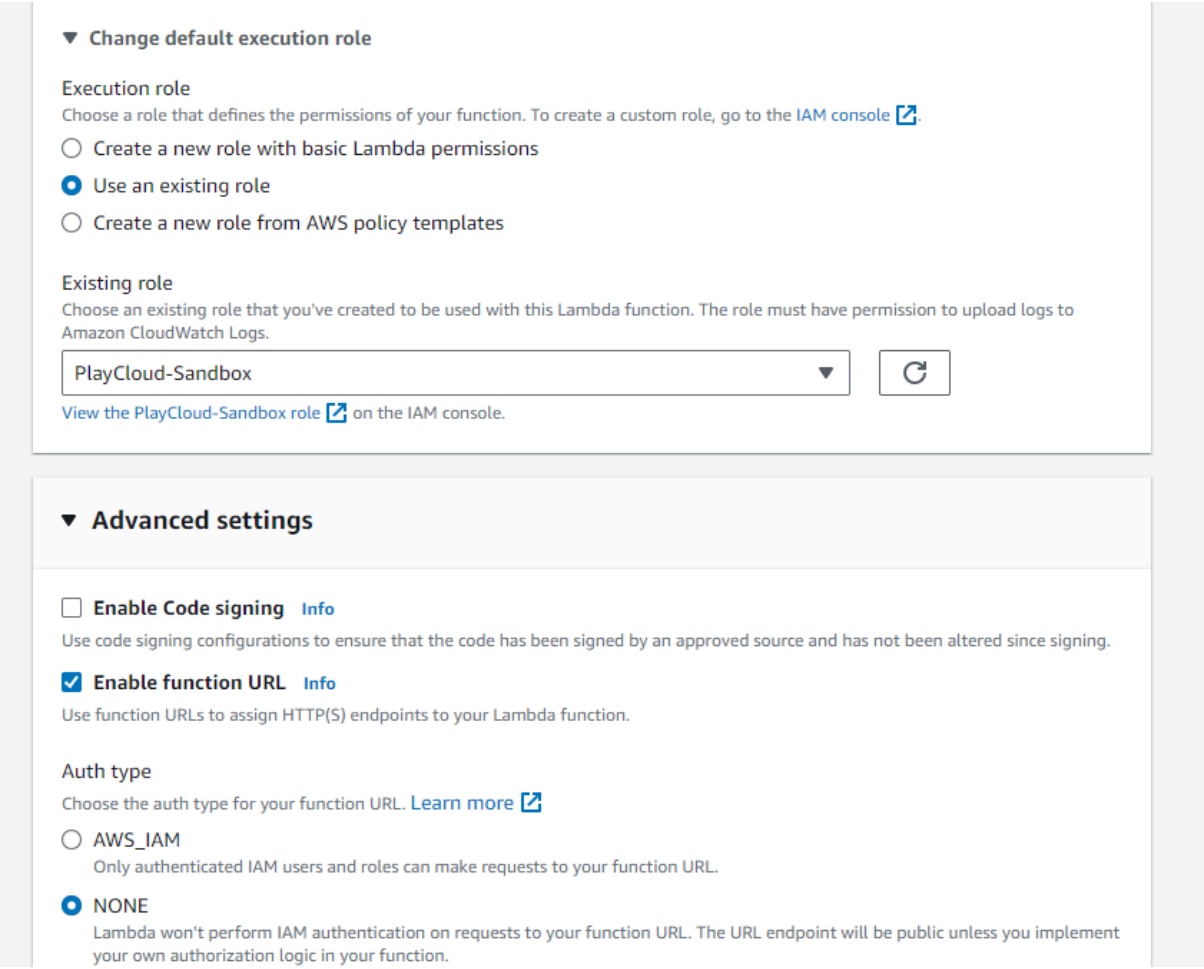


2. Create Function using the following confgurations:

- Choose Author from scratch.
- Function name: HTMLPageLambda
- Select Node.js 20.x as the runtime.



- **Execution role:**
    - Select Use an Existing Role: PlayCloud-Sanbox
- **Advance settings:**
    - Checked **Enable function URL**
        - Auth type: NONE

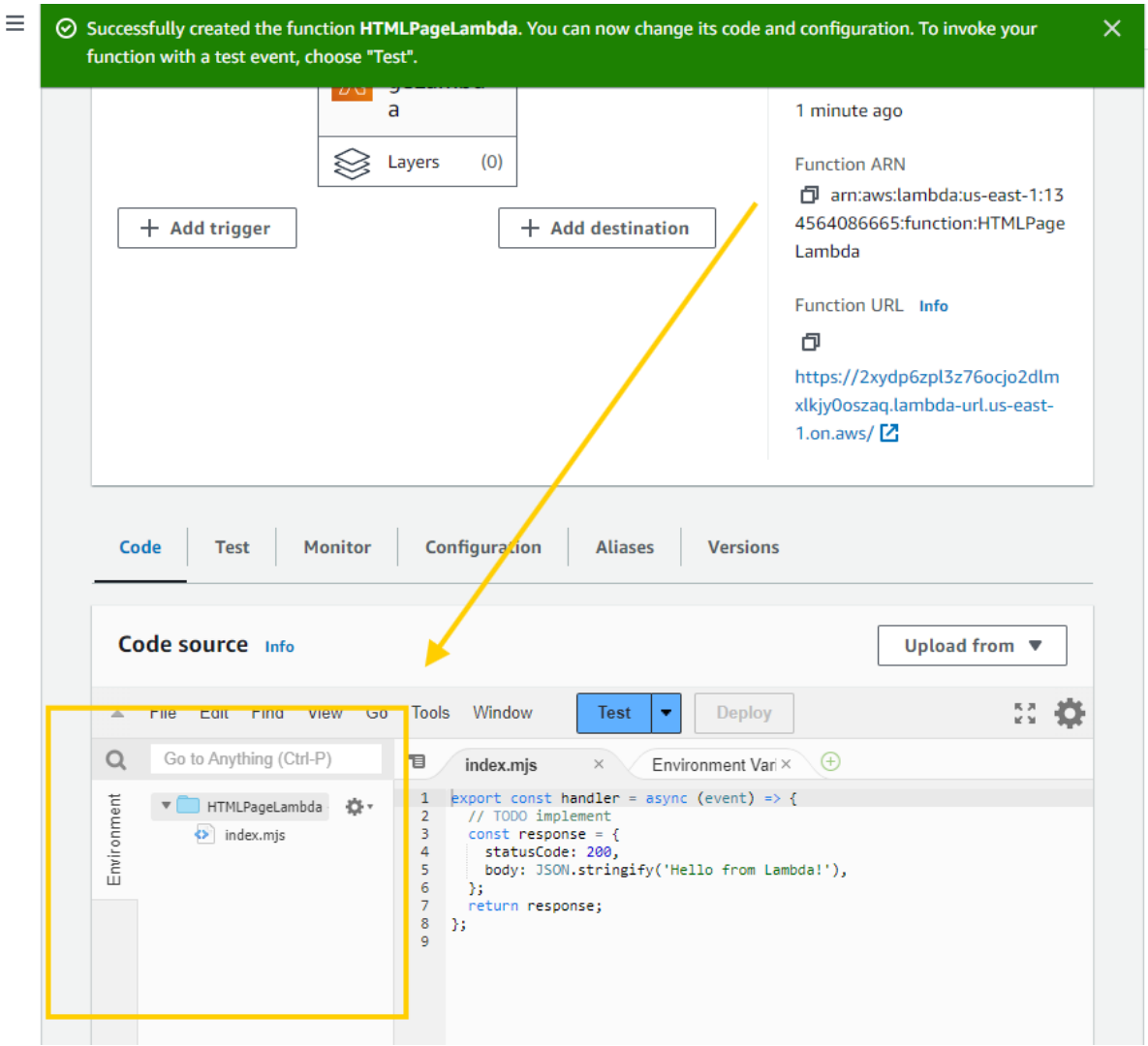


- Click on **Create Function**

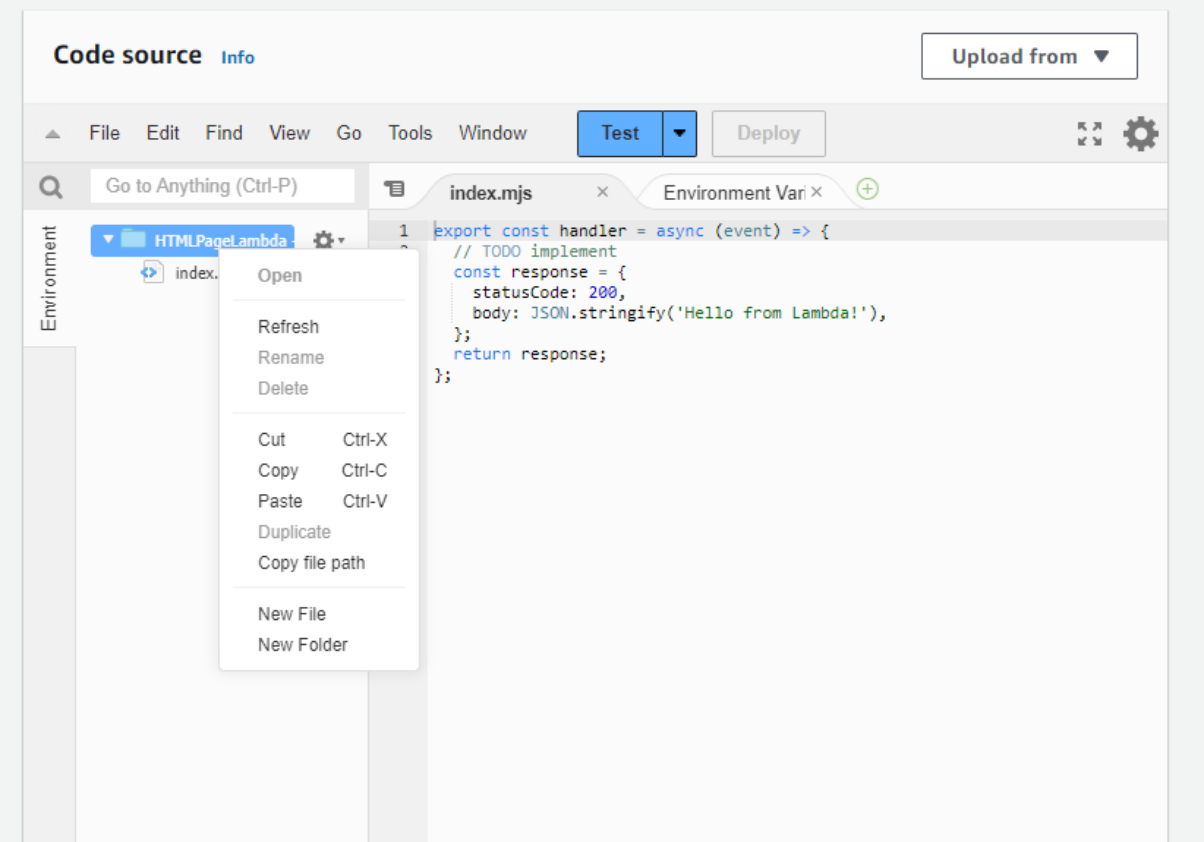
Creating the `index.html` File

1. In the Code editor section, look at the Environment where your index.mjs is:

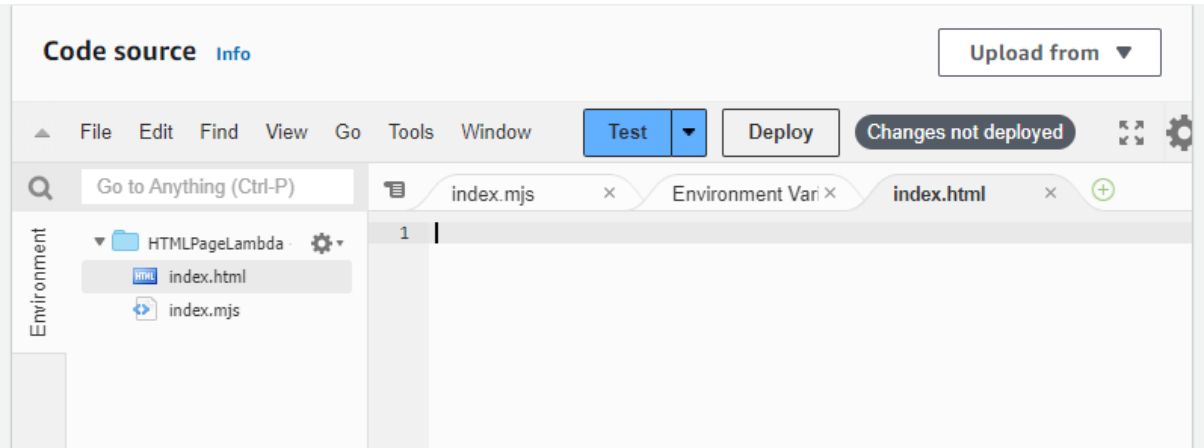
***Note**: We are using the **old console editor** for this lab. You're welcome to use either the old or new editor, whichever you prefer; the steps remain the same, though the interface may have a slightly different appearance in the new editor.*



2. Right click on the `HTMLPageLambda` Folder. Select `New File`



3. Create a new file named `index.html` and then open the file

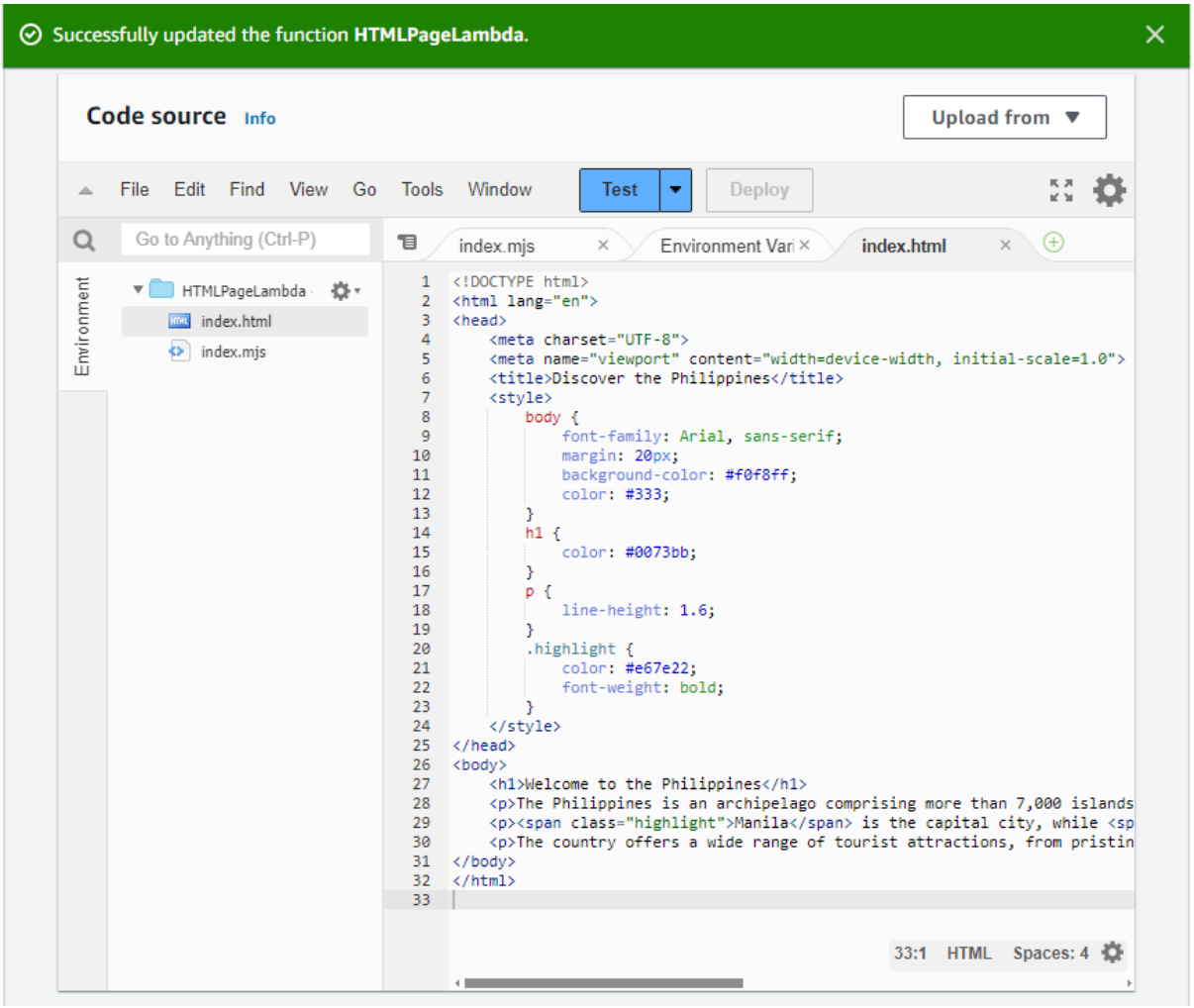


4. Add the following content to the file:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Discover the Philippines</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 20px;
```

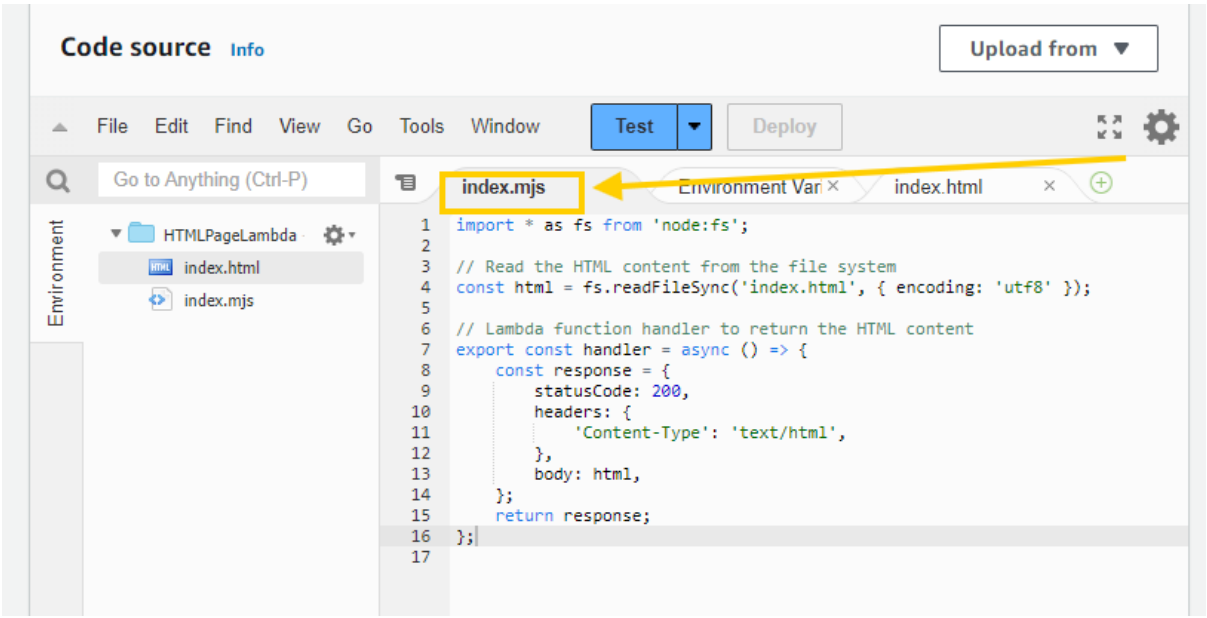
```
        background-color: #f0f8ff;
        color: #333;
    }
    h1 {
        color: #0073bb;
    }
    p {
        line-height: 1.6;
    }
    .highlight {
        color: #e67e22;
        font-weight: bold;
    }
  </style>
</head>
<body>
  <h1>Welcome to the Philippines</h1>
  <p>The Philippines is an archipelago comprising more than 7,000 islands, known for its rich biodiversity, vibrant culture, and stunning landscapes.</p>
  <p><span class="highlight">Manila</span> is the capital city, while <span class="highlight">Cebu</span> and <span class="highlight">Davao</span> are major urban centers.</p>
  <p>The country offers a wide range of tourist attractions, from pristine beaches to historical landmarks.</p>
</body>
</html>
```

5. Click on **Deploy** to save changes.



Modifying the mjs Code

1. **Navigate back to the index.mjs**



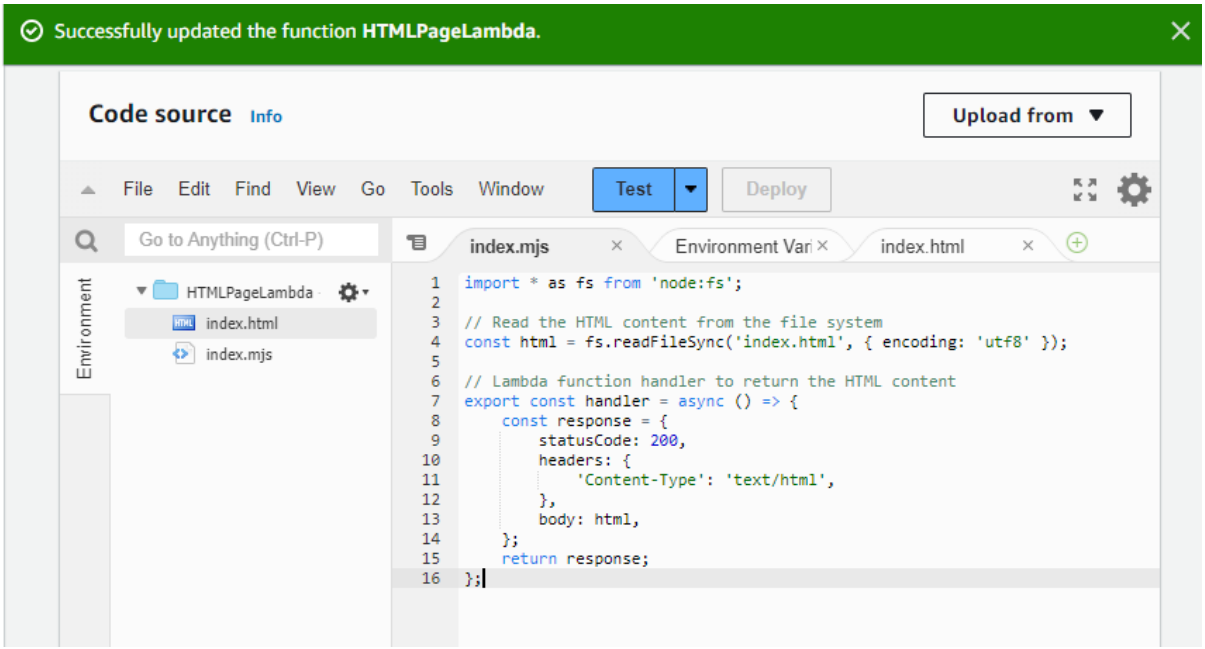
2. **Add the Following Code to** index.mjs:

```
import * as fs from 'node:fs';


// Read the HTML content from the file system
const html = fs.readFileSync('index.html', { encoding: 'utf8' });


// Lambda function handler to return the HTML content
export const handler = async () => {
   const response = {
      statusCode: 200,
      headers: {
         'Content-Type': 'text/html',
      },
      body: html,
   };
   return response;
};
```

- The fs.readFileSync() function reads the contents of the index.html file into a string.
- The Lambda function's handler returns this string as the body of the HTTP response, with the Content-Type set to text/html, ensuring that the browser interprets it as an HTML document.

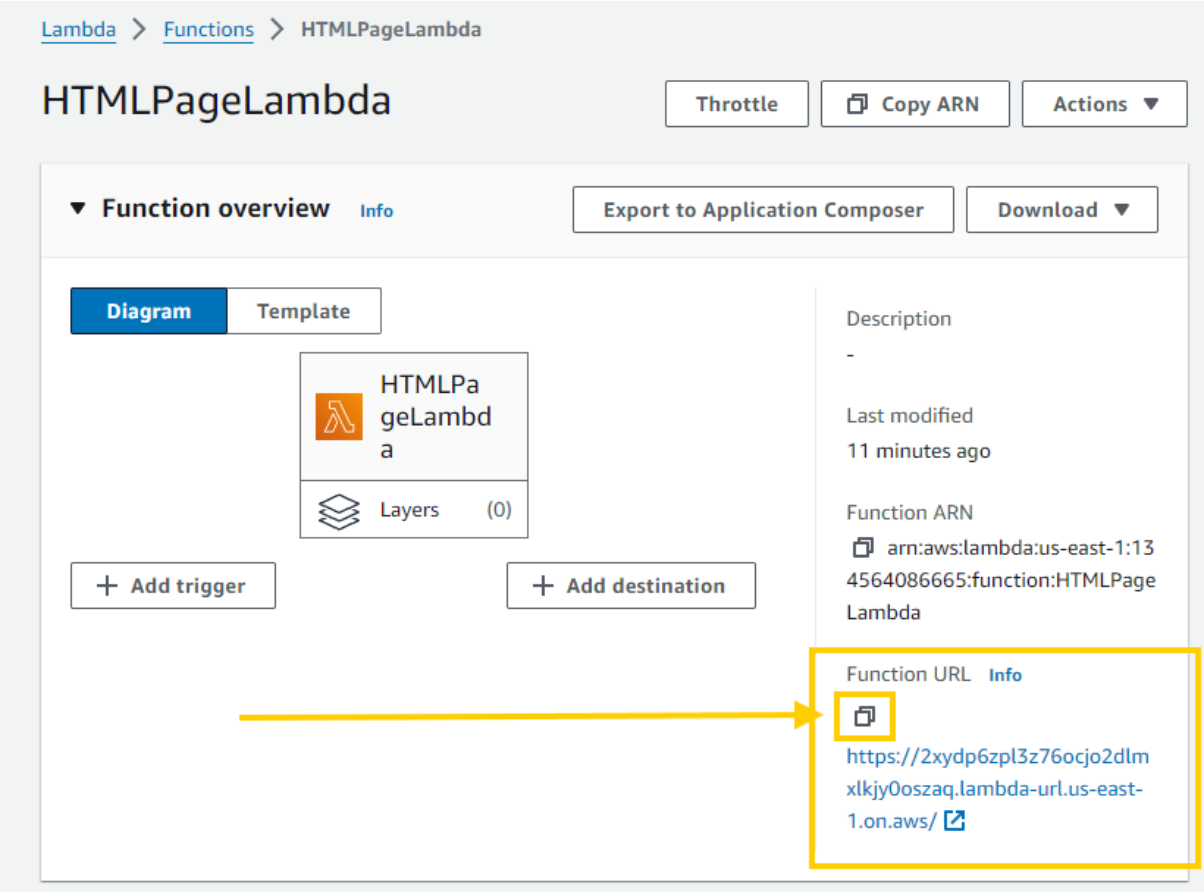
---

3. Click on **Deploy** to save the changes

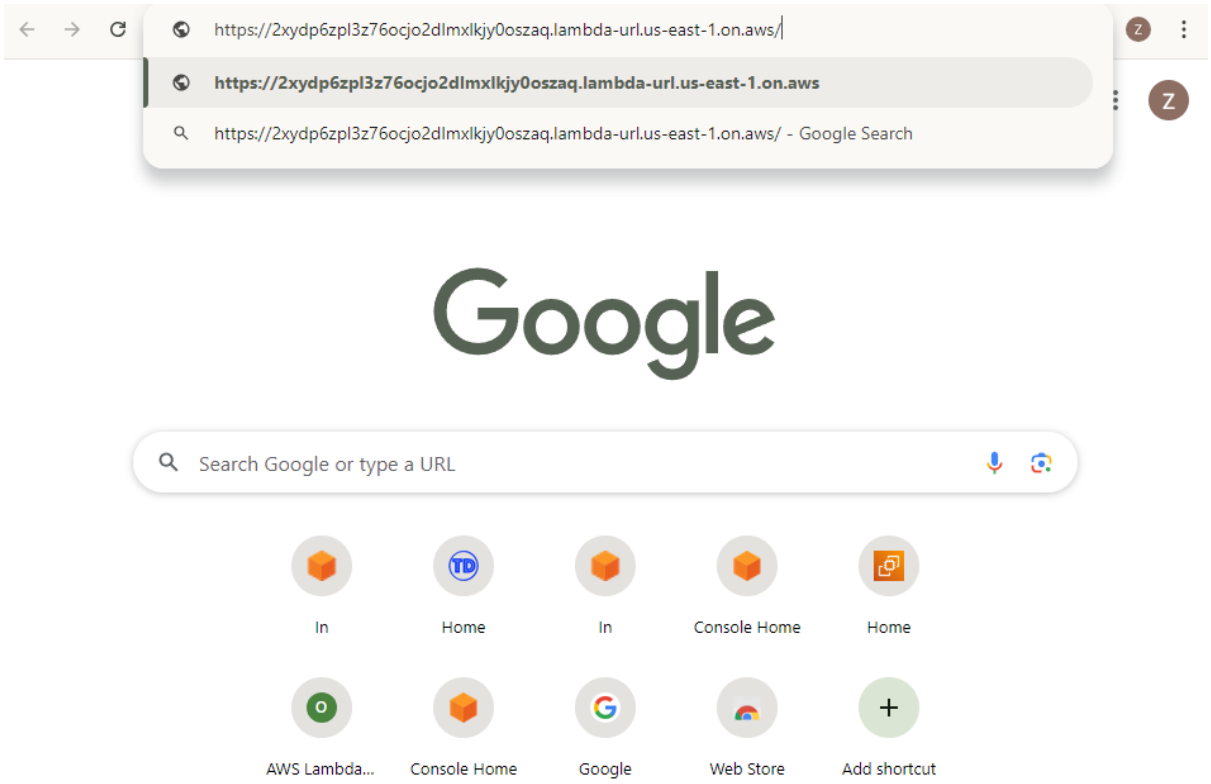


Testing the Lambda Function

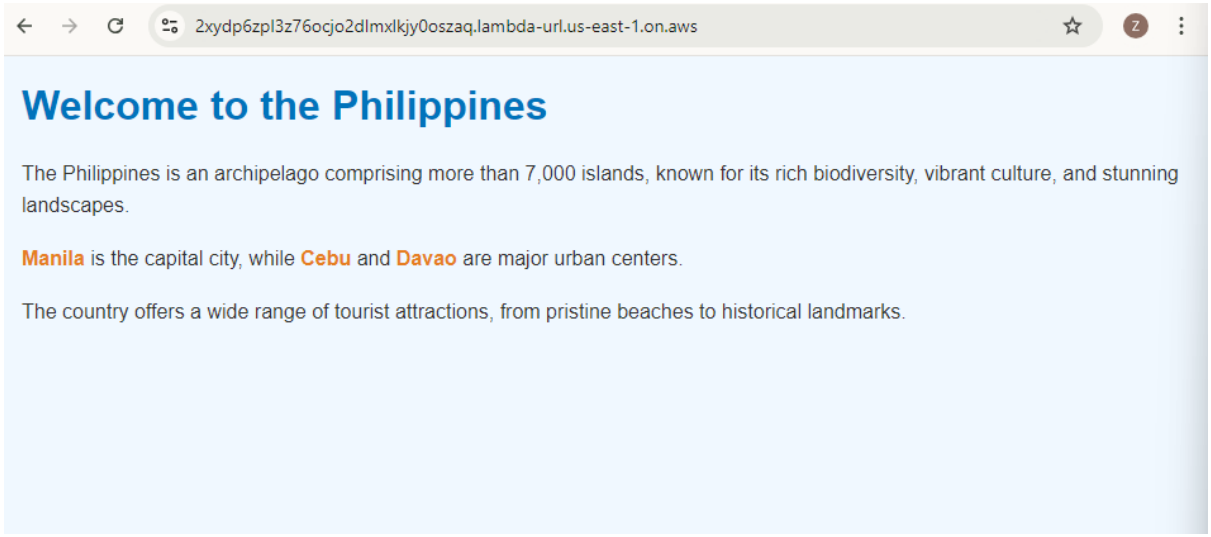
1. Scroll up and copy the Function URL from the Lambda console.



2. Paste it into your web browser's address bar and press **Enter**.



3. You should see the HTML page about the Philippines displayed in your browser.



That's it! Congratualatioons! You have successfully created an AWS Lambda function that serves a static HTML page about the Philippines. You learned how to handle static content by reading it from a file system and how to expose this content using a Function URL, making it publicly accessible via HTTP requests.

This lab demonstrates the simplicity and power of AWS Lambda for serving static content without the need for traditional web servers, showcasing a lightweight and scalable way to deliver web pages. By applying these concepts, you can extend this solution to serve more complex or dynamic content, integrate it with other AWS services, or even build full-fledged web applications using serverless architecture. Happy learning!