

Implemented AWS Lambda to enable serverless computing, running code in response to events, managing compute resources automatically, and scaling applications without server management.

The screenshot displays the AWS Lambda console interface. At the top, the 'Create function' wizard is active, showing three options: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. The 'Basic information' section is expanded, showing the function name 'test', the runtime 'Node.js 22.x', and the architecture 'x86\_64'. The 'Permissions' section is also visible at the bottom. On the right side, a 'Tutorials' sidebar is open, displaying a tutorial for 'Create a simple web app'.

**Get started**

Author a Lambda function from scratch, or choose from one of many preconfigured examples.

[Create a function](#)

**How it works**

[Run](#) [Next: Lambda responds to events](#)

**Basic information**

**Function name**  
Enter a name that describes the purpose of your function.

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).

**Runtime** [Info](#)  
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

**Architecture** [Info](#)  
Choose the instruction set architecture you want for your function code.  
☒ x86\_64  
☐ arm64

**Permissions** [Info](#)

**Info** **Tutorials**

Learn how to implement common use cases in AWS Lambda.

**Create a simple web app**

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

[Start tutorial](#)

Implemented AWS Lambda to enable serverless computing, running code in response to events, managing compute resources automatically, and scaling applications without server management.

The screenshot displays the AWS Lambda console interface for a function named 'test'. The top navigation bar shows the AWS logo, a search bar, and various service icons. The main content area is divided into two sections: 'Function overview' and 'Code source'.

**Function overview:** This section provides a high-level view of the function. It includes a 'Diagram' tab showing the function's architecture and a 'Template' tab. The function is named 'test' and has no layers. A green notification banner at the top states: 'Successfully created the function test. You can now change its code and configuration. To invoke your function with a test event, choose "Test".' The 'Description' field is empty. The 'Last modified' timestamp is '1 second ago'. The 'Function ARN' is 'arn:aws:lambda:ap-south-1:221082205626:function:test'. The 'Function URL' is also empty. Buttons for 'Throttle', 'Copy ARN', 'Actions', 'Export to Infrastructure Composer', and 'Download' are visible.

**Code source:** This section shows the source code for the function. The 'EXPLORER' pane on the left lists the files 'index.mjs' and 'test'. The 'index.mjs' file is selected, and its content is displayed in the main editor. The code is a JavaScript function that returns a JSON response with a status code of 200 and a body of 'Hello from Lambda!'. The 'DEPLOY' section on the left includes buttons for 'Deploy (Ctrl+Shift+U)' and 'Test (Ctrl+Shift+I)'. The 'TEST EVENTS' section is currently empty.

The bottom of the screenshot shows the Windows taskbar with various application icons and the system clock indicating 11:37 on 05-01-2025.