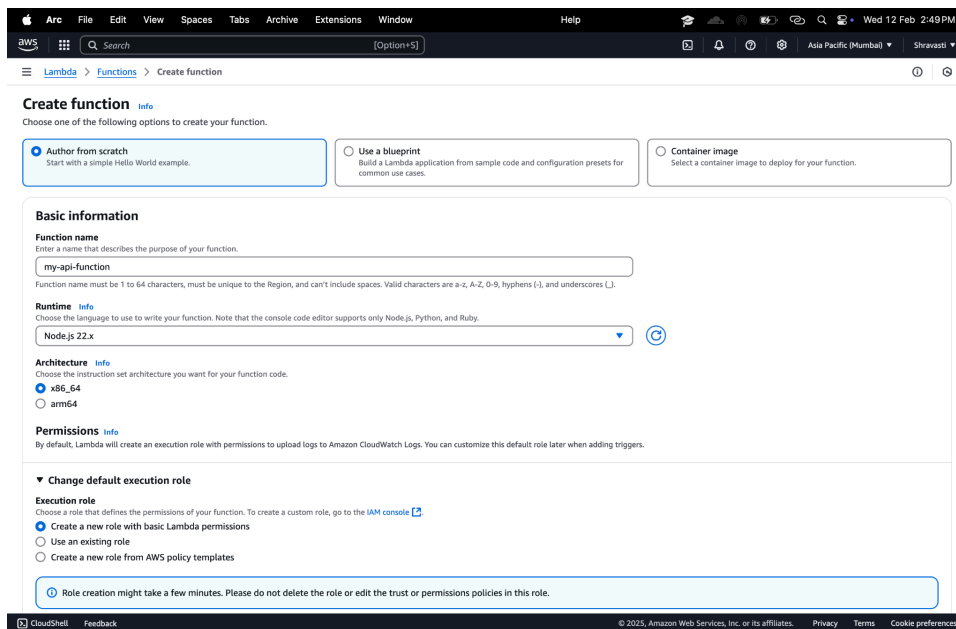


CBE 321 LAB EXAM

2022BCY0012

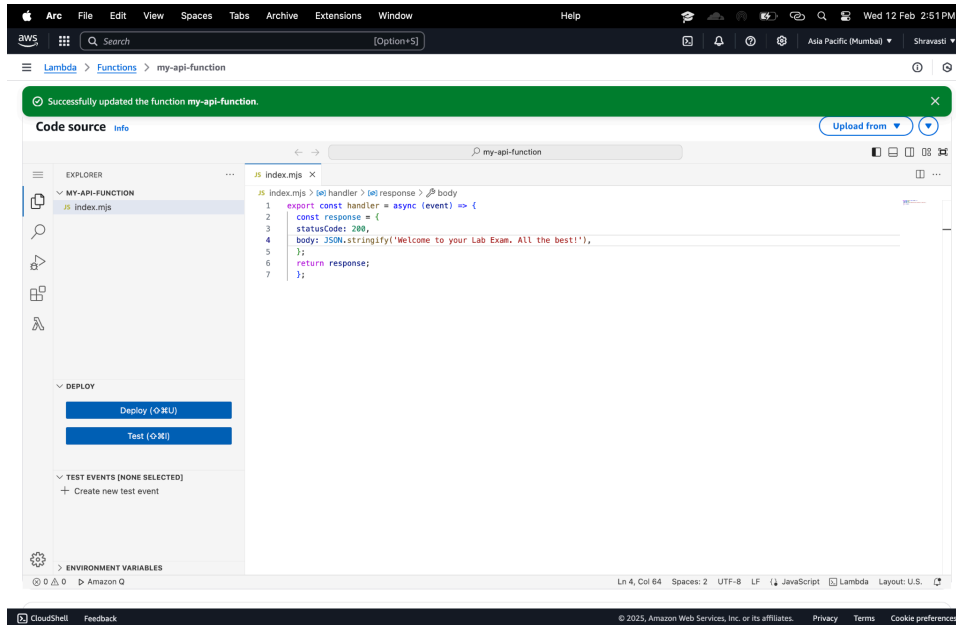
Q. Create an API Gateway that displays 'Welcome' on the browser

1. Create a lambda function, name it (my-api-function) and set its runtime.

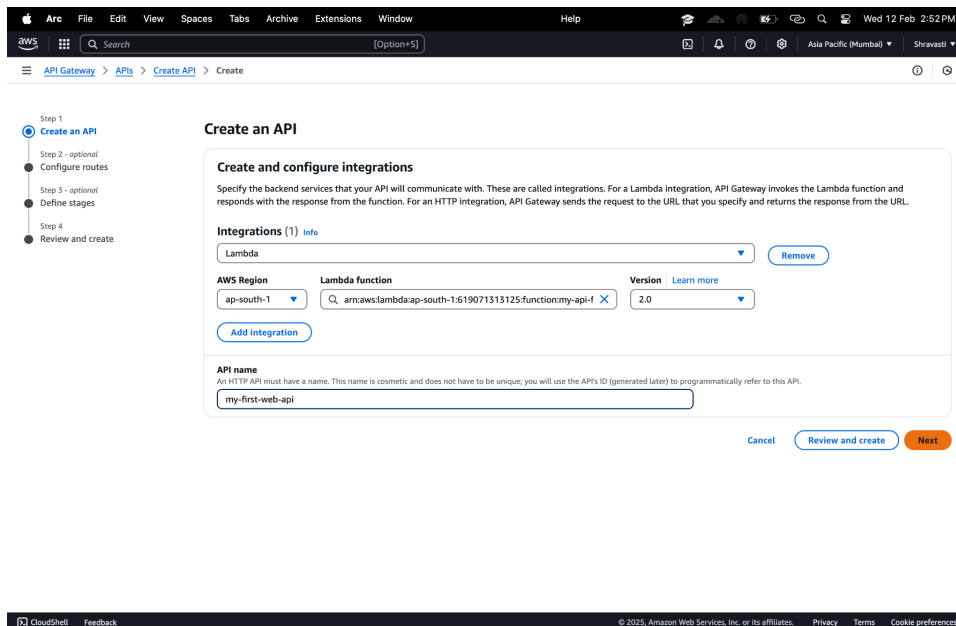


The screenshot shows the AWS Lambda 'Create function' console page. At the top, there's a navigation bar with 'Lambda' and 'Functions' tabs, and a 'Create function' button. Below this, the 'Create function' section has three options: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. The 'Author from scratch' option is highlighted with a blue border. Below the options, the 'Basic information' section contains a 'Function name' field with the value 'my-api-function', a 'Runtime' dropdown menu set to 'Node.js 22.x', and an 'Architecture' dropdown menu set to 'x86_64'. The 'Permissions' section is expanded, showing the 'Change default execution role' section with three options: 'Create a new role with basic Lambda permissions' (selected), 'Use an existing role', and 'Create a new role from AWS policy templates'. A note at the bottom of the permissions section states: 'Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.'

2. Once the function is created, add the code corresponding to displaying the given message on the browser



3. Go to API Gateway and create an HTTP API. Choose integration as 'Lambda', choose your lambda function and name the API.



4. Configure the routes

The screenshot shows the AWS API Gateway console in the 'Create API' wizard. The left sidebar indicates the current step is 'Configure routes' (Step 2 - optional). The main content area is titled 'Configure routes - optional' and includes an 'info' icon. Below the title, a paragraph explains that API Gateway uses routes to expose integrations to consumers of your API, and that routes consist of an HTTP method and a resource path. The form contains three input fields: 'Method' (set to 'ANY'), 'Resource path' (set to '/my-api-function'), and 'Integration target' (set to 'my-api-function'). There are 'Add route' and 'Remove' buttons. At the bottom, there are 'Cancel', 'Review and create', 'Previous', and 'Next' buttons.

Step 1: Create an API
Step 2 - optional: **Configure routes**
Step 3 - optional: Define stages
Step 4: Review and create

Configure routes info

API Gateway uses routes to expose integrations to consumers of your API. Routes for HTTP APIs consist of two parts: an HTTP method and a resource path (e.g., GET /pets). You can define specific HTTP methods for your integration (GET, POST, PUT, PATCH, HEAD, OPTIONS, and DELETE) or use the ANY method to match all methods that you haven't defined on a given resource.

Method: ANY Resource path: /my-api-function Integration target: my-api-function

[Add route](#) [Remove](#)

[Cancel](#) [Review and create](#) [Previous](#) [Next](#)

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

5. Define the stages

The screenshot shows the AWS API Gateway console in the 'Create API' wizard. The left sidebar indicates the current step is 'Define stages' (Step 3 - optional). The main content area is titled 'Define stages - optional' and includes an 'info' icon. Below the title, a paragraph explains that stages are independently configurable environments that your API can be deployed to, and that you must deploy to a stage for API configuration changes to take effect. The form contains a 'Stage name' input field (set to '\$default') and an 'Auto-deploy' toggle switch (turned on). There are 'Add stage' and 'Remove' buttons. At the bottom, there are 'Cancel', 'Previous', and 'Next' buttons.

Step 1: Create an API
Step 2 - optional: Configure routes
Step 3 - optional: **Define stages**
Step 4: Review and create

Define stages - optional info

Stages are independently configurable environments that your API can be deployed to. You must deploy to a stage for API configuration changes to take effect, unless that stage is configured to autodeploy. By default, all HTTP APIs created through the console have a default stage named \$default. All changes that you make to your API are autodeployed to that stage. You can add stages that represent environments such as development or production.

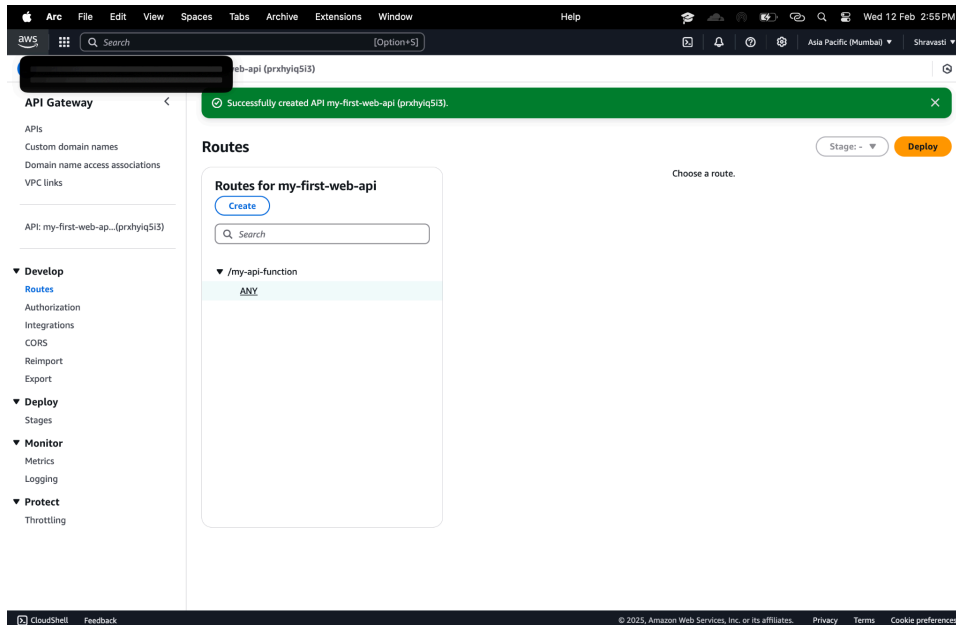
Stage name: \$default Auto-deploy: ☒

[Add stage](#) [Remove](#)

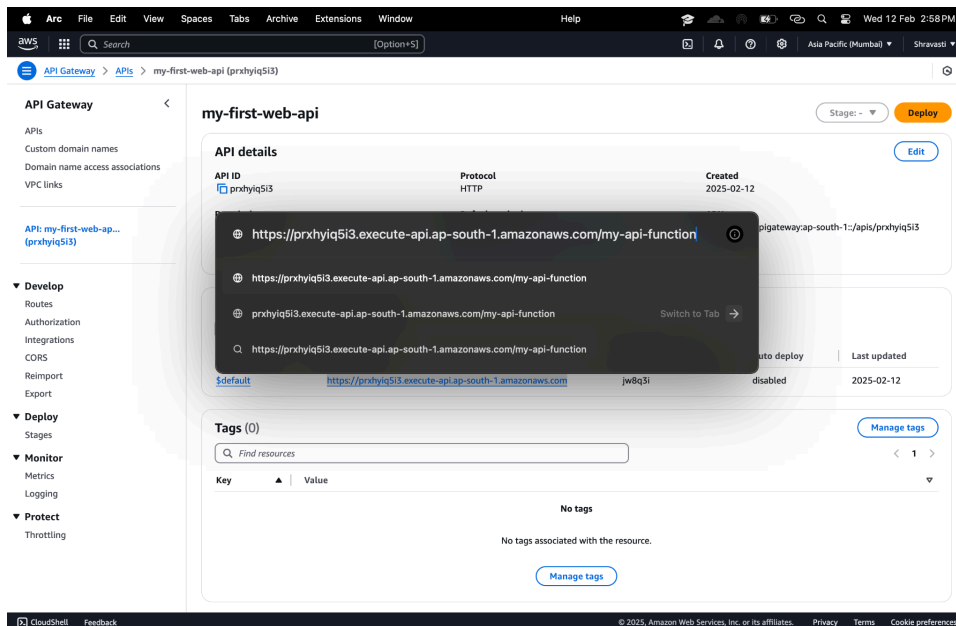
[Cancel](#) [Previous](#) [Next](#)

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

6. API will be created successfully



7. Copy the API URL from the left corner and paste it into the search tab along with your lambda function's name



8. Message is displayed

