



Serverless Registration Portal

03.19.2024

Rtt-41-2023

AWS re/start Cloud Practitioner Course

CLF-C02 Exam

Per Scholas

November 2023 - March 2024

Instructors: Sajini George & Jose Santos

Overview

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper.

Goals

1. Lorem ipsum dolor sit amet, consectetuer adipiscing elit
2. Sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Specifications

Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius.

Lorem Ipsum

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan.

Steps

Console

Part 1: S3 Static Hosting

1. Go to your console
2. Go to S3
3. Create s3 bucket
4. Choose default settings

- 
5. For 'bucket name' choose anything. E.g. bucket name: "my s3 bucket"
 6. unblock public access, to make it public
 7. click 'i acknowledge'
 8. create bucket
 9. click on the newly created bucket
 10. for objects tab, click 'upload' file "home.html" (this is home.html script) (Visual Code Studio to view this code easier)

```
1. 1. <!DOCTYPE html>
2. 2. <html lang="en">
3. 3. <head>
4. 4.   <meta charset="UTF-8">
5. 5.   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6. 6.   <title>A Helping Hand: Community Volunteers for Senior Support</title>
7. 7. <style>
8. 8.   body {
9. 9.     font-family: Arial, sans-serif;
10. 10.    margin: 0;
11. 11.    padding: 0;
12. 12.    background-color: #f4f4f4;
13. 13. }
14. 14. header {
15. 15.     background-color: #002657;
16. 16.     color: #fff;
17. 17.     padding: 20px;
18. 18.     text-align: center;
19. 19. }
20. 20. h1 {
21. 21.     margin-top: 0;
22. 22. }
23. 23. .container {
24. 24.     max-width: 800px;
25. 25.     margin: 20px auto;
26. 26.     padding: 20px;
27. 27.     background-color: #fff;
28. 28.     border-radius: 5px;
29. 29.     box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
30. 30. }
```

```
31.     p {
32.         line-height: 1.6;
33.     }
34.
35.     #welcomeHeading {
36.         color: #01377D; /* Change 'blue' to any color you desire */
37.     }
38.
39.     #welcomeHeading1 {
40.         color:white; /* Change 'blue' to any color you desire */
41.     }
42.     nav ul {
43.         list-style-type: none;
44.         margin: 0;
45.         padding: 0;
46.     }
47.     nav ul li {
48.         display: inline;
49.         margin-right: 20px;
50.     }
51.     nav ul li a {
52.         color: #fff;
53.         text-decoration: none;
54.     }
55. </style>
56. </head>
57. <body>
58.   <header>
59.     <h1 id="welcomeHeading1">A Helping Hand: Community Volunteers for
  Senior Support</h1>
60.
61.   </header>
62.   <div class="container">
63.     <h2 id="welcomeHeading">Welcome to our Community!</h2>
64.     <p>We provide assistance and support to seniors in our community
  through dedicated volunteer efforts. Our volunteers offer various
```

```
services such as companionship, assistance with daily tasks,
transportation, and more.</p>
65.
66.      
67.
68.      <h3 id="welcomeHeading">How to Get Involved</h3>
69.
70.      <p>If you're interested in becoming a volunteer or if you're a
senior in need of assistance, please contact us using the information
below:</p>
71.      <p><a href="Profile.html">Register by Submitting the Form</a></p>
72.
73.      <p>OR</p>
74.
75.      <p>Contact Email: volunteer@helpinghand.org</p>
76.      <p>Phone: (555) 123-4567</p>
77.      <p>Address: 123 Main Street, City, State, ZIP</p>
78.      </div>
79. </body>
80. </html>
```

11. Add this image:



12. click on add files.

13. add those files

14. go down and the bottom corner will show 'upload' button

15. Click it

16. after uploading, click 'close'

17. on the bucket 'properties' tab, click on that tab

18. scroll down

The screenshot shows the AWS S3 console with the URL s3.console.aws.amazon.com/s3/buckets/s3finalproject?region=eu-north-1&bucketType=general&tab=properties. The 'Static website hosting' section is expanded, and the 'Edit' button next to it is being clicked.

1. for 'static website hosting', click 'edit' and enable it
- 2.

19. For 'Index Document' write "Home.html" [MAKE SURE IT IS WRITTEN WITH CAPITAL "H" !!!]

The screenshot shows the AWS S3 console interface for editing a website endpoint. At the top, the URL is `s3.console.aws.amazon.com/s3/bucket/s3finalproject/property/website/edit?region=eu-north-1&bucketType=general`. The 'Services' tab is selected. In the main area, under 'Hosting type', 'Host a static website' is selected. A note indicates that customers must make content publicly readable. Below this, the 'Index document' field contains 'index.html'. The 'Error document - optional' field contains 'error.html'. A section for 'Redirection rules - optional' is present but empty. The bottom navigation bar includes CloudShell, Feedback, and various icons for other services like Lambda, CloudWatch, and CloudFront.

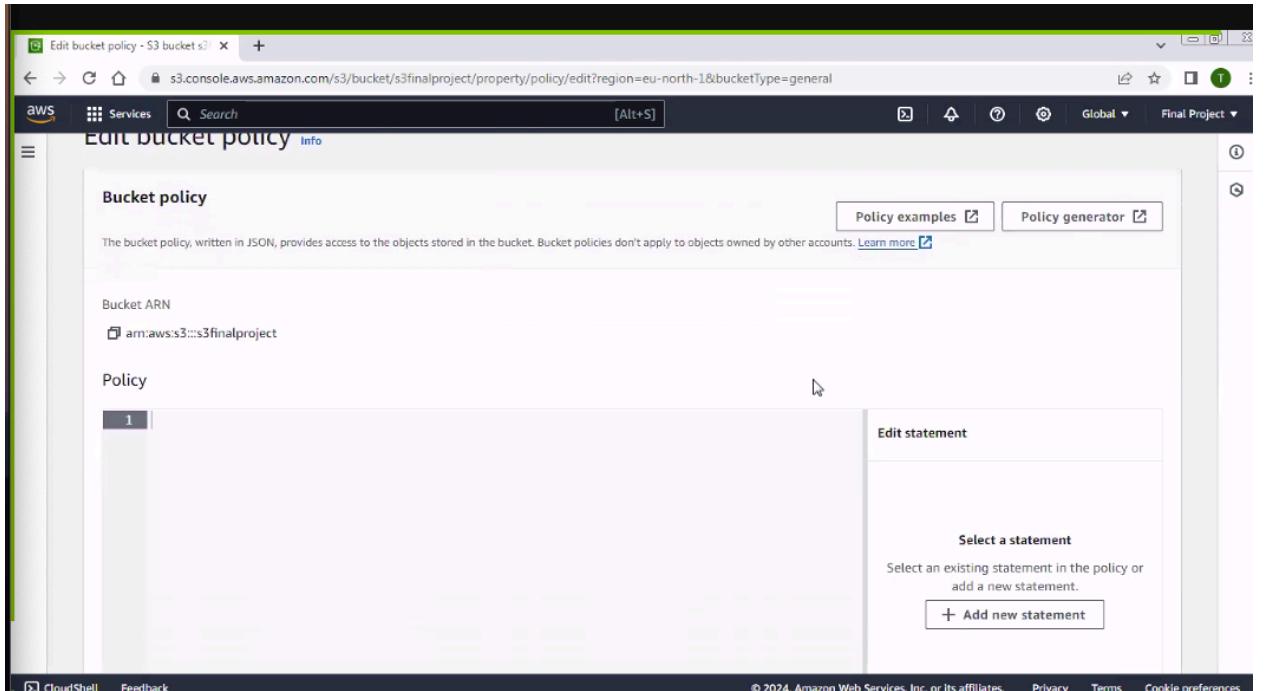
20. scroll down 'save changes'

21. go 'permissions' tab

The screenshot shows the AWS S3 console interface for the 's3finalproject' bucket. The browser address bar indicates the URL is s3.console.aws.amazon.com/s3/buckets/s3finalproject?region=eu-north-1&bucketType=general&tab=properties. The top navigation bar includes the AWS logo, Services dropdown, a search bar with placeholder 'Search [Alt+S]', and a close button. A green success message box says 'Successfully edited static website hosting.' Below the header, the breadcrumb navigation shows 'Amazon S3 > Buckets > s3finalproject'. The main content area displays the bucket details for 's3finalproject'. The 'Permissions' tab is currently selected, indicated by a blue underline and a cursor icon. Other tabs include 'Objects', 'Properties', 'Metrics', 'Management', and 'Access Points'. The 'Bucket overview' section shows the AWS Region as 'Europe (Stockholm) eu-north-1', the Amazon Resource Name (ARN) as 'arn:aws:s3:::s3finalproject', and the Creation date as 'March 19, 2024'. The 'Bucket Versioning' section contains a brief description of versioning and a link to 'Learn more'. The bottom of the page has a footer with links to 'AWS Documentation', 'AWS Support', and 'AWS Community'.

The screenshot shows the AWS S3 console interface for the bucket 's3finalproject'. The top navigation bar includes tabs for 'Objects', 'Properties', 'Permissions' (which is selected and highlighted in blue), 'Metrics', 'Management', and 'Access Points'. Below this, the 'Permissions overview' section indicates that objects can be public. The 'Block public access (bucket settings)' section is expanded, showing that 'Block all public access' is currently off. A link to 'Individual Block Public Access settings for this bucket' is provided. At the bottom of the page, there is a toolbar with icons for CloudShell, Feedback, and various AWS services like Lambda, CloudWatch, and S3.

22. go down click 'edit' on bucket policy (Take note of 'Bucket ARN')



23. paste this script on the policy (NAMEOFTHEBUCKET should be your bucket name) Note:
above 'policy' you will see "bucket ARN" copy and paste that into the 'resource' section below

```

1. {
2.   "Version": "2012-10-17",
3.   "Statement": [
4.     {
5.       "Effect": "Allow",
6.       "Principal": "*",
7.       "Action": "s3:GetObject",
8.       "Resource": "arn:aws:s3:::NAMEOFTHEBUCKET/*"
9.     }
10.   ]
11. }
```

24. then save changes

25. go to properties tab

26. Go down

s3finalproject - S3 bucket | S3 | +

s3.console.aws.amazon.com/s3/buckets/s3finalproject?region=eu-north-1&bucketType=general&tab=properties

aws Services Search [Alt+S]

Enabled

Requester pays

When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. [Learn more](#)

Requester pays

Disabled

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

Enabled

Hosting type

Bucket hosting

Bucket website endpoint

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)

<http://s3finalproject.s3-website.eu-north-1.amazonaws.com>

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 10:53 AM 3/19/2024

27. Copy that URL below 'bucket website endpoint'. Note: the bucket name will vary, it depends on what you named your bucket.

28. The link should take you to this:

A Helping Hand: Community Volunteers for Senior Support

Welcome to our Community!

We provide assistance and support to seniors in our community through dedicated volunteer efforts. Our volunteers offer various services such as companionship, assistance with daily tasks, transportation, and more.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 10:54 AM 3/19/2024

29. Finished Static Hosting part!

Part 2: SNS notifications

1. in search bar type "SNS" in the top of console
2. type in 'create topic' : "s3-notifications" and create it
3. The Name in our example is " s3-notifications", but you can name it whatever you want. Just make sure you remember

The screenshot shows the 'Create topic' page in the Amazon SNS console. At the top, a banner informs users that SNS now supports in-place message archiving and replay for FIFO topics, with a 'Learn more' link. Below the banner, the navigation path is 'Amazon SNS > Topics > Create topic'. The main section is titled 'Create topic' and contains a 'Details' tab. Under the 'Type' section, there are two options: 'FIFO (first-in, first-out)' and 'Standard'. The 'Standard' option is selected, indicated by a blue circle and a checked radio button. The 'Standard' section lists the following features:

- Best-effort message ordering
- At-least once message delivery
- Highest throughput in publishes/second
- Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Below the type selection, the 'Name' field is populated with 's3-notifications'. A note below the name field states: 'Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (_.)'. The 'Display name - optional' field is present but empty, with a note: 'To use this topic with SMS subscriptions, enter a display name. Only the first 10 characters are displayed in an SMS message.'

4. Click on Access Policy

The screenshot shows the AWS Management Console interface for creating a new SNS topic. The top navigation bar includes links for 'Create topic', 'Topics', and 'Simple Notification'. The URL in the address bar is 'eu-north-1.console.aws.amazon.com/sns/v3/home?region=eu-north-1#/create-topic'. The main content area is titled 'Create topic' and contains several optional configuration sections:

- Encryption - optional**: Describes in-transit encryption by default.
- Access policy - optional**: Defines who can access the topic. By default, only the topic owner can publish or subscribe.
- Data protection policy - optional**: Describes monitoring and prevention of sensitive data exchange.
- Delivery policy (HTTP/S) - optional**: Describes how Amazon SNS retries failed deliveries to HTTP/S endpoints.
- Delivery status logging - optional**: Configures message delivery status logging to CloudWatch Logs.

At the bottom of the page, there are links for 'Command Line Interface', 'Feedback', and 'AWS Support'. The footer includes copyright information: '© 2024, Amazon Web Services, Inc. or its affiliates.' and links for 'Privacy', 'Terms', and 'Conditions'.

5. put in this code in JSON preview

```

1. {
2.   "Version": "2012-10-17",
3.   "Id": "example-ID",
4.   "Statement": [
5.     {
6.       "Sid": "S3_notification",
7.       "Effect": "Allow",
8.       "Principal": {
9.         "Service": "s3.amazonaws.com"
10.      },
11.      "Action": "SNS:Publish",
12.      "Resource": "arn:aws:sns:us-east-1:339712730992:S3_notification",
13.      "Condition": {
14.        "ArnLike": {
15.          "aws:SourceArn": "arn:aws:s3:::registration-portal-bucket"
16.        }
17.      }
18.    }
19.  ]
20. }
21.

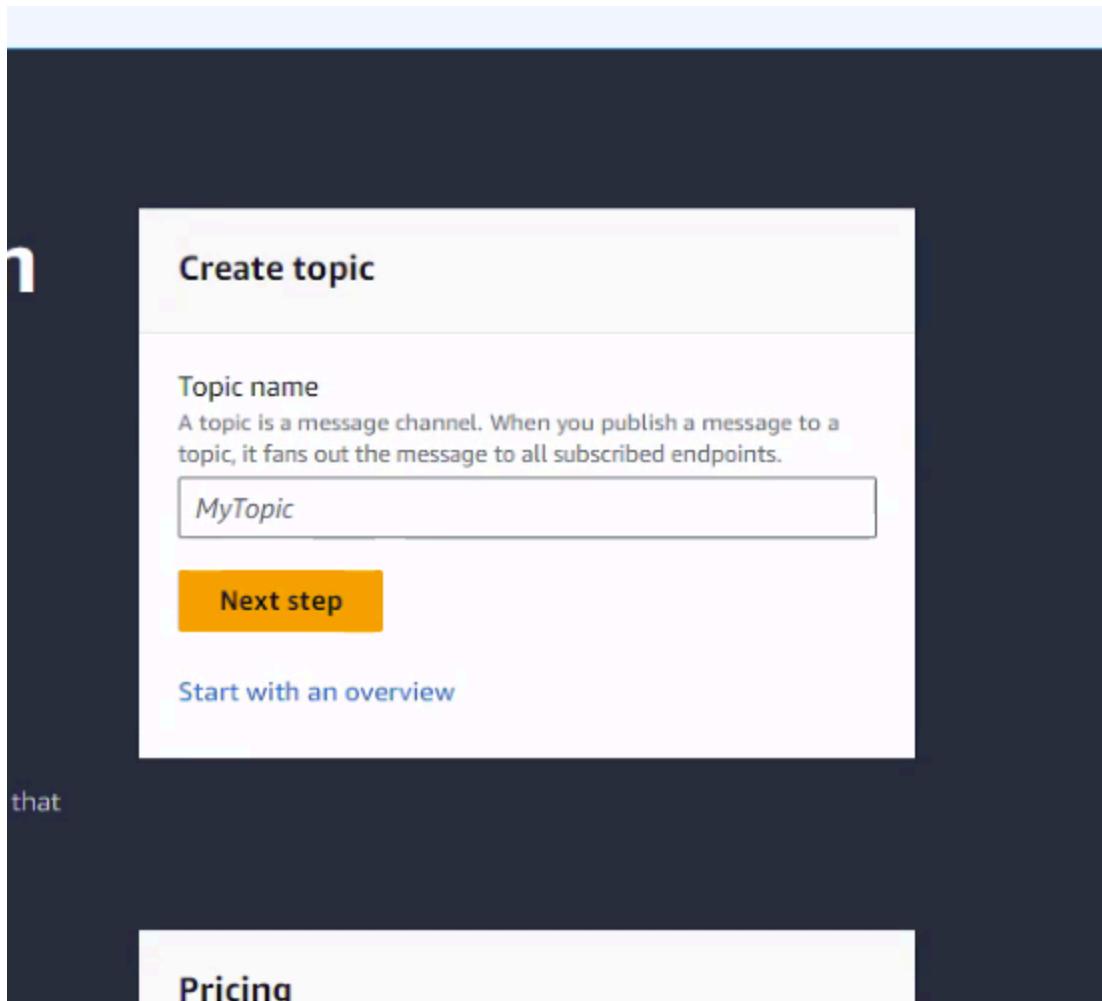
```

6. It should look like this:

The screenshot shows the AWS Management Console interface for configuring an access policy. The top navigation bar includes 'Services', a search bar, and a location bar set to 'Stockholm'. The main content area is titled 'Access policy - optional' with a note: 'This policy defines who can access your topic. By default, only the topic owner can publish or subscribe to the topic.' Below this, there are two options: 'Basic' (selected) and 'Advanced'. Under 'Basic', the 'Publishers' section is set to 'Only the topic owner'. The 'Subscribers' section is also set to 'Only the topic owner'. To the right, a 'JSON preview' window displays the generated JSON policy document:

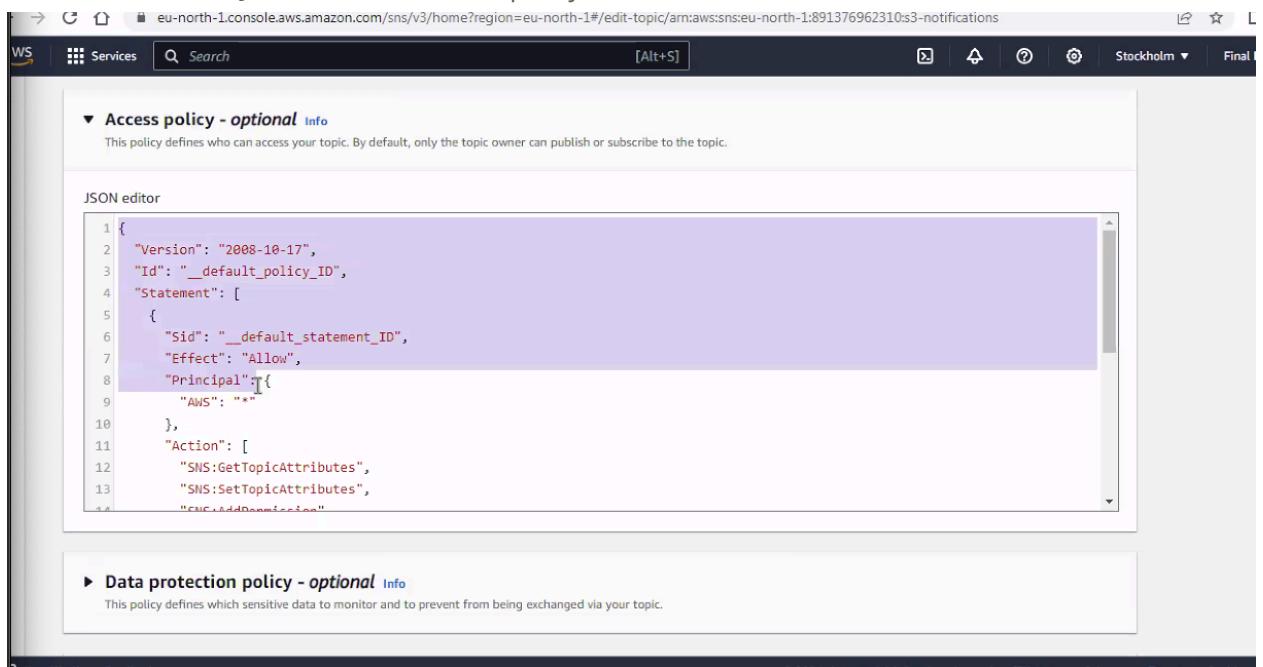
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "SNS:AddPermission",
                "SNS:Subscribe"
            ],
            "Resource": "arn:aws:sns:eu-north-1:891376962310:s3-notifications",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceOwner": "891376962310"
                }
            }
        }
    ]
}
```

7. click 'create topic':



8. then click on 'edit' on that topic

9. insert that into the JSON editor in access policy:



The screenshot shows the AWS SNS Access Policy JSON editor. The JSON code is as follows:

```

1 {
2   "Version": "2008-10-17",
3   "Id": "__default_policy_ID",
4   "Statement": [
5     {
6       "Sid": "__default_statement_ID",
7       "Effect": "Allow",
8       "Principal": "*",
9       "AWS": "*"
10    },
11    {
12      "Action": [
13        "SNS:GetTopicAttributes",
14        "SNS:SetTopicAttributes",
15        "SNS:AddPermission"
16      ]
17    }
18  ]
19 }

```

10. scroll up

11. make that change to the code. it needs to match exactly!

12. make sure Sid matches with the name of the Topic you created earlier:



The screenshot shows the AWS SNS Access Policy JSON editor with the following modified JSON code:

```

1 {
2   "Version": "2012-10-17",
3   "Id": "example-ID",
4   "Statement": [
5     {
6       "Sid": "s3-notifications",
7       "Effect": "Allow",
8       "Principal": {
9         "Service": "s3.amazonaws.com"
10      },
11      "Action": "SNS:Publish",
12      "Resource": "arn:aws:sns:us-east-1:339712730992:S3_notification",
13      "Condition": {
14        "ArnLike": "arn:aws:s3:::mybucket"
15      }
16    }
17  ]
18 }

```

13. go to SNS service.

14. click on Topic tab on the left hand screen

15. click on your topic (in our example 's3-notifications')

16. copy that ARN:

New Feature
Amazon SNS now supports in-place message archiving and replay for FIFO topics. [Learn more](#)

Amazon SNS > Topics > s3-notifications

s3-notifications

Details

Name	s3-notifications	Display name	s3-notifications
ARN	arn:aws:sns:eu-north-1:891376962310:s3-notifications	Topic owner	891376962310
Type	Standard		

Subscriptions Access policy Data protection policy Delivery policy (HTTP/S)

17.

18. then go back to that JSON editor and add that ARN to the 'resource' section of code:

```

1  {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Principal": {
7                 "Service": "s3.amazonaws.com"
8             },
9             "Action": "SNS:Publish",
10            "Resource": "arn:aws:sns:eu-north-1:891376962310:s3-notifications",
11            "Condition": {
12                "ArnLike": "arn:aws:s3:::mybucket/*"
13            }
14        }
15    ]
16}
  
```

Data protection policy - optional [Info](#)
This policy defines which sensitive data to monitor and to prevent from being exchanged via your topic.

19. go to s3

20. Click on your bucket:

Name	AWS Region	Access	Creation date
s3finalproject	Europe (Stockholm) eu-north-1	Public	March 19, 2024, 10:40:22 (UTC-04:00)

21. copy the name of that bucket exactly

22. then go back to SNS service

23. then in JSON editor, paste that bucket name in SourceARN:

```

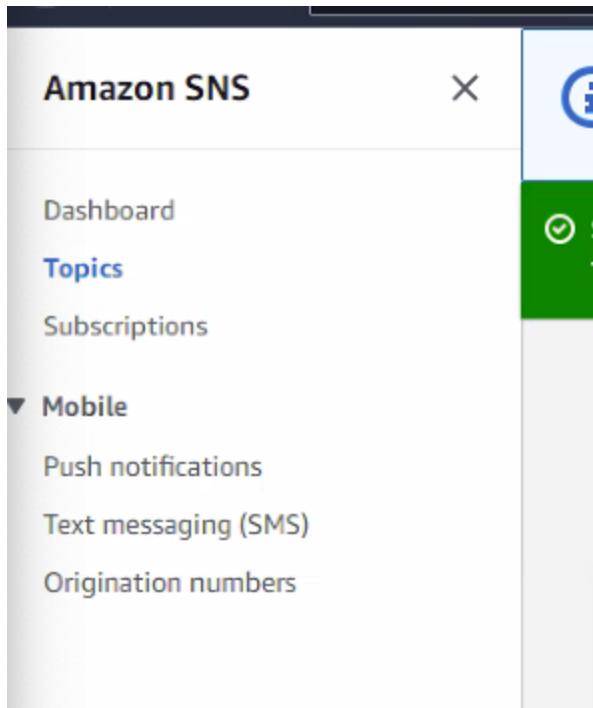
10 },
11 "Action": "SNS:Publish",
12 "Resource": "arn:aws:sns:eu-north-1:891376962310:s3-notifications",
13 "Condition": {
14     "ArnLike": {
15         "aws:SourceArn": "arn:aws:s3:::s3finalproject"
16     }
17 }

```

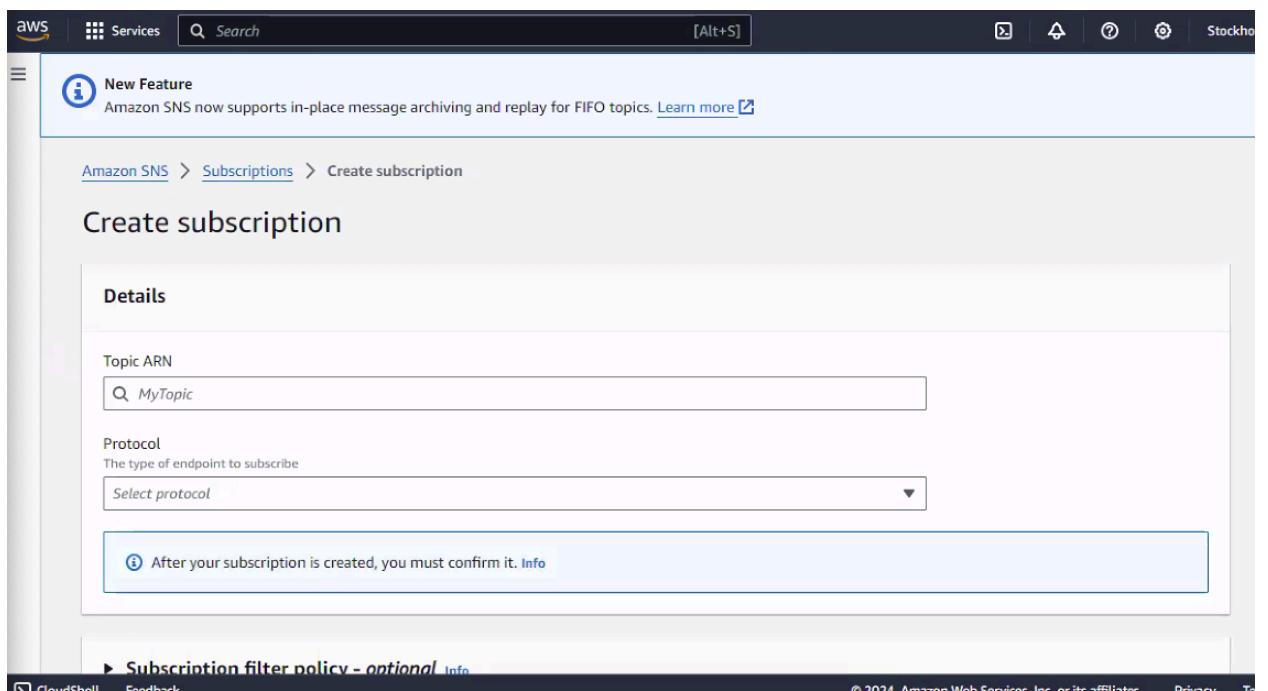
► Data protection policy - optional [Info](#)

24. then go down to the page and save changes

25. then click on subscriptions on the left hand side (below Topics):



26. Create one:



27. 'Topic ARN' should be there in the drop-down menu

28. 'Protocol' should be 'email' in the selection drop-down

29. For 'Endpoint', let it be your own personal email to receive notification:

Topic ARN
arn:aws:sns:eu-north-1:891376962310:s3-notifications

Protocol
The type of endpoint to subscribe
Email

Endpoint
An email address that can receive notifications from Amazon SNS.
test@example.com

Subscription filter policy - optional Info
This policy filters the messages that a subscriber receives.

Redrive policy (dead-letter queue) - optional Info
Send undeliverable messages to a dead-letter queue.

30. then create the subscription

31. Go to your email e.g. :

View notes as list

New

From: Amazon SNS <amazon-sns@amazonaws.com>

Subject: Amazon SNS has a new message for you

Date: Mon, 12 Dec 2022 10:30:00 +0000

Message ID: <AMAZON-SNS-20221212T103000Z-1891376962310>

Amazon SNS has a new message for you

Amazon SNS has a new message for you

Google Workspace

If you're having trouble loading, visit the [Gmail help center](#)

W X E Y Z C M A D ZM

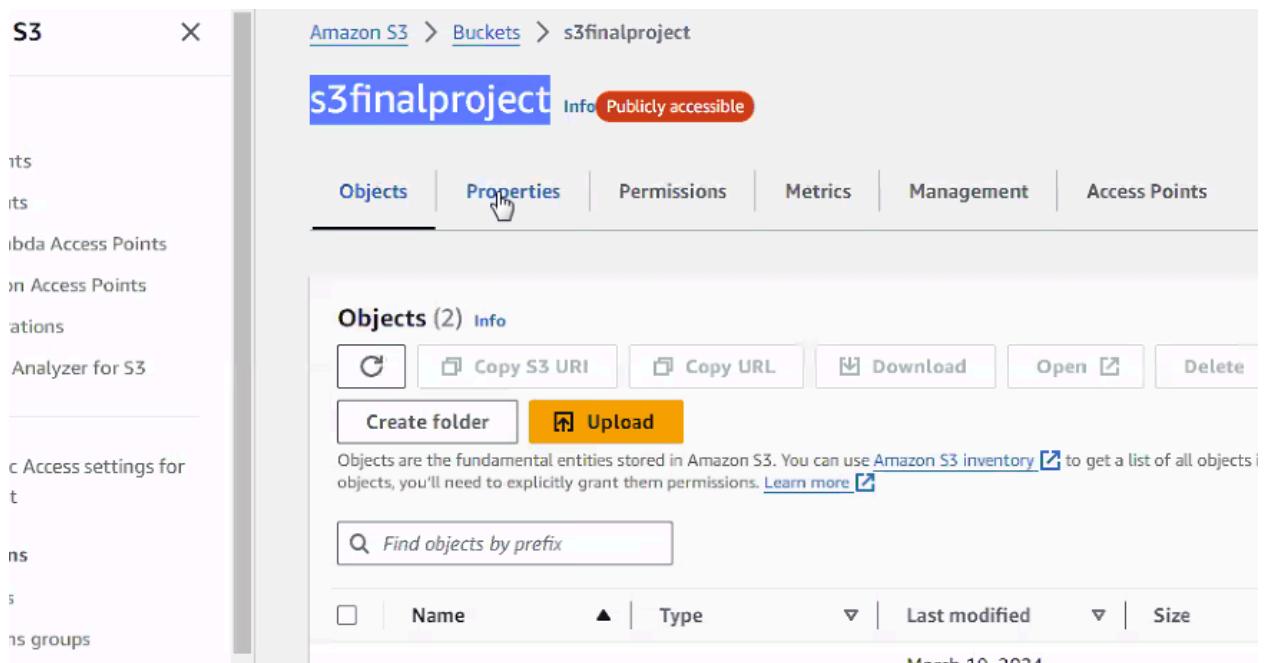
32. Go to your personal email and click on the email from s3 notifications:

The screenshot shows an email from 's3-notifications <no-reply@sns.amazonaws.com>' to the user. The subject is 'AWS Notification - Subscription Confirmation'. The email body contains the following text:
You have chosen to subscribe to the topic:
arn:aws:sns:eu-north-1:891376962310:s3-notifications
To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
[Confirm subscription](#)
Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests, please click [here](#).

33. Click on the link to 'confirm subscription'

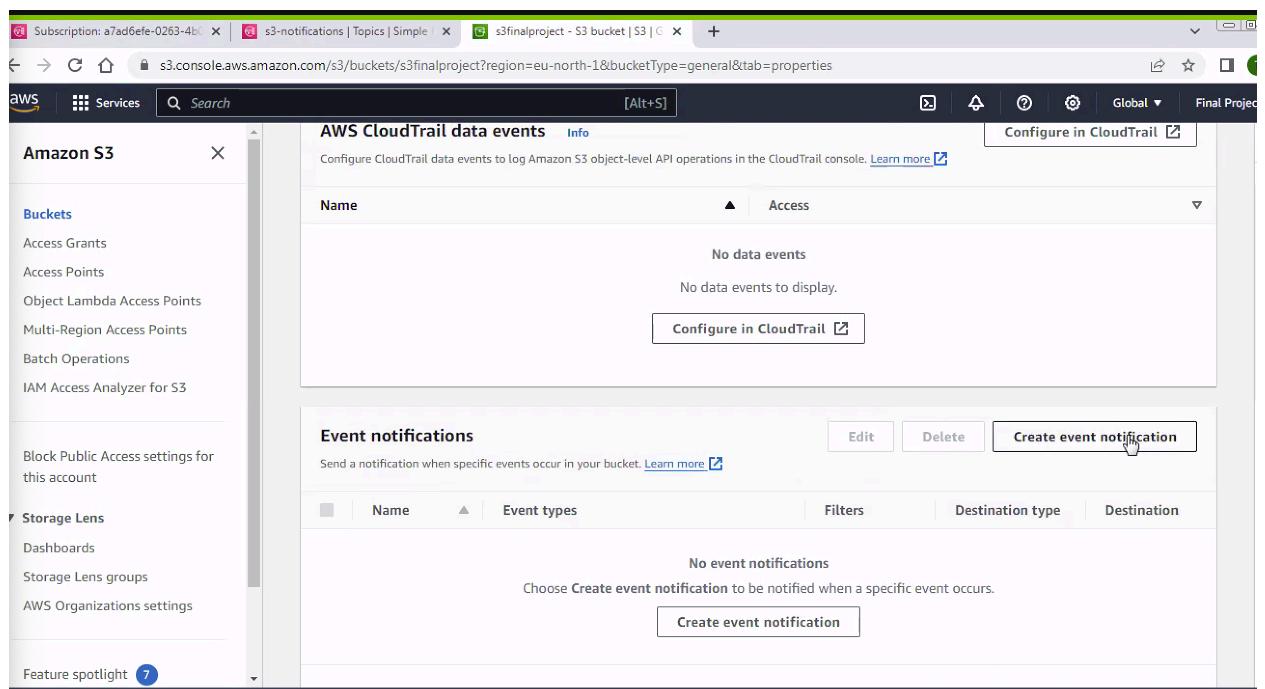
The screenshot shows a web browser window with the title 'AWS Notification - Subscription' and the tab 'Subscription confirm'. The URL in the address bar is 'sns.eu-north-1.amazonaws.com/confirmation.html?TopicArn=arn:aws:sns:eu-north-1:891376962310:s3-notifications'. The page content includes the AWS logo and the text 'Simple Notification Service'. A green box contains the following message:
Subscription confirmed!
You have successfully subscribed.
Your subscription's id is:
arn:aws:sns:eu-north-1:891376962310:s3-notifications:a7ad6efe-0263-4b03-9df9-a3996d5c46f9
If it was not your intention to subscribe, [click here to unsubscribe](#).

34. Go back to your console:



35. click on properties tab

36. Create event notifications button:



37. It should look like this:

The screenshot shows the 'Create event notification' page in the AWS Management Console. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, and a search bar. Below the navigation, the breadcrumb trail reads: 'Amazon S3 > Buckets > s3finalproject > Create event notification'. The main title 'Create event notification' is centered at the top of the page. A descriptive text below it says: 'To enable notifications, you must first add a notification configuration that identifies the publish and the destinations where you want Amazon S3 to send the notifications.' There are several tabs at the bottom of the page, though they are mostly cut off by the image's edge.

38. and name it something, in this case its "email notifications":

This screenshot shows the 'General configuration' section of the 'Create event notification' form. On the left, there's a sidebar with a vertical list of icons. The main area has a header 'General configuration'. Underneath, there's a 'Event name' field containing 'email-notifications', with a note below stating 'Event name can contain up to 255 characters.' Below that is a 'Prefix - optional' field containing 'images/' with a note 'Limit the notifications to objects with key starting with specified characters.' To the right of this field is a cursor icon. Further down is a 'Suffix - optional' field containing '.jpg' with a note 'Limit the notifications to objects with key ending with specified characters.' At the bottom of the configuration section is a heading 'Event types' with a note: 'Specify at least one event for which you want to receive notifications. For each group, you can choose an event type for all events, or you can choose one or more individual events.'

39. scroll down to "event types":

The screenshot shows the AWS Lambda console interface. At the top, there are tabs for 'Services' and 'Search'. Below the tabs, a message says 'Event name can contain up to 255 characters.' The main area is titled 'Event types' with the sub-section 'Object creation'. Under 'Object creation', there is a checkbox labeled 'All object create events' which is checked, and a sub-item 's3:ObjectCreated:*'. To the right of this, there are three other checkbox options: 'Put' (s3:PutObject), 'Post' (s3:PutObjectAcl), and 'Copy' (s3:CopyObject).

40. on object creation part

41. click on 'all object create events'

42. click on that object removal:

Event types
Specify at least one event for which you want to receive notifications. For each group, can choose one or more individual events.

Object creation

All object create events
s3:ObjectCreated:*

Put
s3:Ob

Post
s3:Ob

Copy
s3:Ob

Multi
s3:Ob

Object removal

All object removal events
s3:ObjectRemoved:*

Perm
s3:Ob

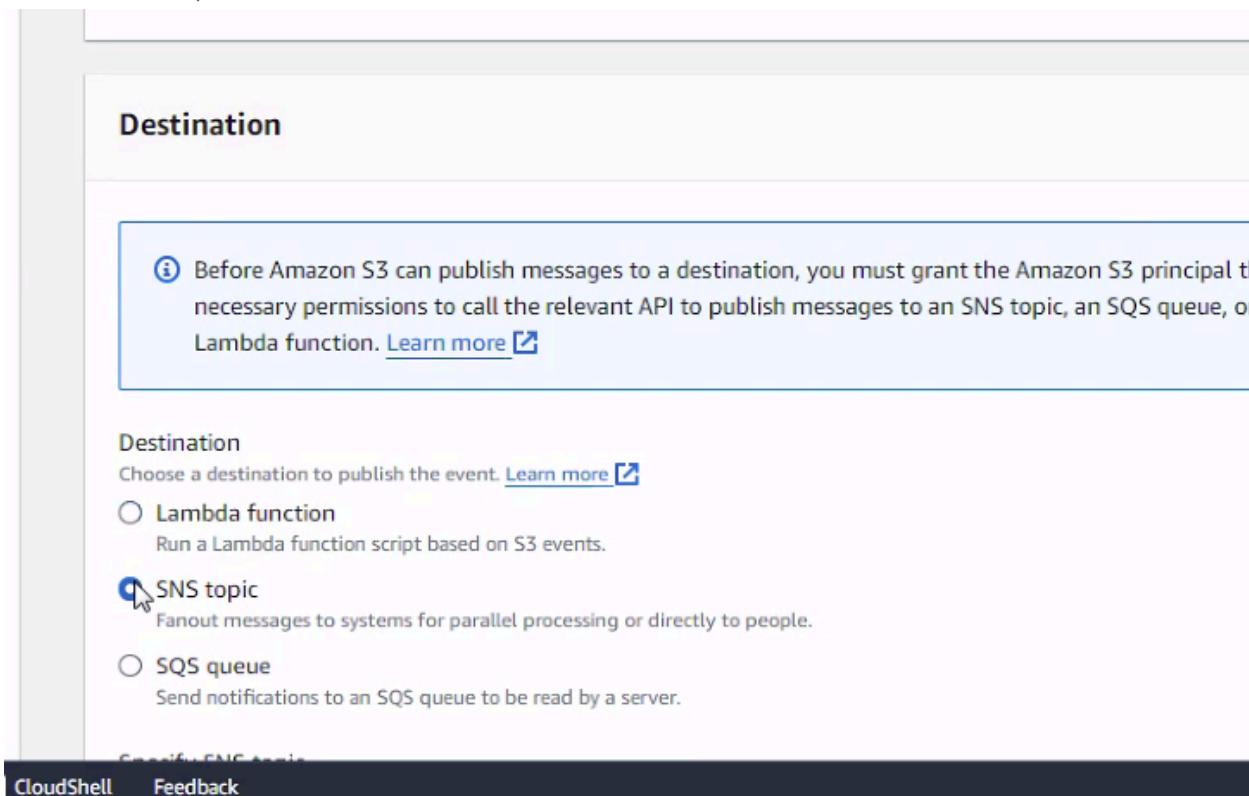
Delete
s3:Ob

CloudShell Feedback

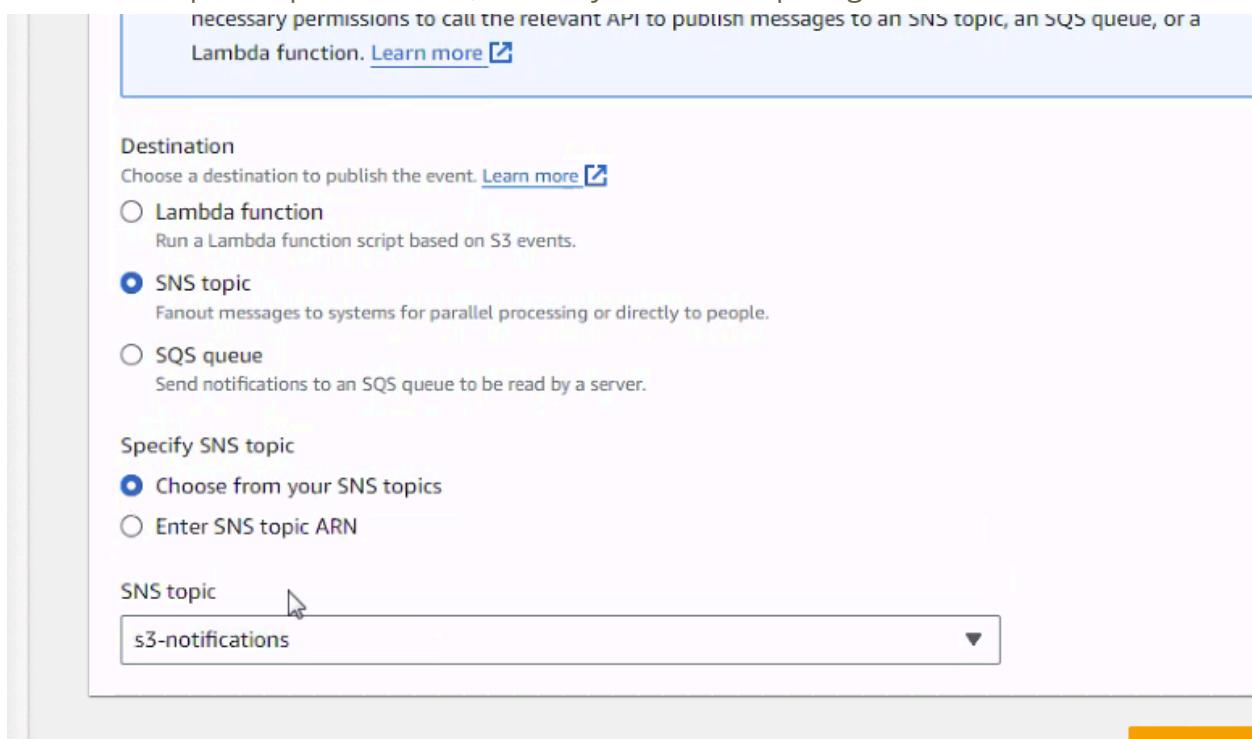
43. Click on 'object restore' checkbox too :



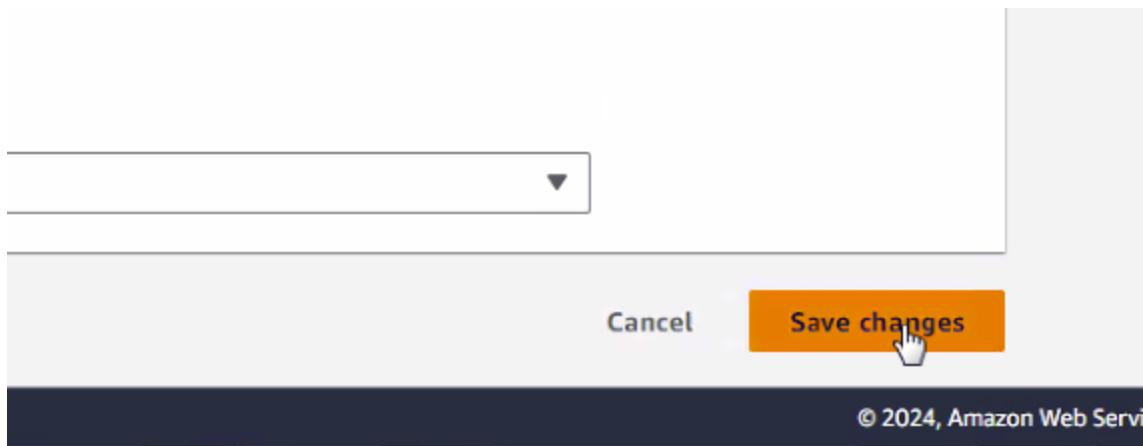
44. Click on 'SNS Topic'



45. In the 'SNS Topic' dropdown menu, choose your saved topic e.g. "s3-notifications":



46. Save changes



47. Go to your S3 page

48. Go to your S3 bucket (e.g. s3finalproject)

49. Click on 'Objects' tab

50. Click on 'Upload' button on the far right:

The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and various global settings. Below the navigation bar, the path 'Amazon S3 > Buckets > s3finalproject' is displayed. The main area is titled 's3finalproject' and shows it is 'Publicly accessible'. There are tabs for 'Objects', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points', with 'Objects' currently selected. A sub-header 'Objects (2) Info' is shown. Below this are several action buttons: 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', 'Actions', 'Create folder', and a large orange 'Upload' button which has a cursor icon pointing to it. A note below the buttons says 'Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions.' A search bar labeled 'Find objects by prefix' is also present. The main list displays two objects: 'Home.html' (html type, 2.5 KB, Standard storage class) and 'senior_support_image1.jpg' (jpg type, 73.0 KB, Standard storage class).

51. put these 3 files in the s3 bucket (use Visual Code Studio to view their code easier)

The screenshot shows a Microsoft Teams message from a user named 'Archana Allishe' at 2:47 PM. The message contains three attachments, each represented by a red rounded square icon with white code symbols (either '</>' or '<>') and a file name: 'InformationSaved.html' (HTML), 'Profile.html' (HTML), and 'registration.js' (JavaScript). The background of the message area is dark.

InformationSaved.html: (be sure you copy the whole code)

```

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4.   <meta charset="UTF-8">
5.   <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```

6.    <title>Registration Submitted</title>
7.    <style>
8.        body {
9.            font-family: Arial, sans-serif;
10.           margin: 0;
11.           padding: 20px;
12.       }
13.       .message-container {
14.           border: 1px solid #ccc;
15.           padding: 20px;
16.           max-width: 600px;
17.           margin: 0 auto;
18.           text-align: center;
19.       }
20.       h1 {
21.           margin-bottom: 20px;
22.       }
23.   </style>
24. </head>
25. <body>
26.
27. <div class="message-container">
28.     <h1>Your Registration Form is Submitted</h1>
29.     <p>We will contact you soon!</p>
30.     <a href="Home.html">Go back to Home</a>
31. </div>
32.
33. </body>
34. </html>
35.

```

52. Profile.html:

```

1.  <!DOCTYPE html>
2.  <html lang="en">
3.  <head>
4.      <meta charset="UTF-8">
5.      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6.
7.      <style>
8.          body {

```

```
9.         font-family: Arial, sans-serif;
10.        background-color: #f2f2f2;
11.        margin: 0;
12.        padding: 0;
13.    }
14.    .container {
15.        max-width: 600px;
16.        margin: 0 auto;
17.        padding: 20px;
18.        background-color: #fff;
19.        border-radius: 8px;
20.        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
21.    }
22.    h1 {
23.        text-align: center;
24.        color:#01377D;
25.    }
26.    label {
27.        display: block;
28.        margin-bottom: 5px;
29.        color:#01377D;
30.    }
31.    input[type="text"] {
32.        width: 100%;
33.        padding: 8px;
34.        margin-bottom: 10px;
35.        border: 1px solid #ccc;
36.        border-radius: 5px;
37.        box-sizing: border-box;
38.    }
39.    input[type="submit"] {
40.        width: 100%;
41.        padding: 10px;
```

```
42.         background-color: #01377D;
43.         color: #fff;
44.         border: none;
45.         border-radius: 5px;
46.         cursor: pointer;
47.         font-size: 16px;
48.     }
49.     input[type="submit"]:hover {
50.         background-color:#01377D;
51.     }
52.     #DataSaved {
53.         text-align: center;
54.         margin-top: 20px;
55.         color: green;
56.         font-weight: bold;
57.     }
58.
59. </style>
60.</head>
61.<body>
62. <div class="container">
63.     <h1>Volunteer Registration Form</h1>
64.     <form id="registrationForm">
65.         <label for="useremail">User email:</label>
66.         <input type="text" name="useremail" id="useremail" required>
67.
68.         <label for="firstname">First name:</label>
69.         <input type="text" name="firstname" id="firstname" required>
70.
71.         <label for="lastname">Last name:</label>
72.         <input type="text" name="lastname" id="lastname" required>
73.
74.         <label for="usergender">Gender:</label>
```

```

75.      <input type="text" name="usergender" id="usergender" required>
76.
77.      <label for="userage">User Age:</label>
78.      <input type="text" name="userage" id="userage" required>
79.
80.      <input type="submit" id="Submit" value="Save Profile">
81.    </form>
82.    <p id="DataSaved"></p>
83.  </div>
84.  <script>
85.    src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></
86. script>
85.  <script src="registration.js"></script>
86.</body>
87.</html>

```

53. Registration.js:

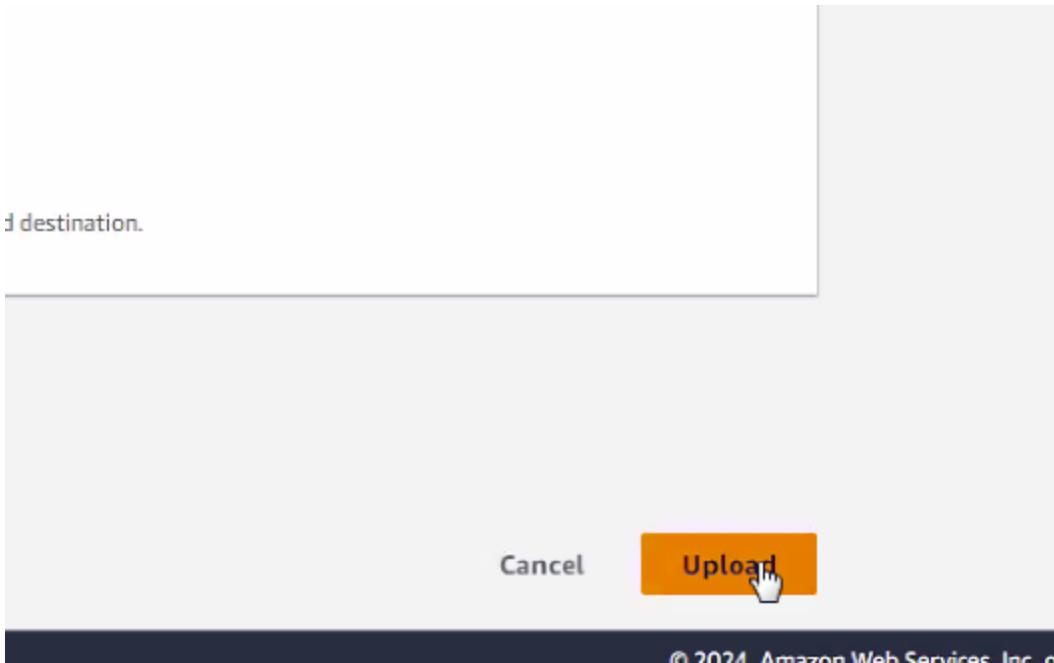
```

1.
2. var API_ENDPOINT =
3.   "https://7qmc7yvqc1.execute-api.us-east-1.amazonaws.com/prod";
4. $(document).ready(function() {
5.   $('#Submit').click(function(event) {
6.     event.preventDefault(); // Prevent default form submission
7.     behavior
8.     var inputData = {
9.       "userEmail": $('#useremail').val(),
10.      "userFirstName": $('#firstname').val(),
11.      "userLastName": $('#lastname').val(),
12.      "userGender": $('#usergender').val(),
13.      "userAge": $('#userage').val()
14.    };
15.
16.    $.ajax({
17.      url: API_ENDPOINT,
18.      type: 'POST',
19.      data: JSON.stringify(inputData),
20.      contentType: 'application/json; charset=utf-8',

```

```
21.         success: function(response) {
22.             $('#DataSaved').text('Data Submitted');
23.             window.location.href = "InformationSaved.html"; // Redirect to the next page
24.         },
25.         error: function() {
26.             alert("Error occurred while submitting data.");
27.         }
28.     });
29. });
30. });
31.
32.
```

54. Upload it :



55. Go check your email for the notification:

The image shows three screenshots of an email inbox from the Amazon S3 Notifications service. The first screenshot shows a single email from 's3-notifications' at 'no-reply@sns.amazonaws.com' with the subject 'Amazon S3 Notification'. The second and third screenshots show a follow-up message with the subject 'Amazon S3: ObjectCreated:Put'.

Email 1: Amazon S3 Notification

From: s3-notifications <no-reply@sns.amazonaws.com>
To: me <>
Subject: Amazon S3 Notification
Date: 2:45 PM (5 minutes ago)

Service: Amazon S3, Event: s3:TestEvent, Time: 2024-03-19T18:45:36.644Z, Bucket: s3finalproject, RequestId: PK1J5N48NGK9T66S, HostId: 3y+qy69ilk5cktqDxeVaNORBSrbo3E0YtQne/Yq0GWiscPLcwNmppDCtNupicj/o1Db4TJ0su9PQ=

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
<https://sns.eu-north-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:eu-north-1:891376962310:s3-notifications:a7ad6efe-0263-4b03-9df9-a3996d5c46f9&Endpoint=mharrati1967@gmail.com>

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at <https://aws.amazon.com/support>

Email 2: Amazon S3: ObjectCreated:Put

From: s3-notifications <no-reply@sns.amazonaws.com>
To: me <>
Subject: Amazon S3: ObjectCreated:Put
Date: 2:51PM (0 minutes ago)

Records: [{"eventVersion": "2.1", "eventSource": "aws:s3", "awsRegion": "eu-north-1", "eventTime": "2024-03-19T18:51:17.494Z", "eventName": "ObjectCreated:Put", "userIdentity": {"principalId": "A12CNQWLWKW3TU"}, "requestParameters": {"sourceIPAddress": "154.183.233.26"}, "responseElements": {"x-amz-request-id": "7PEF9RKQTWEJ50Y", "x-amz-id-2": "ml4AGCZlwO4uEWQEXXdz7GxbOtcVa9M/WOc7vAJS|LbwgkUeF3OB4KBkbBqZGxu0XUatRBRhIyb1+zXyqCQBybaNow7MuZ"}, "s3": {"s3SchemaVersion": "1.0", "configurationId": "email-notifications", "bucket": {"name": "s3finalproject"}, "ownerIdentity": {"principalId": "A12CNQWLWKW3TU"}, "arn": "arn:aws:s3:::s3finalproject"}, "object": {"key": "registration.js", "size": 1074, "eTag": "7d192230eaa5b00cf02d25e2cb73ea6", "sequencer": "0065F9DEA573568F8B"}}]]

Email 3: Amazon S3: ObjectCreated:Put

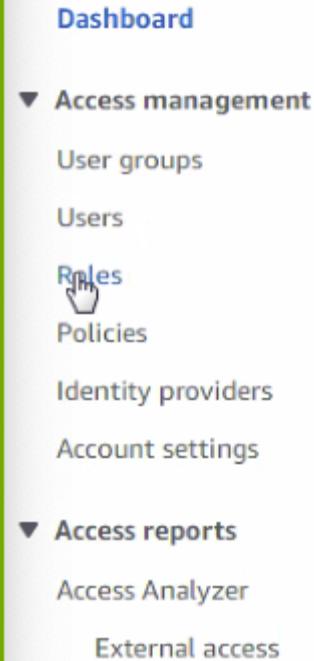
From: s3-notifications <no-reply@sns.amazonaws.com>
To: me <>
Subject: Amazon S3: ObjectCreated:Put
Date: 2:51PM (0 minutes ago)

9T18:51:17.494Z", "eventName": "ObjectCreated:Put", "userIdentity": {"principalId": "A12CNQWLWKW3TU", "arn": "arn:aws:s3:::s3finalproject"}, "requestParameters": {"sourceIPAddress": "154.183.233.26"}, "responseElements": {"x-amz-request-id": "7PEF9RKQTWEJ50Y", "x-amz-id-2": "ml4AGCZlwO4uEWQEXXdz7GxbOtcVa9M/WOc7vAJS|LbwgkUeF3OB4KBkbBqZGxu0XUatRBRhIyb1+zXyqCQBybaNow7MuZ"}, "s3": {"s3SchemaVersion": "1.0", "configurationId": "email-notifications", "bucket": {"name": "s3finalproject"}, "ownerIdentity": {"principalId": "A12CNQWLWKW3TU"}, "arn": "arn:aws:s3:::s3finalproject"}, "object": {"key": "registration.js", "size": 1074, "eTag": "7d192230eaa5b00cf02d25e2cb73ea6", "sequencer": "0065F9DEA573568F8B"}}]]

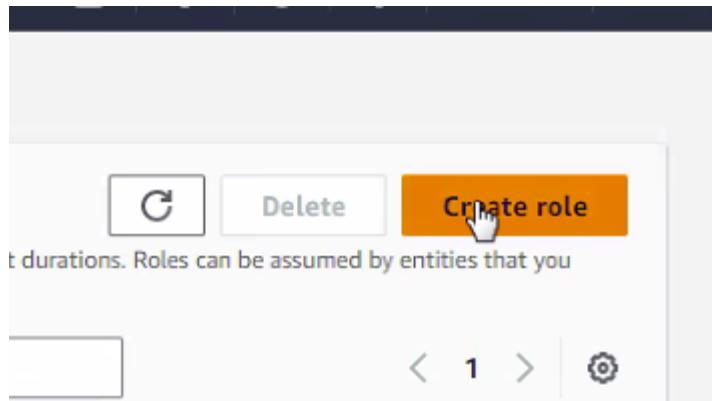
56. this shows that the files have been uploaded and you received an email notification "registration.js" is the name of the file that it uploaded to s3. and SNS notified us. it means it has the data and it is taking the front end info from the text boxes and connect it to the API

Part 3: IAM setup

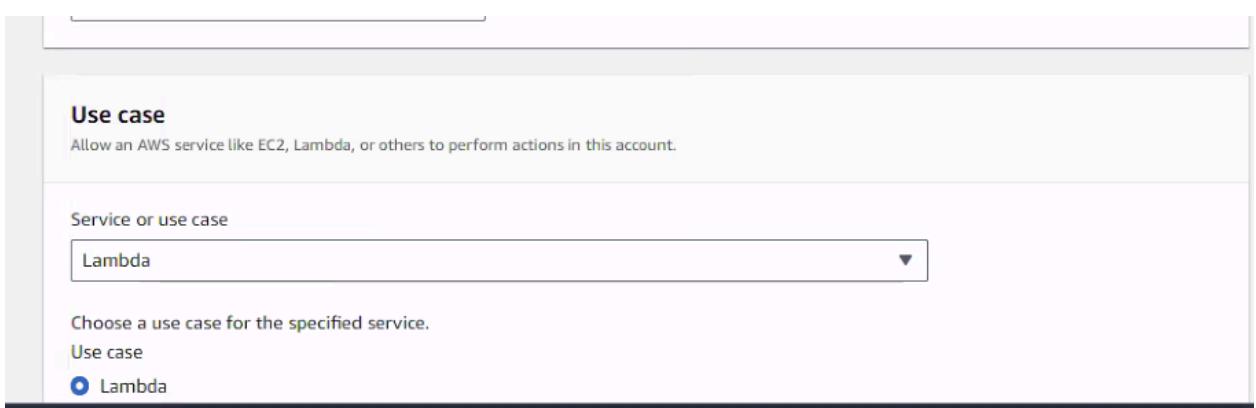
1. Go to IAM on your console:

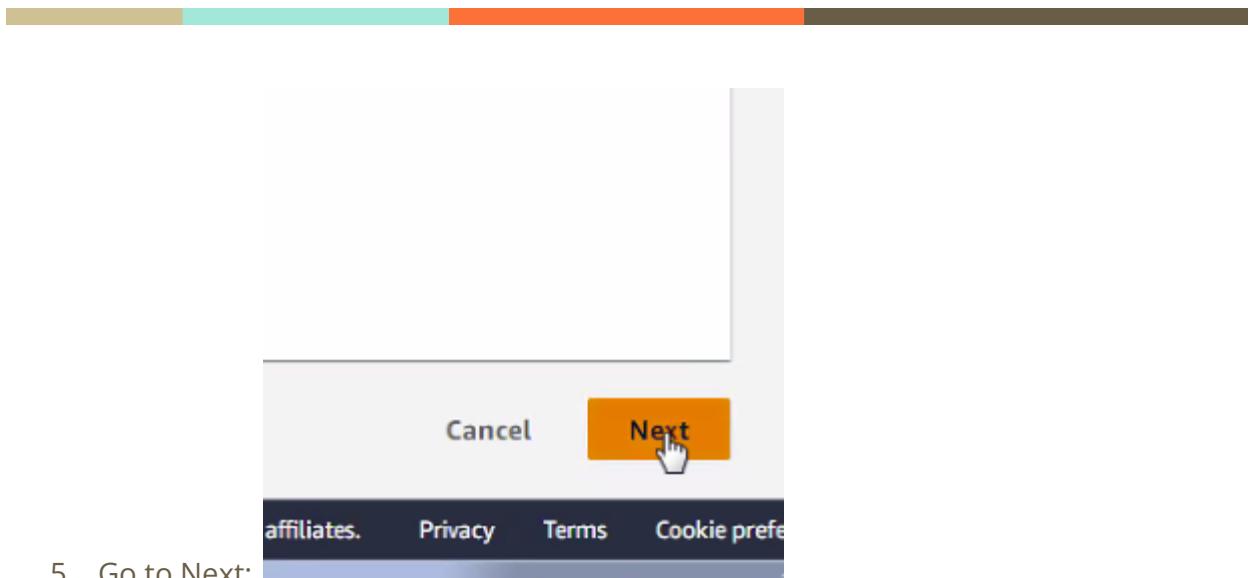


2. Go to 'Roles' on the left hand side of the console:



3. Create Role:
4. For 'Use Case' select Lambda:





5. Go to Next:
6. Add permissions
7. In the Search box click on it:

Step 1
[Select trusted entity](#)

Step 2
Add permissions

Step 3
Name, review, and create

Add permissions [Info](#)

Permissions policies (912) [Info](#)

Choose one or more policies to attach to your new role.

Filter by Type

Search All types ▾

<input type="checkbox"/>	Policy name	Type
<input type="checkbox"/>	AdministratorAccess	AWS managed - job function
<input type="checkbox"/>	AdministratorAccess-Amp...	AWS managed

8. Type in "AmazonDynamoDB":

The screenshot shows the 'Add permissions' interface in the AWS IAM console. A search bar at the top contains the text 'amazondynamo'. Below the search bar, a table lists two AWS managed policies:

Policy name	Type	Description
AmazonDynamoDBFullAccess	AWS managed	Provides
AmazonDynamoDBReadOnlyAccess	AWS managed	Provides

9. Then select the one that has "AmazonDynamoDBFullAccess":

The screenshot shows a simplified view of the policy selection interface. It displays a single row of data with a checked checkbox next to the policy name 'AmazonDynamoDBFullAccess'. The other columns show the policy type as 'AWS managed'.

Policy name	Type
AmazonDynamoDBFullAccess	AWS managed

10. Search for "CloudWatchLogsReadOnlyAccess":

The screenshot shows the 'Add permissions' interface in the AWS IAM console. A search bar at the top contains the text 'cloudwatchlogs'. Below the search bar, a table lists seven AWS managed policies. The columns are 'Policy name', 'Type', and 'Description'. The policies listed are:

Policy name	Type	Description
AmazonAPIGatewayPushToCloudW...	AWS managed	Allows
AmazonDMSCloudWatchLogsRole	AWS managed	Provides
AWSAppSyncPushToCloudWatchLogs	AWS managed	Allows
AWSOpsWorksCloudWatchLogs	AWS managed	Enables
CloudWatchLogsCrossAccountShari...	AWS managed	Provides
CloudWatchLogsFullAccess	AWS managed	Provides

11. Go next

12. Then in Role Name: "InsertDataRole" (you can use whatever name you desire, just keep that in mind when using this lab's code):

The screenshot shows the 'Role details' configuration page. The 'Role name' field is filled with 'InsertDataRole'. The 'Description' field contains the text 'Allows Lambda functions to call AWS services on your behalf.' Both fields have character limits and character count indicators.

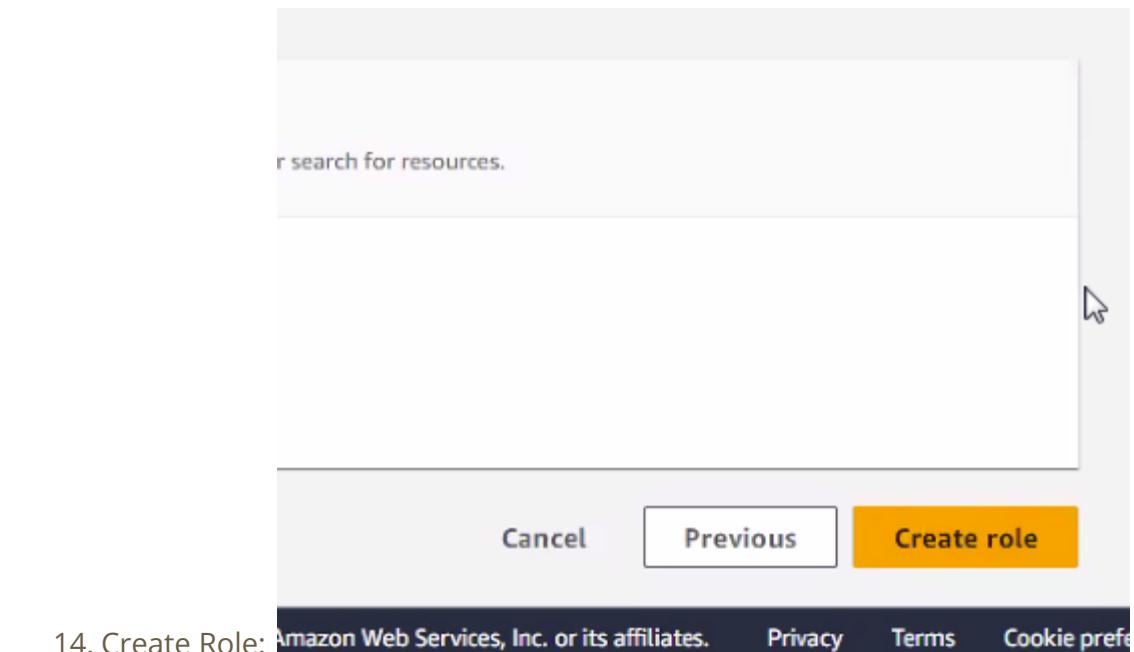
Role details

Role name
Enter a meaningful name to identify this role.
InsertDataRole
Maximum 64 characters. Use alphanumeric and '+,-,@-_ characters.

Description
Add a short explanation for this role.
Allows Lambda functions to call AWS services on your behalf.
Maximum 1000 characters. Use alphanumeric and '+,-,@-_ characters.

Step 1: Select trusted entities Edit

13. Scroll down



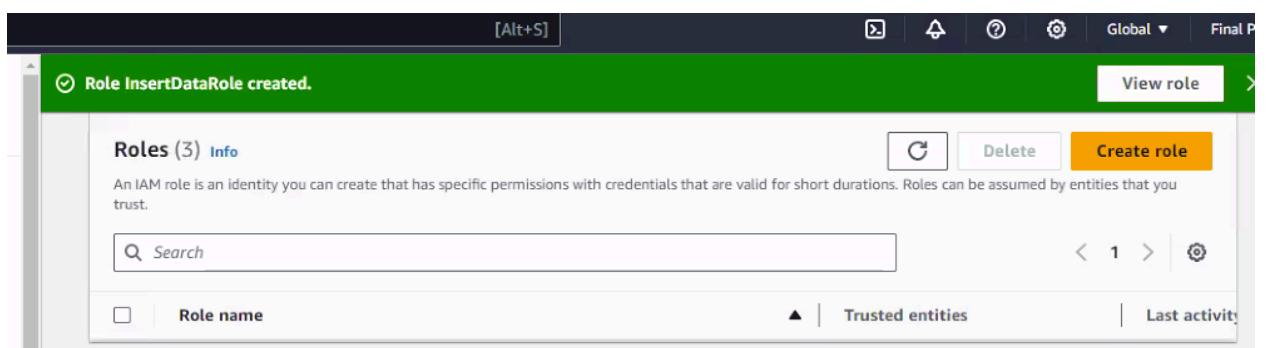
14. Create Role:

Privacy

Terms

Cookie preferences

15. Success:



16. Section complete!

Part 4: DynamoDB

1. Go to DynamoDB on your console:

The screenshot shows the AWS IAM console search results for 'dynamodb'. The search bar at the top contains the query 'dynamodb'. Below the search bar, there is a sidebar with links to 'Services (2)', 'Features (14)', 'Resources New', 'Documentation (6,813)', 'Knowledge Articles (49)', 'Marketplace (343)', 'Blogs (548)', 'Events (11)', and 'Tutorials (5)'. The main content area displays a search result for 'DynamoDB' under the 'Services' category. The result card for DynamoDB includes the icon, name, description ('Managed NoSQL Database'), and a 'Top features' section with links to 'Tables', 'Imports from S3', 'Explore Items', 'Clusters', and 'Reserved Capacity'. Below this, another service card for 'Athena' is shown. At the bottom right of the main content area, there is a link 'See all 14 results ▶'.

2. Create a Table:

The screenshot shows the Amazon DynamoDB Management Console home page. The URL in the browser is 'eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#service'. The left sidebar lists navigation options: Dashboard, Tables, Explore items, PartiQL editor, Backups, Exports to S3, Imports from S3, Reserved capacity, and Settings. The main content area features a large heading 'Amazon DynamoDB' with the subtext 'A fast and flexible NoSQL database service for any scale'. It includes a brief description of DynamoDB as a fully managed, key-value, and document database. A 'Get started' section with a 'Create table' button is visible, along with a 'Pricing' section. A feedback survey banner at the top right encourages users to share their feedback. The bottom of the screen shows a standard Windows taskbar with various icons.

3. Name Table name: 'RegistrationTable'

Create table

Table details Info

DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

Table name
This will be used to identify your table.
 Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.)

Partition key

4. Partition Key: 'EmailID' :

Partition key
The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.

1 to 255 characters and case sensitive.

Sort key - optional

5. Scroll down and create table:

© 2024, Amazon Web Service

6. Success:

The screenshot shows the AWS DynamoDB 'Tables' page. At the top, there is a blue header bar with a feedback survey message: 'Share your feedback on Amazon DynamoDB' and a 'Share feedback' button. Below the header, a green success message box displays: 'The RegistrationTable table was created successfully.' The main content area shows a table titled 'Tables (1) Info'. The table has columns: Name, Status, Partition key, Sort key, Indexes, Deletion protection, and Read capacity mode. One row is listed: 'RegistrationTable' (Status: Active, Partition key: EmailID (S), Sort key: -, Indexes: 0, Deletion protection: Off, Read capacity mode: Provisioned (5)). There are also search and filter options at the top of the table.

Part 5: Creating a Lambda

1. Go to Lambda on your console
2. Click on 'Create a function' on the right side of the screen (yellow-orange button):

The screenshot shows the AWS Lambda console. The top navigation bar includes the AWS logo, Services, a search bar, and account information for Stockholm. The main content area features a dark background with white text. It says 'Compute' and 'AWS Lambda' in large letters, followed by the tagline 'lets you run code without thinking about servers.' Below this, a paragraph explains Lambda's pay-as-you-go model and serverless nature. To the right, a 'Get started' box contains the text 'Author a Lambda function from scratch, or choose from one of many preconfigured examples.' with a prominent yellow 'Create a function' button. At the bottom, a 'How it works' section lists runtime options: .NET, Java, Node.js (which is selected), Python, Ruby, and Custom runtime. There are also 'Run' and 'Next: Lambda responds to events' buttons.

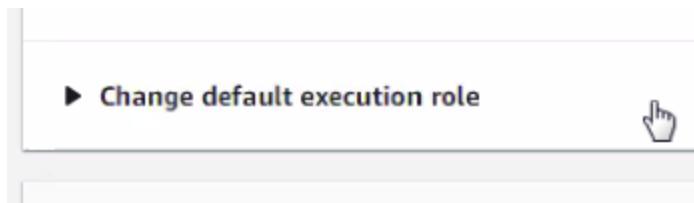
3. For 'Function name' : "InsertLambda"

The screenshot shows the AWS Lambda 'Author from scratch' wizard. At the top, there are four options: 'Author from scratch' (selected), 'Use a blueprint', 'Container image', and 'Browse serverless app repository'. Below this, the 'Basic information' section is displayed. It includes fields for 'Function name' (set to 'InsertLambda') and 'Runtime' (set to 'Node.js 20.x'). There are also sections for 'Architecture' and 'Architecture Info'.

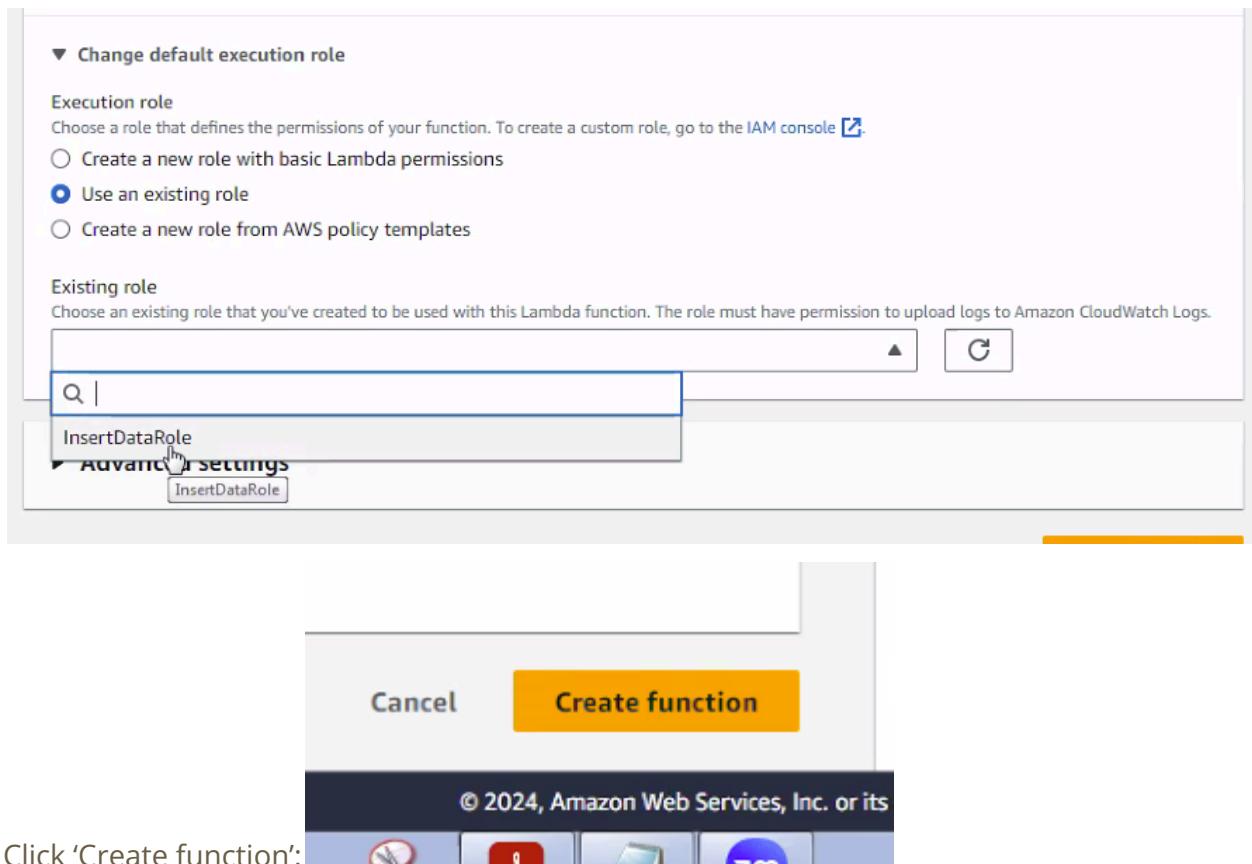
4. Runtime: 'Python 3.12'

The screenshot shows the 'Runtime' configuration step. It includes a dropdown menu set to 'Python 3.12' and a 'Change' button. Below the dropdown is an 'Architecture' section.

5. Click on 'Change default execution role' to expand it:

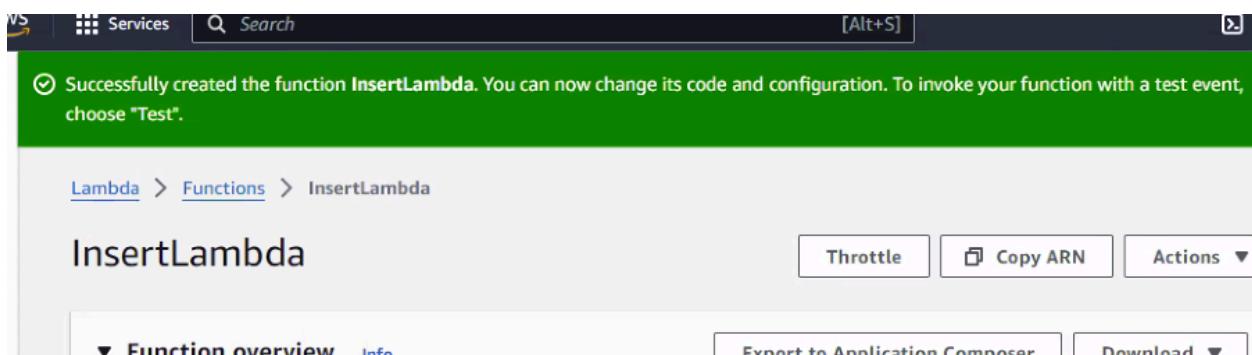


6. For 'Existing role' select the Role that you created previously (e.g. InsertDataRole)):

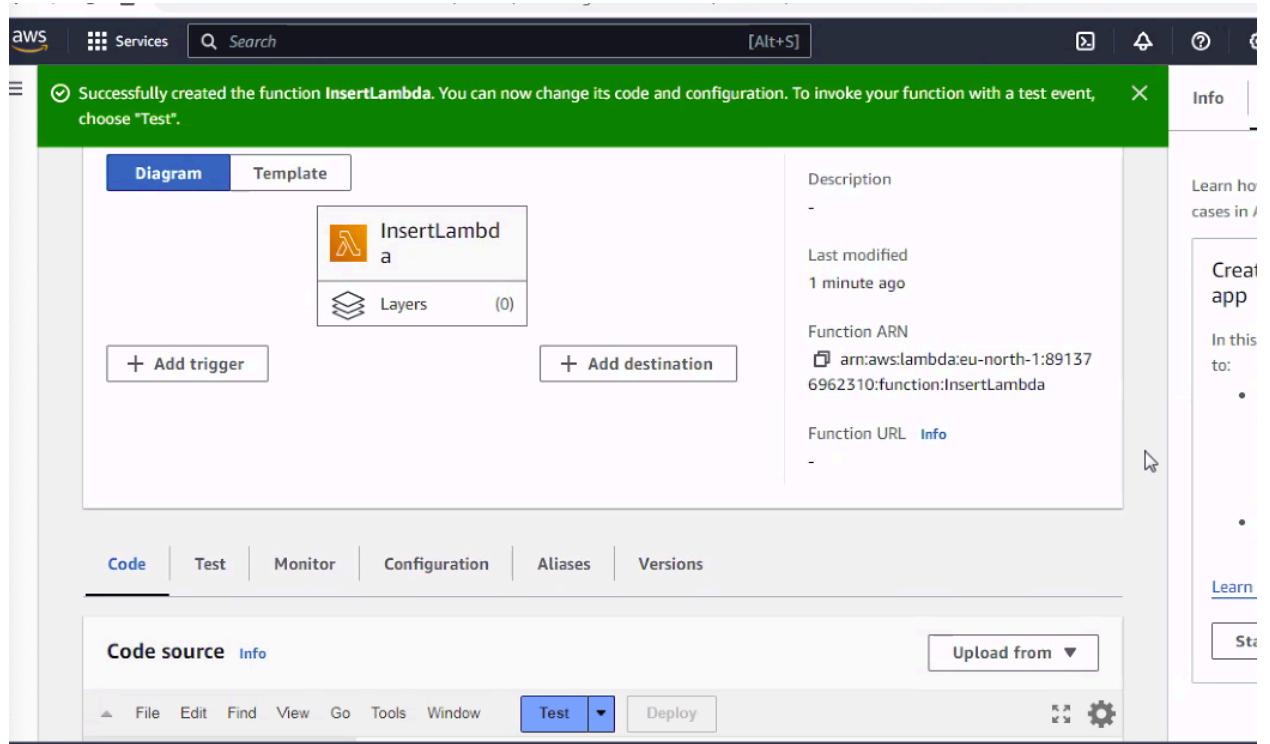


7. Click 'Create function':

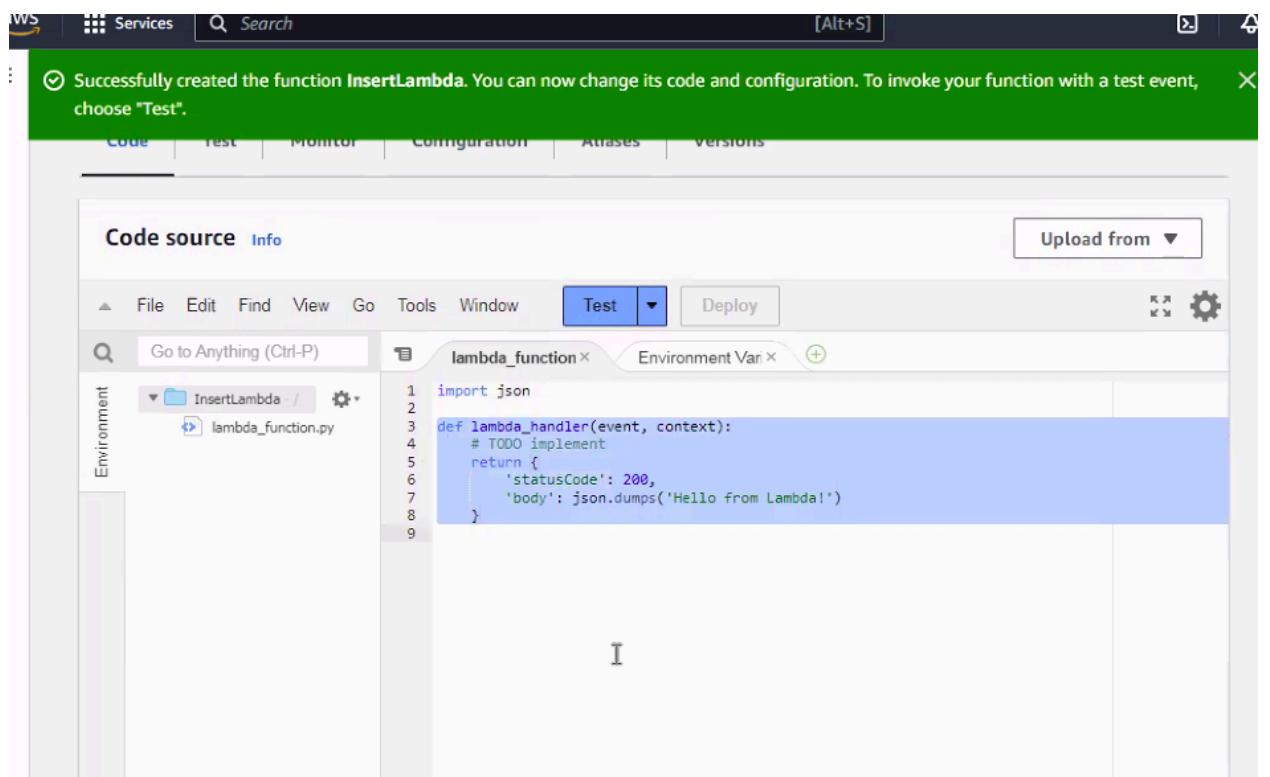
8. Success



9. Scroll down to 'Source Code'



10. Then delete the default code in the box:



11. Insert the following code into that code editor box (the “lambda_function” tab)Insert this

```

1 import json
2 import boto3
3 # create a DynamoDB object using the AWS SDK
4 dynamodb = boto3.resource('dynamodb')
5 # use the DynamoDB object to select our table
6 table = dynamodb.Table('UserDynamoTable')
7 # define the handler function that the Lambda service will use as an entry point
8 def lambda_handler(event, context):
9     # extract values from the event object we got from the Lambda service and store in a variable
10    user_email = event['userEmail']
11    user_first_name = event['userFirstName']
12    user_last_name = event['userLastName']
13    user_gender = event['userGender']
14    user_age = event['userAge']
15    # write name and time to the DynamoDB table using the object we instantiated and save response in a variable
16    response = table.put_item(
17        Item={
18            'userEmail': user_email,
19            'userFirstName': user_first_name,
20            'userLastName': user_last_name,
21            'userGender': user_gender,
22            'userAge': user_age
23        })
24    # return a properly formatted JSON object
25    return {
26        'statusCode': 200,
27        'body': json.dumps('Hello from Lambda, ' + user_email)
28    }

```

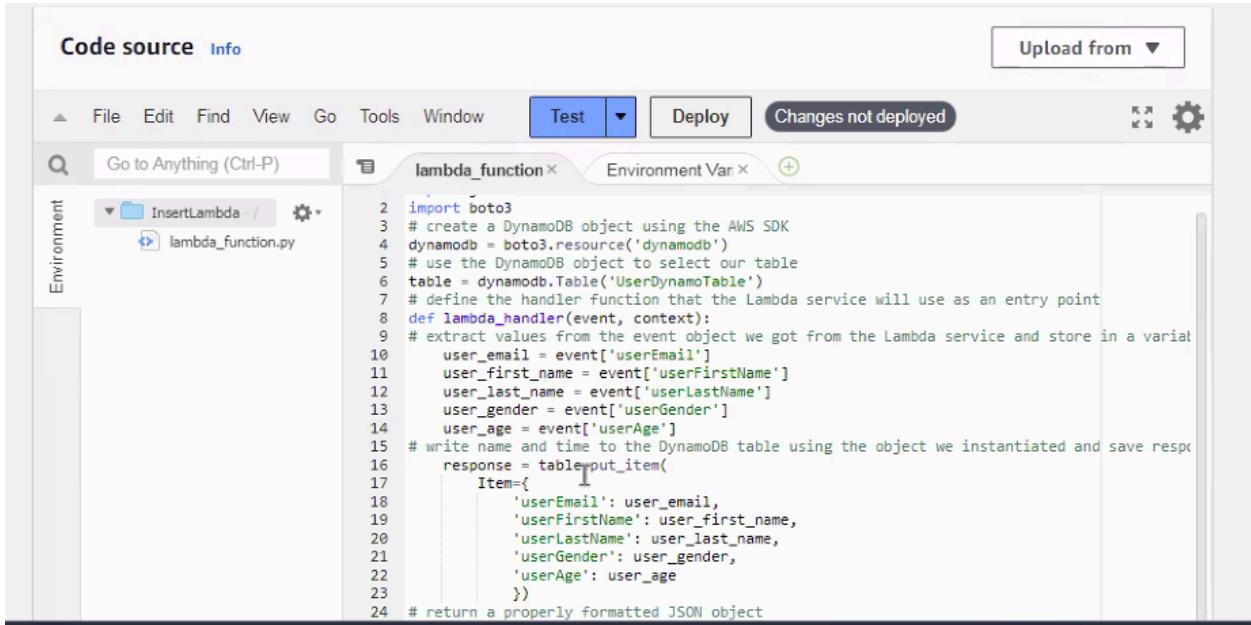
1. import json
2. import boto3
3. # create a DynamoDB object using the AWS SDK
4. dynamodb = boto3.resource('dynamodb')
5. # use the DynamoDB object to select our table
6. table = dynamodb.Table('UserDynamoTable')
7. # define the handler function that the Lambda service will use as an entry point
8. def lambda_handler(event, context):
9. # extract values from the event object we got from the Lambda service and store in a variable
10. user_email = event['userEmail']
11. user_first_name = event['userFirstName']
12. user_last_name = event['userLastName']
13. user_gender = event['userGender']
14. user_age = event['userAge']

```

15. # write name and time to the DynamoDB table using the object we
   instantiated and save response in a variable
16. response = table.put_item(
17.     Item={
18.         'userEmail': user_email,
19.         'userFirstName': user_first_name,
20.         'userLastName': user_last_name,
21.         'userGender': user_gender,
22.         'userAge': user_age
23.     })
24. # return a properly formatted JSON object
25. return {
26.     'statusCode': 200,
27.     'body': json.dumps('Hello from Lambda, ' + user_email)
28. }

```

12. Paste, and it should look like this:



The screenshot shows the AWS Lambda function editor interface. The top navigation bar includes 'Code source' and 'Info' tabs, along with 'Test', 'Deploy', and 'Changes not deployed' buttons. The main workspace displays the 'lambda_function' code. The code itself is a Python script named 'lambda_function.py' located in a folder 'InsertLambda'. The script imports the 'boto3' library and defines a handler function 'lambda_handler' that processes an event to extract user information and save it to a DynamoDB table.

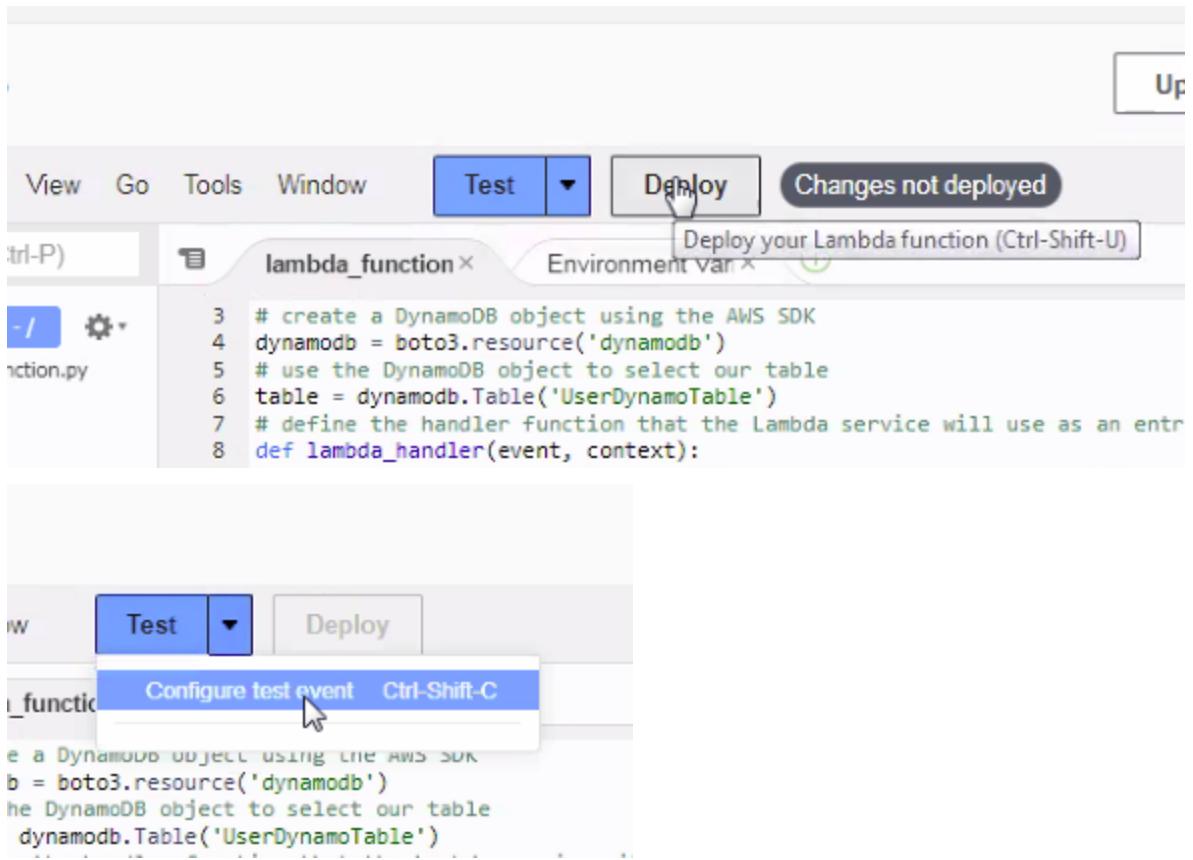
```

2 import boto3
3 # create a DynamoDB object using the AWS SDK
4 dynamodb = boto3.resource('dynamodb')
5 # use the DynamoDB object to select our table
6 table = dynamodb.Table('UserDynamoTable')
7 # define the handler function that the Lambda service will use as an entry point
8 def lambda_handler(event, context):
9     # extract values from the event object we got from the Lambda service and store in a variable
10    user_email = event['userEmail']
11    user_first_name = event['userFirstName']
12    user_last_name = event['userLastName']
13    user_gender = event['userGender']
14    user_age = event['userAge']
15    # write name and time to the DynamoDB table using the object we instantiated and save response
16    response = table.put_item(
17        Item={
18            'userEmail': user_email,
19            'userFirstName': user_first_name,
20            'userLastName': user_last_name,
21            'userGender': user_gender,
22            'userAge': user_age
23        })
24    # return a properly formatted JSON object

```

13. ctrl+s to save

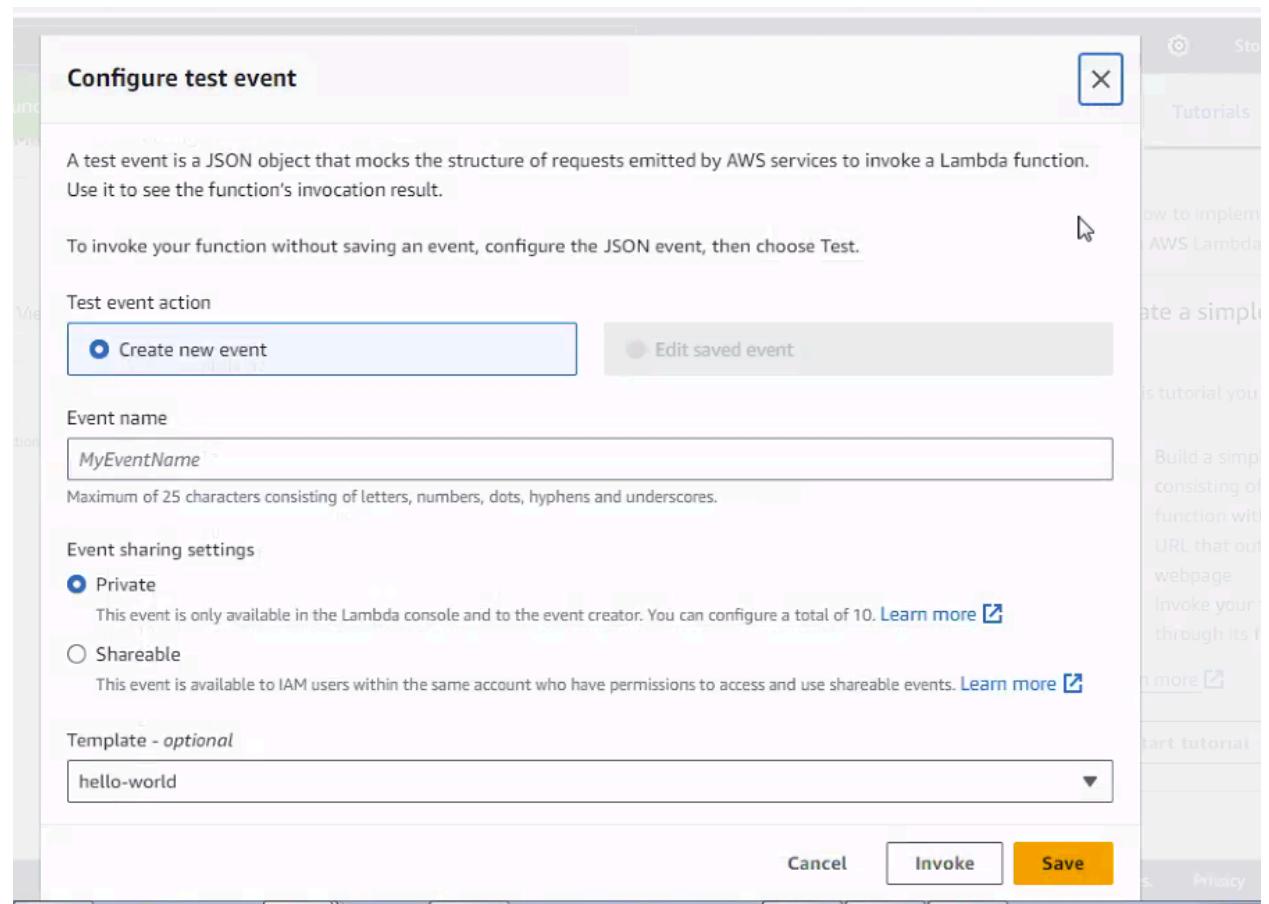
14. then deploy:



The screenshot shows the AWS Lambda function editor interface. At the top, there's a toolbar with 'View', 'Go', 'Tools', 'Window', 'Test' (which is currently selected), 'Deploy' (which has a tooltip 'Deploy your Lambda function (Ctrl-Shift-U)'), and a status message 'Changes not deployed'. Below the toolbar, there's a file navigation bar with 'lambda_function' selected. The main area contains a code editor with Python code for a Lambda function named 'lambda_handler'. The code creates a DynamoDB object, selects a table, and defines the handler function. At the bottom of the editor, there's another toolbar with 'Test' (selected) and 'Deploy' buttons, and a dropdown menu with the option 'Configure test event Ctrl-Shift-C' highlighted.

```
3 # create a DynamoDB object using the AWS SDK
4 dynamodb = boto3.resource('dynamodb')
5 # use the DynamoDB object to select our table
6 table = dynamodb.Table('UserDynamoTable')
7 # define the handler function that the Lambda service will use as an entr
8 def lambda_handler(event, context):
```

15.



16.

17. Event name: 'EventTest'

The screenshot shows the 'Event name' input field in the AWS Lambda configuration dialog. The field contains 'EventTest'. A note below it says 'Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.' Under 'Event sharing settings', 'Private' is selected. At the bottom right are 'Cancel', 'Invoke' (disabled), and 'Save' buttons.

18. Proceed

19. For Event JSON box, delete the default code already there:

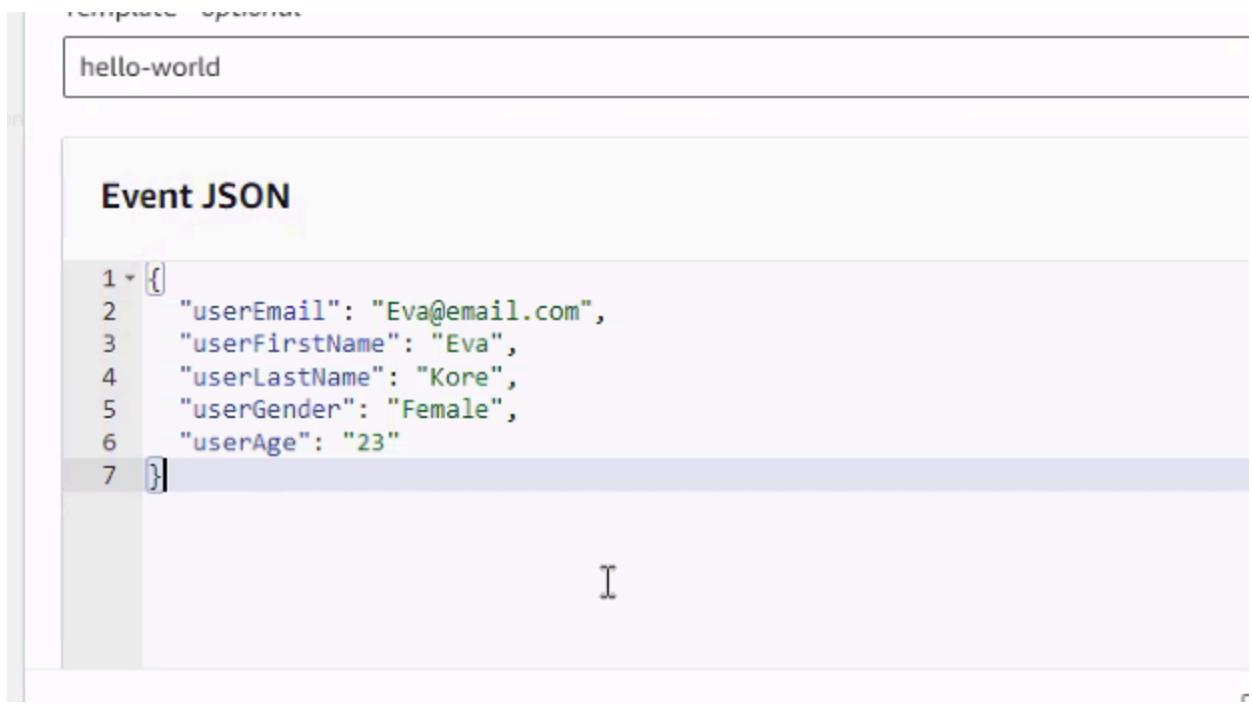


```
1 {  
2   "key1": "value1",  
3   "key2": "value2",  
4   "key3": "value3"  
5 }
```

20. Delete that code

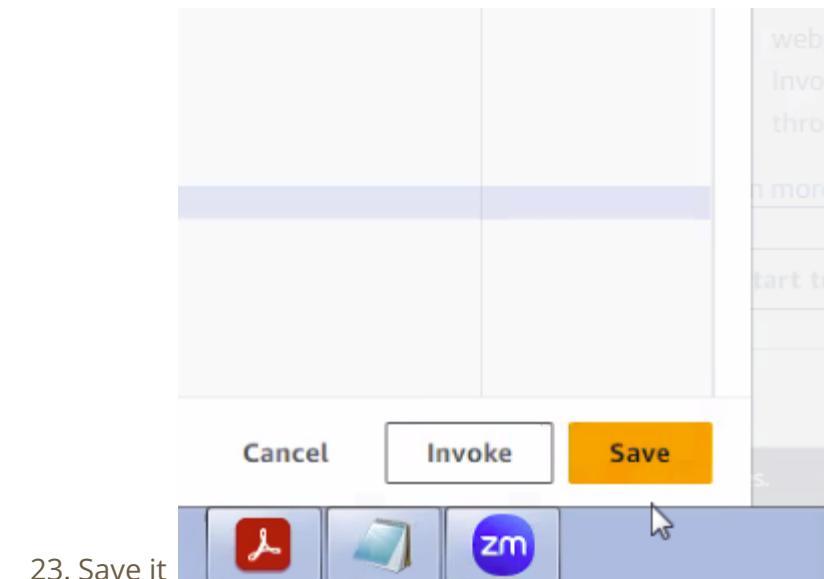
21. Then insert this code in its place:

```
1. {  
2.   "userEmail": "Eva@email.com",  
3.   "userFirstName": "Eva",  
4.   "userLastName": "Kore",  
5.   "userGender": "Female",  
6.   "userAge": "23"  
7. }
```

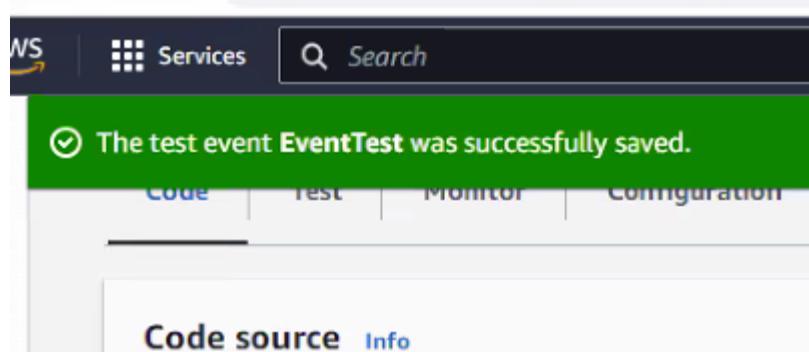


```
1 {  
2   "userEmail": "Eva@email.com",  
3   "userFirstName": "Eva",  
4   "userLastName": "Kore",  
5   "userGender": "Female",  
6   "userAge": "23",  
7   "userCity": "Berlin",  
8   "userCountry": "Germany"  
9 }
```

22.



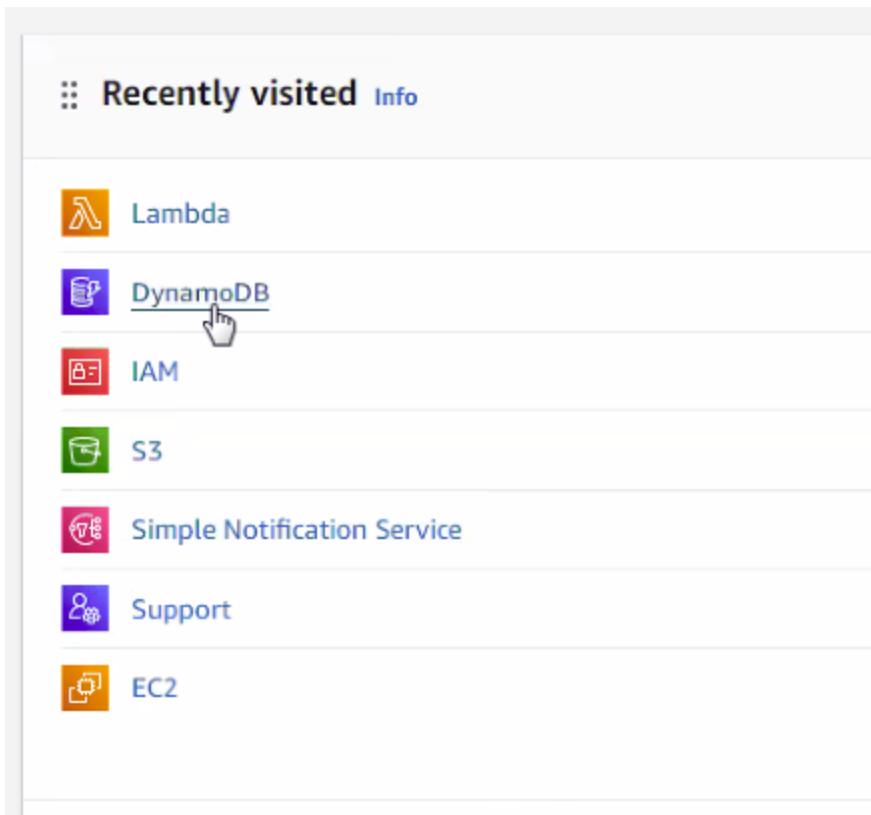
23. Save it



24. Success !

Part 6: Returning back to DynamoDB

1. Go back to dynamoDB

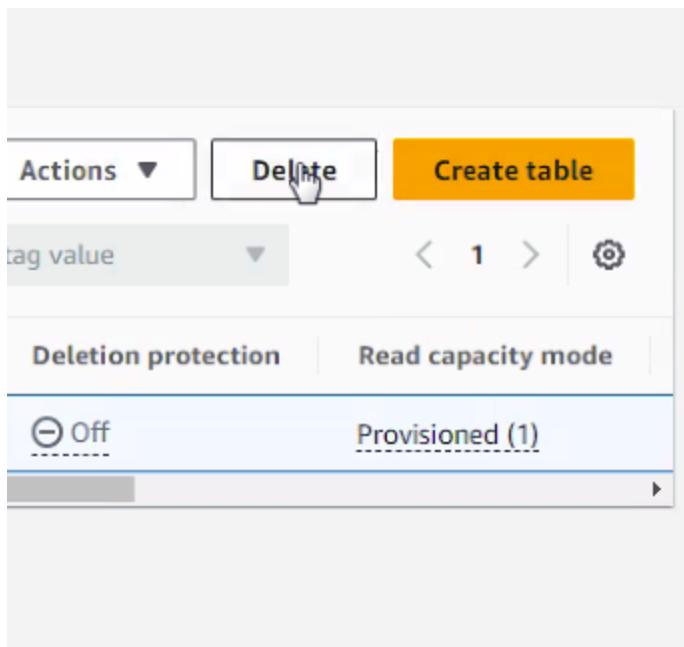


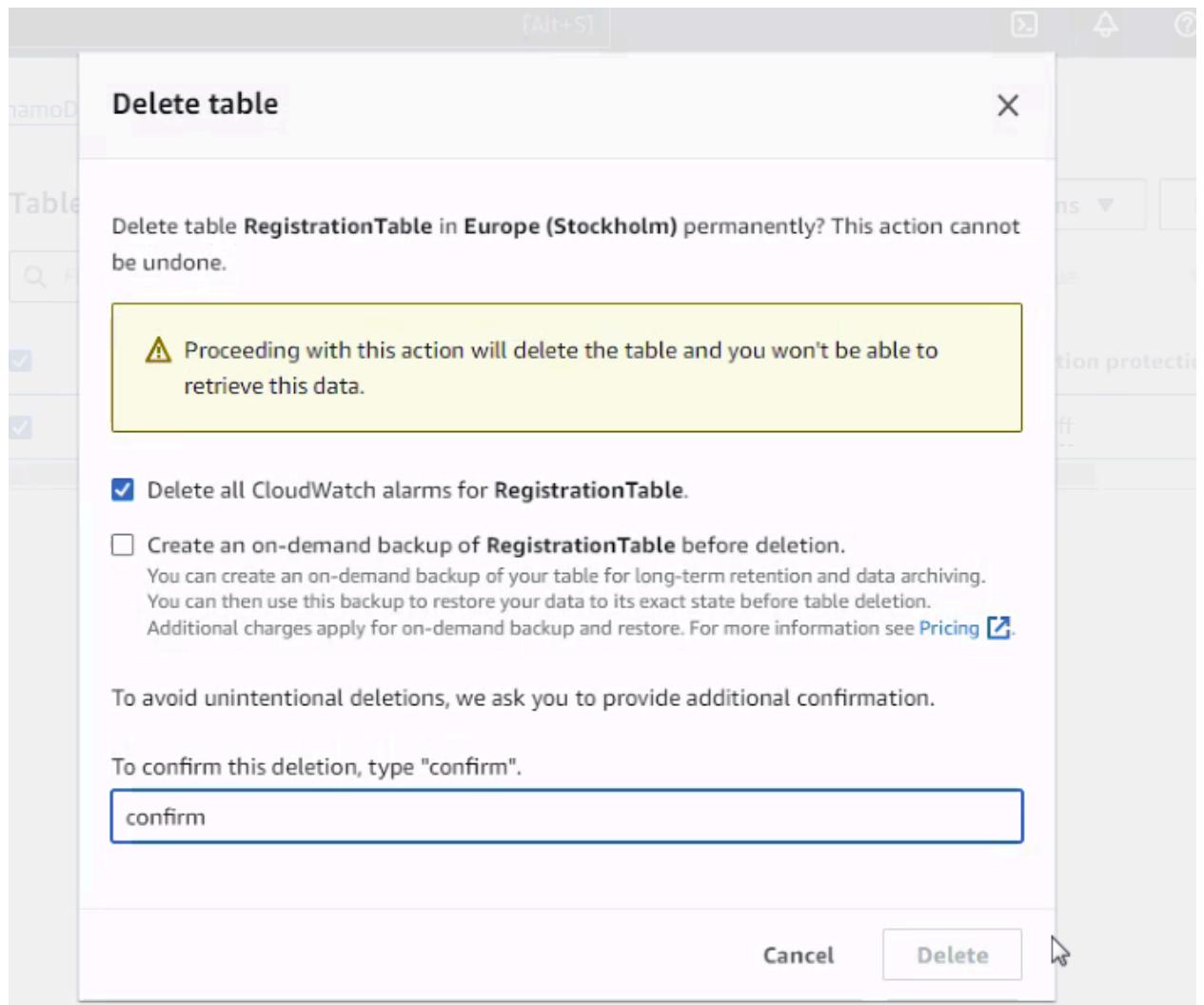


A screenshot of the AWS DynamoDB console. The top navigation bar includes the AWS logo, a services menu, and a search bar. The main title is "DynamoDB". On the left, a sidebar lists several options: "Dashboard" (blue link), "Tables" (selected, indicated by a cursor icon over the link), "Explore items", "PartiQL editor", "Backups", "Exports to S3", "Imports from S3", "Reserved capacity", and "Settings".

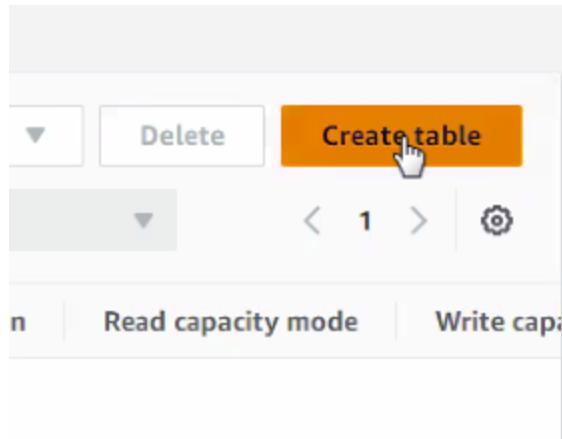
2. Go to tables

3. [are these steps necessary to keep? Or delete?] Click on your table that you created, click its check box and delete the table

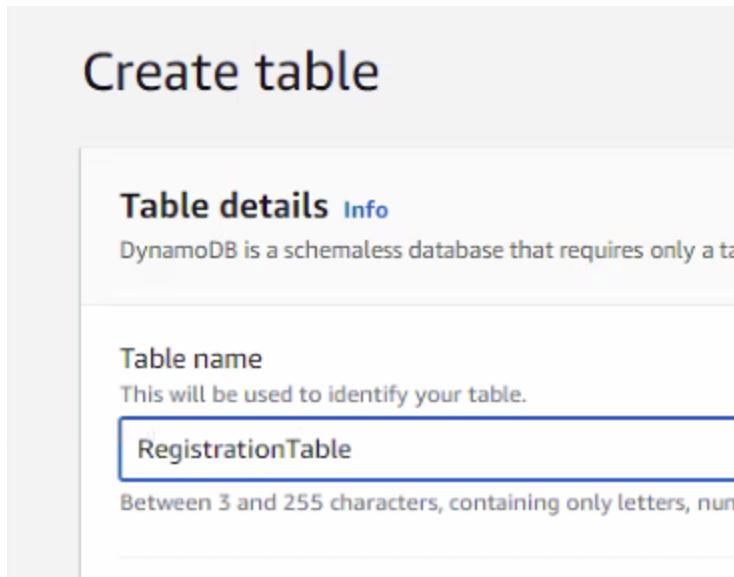




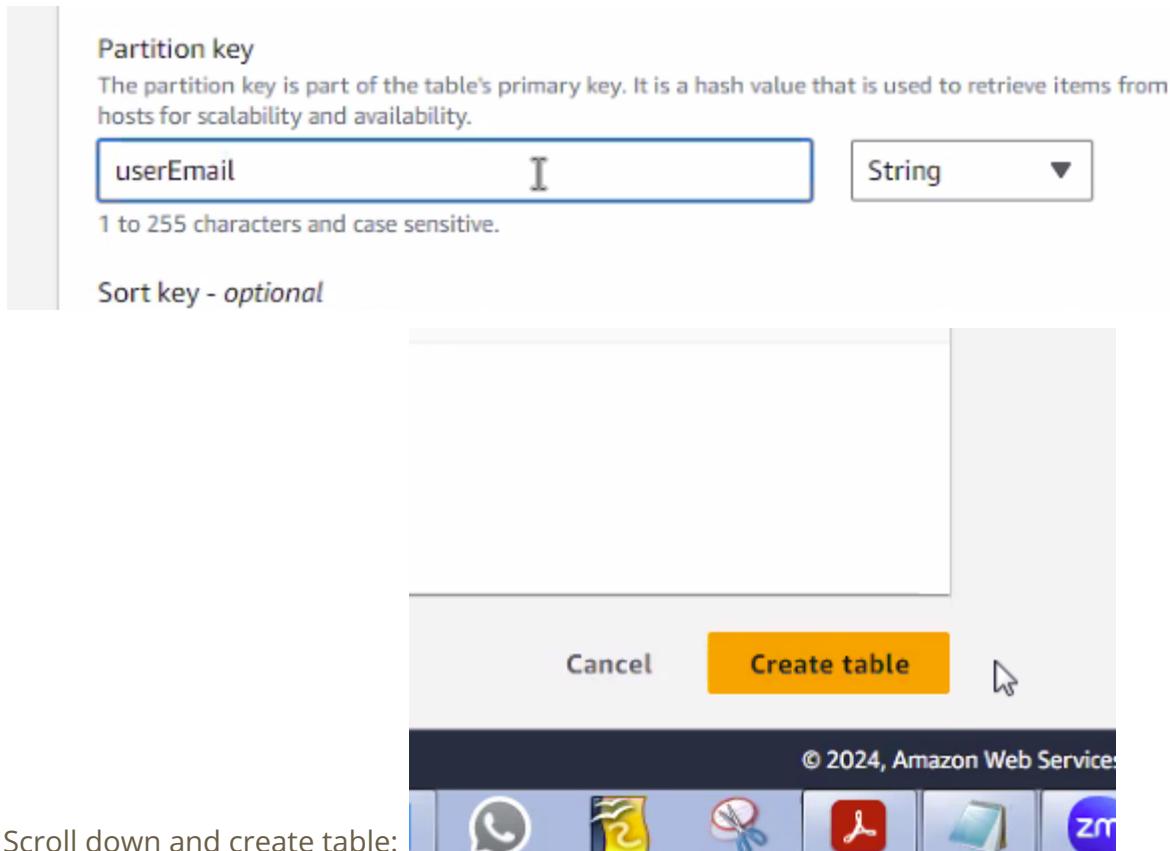
- 4.
5. [due to project not working as initially planned] Create the table again



6. Name the Registration table "RegistrationTable"



7. Name 'Partition Key' as "userEmail"



8. Scroll down and create table:

9. Success :

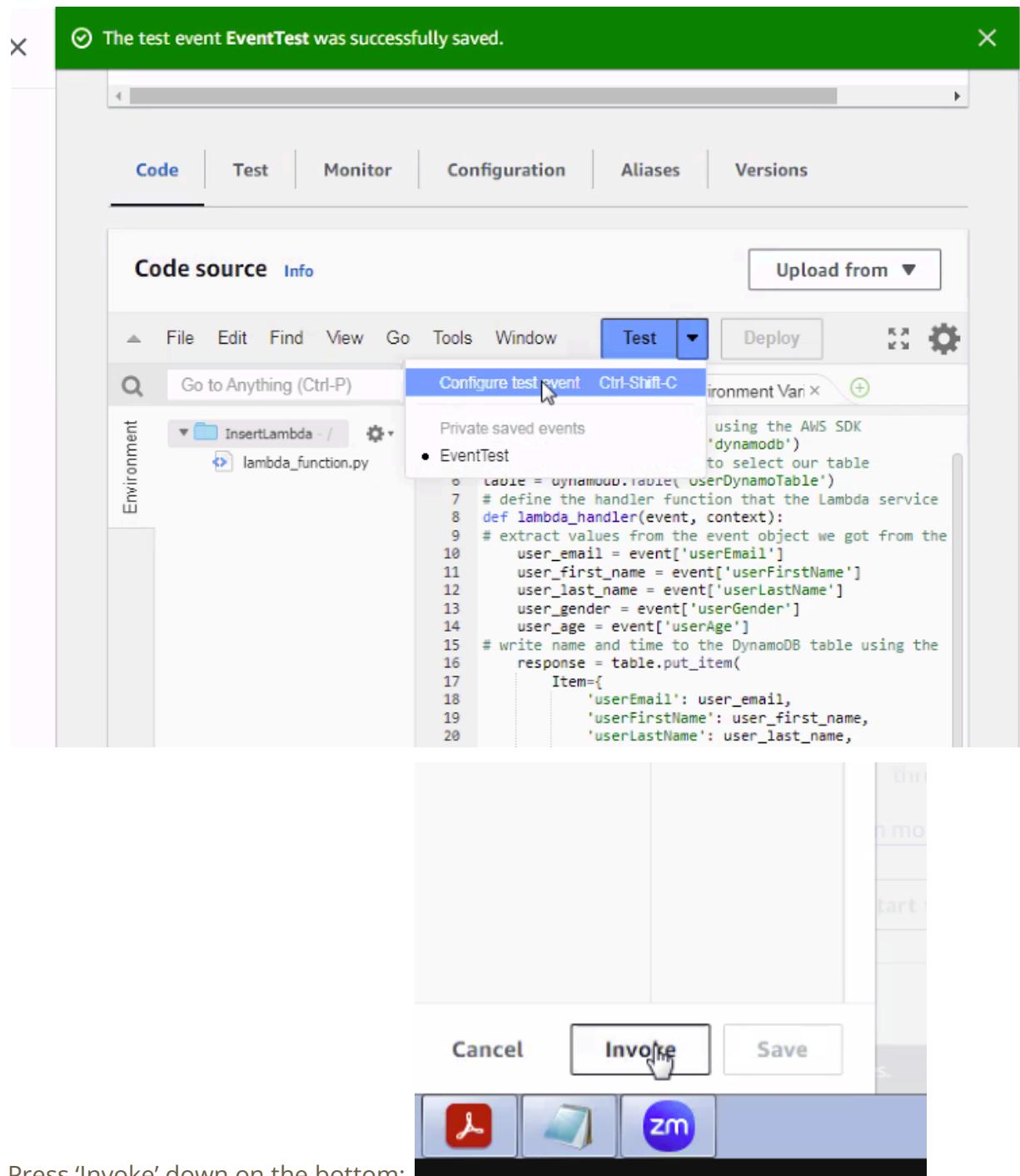
The screenshot shows the AWS DynamoDB service dashboard. On the left, there's a sidebar with links like Dashboard, Tables, Explore items, PartiQL editor, Backups, Exports to S3, Imports from S3, Reserved capacity, and Settings. The main area is titled 'Tables' and shows a table named 'RegistrationTable'. A green banner at the top right says 'The RegistrationTable table was created successfully.' The table details are as follows:

Name	Status	Partition key	Sort key	Indexes	Deletion protection	Read capacity mode
RegistrationTable	Active	userEmail (\$)	-	0	Off	Provisioned (5)

10. Go to your Lambda service tab InsertLambda:

The screenshot shows the AWS Lambda service tab. The title bar says 'InsertLambda - Lambda'. Below it, there's a browser-style address bar with a lock icon and the URL 'eu-north-1.console.aws.amazon.com'. The main navigation bar includes the AWS logo, 'Services' link, and a search bar.

11. Configure test event



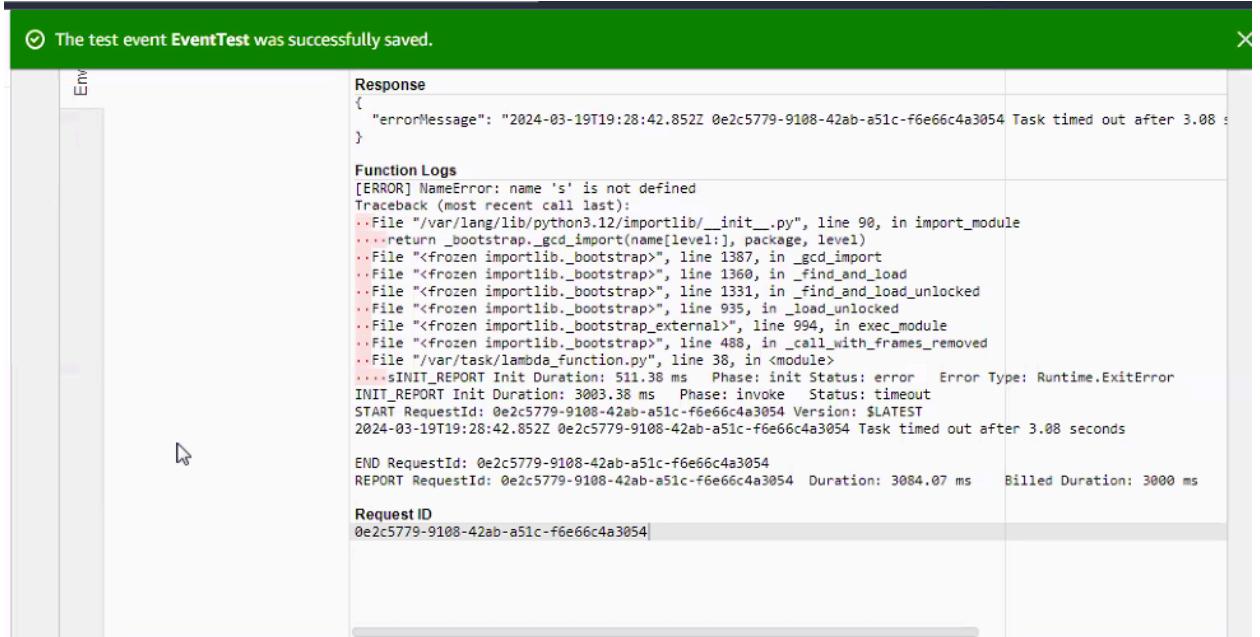
12. Press 'Invoke' down on the bottom:

13. [error! You need to investigate what went wrong!]

The screenshot shows a Lambda function configuration page. At the top, a green banner displays the message: "The test event EventTest was successfully saved." Below the banner, there are tabs for Code, Test, Monitor, Configuration, Aliases, and Versions. The Code tab is selected. In the main area, under the heading "Code source", there is an "Upload from" button and a toolbar with File, Edit, Find, View, Go, Tools, Window, Test (which is highlighted in blue), Deploy, and settings icons. A search bar labeled "Go to Anything (Ctrl-P)" is present. On the left, an "Environment" sidebar shows a folder named "InsertLambda - /" containing "lambda_function.py". The main content area displays the "Execution result" for a test event named "EventTest". The status is "Failed" with a time of 511.38 ms. The response body contains a JSON object with a key "errorMessage": "2024-03-19T19:28:42.85Z 0e2c5779-9108-42ab-a...". The "Function Logs" section shows a detailed traceback for a NameError:

```
[ERROR] NameError: name 's' is not defined
Traceback (most recent call last):
  File "/var/lang/lib/python3.12/importlib/_init_.py", line 90
    ...return _bootstrap._gcd_import(name[level:], package, level)
  File "<frozen importlib._bootstrap>", line 1387, in _gcd_impor
  File "<frozen importlib._bootstrap>", line 1360, in _find_and_
  File "<frozen importlib._bootstrap>", line 1331, in _find_and_
  File "<frozen importlib._bootstrap>", line 935, in _load_unlo
  File "<frozen importlib._bootstrap_external>", line 994, in e
```

14. Status: FAILED! Read the Function Logs to see the problem. What do you think went wrong?



The screenshot shows the AWS Lambda Test event details page. At the top, a green banner says "The test event EventTest was successfully saved." Below the banner, there are two tabs: "Env" and "Response". The "Response" tab is selected, displaying a JSON object with a single key-value pair: "errorMessage": "2024-03-19T19:28:42.852Z 0e2c5779-9108-42ab-a51c-f6e66c4a3054 Task timed out after 3.08 seconds". The "Function Logs" tab is also visible, showing a detailed traceback of the error. The logs indicate a NameError: name 's' is not defined, which occurred at line 38 of the lambda_function.py file. The logs also show the task timing out after 3.08 seconds.

```

{
  "errorMessage": "2024-03-19T19:28:42.852Z 0e2c5779-9108-42ab-a51c-f6e66c4a3054 Task timed out after 3.08 seconds"
}

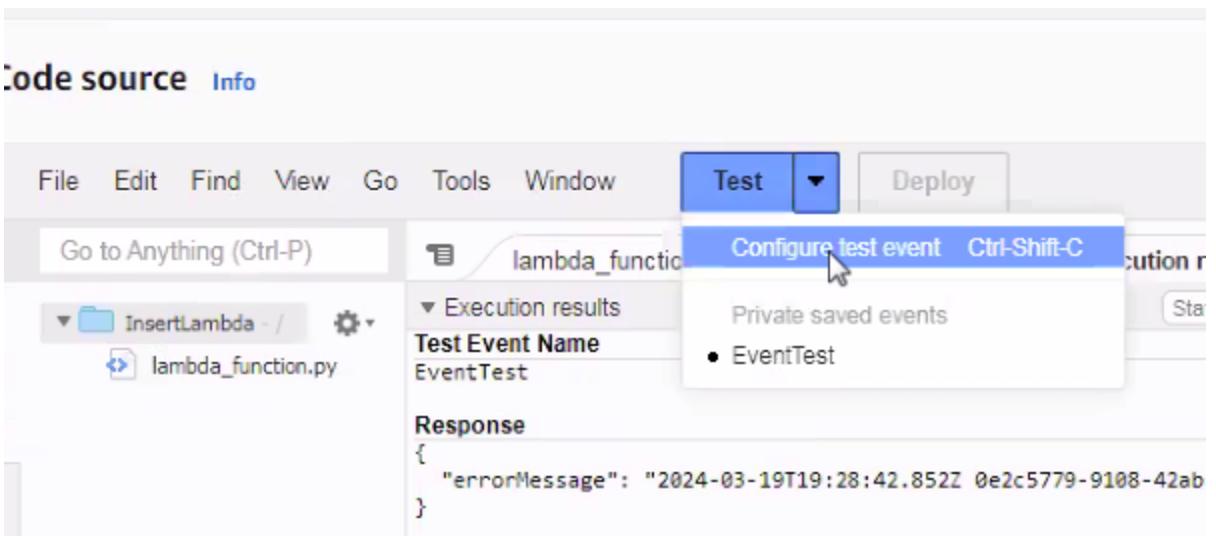
[ERROR] NameError: name 's' is not defined
Traceback (most recent call last):
  File "/var/lang/lib/python3.12/importlib/_bootstrap/__init__.py", line 90, in import_module
    return _bootstrap._gcd_import(name[level:], package, level)
  File "<frozen importlib._bootstrap>", line 1387, in _gcd_import
  File "<frozen importlib._bootstrap>", line 1360, in _find_and_load
  File "<frozen importlib._bootstrap>", line 1331, in _find_and_load_unlocked
  File "<frozen importlib._bootstrap>", line 935, in _load_unlocked
  File "<frozen importlib._bootstrap_external>", line 994, in exec_module
  File "<frozen importlib._bootstrap>", line 488, in _call_with_frames_removed
  File "/var/task/lambda_function.py", line 38, in <module>
    ...INIT_REPORT Init Duration: 511.38 ms Phase: init Status: error Error Type: Runtime.ExitError
INIT_REPORT Init Duration: 3003.38 ms Phase: invoke Status: timeout
START RequestId: 0e2c5779-9108-42ab-a51c-f6e66c4a3054 Version: $LATEST
2024-03-19T19:28:42.852Z 0e2c5779-9108-42ab-a51c-f6e66c4a3054 Task timed out after 3.08 seconds

END RequestId: 0e2c5779-9108-42ab-a51c-f6e66c4a3054
REPORT RequestId: 0e2c5779-9108-42ab-a51c-f6e66c4a3054 Duration: 3084.07 ms Billed Duration: 3000 ms

Request ID
0e2c5779-9108-42ab-a51c-f6e66c4a3054

```

15. Check the log, it mentions "line 38" what is on line 38? (we will find out in the next few steps)
16. Go back here to investigate the error:



The screenshot shows the AWS Lambda function configuration interface. The top navigation bar includes "File", "Edit", "Find", "View", "Go", "Tools", "Window", "Test" (which is currently selected), and "Deploy". A dropdown menu is open under "Test", showing options like "Configure test event" and "Ctrl-Shift-C". The main area displays the "Execution results" section for a test event named "EventTest". The "Response" field contains the same JSON object as the previous screenshot. The left sidebar shows the project structure with "InsertLambda" and "lambda_function.py".

17. Click on 'lambda_function' tab in the Code Source editor

The screenshot shows the AWS Lambda Code Source editor interface. At the top, there are tabs for 'Code source' and 'Info'. Below the tabs is a menu bar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', and 'Window'. To the right of the menu is a toolbar with 'Test' and 'Deploy' buttons. A search bar labeled 'Go to Anything (Ctrl-P)' is positioned above the code editor. On the left, there's a sidebar titled 'Environment' with a dropdown menu showing 'InsertLambda - /' and a file icon next to 'lambda_function.py'. The main area contains the following Python code:

```

3 # create a DynamoDB object using the AWS SDK
4 dynamodb = boto3.resource('dynamodb')
5 # use the DynamoDB object to select our
6 table = dynamodb.Table('UserDynamoTable')
7 # define the handler function that the
8 def lambda_handler(event, context):
9 # extract values from the event object
10    user_email = event['userEmail']
11    user_first_name = event['userFirstName']
12    user_last_name = event['userLastName']
13    user_gender = event['userGender']
14    user_age = event['userAge']

```

18. Line 6! fix that. The code had a table name from another specific user's account. Change it to yours. Name the table what you named the table in your system, the script has named it "UserDynamoTable" by default

The screenshot shows the AWS Lambda Code Source editor interface with the 'lambda_function' tab selected. The code editor displays the same Python script as the previous screenshot. A specific line of code, 'table = dynamodb.Table('UserDynamoTable')', is highlighted with a blue selection bar, indicating it is being edited. The rest of the code remains the same.

19. Check what you named your DynamoDB table? E.g. this lab has name its table 'RegistrationTable'

The screenshot shows the AWS DynamoDB 'Tables' page. At the top, a green success message reads: 'The RegistrationTable table was created successfully.' Below the message, the navigation bar shows 'DynamoDB > Tables'. The main area is titled 'Tables (1) Info'. A search bar contains the placeholder 'Find tables by table name'. To the right of the search bar are filters for 'Any tag key' and 'Any tag value'. A table lists one table entry:

	Name	Status	Partition key	Sort key	Indexes	Delet
<input type="checkbox"/>	RegistrationTable	Active	userEmail (S)	-	0	Of

20. Click 'deploy' button

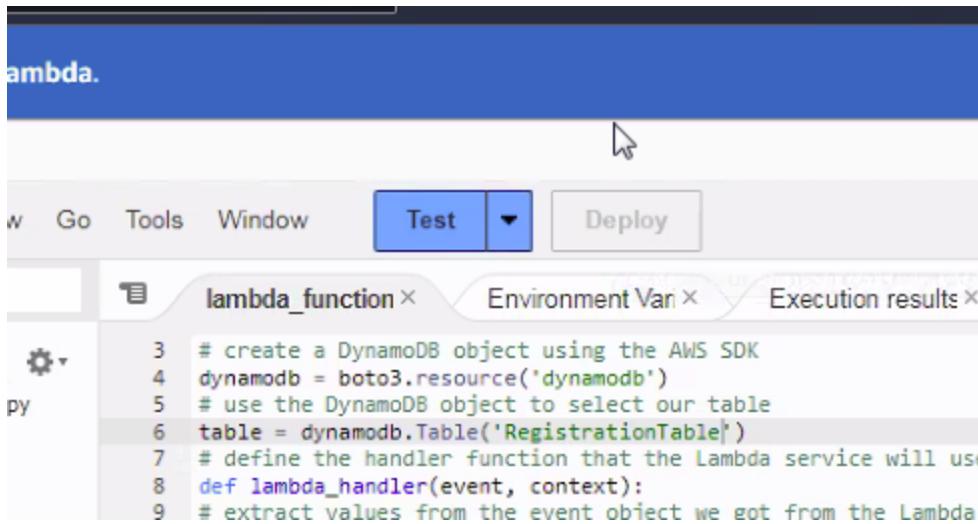
The screenshot shows the AWS Lambda function editor. The top navigation bar includes 'Tools', 'Window', 'Test' (which is selected), 'Deploy', and a status message 'Changes not deployed'. Below the navigation bar, there are tabs for 'lambda_function', 'Environment Vari', and 'Execution results'. The code editor displays a Python script for a Lambda function:

```

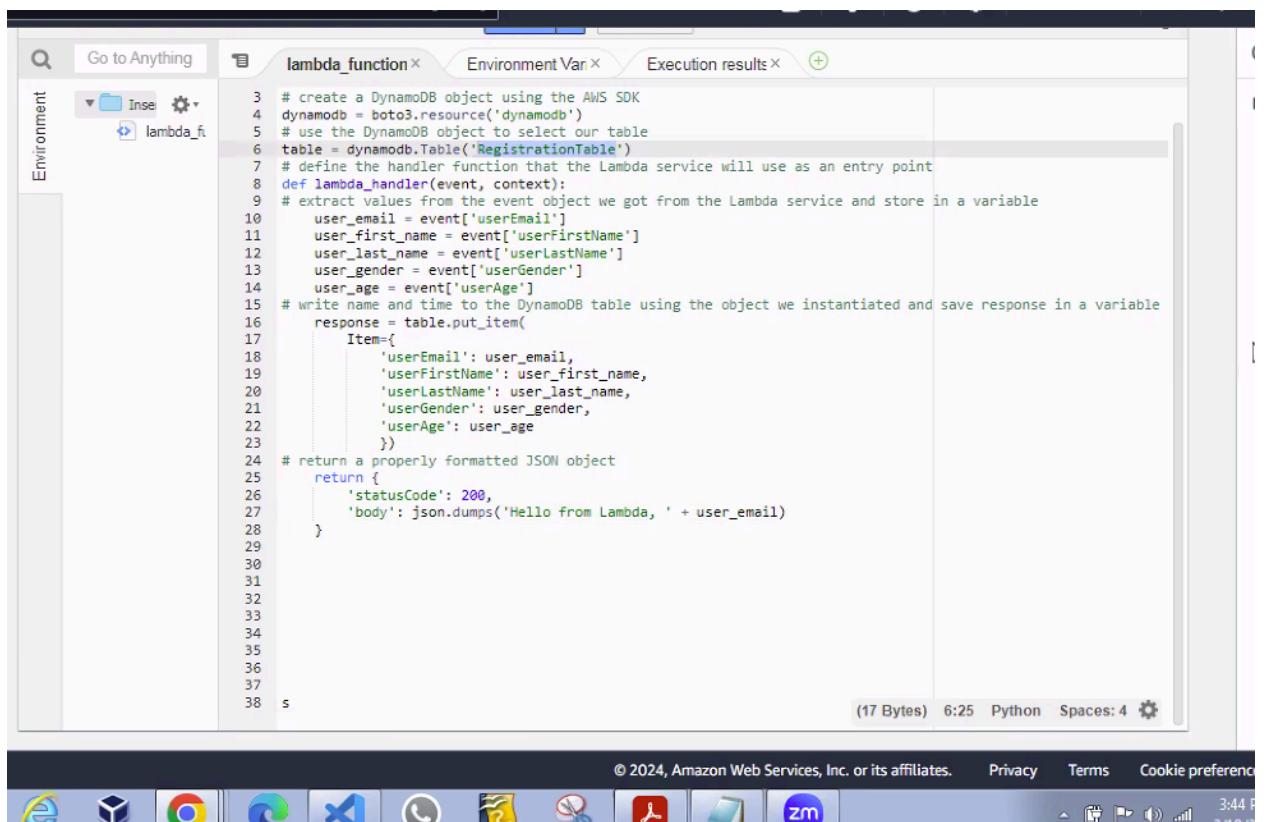
3 # create a DynamoDB object using the AWS SDK
4 dynamodb = boto3.resource('dynamodb')
5 # use the DynamoDB object to select our table
6 table = dynamodb.Table('RegistrationTable')
7 # define the handler function that the Lambda service will use as an entry point
8 def lambda_handler(event, context):
9     # extract values from the event object we got from the Lambda service and store in a
10     user_email = event['userEmail']
11     user_first_name = event['userFirstName']
12     user_last_name = event['userLastName']
13     user_gender = event['userGender']
14     user_age = event['userAge']
15     # write name and time to the DynamoDB table using the object we instantiated and save
16     response = table.put_item(
17         Item={
18             'userEmail': user_email,
19             'userFirstName': user_first_name,
20             'userLastName': user_last_name,
21             'userGender': user_gender,
22             'userAge': user_age
23         })
24     # return a properly formatted JSON object
25     return {

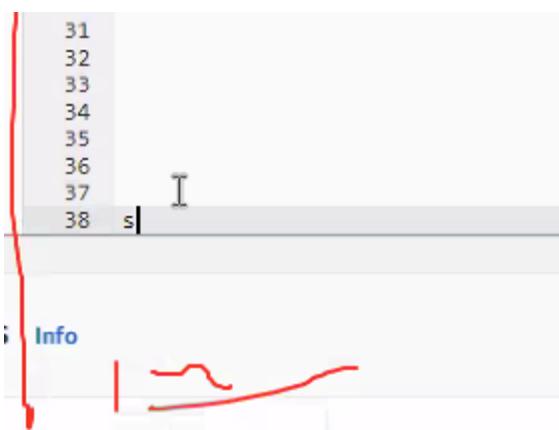
```

21. Now click 'Test'



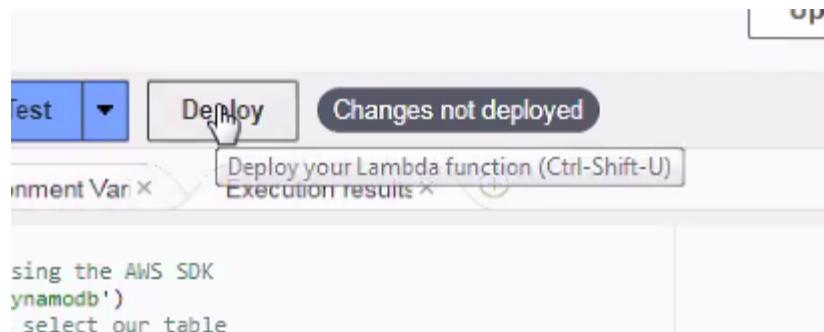
22. See anything odd?



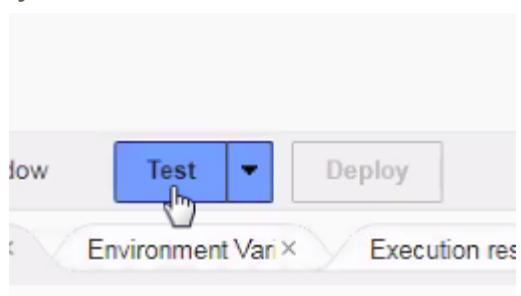


31
32
33
34
35
36
37
38 s|

23. Look at line 38:
24. Delete that letter 's' on line 38

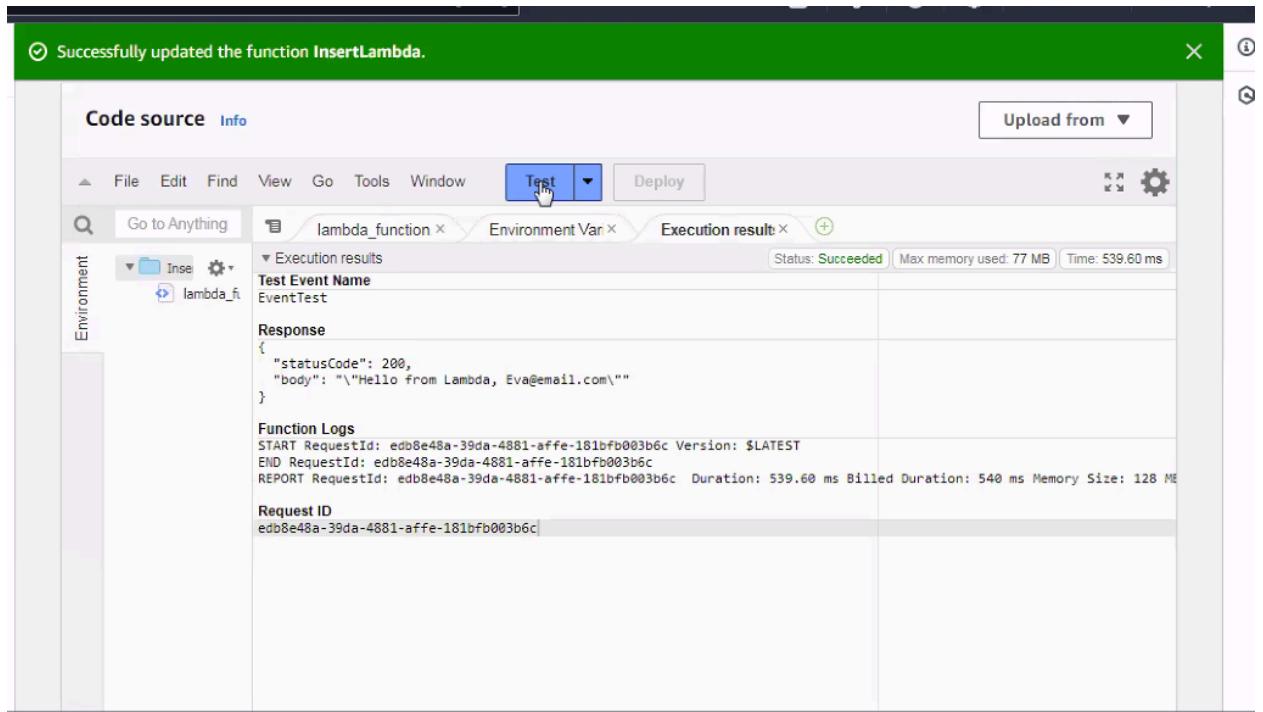


25. Click 'deploy'

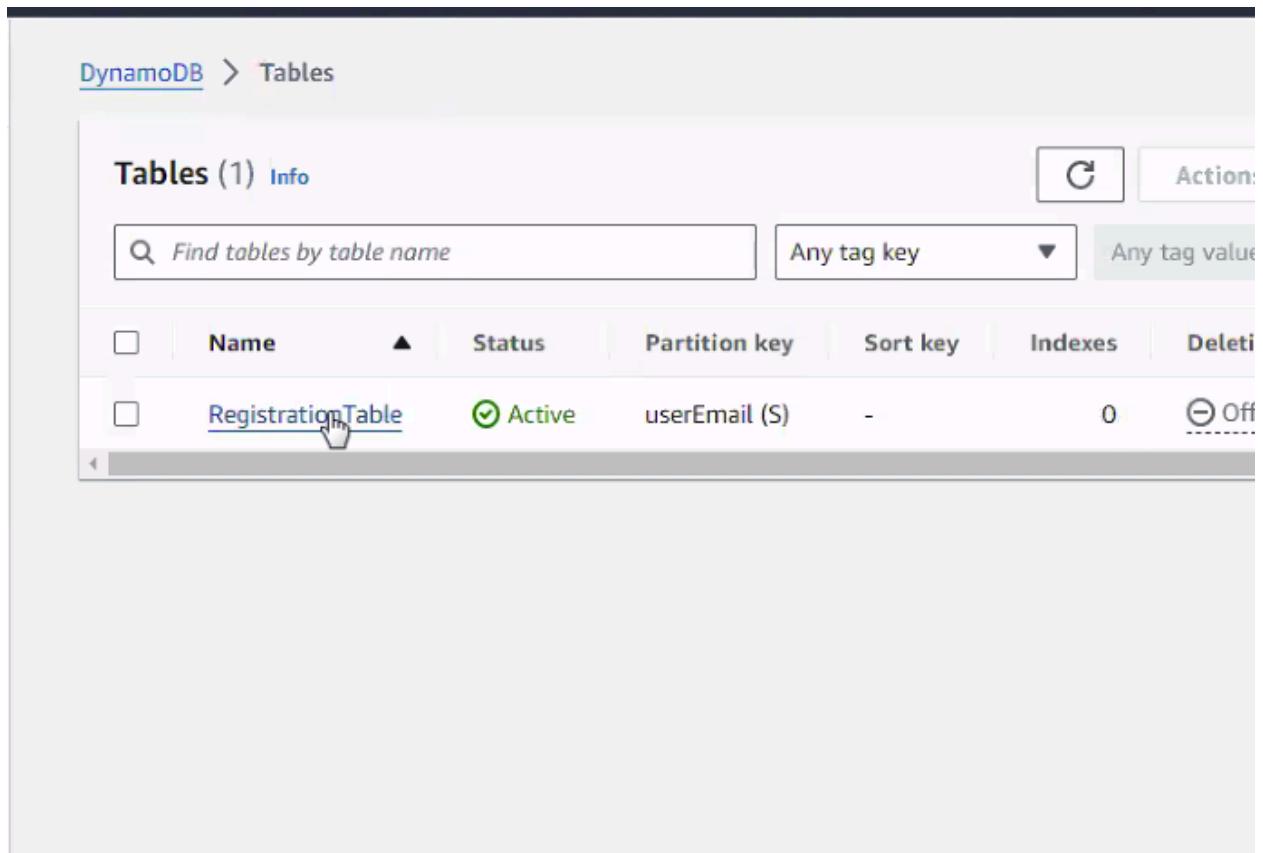


26. Click 'test'

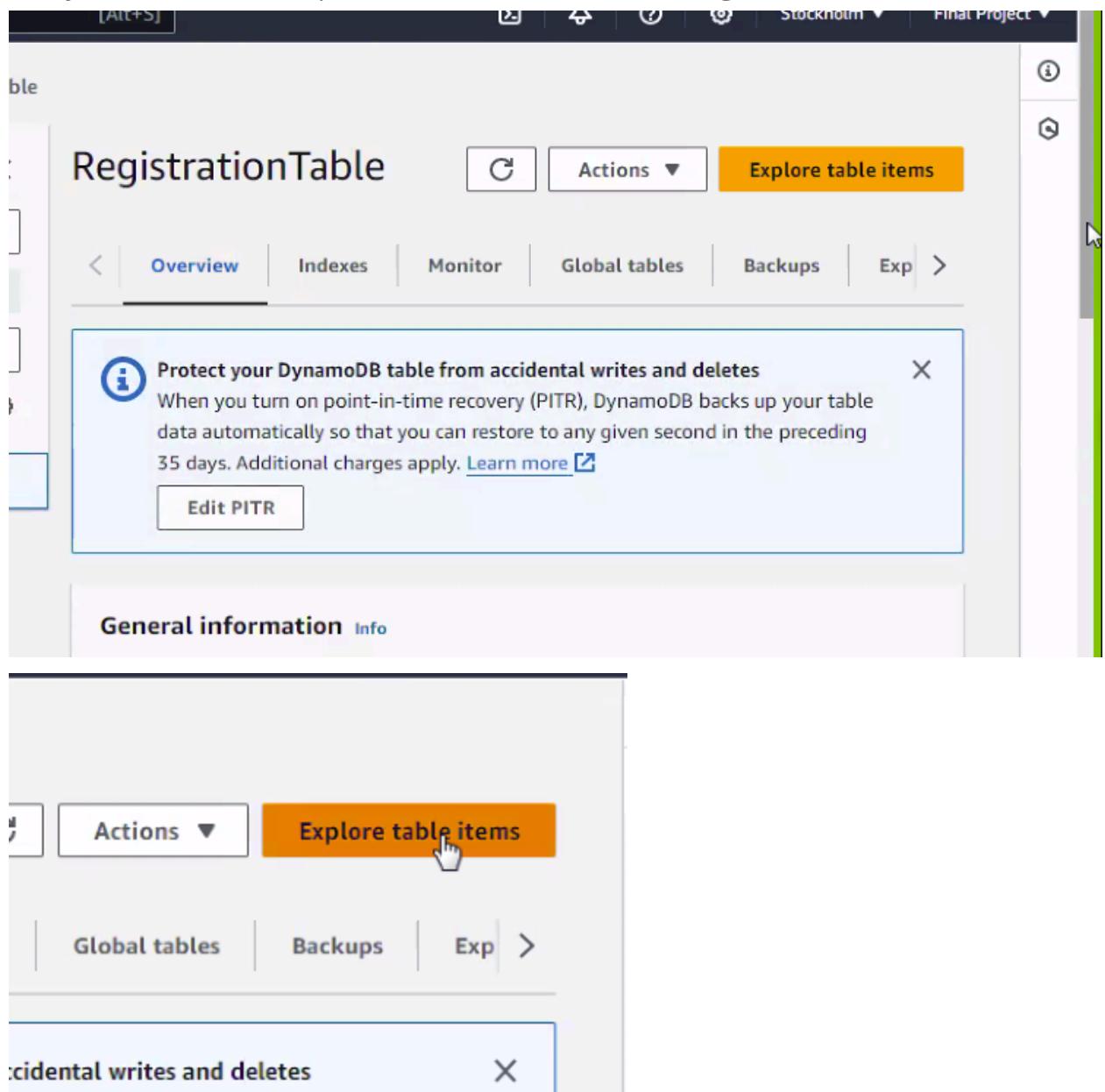
27. Success!



28. Go to DynamoDB and click on your table:



29. On DynamoDB click on 'explore table items' button on the right:



30. Press 'Run'

The screenshot shows the AWS DynamoDB Management Console. On the left, there's a sidebar with options like Dashboard, Tables, Explore items, Partition editor, Backups, Exports to S3, Imports from S3, Reserved capacity, and Settings. The main area is titled 'Tables (1)' and shows a single table named 'RegistrationTable'. To the right, the 'RegistrationTable' details page is displayed. It has sections for 'Scan or query items' (with 'Scan' selected), 'Select a table or index' (set to 'Table - RegistrationTable'), 'Select attribute projection' (set to 'All attributes'), and 'Filters'. At the bottom, there's a 'Run' button which is being clicked, and a message box indicating 'Completed. Read capacity units consumed: 0.5'. Below the table details, there are buttons for 'Actions' and 'Create item'.

31. [this may be repeat from step 29] Click 'explore table items'

The screenshot shows the 'RegistrationTable' overview page. At the top, there are buttons for 'Actions' and 'Explore table items' (which is highlighted with a cursor). Below the table name, there are tabs for 'Overview' (which is selected), 'Indexes', 'Monitor', 'Global tables', 'Backups', and 'Exp'. A modal window is open, titled 'Protect your DynamoDB table from accidental writes and deletes'. It explains that turning on PITR backs up data automatically for 35 days and includes a 'Learn more' link. There's also a 'Edit PITR' button. At the bottom, there's a 'General information' section with a 'Info' link.

32. Then press 'Run'. It should say 'completed',

The screenshot shows the AWS DynamoDB console with the 'RegistrationTable' selected. In the 'Scan or query items' section, the 'Scan' button is highlighted. Below it, the table 'Table - RegistrationTable' and attribute projection 'All attributes' are selected. At the bottom, a green success message indicates the operation completed with 0.5 read capacity units consumed.

33. Scroll down

The screenshot shows the 'Items returned' section of the AWS DynamoDB console. It displays a single item with the following attributes and values:

	userEmail (String)	userAge	userFirstName	userGender
	Eva@email.com	23	Eva	Female

34. Check the 'Items returned' section. There should be one entry there, the email you previously input:

The screenshot shows the AWS Lambda function configuration interface. At the top, there is a green success message: "Completed. Read capacity units consumed: 0.5". Below this, there is a table titled "Items returned (1)". The table has four columns: "userEmail (String)", "userAge", "userFirstName", and "userGender". The first row contains the values: "Eva@email.com", "23", "Eva", and "Female". The table includes standard navigation controls like a header bar with "Actions" and "Create item" buttons, and a footer with copyright information.

userEmail (String)	userAge	userFirstName	userGender
Eva@email.com	23	Eva	Female

35. Success. This shows that it is being stored in this table.

Part 7: API Gateway

1. Search in your console for 'API gateway'

The screenshot shows the AWS Management Console with the URL `eu-north-1.console.aws.amazon.com/apigateway/main/apis?region=eu-north-1`. The page title is "Amazon API Gateway" under "Networking & Content Delivery". Below the title, it says "create, maintain, and secure APIs at any scale". A description follows: "Amazon API Gateway helps developers to create and manage APIs to back-end systems running on Amazon EC2, AWS Lambda, or any publicly addressable web service. With Amazon API Gateway, you can generate custom client SDKs for your APIs, to connect your back-end systems to mobile, web, and server applications or services." The navigation path is "API Gateway > APIs > Create API". The main content area is titled "Choose an API type" and contains a box for "HTTP API" which is described as "Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support".

2. Scroll down to REST API
3. Click 'Build'

The screenshot shows the "REST API" section of the AWS API Gateway. It includes a brief description: "Develop a REST API where you gain complete control over the request and response along with API management capabilities." Below this, it states "Works with the following: Lambda, HTTP, AWS Services". At the bottom right, there are two buttons: "Import" and "Build". The "Build" button is highlighted with a mouse cursor icon.

4. In the 'API name' box, name it 'Registration-API'

Create REST API

API details

New API
Create a new REST API.

Clone existing API
Create a copy of an API in this AWS account.

Import API
Import an API from an OpenAPI definition.

Example API
Learn about API Gateway with an example API.

API name: Registration-API

Description - *optional*

API endpoint type: Regional

Cancel Create API

5. Go down and click 'Create API'

Registration-API

Description - *optional*

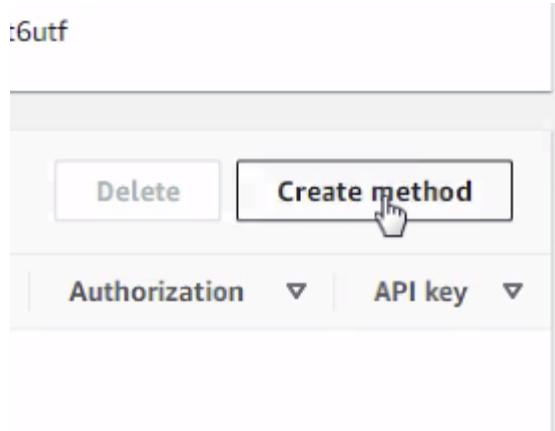
API endpoint type: Regional

Cancel Create API

6. Success

The screenshot shows a software interface for managing APIs. At the top, there's a green header bar with the message "Successfully created REST API 'Registration-API (r5xhyu8osf)'". Below the header, the main area is titled "Resources". On the left, there's a sidebar with a "Create resource" button and a path entry field containing "/". The main content area is divided into two sections: "Resource details" and "Methods (0)". In the "Resource details" section, the path is set to "/" and the resource ID is "t5jc9t6utf". There are buttons for "Update documentation" and "Enable CORS". In the "Methods (0)" section, there are buttons for "Delete" and "Create method". A sub-section below shows filter options for "Method type", "Integration type", "Authorization", and "API key". The status message "No methods" and "No methods defined." is displayed.

7. Below 'Resource details' in the 'Methods' section, click 'Create method' button on the



right side

8. In the 'Create method' section, click 'Method type' dropdown menu

The screenshot shows the 'Create method' interface. In the 'Method details' section, the 'Method type' dropdown is open, showing 'Select a method type' at the top, followed by three options: 'ANY', 'DELETE', and 'GET'. A cursor is positioned over the dropdown menu. To the right of the dropdown, there is a small 'Mock' button with the text 'generate a response'.

9. Select 'POST'

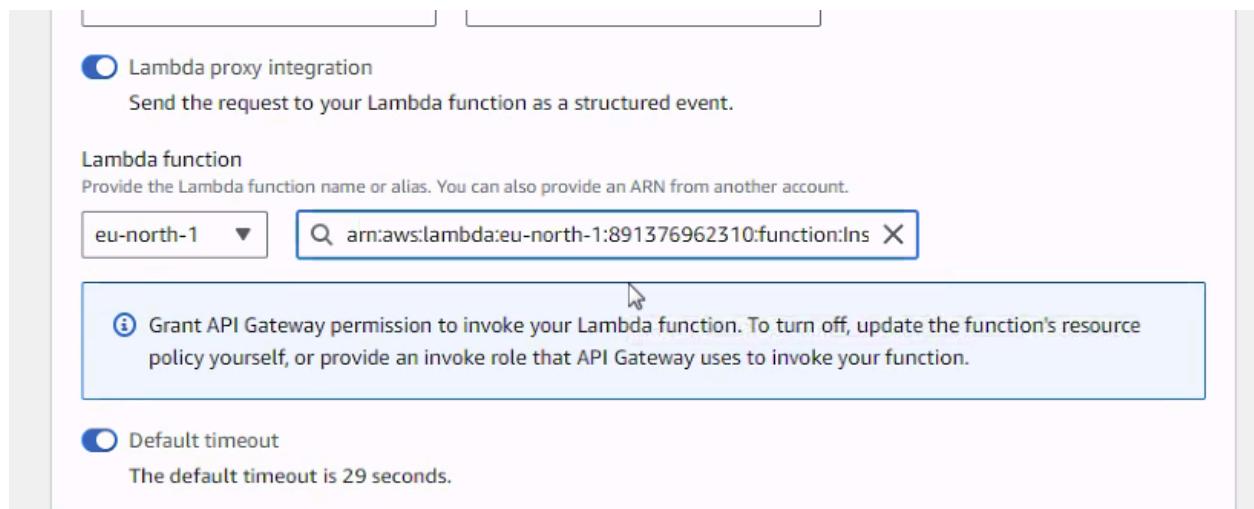
The screenshot shows the 'Create method' interface. In the 'Method details' section, the 'Method type' dropdown is now set to 'POST'. Below it, the 'Integration type' dropdown is also visible.

10. Scroll down to 'Lambda proxy integration' and click on it to activate it

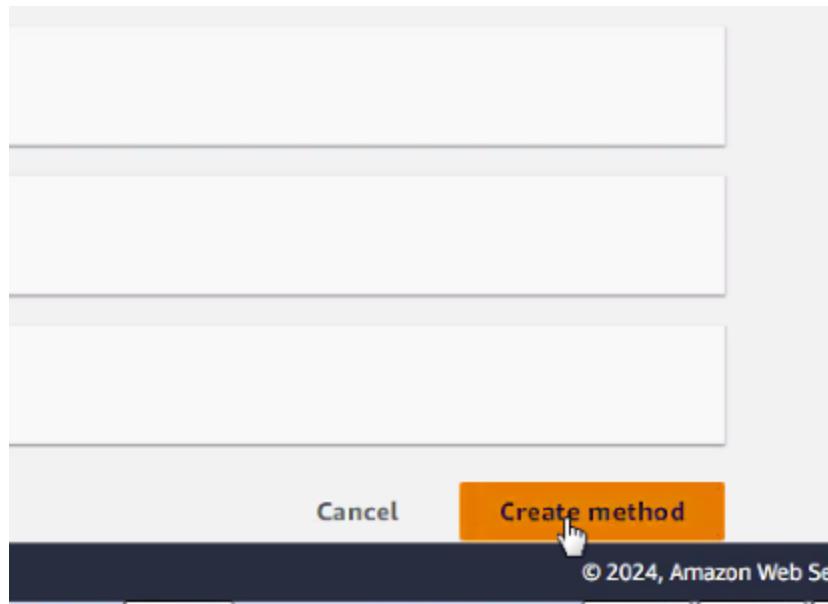
The screenshot shows the 'Create method' interface. The 'Lambda proxy integration' section is visible, featuring a toggle switch that is turned on (blue). Below the switch, the text 'Send the request to your Lambda function as a structured event.' is displayed.

11. In 'Lambda function', click on the search bar and your ARN should show up automatically in the drop down menu

e.g.arn:aws:lambda:eu-north-1:891376962310:function.....

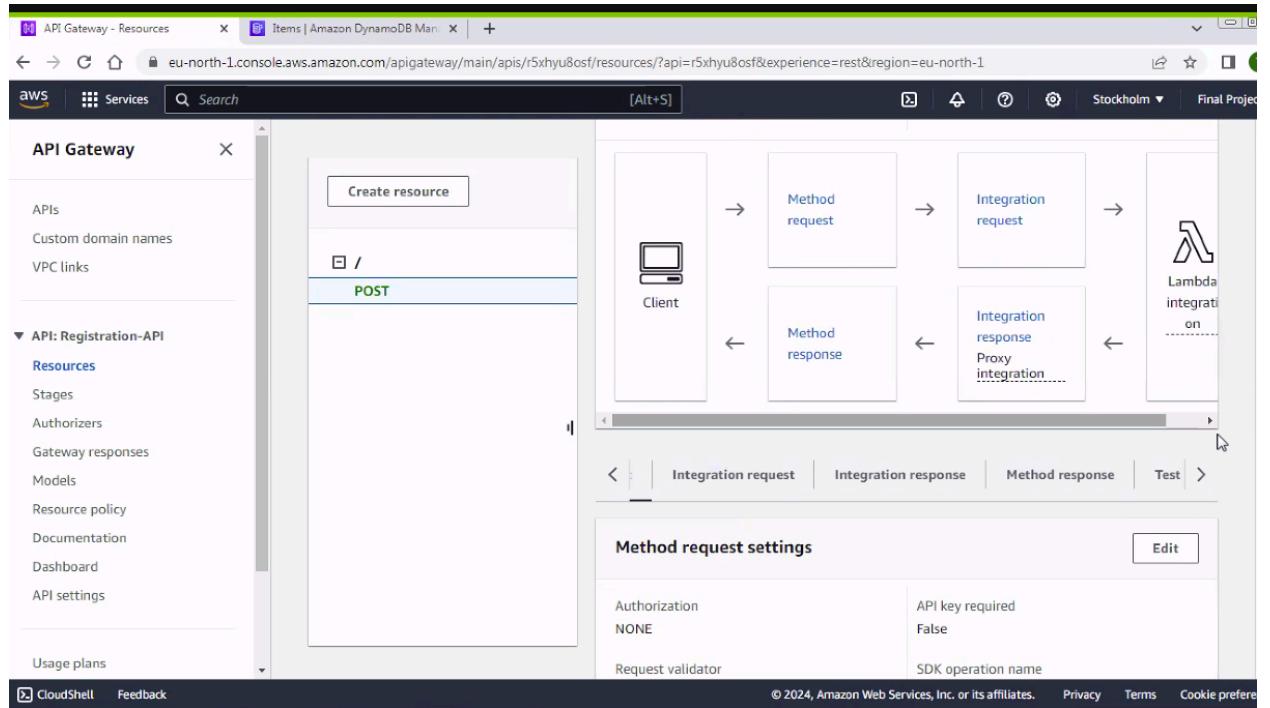


12. Go to the bottom and click 'create method'

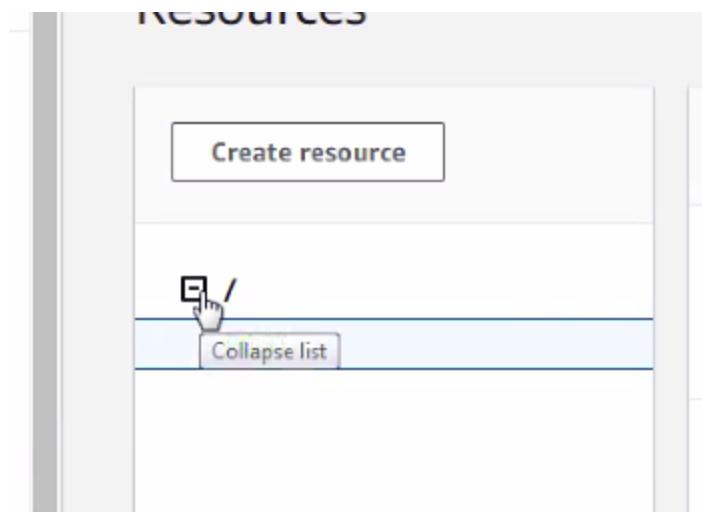


13. Success

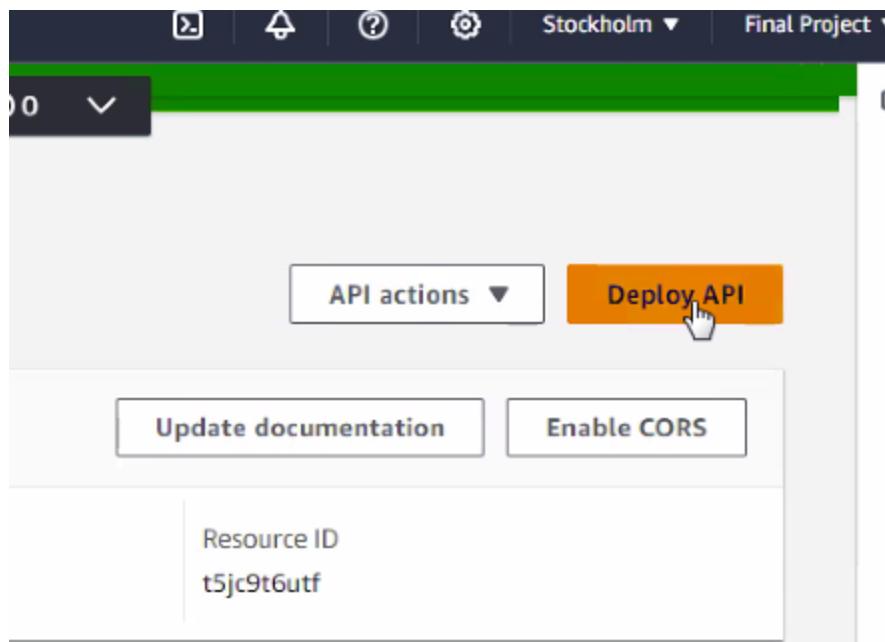
14. Go back to API gateway page on console



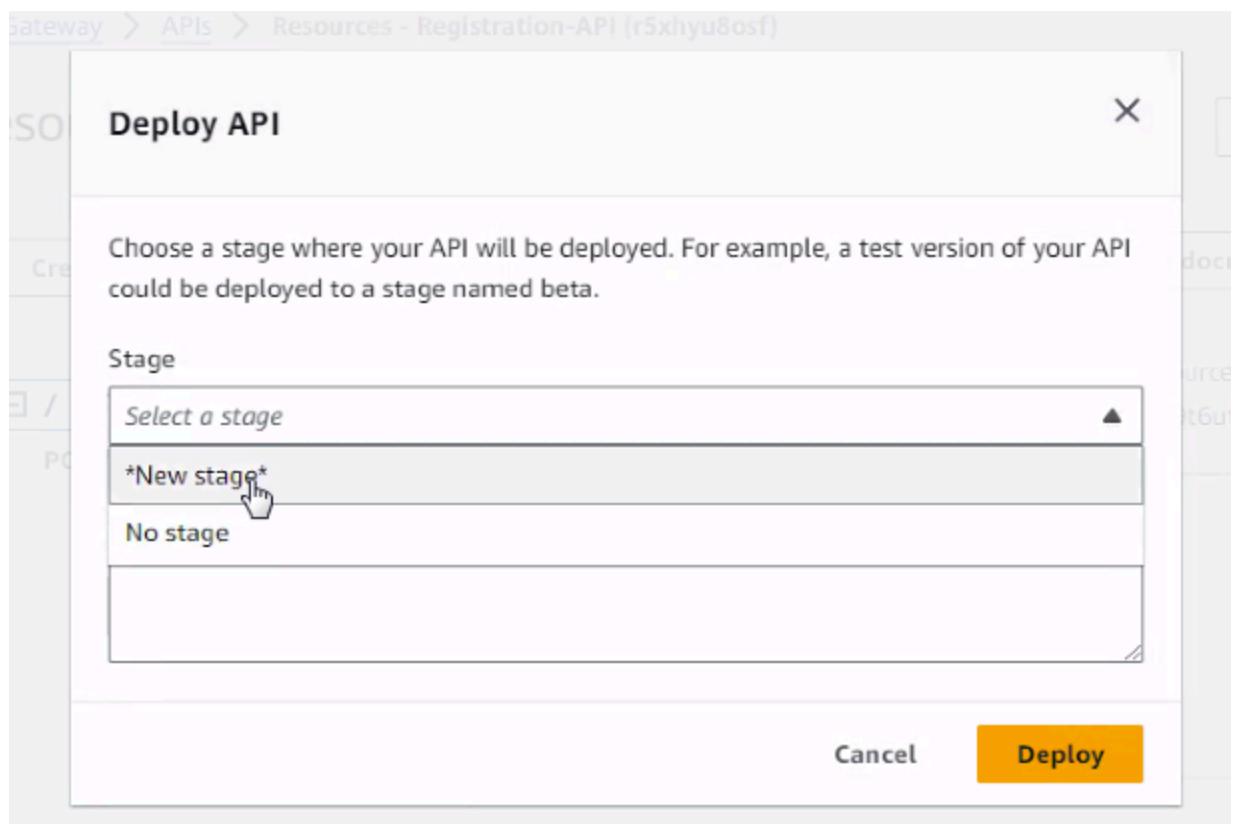
15. Below the 'create resource' button, click on the “-” sign to collapse the list



16. Click 'Deploy API' on the right side



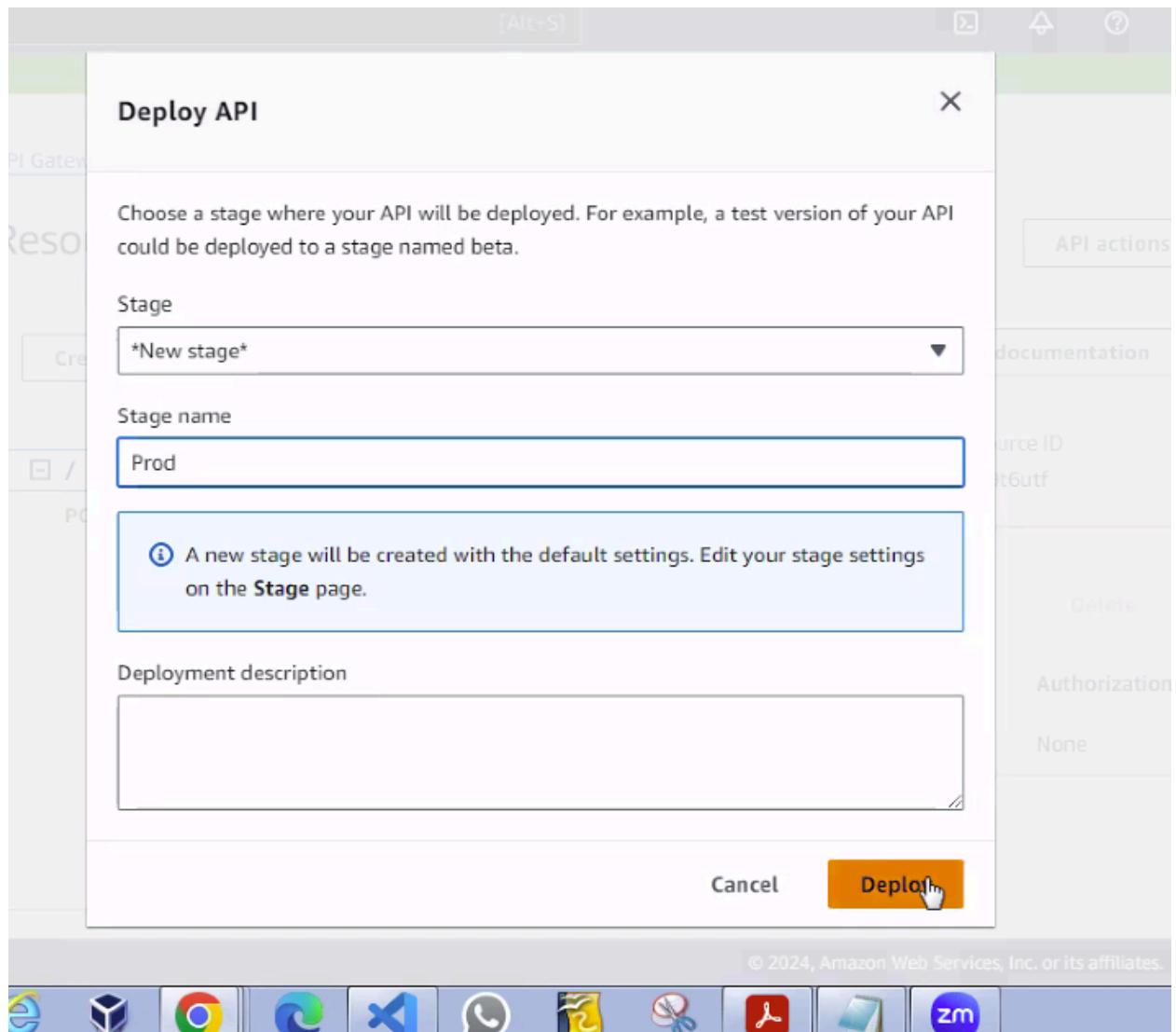
17. In the 'Stage' box, select 'New stage' in the drop down menu



18. Under 'Stage name' type in "Prod"

Stage name

19. Then click on the 'Deploy' button

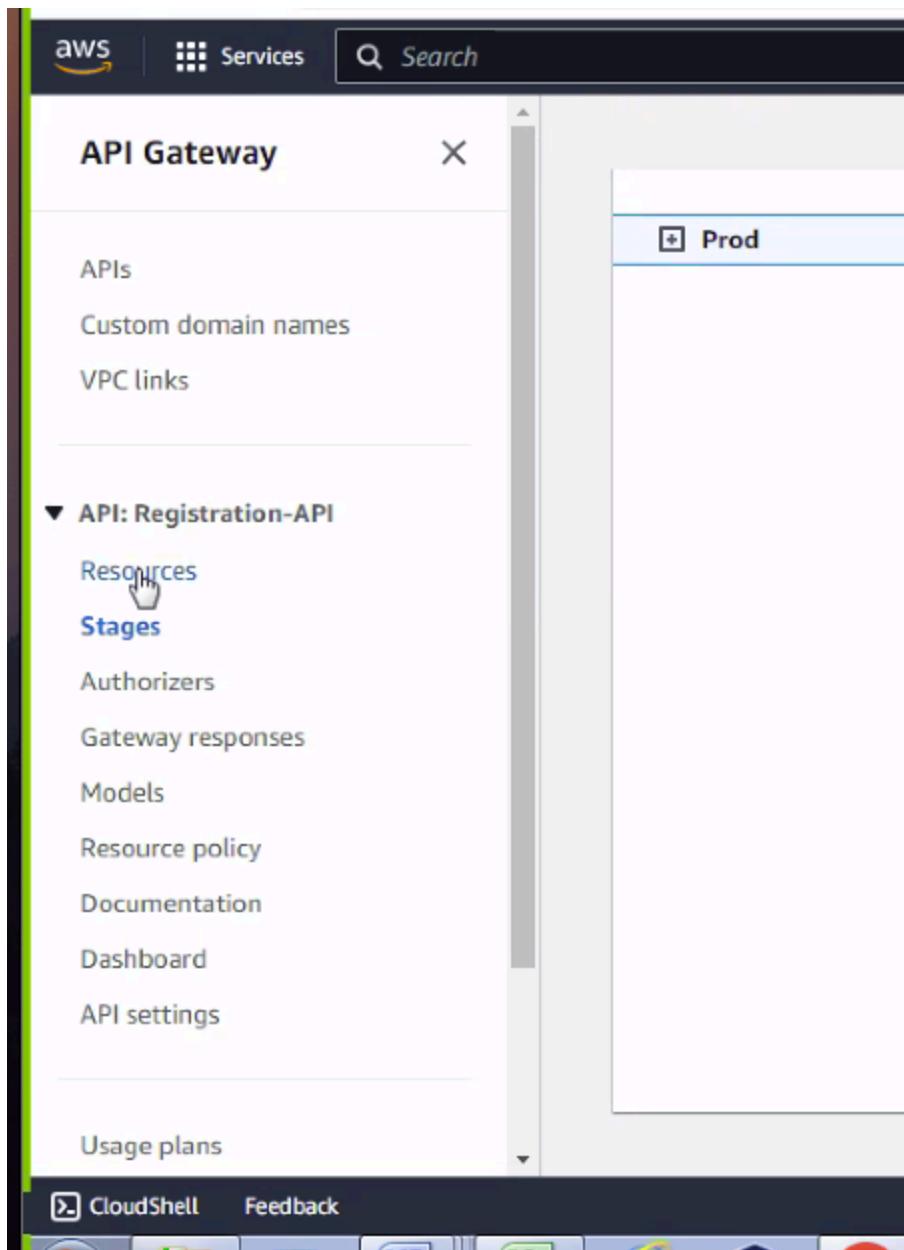


20. Success

The screenshot shows the AWS API Gateway interface. At the top, there is a green success message: "Successfully created deployment for Registration-API. This deployment is active for Prod." Below the message, the navigation path is: API Gateway > APIs > Registration-API (r5xhyu8osf) > Stages. On the right side, there are buttons for "Stage actions" and "Create stage". The main area displays the "Stages" section, which lists a single stage named "Prod". To the right of the stage name, under "Stage details", is an "Edit" button. The "Stage details" table contains the following information:

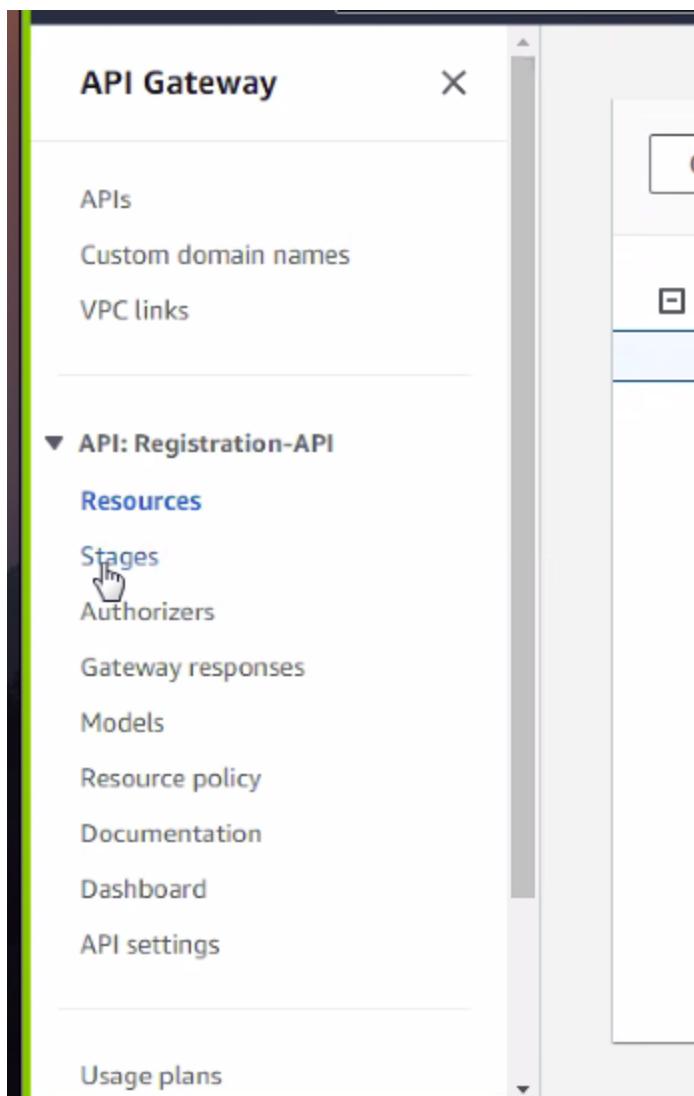
Stage details	
Stage name	Prod
Web ACL	-
Burst Info	-
Default method-level caching	⊖ Inactive
Invoke URI	-
Rate Info	-
Cache cluster Info	-
Client certificate	-

21. Go to the Resources section of the left hand side



22. Nevermind

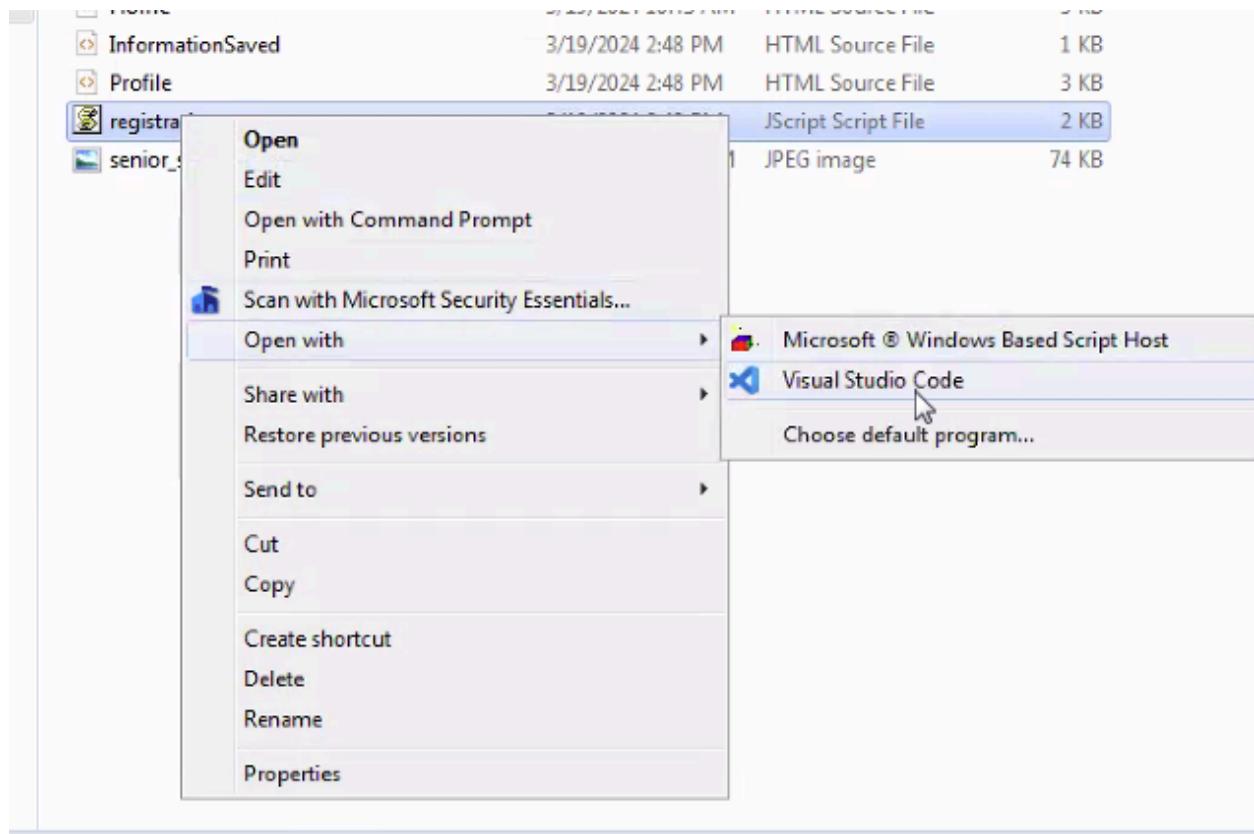
23. Click on 'Stages' link on the left hand side



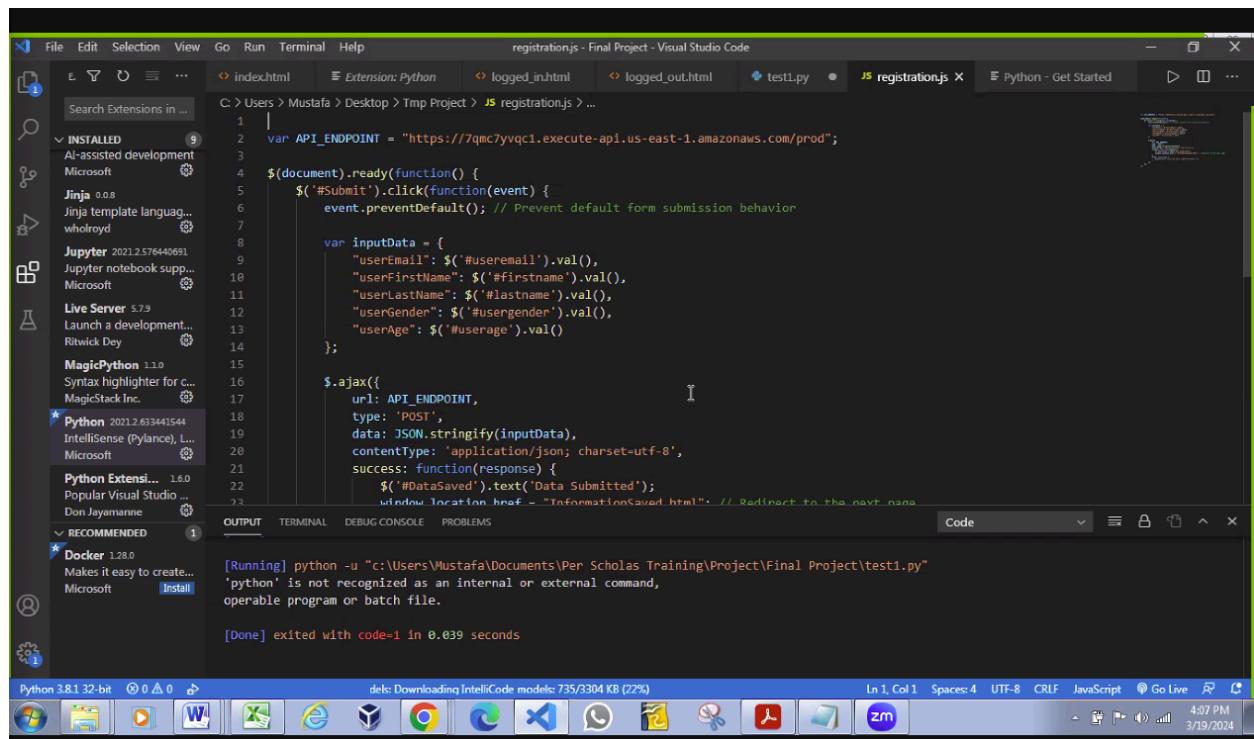
24. In the 'Stage details' click to copy the URL

The screenshot shows the AWS API Gateway Stages page. On the left, there's a sidebar with options like APIs, Custom domain names, VPC links, and a list for the API: Registration-API, which includes Stages, Authorizers, Gateway responses, Models, Resource policy, Documentation, Dashboard, and API settings. The Stages section is currently selected. On the right, under the 'Stages' heading, there's a table with one row for 'Prod'. The 'Stage details' section for 'Prod' shows the Stage name as 'Prod', Web ACL as 'None', and a 'Copied' message next to the URL link. The URL is <https://r5xhyu8osf.execute-api.eu-north-1.amazonaws.com/Prod>. Below the URL, it says 'Active deployment grid01 on March 19, 2024, 16:03 (UTC-04:00)'. At the bottom of the page, there's a tooltip with the text 'Copied' and a link to the same URL.

25. Then go to your desktop on your computer and open 'registration.js' document with Visual Studio Code



26. Visua Studio Code:



27. Do you see line 2? Replace that default address with your own URL link you copied earlier (e.g. <https://r5xhyu8osf....amazobnaws.com/Prod>)
 28. Replace the old default code in line 2 with your address:

Go Run Terminal Help registrationjs - Final Project - Visual Studio Code

index.html Extension: Python logged_in.html logged_out.html test1.py JS registration.js

C: > Users > Mustafa > Desktop > Tmp Project > JS registration.js > API_ENDPOINT

```
1 var API_ENDPOINT = "https://qmc7yvqc1.execute-api.us-east-1.amazonaws.com/prod";
2
3 $(document).ready(function() {
4     $('#Submit').click(function(event) {
5         event.preventDefault(); // Prevent default form submission behavior
6
7         var inputData = {
8             "userEmail": $('#useremail').val(),
9             "userFirstName": $('#firstname').val(),
10            "userLastName": $('#lastname').val(),
11            "userGender": $('#usergender').val(),
12            "userAge": $('#userage').val()
13        };
14
15        $.ajax({
16            url: API_ENDPOINT,
17            type: 'POST',
18            data: JSON.stringify(inputData),
19            contentType: 'application/json; charset=utf-8',
20            success: function(response) {
21                $('#DataSaved').text('Data Submitted');
22                window.location.href = "InformationSaved.html"; // Redirect to the next page
23            }
24        });
25    });
26
27 });
28
29 
```

- ### 29. See the change?

```
File Edit Selection View Go Run Terminal Help registrationjs - Final Project - Visual Studio Code

index.html Extension: Python logged_in.html logged_out.html test1.py JS registration.js X Python - G

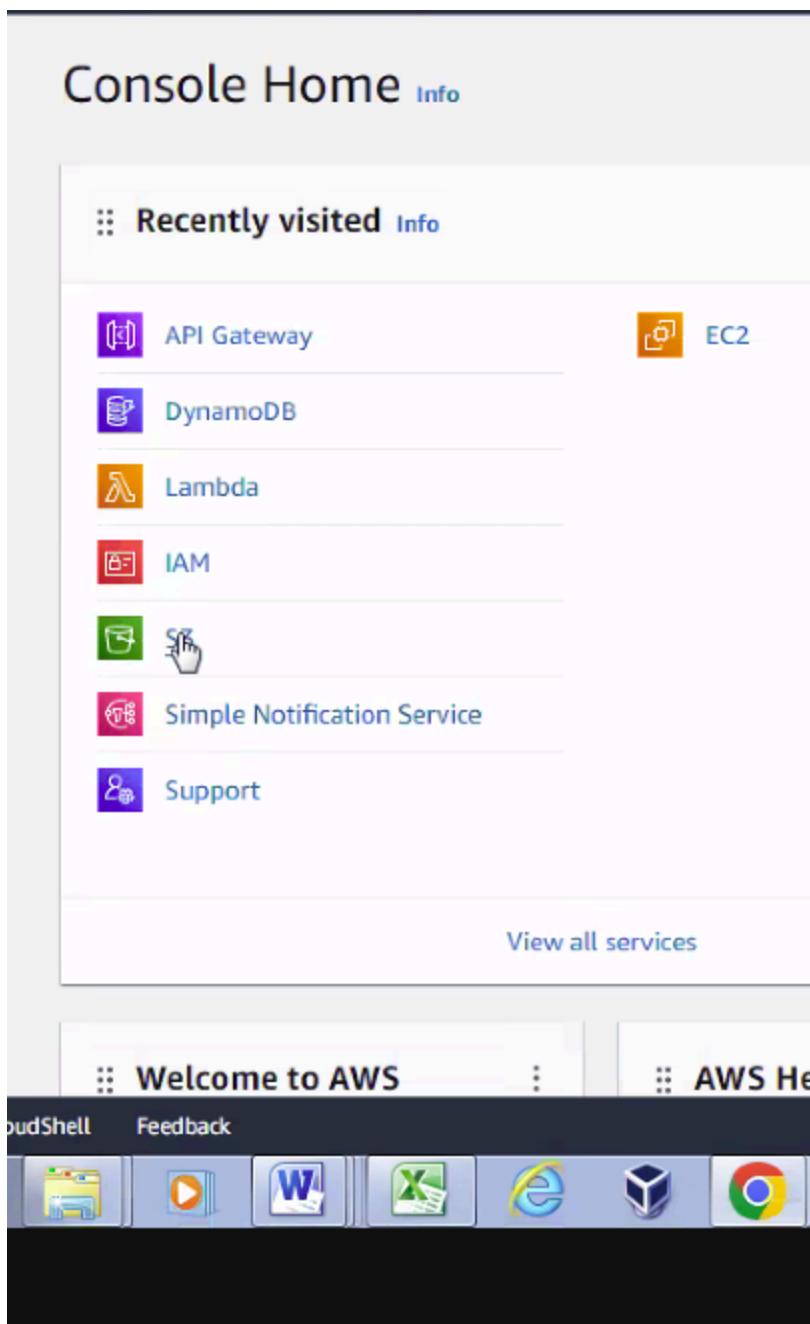
Search Extensions in ...

INSTALLED 9
AI-assisted development
Microsoft
Jinja 0.0.8
Jinja template languag...
wholroyd
Jupyter 2021.2.576440691
Jupyter notebook supp...
Microsoft
Live Server 5.7.9
Launch a development...
Ritwick Dey

C: > Users > Mustafa > Desktop > Tmp Project > JS registration.js > API_ENDPOINT

1 var API_ENDPOINT = "https://r5xhyu8osf.execute-api.eu-north-1.amazonaws.com/Prod";
2
3
4 $(document).ready(function() {
5     $('#Submit').click(function(event) {
6         event.preventDefault(); // Prevent default form submission behavior
7
8             var inputData = {
9                 "userEmail": $('#useremail').val(),
10                "userFirstName": $('#firstname').val(),
11                "userLastName": $('#lastname').val(),
12                "userGender": $('#usergender').val(),
13                "userAge": $('#userage').val()
14            };
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
59
```

30. Now travel back to your S3 bucket.



31. Go click on your bucket

The screenshot shows the AWS S3 console interface. The URL in the browser is s3.console.aws.amazon.com/s3/buckets/s3finalproject?region=eu-north-1&bucketType=general&tab=objects. The bucket name 's3finalproject' is visible in the path. The 'Objects' tab is selected. At the top of the objects list, there is a toolbar with several buttons: Copy S3 URI, Copy URL, Download, Open, Delete, Actions (with a dropdown arrow), Create folder, and a prominent orange 'Upload' button. A yellow box highlights the 'Upload' button. Below the toolbar, a table lists three objects: 'Home.html' (html type, 2.5 KB, Standard storage class), 'registration.js' (js type, 1.0 KB, Standard storage class), and 'senior_support_image1.jpg' (jpg type, 73.0 KB, Standard storage class). The table has columns for Name, Type, Last modified, Size, and Storage class.

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	Home.html	html	March 19, 2024, 10:45:37 (UTC-04:00)	2.5 KB	Standard
<input type="checkbox"/>	registration.js	js	March 19, 2024, 14:51:18 (UTC-04:00)	1.0 KB	Standard
<input type="checkbox"/>	senior_support_image1.jpg	jpg	March 19, 2024, 10:45:38 (UTC-04:00)	73.0 KB	Standard

32. On the right side, click on that orange button that says 'Upload'

This screenshot is identical to the previous one, showing the AWS S3 console with the 's3finalproject' bucket. The 'Upload' button in the toolbar is again highlighted with a yellow box. The table below the toolbar shows the same three objects: 'Home.html', 'registration.js', and 'senior_support_image1.jpg'. The interface includes standard AWS navigation elements like CloudShell, Feedback, and various icons at the bottom.

33. In 'Upload', click on the 'add files' button over to the right:

The screenshot shows the 'Upload' page in the AWS S3 console. The URL in the top navigation bar is 'Amazon S3 > Buckets > s3finalproject > Upload'. The main area has a large dashed box with the instruction 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.' Below this is a table titled 'Files and folders (0)' with one row: 'All files and folders in this table will be uploaded.' At the top right of the table are three buttons: 'Remove', 'Add files' (which has a hand cursor icon), and 'Add folder'. A search bar with placeholder text 'Find by name' is above the table. The table has columns for Name, Folder, and Type. A message 'No files or folders' is displayed below the table, followed by the note 'You have not chosen any files or folders to upload.'

34. Select the 'registration.js' file from your computer files

The screenshot shows a file explorer window with a sidebar on the left. The main area displays a table of files with columns: Name, Date modified, Type, and Size. The files listed are: 'Home' (3/19/2024 10:43 AM, HTML Source File, 3 KB), 'InformationSaved' (3/19/2024 2:48 PM, HTML Source File, 1 KB), 'Profile' (3/19/2024 2:48 PM, HTML Source File, 3 KB), 'registration' (3/19/2024 4:07 PM, JScript Script File, 2 KB), and 'senior_support_image1' (3/19/2024 10:42 AM, JPEG image, 74 KB). The 'registration' file is highlighted with a blue selection bar and has a small hand cursor icon pointing at it.

35. Add it

The screenshot shows the AWS S3 'Upload' interface. At the top, there's a navigation bar with 'Services' and a search bar. Below that, the path 'Amazon S3 > Buckets > s3finalproject > Upload' is displayed. The main area is titled 'Upload' with an 'Info' link. A large dashed box contains the instruction 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.' Below this, a table lists 'Files and folders (1 Total, 1.0 KB)'. The table has columns for 'Name', 'Folder', and 'Type'. One item, 'registration.js', is listed with a type of 'text/javascript'. There are 'Remove', 'Add files', and 'Add folder' buttons at the top of the table. A search bar labeled 'Find by name' is also present.

36. Scroll down and click 'Upload'

This screenshot shows the same AWS S3 'Upload' interface as the previous one, but with a focus on the bottom right. The 'Upload' button is highlighted with a white arrow pointing to it from the left. The 'Cancel' button is also visible. At the very bottom of the screen, there's a footer bar with icons for various services like CloudWatch, Lambda, and S3, along with the text '© 2024, Amazon Web Services, Inc. or its affiliates.' and 'Privacy'.

37. Success

Destination	Succeeded	Failed
s3://s3finalproject	1 file, 1.0 KB (100.00%)	0 files, 0 B (0%)

Name	Folder	Type	Size	Status	Error
registration.js	-	text/javascript	1.0 KB	Succeeded	-

38. In your console search bar on the top, type in 'S3'

39. In S3, click on your bucket link (e.g. s3finalproject)

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with various settings and points-related options. The main area is titled 'Amazon S3' and shows an 'Account snapshot' with a link to 'View Storage Lens dashboard'. Below this, there are tabs for 'General purpose buckets' and 'Directory buckets', with 'General purpose buckets' selected. It displays a table with one row for the bucket 's3finalproject'. The table columns include Name, AWS Region, Access, and Creation date. The bucket details are: Name - s3finalproject, AWS Region - Europe (Stockholm) eu-north-1, Access - Public, Creation date - March 19, 2024, 10:40:27 (UTC-04:00). At the bottom of the table, there are buttons for Copy ARN, Empty, Delete, and Create bucket. The status bar at the bottom right shows the date as March 19, 2024, and the time as 10:40:27 (UTC-04:00).

40. While in that bucket, click on the 'Properties' tab

This screenshot shows the properties page for the 's3finalproject' bucket. The title bar says 's3finalproject [Alt+S]'. Below it, the bucket name 's3finalproject' is displayed with a 'Publicly accessible' button. The navigation tabs are 'Objects', 'Properties' (which is highlighted), 'Permissions', 'Metrics', and 'Manager'. Under the 'Objects' tab, it shows 'Objects (3) Info'. There are four buttons: 'Copy S3 URI', 'Copy URL', 'Download', and 'Upload'. Below these buttons are 'Create folder' and another 'Upload' button.

41. Scroll down

42. In the 'Static website hosting', click on the URL link to copy it

The screenshot shows the 'Static website hosting' section of an AWS S3 bucket configuration. At the top, there is a note about requester pays and a 'Disabled' status. Below this, the 'Static website hosting' section is set to 'Enabled'. It specifies 'Hosting type' as 'Bucket hosting' and 'Bucket website endpoint' as 'http://s3finalproject.s3-website.eu-north-1.amazonaws.com'. A tooltip indicates that when you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. The URL 'http://s3finalproject.s3-website.eu-north-1.amazonaws.com' is underlined and has a small cursor icon pointing to it.

43. You should be taken to a webpage that looks like this:

The screenshot shows a web browser window displaying a static website. The title bar says 'A Helping Hand: Community Volunteers for Senior Support'. The main content area features a dark blue header with the text 'A Helping Hand: Community Volunteers for Senior Support'. Below this is a white section with the heading 'Welcome to our Community!'. It includes a paragraph about providing assistance and support to seniors through dedicated volunteer efforts, mentioning services like companionship, assistance with daily tasks, transportation, and more. Below the text are six small illustrations showing various volunteer interactions with seniors, such as walking, eating, pushing a wheelchair, grocery shopping, and medical assistance. The browser's address bar shows the URL 'http://s3finalproject.s3-website.eu-north-1.amazonaws.com'. The taskbar at the bottom of the screen shows various application icons.

44. Scroll down and click on the link 'Register by Submitting the Form'



How to Get Involved

If you're interested in becoming a volunteer or if you're a senior in the information below:

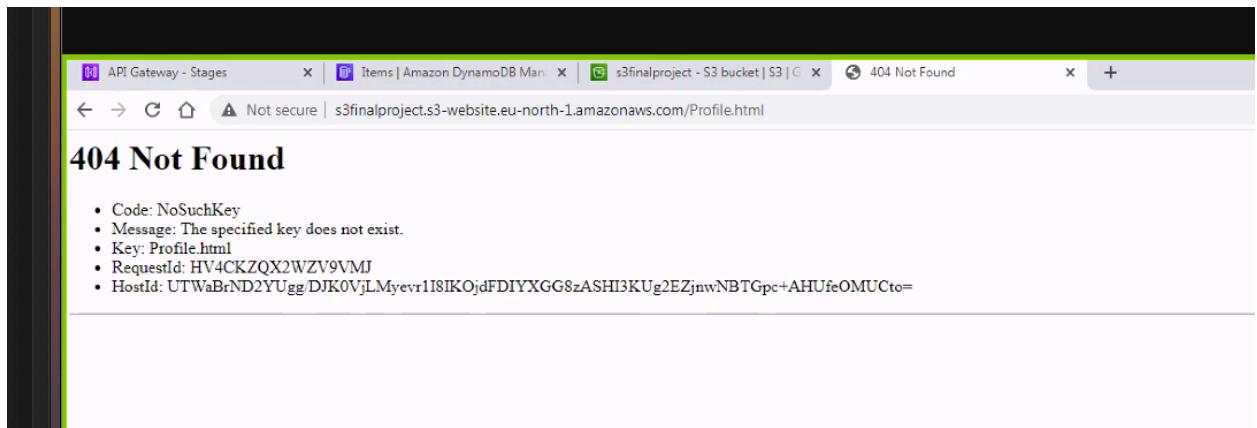
[Register by Submitting the Form](#)

OR

Contact Email: volunteer@helpinghand.org

Phone: (555) 123-4567

45. [404 error! Mistake is: profile.html document was not added to the S3 bucket!]

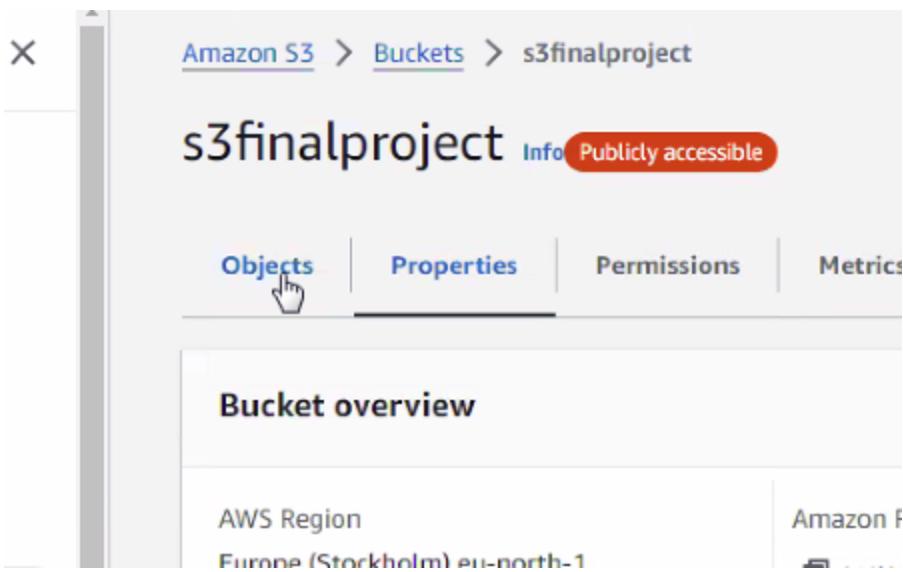


Not secure | s3finalproject.s3-website.eu-north-1.amazonaws.com/Profile.html

404 Not Found

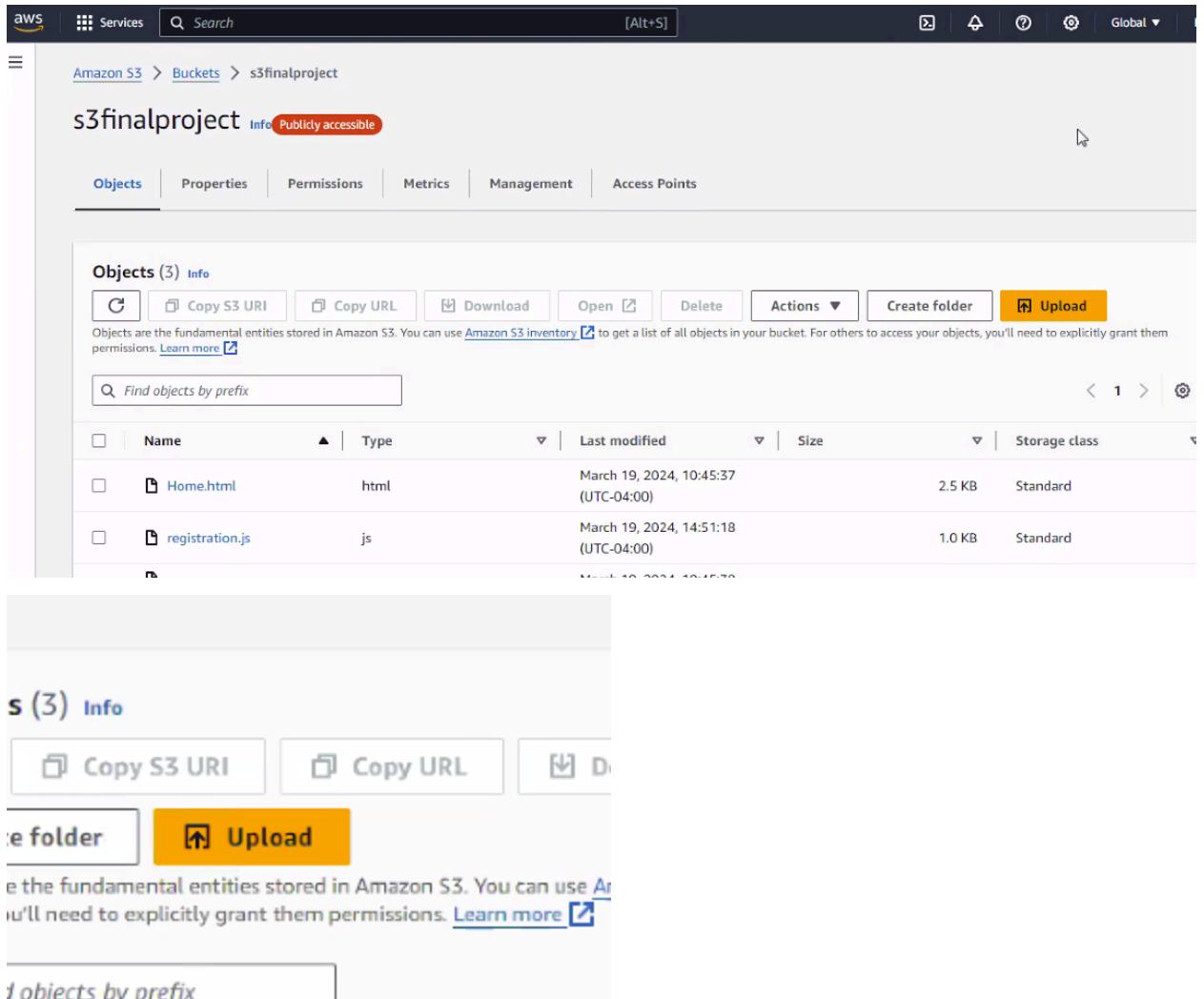
- Code: NoSuchKey
- Message: The specified key does not exist.
- Key: Profile.html
- RequestId: HV4CKZQX2WZV9VMJ
- HostId: UTWaBrND2YUggDJK0VjLMyevr1I8IKOjdFDIYXGG8zASHI3KUg2EZJnwNBTGpc+AHUfeOMUCto=

46. Go back to S3 bucket



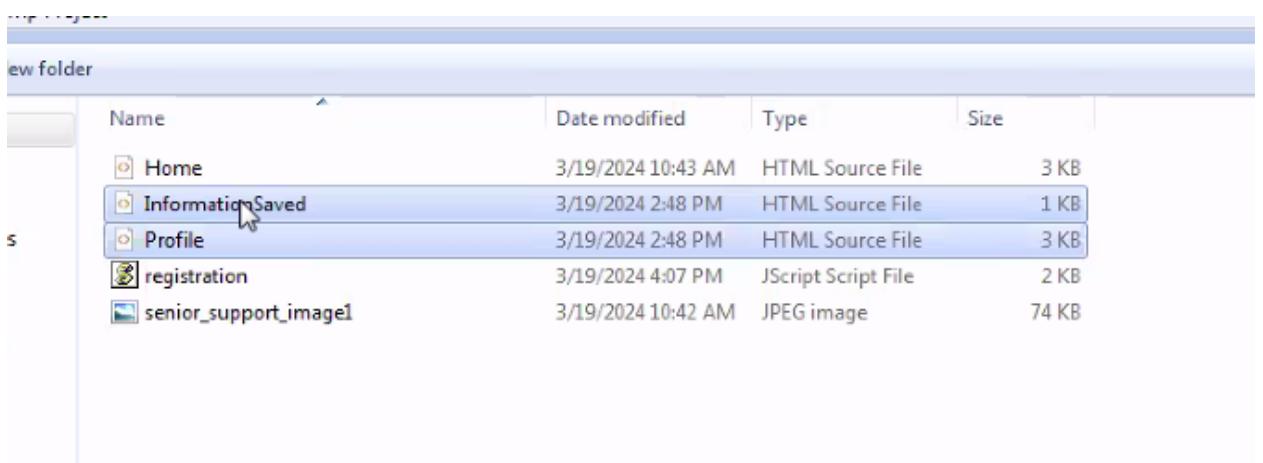
47. Click on 'Objects tab'

48. Click on 'Upload' orange button on the right side



The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with the AWS logo, 'Services', a search bar, and various global settings. Below the navigation bar, the path 'Amazon S3 > Buckets > s3finalproject' is displayed. The main area shows the 's3finalproject' bucket with the status 'Publicly accessible'. A red box highlights the 'Actions' dropdown menu, specifically the 'Upload' option, which is highlighted in orange. Below this, there's a table of objects: 'Home.html' (html, 2.5 KB, Standard) and 'registration.js' (js, 1.0 KB, Standard). At the bottom of the object list, there's a note about using Amazon S3 inventory and a link to learn more.

49. Add Profile.html and InformationSaved.html



The screenshot shows the contents of a folder named 's'. The table lists the following files:

Name	Date modified	Type	Size
Home	3/19/2024 10:43 AM	HTML Source File	3 KB
InformationSaved	3/19/2024 2:48 PM	HTML Source File	1 KB
Profile	3/19/2024 2:48 PM	HTML Source File	3 KB
registration	3/19/2024 4:07 PM	JScript Script File	2 KB
senior_support_image1	3/19/2024 10:42 AM	JPEG image	74 KB

A red box highlights the 'InformationSaved' file, indicating it is selected.

50. It should look like this:

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (2 Total, 3.4 KB)

All files and folders in this table will be uploaded.

<input type="checkbox"/>	Name	Folder	Type
<input type="checkbox"/>	InformationSaved.html	-	text/html
<input type="checkbox"/>	Profile.html	-	text/html

51. Scroll down to click 'Upload' button

52. In your S3 bucket, click on 'properties' tab

Amazon S3 > Buckets > s3finalproject

s3finalproject

[Info](#) Publicly accessible

Objects **Properties** **Permissions** **Metrics** **Management**

Objects (5) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 invento](#) permissions. [Learn more](#)

Find objects by prefix

<input type="checkbox"/>	Name	Type
--------------------------	------	------

53. Scroll down and go click on that very same URL we visited earlier

The screenshot shows the AWS S3 console with the bucket 's3finalproject' selected. In the 'Requester pays' section, 'Requester pays' is set to 'Disabled'. In the 'Static website hosting' section, 'Static website hosting' is set to 'Enabled', 'Hosting type' is 'Bucket hosting', and the 'Bucket website endpoint' is listed as <http://s3finalproject.s3-website.eu-north-1.amazonaws.com>.

54. Back on the website, scroll down to click on that same link 'Register by Submitting the Form'

How to Get Involved

If you're interested in becoming a volunteer or if you're a senior in need of the information below:

[Register by Submitting the Form](#)

OR

Contact Email: volunteer@helpinghand.org

Phone: (555) 123-4567

55. Success

56. Now enter any details in the registration form

Volunteer Registration Form

User email:

First name:

 mustafa.harrati@gmail.com
Mustafa Harrati

Last name:

 Manage...

Gender:

Volunteer Registration Form

User email:

 MyTest@gmail.com

First name:

 Mustafa

Last name:

 Test

Gender:

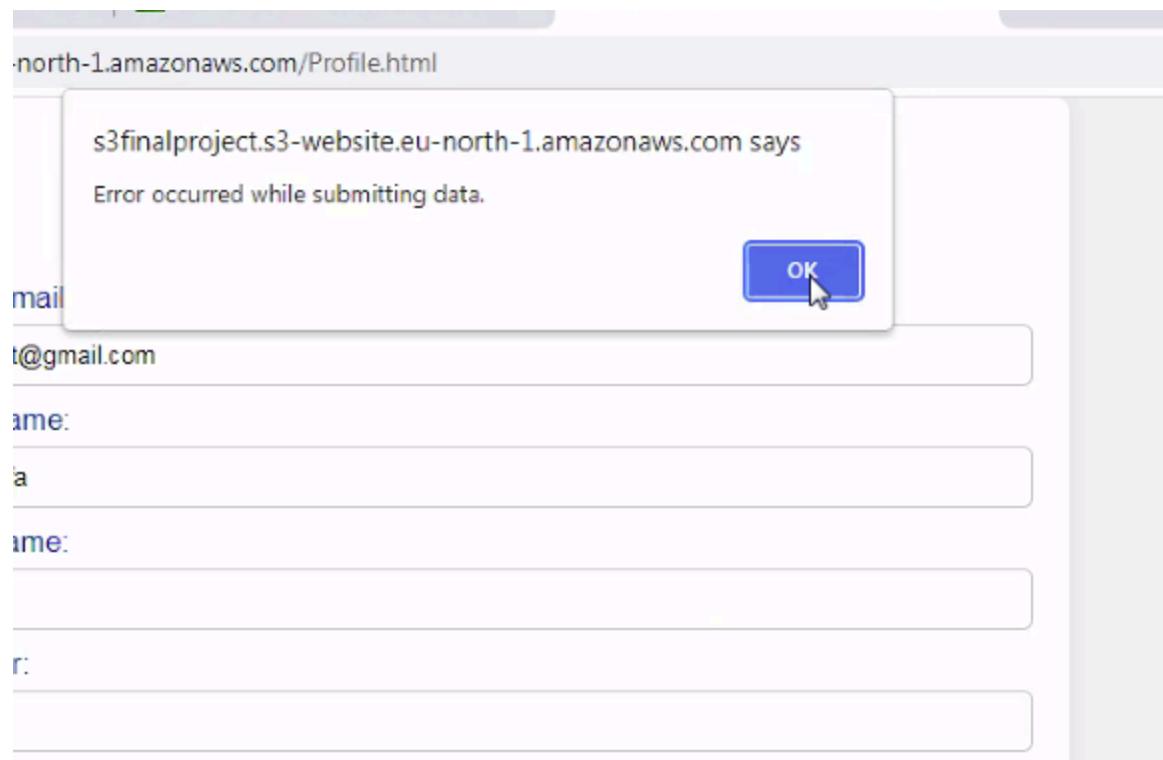
 Male

User Age:

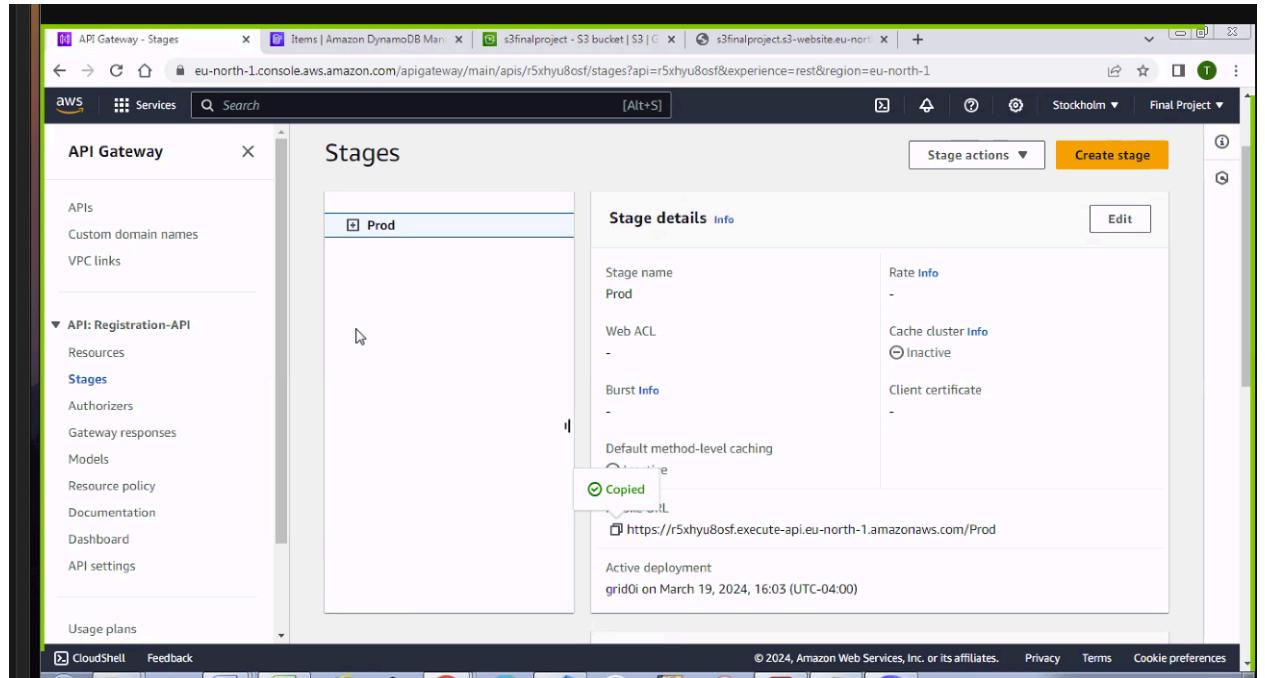
 123

Save Profile

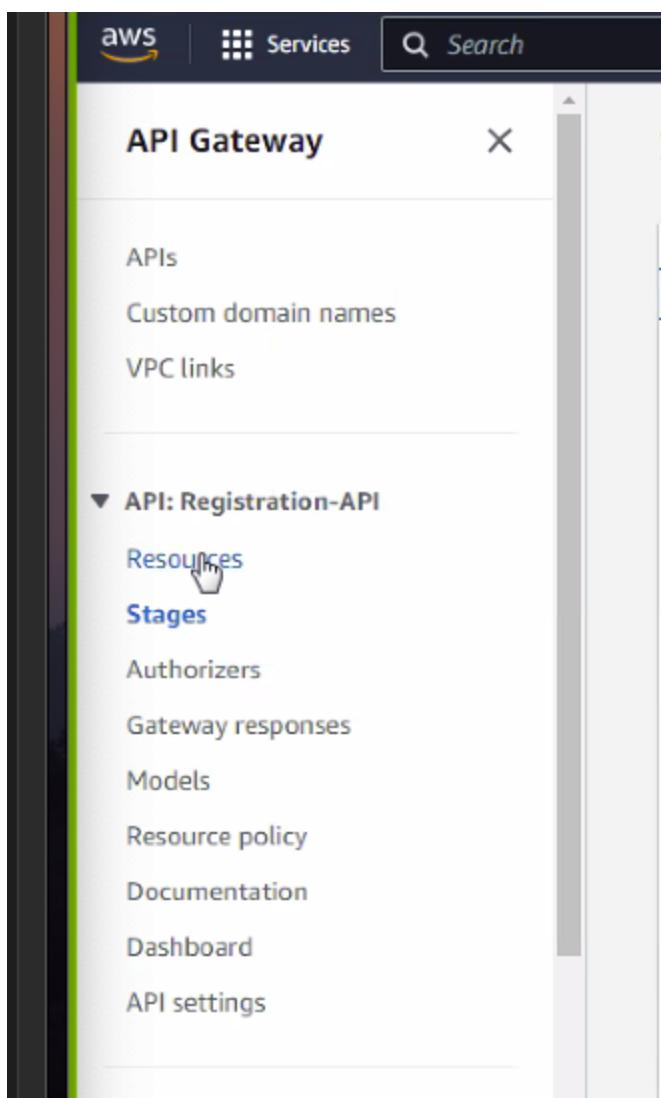
57. [ERROR!]



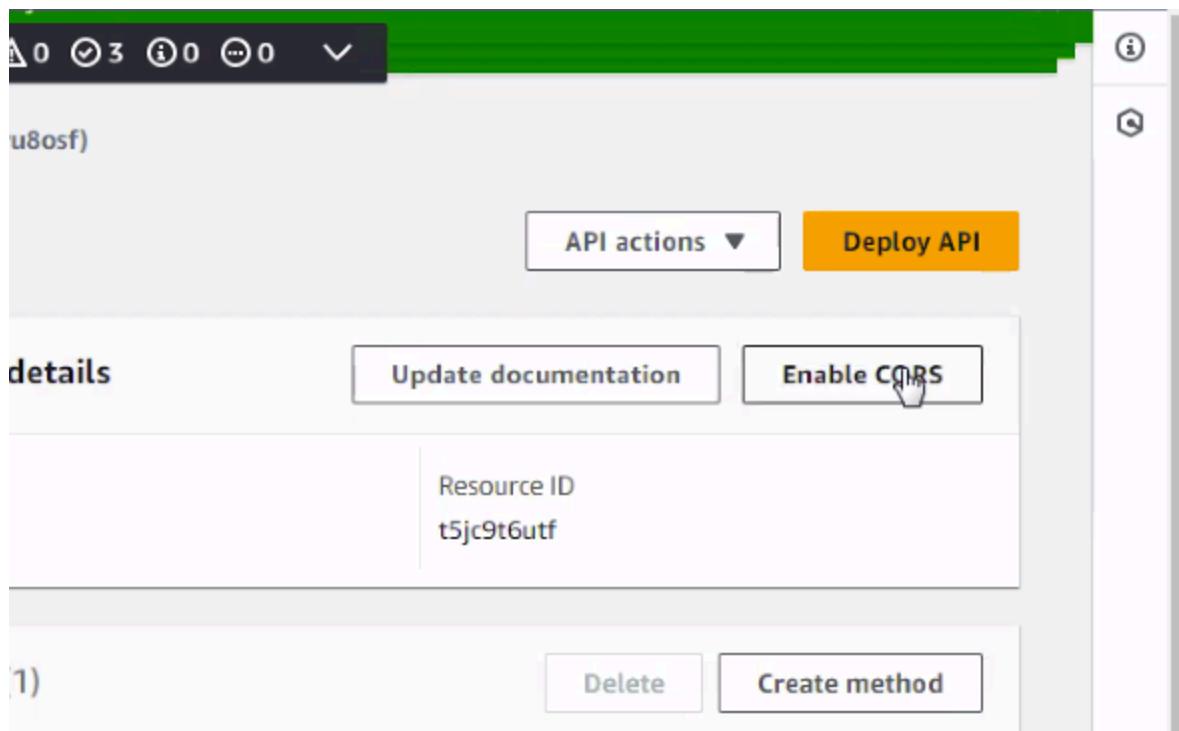
58. Go back to the API Gateway



59. On the left side, click on 'Resources'



60. Enable CORS by clicking on the button



61. In the 'Enable CORS' section, select both "Default 4XX" and "Default 5XX"

The screenshot shows the AWS API Gateway 'Enable CORS' configuration page. At the top, there is a green success message: 'Successfully created deployment for Registration-API. This deployment is active for Prod.' Below the message, the breadcrumb navigation shows: API Gateway > APIs > Resources - Registration-API (r5xhyu8osf) > Enable CORS.

Enable CORS

CORS settings Info

To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API.

Gateway responses
API Gateway will configure CORS for the selected gateway responses.

Default 4XX
 Default 5XX

Access-Control-Allow-Methods

OPTIONS
 POST

Access-Control-Allow-Headers
API Gateway will configure CORS for the selected gateway responses.

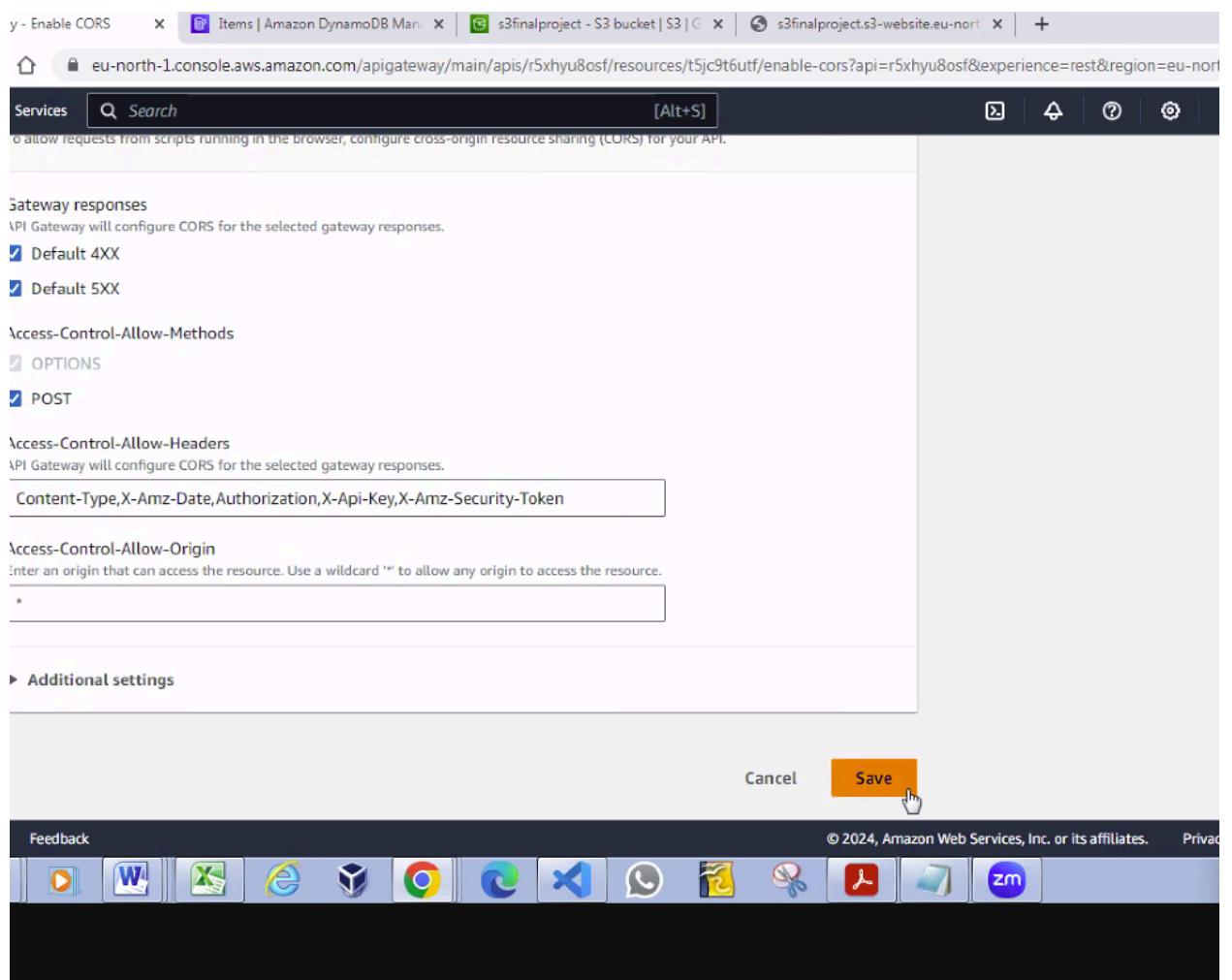
Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token

CloudShell Feedback © 2024, Amazon Web Services, Inc. and its affiliates. All rights reserved.

62. Select 'POST' option too

The screenshot shows the AWS API Gateway CORS settings page. At the top, there are navigation icons (back, forward, search) and a URL bar showing the path to the CORS settings. Below the header, the title 'CORS settings' is displayed with an 'Info' link. A note states: 'To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API.' Under the 'Gateway responses' section, it says 'API Gateway will configure CORS for the selected gateway responses.' with two checked options: 'Default 4XX' and 'Default 5XX'. In the 'Access-Control-Allow-Methods' section, 'OPTIONS' is checked, and 'POST' is also checked and highlighted with a cursor icon. The 'Access-Control-Allow-Headers' section contains a list: 'Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token'. The 'Access-Control-Allow-Origin' section has a placeholder 'Enter an origin that can access the resource. Use a wildcard '*' to allow any origin to access the resource.' with a single asterisk (*) entered. At the bottom, there is a '► Additional settings' link.

63. Scroll down and click on "Save"



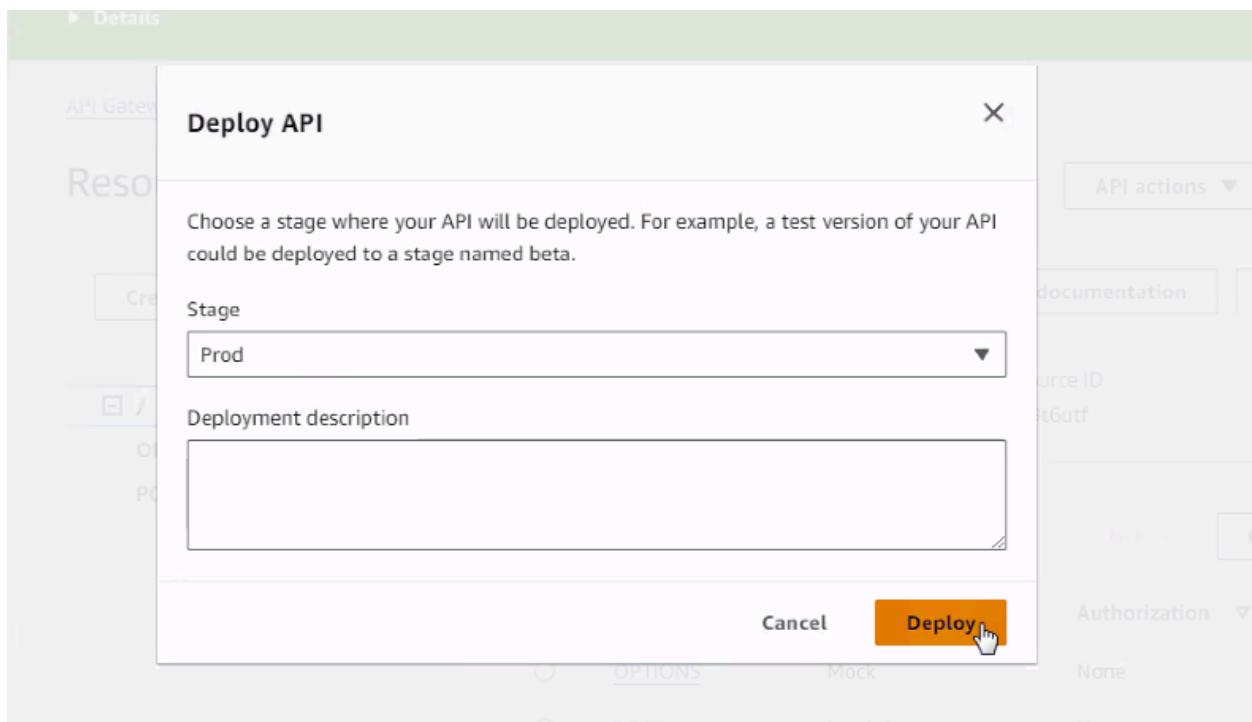
64. Success

The screenshot shows the AWS API Gateway Resources page. At the top, a green banner indicates "Successfully enabled CORS". Below this, the "Resource details" section shows a path of "/" and a resource ID of "t5jc9t6utf". The "Methods" section lists two methods: "OPTIONS" (Mock, None, Not required) and "POST" (Lambda, None, Not required). A large orange "Deploy API" button is prominently displayed at the bottom right of the main content area.

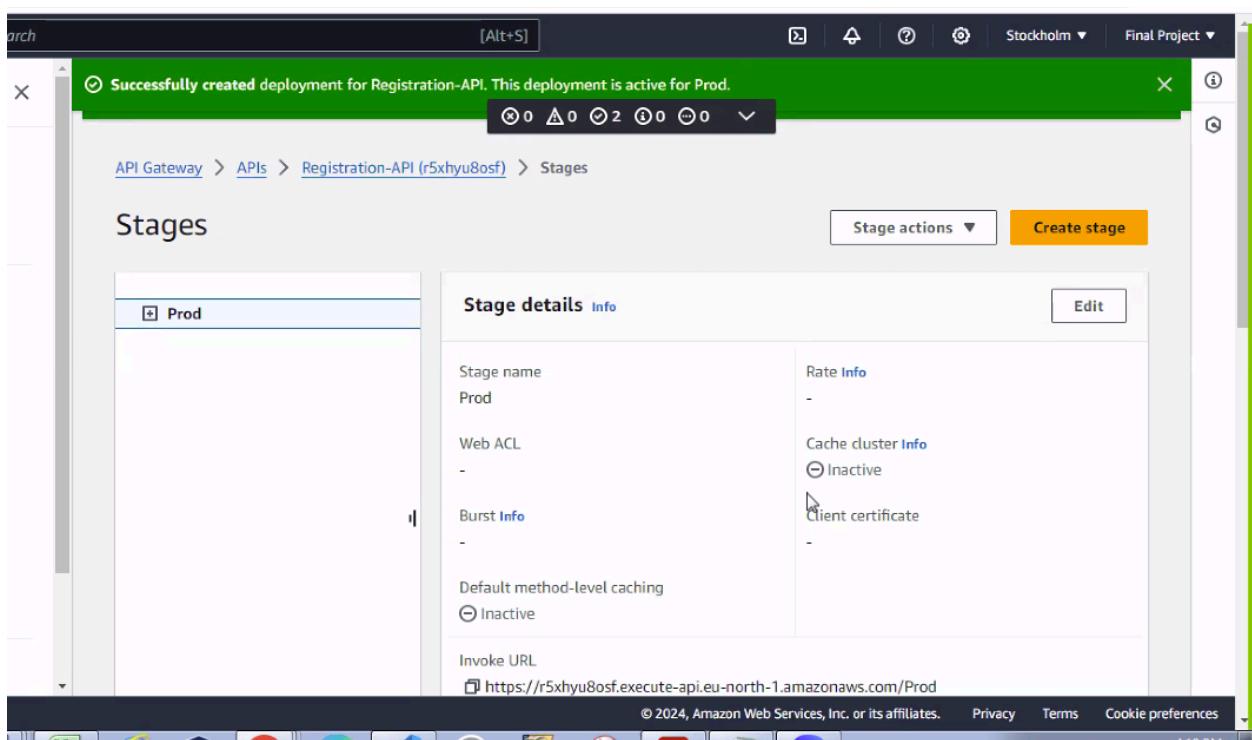
65. Click on 'Deploy API'

66. Select the 'Stage' as the name you gave it earlier "Prod"

67. Click 'Deploy' button in orange at the bottom right corner



68. Success



69. Go back to website and see if you can register...

The screenshot shows a web browser window with three tabs at the top: 'Items | Amazon DynamoDB Man...', 's3finalproject - S3 bucket | S3 | G...', and 's3finalproject.s3-website.eu-nort...'. The active tab is the third one, showing the URL 's3finalproject.s3-website.eu-north-1.amazonaws.com/Profile.html'. The page content is a 'Volunteer Registration Form' with the following fields:

- User email: MyTest@gmail.com
- First name: Mustafa
- Last name: Test
- Gender: Male
- User Age: 1234

A large blue 'Save Profile' button is at the bottom.

70. [Error!]

The screenshot shows a web browser window with three tabs at the top: 'Items | Amazon DynamoDB Man...', 's3finalproject - S3 bucket | S3 | G...', and 's3finalproject.s3-website.eu-nort...'. The active tab is the third one, showing the URL 's3finalproject.s3-website.eu-north-1.amazonaws.com/Profile.html'. A modal dialog box is displayed, containing the text:

s3finalproject.s3-website.eu-north-1.amazonaws.com says
Error occurred while submitting data.

Below the dialog, parts of the registration form are visible:

- User email: MyTest@gmail.com
- First name: Mustafa
- Last name: Test

An 'OK' button is visible in the bottom right corner of the dialog.

71.

