**Text Narrator**

## **Project Overview**

In my project, I have used **Amazon Polly**, a cloud-based text-to-speech service by AWS, to convert written text into lifelike speech.

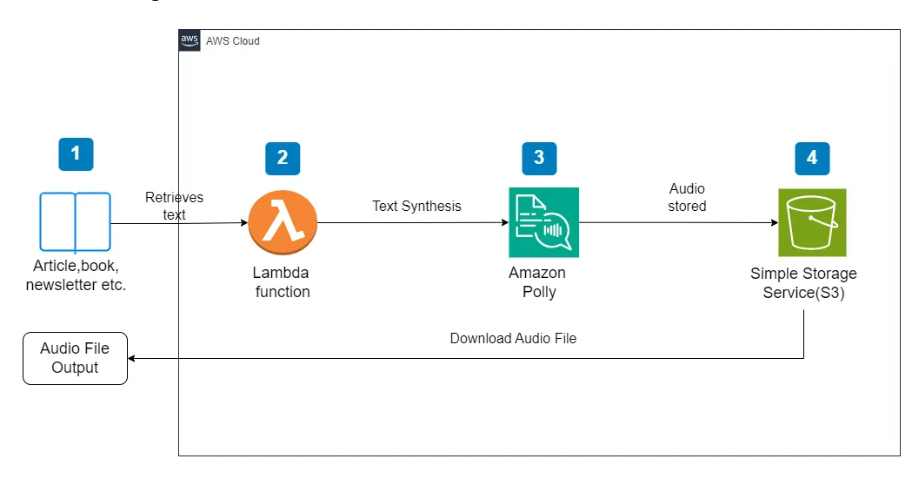
The core component of the project is an **AWS Lambda function**, a serverless compute service that automatically runs our backend logic in response to specific events.

When a user submits text through a web interface or API, the Lambda function is triggered. It processes the input and sends it to **Amazon Polly**, which then synthesizes the text into speech using a selected voice and language.

The output audio is then stored in **Amazon S3**, making it easy to stream or download the narrated file.

This solution is highly scalable, cost-effective, and perfect for applications like e-learning platforms, voice-enabled apps, or automated announcements.

**Architecture Diagram**



## **Tech Stack**

## Backend Language: node.js

## Text-to-Speech API: AWS Polly

## Deployment Platform: AWS lambda

## Access Management: AWS IAM Roles and Policies

## **Step-by-Step Guide**

**Create a new role with appropriate permissions:**

* AmazonPollyFullAccess
* AmazonS3FullAccess
* AWSLambdaBasicExecutionRole

**1. Sign in to AWS Console**

* Go to <https://console.aws.amazon.com/>
* In the search bar, type IAM and select **IAM**.

**2. Create a New Role**

* In the IAM dashboard, from the **left menu**, click on **Roles**.
* Click the **"Create role"** button.

**3. Select Trusted Entity**

* Under **Select trusted entity**, choose **AWS service**.
* Under **Use case**, select **Lambda** (since the role will be used with AWS Lambda).
* Click **Next**.

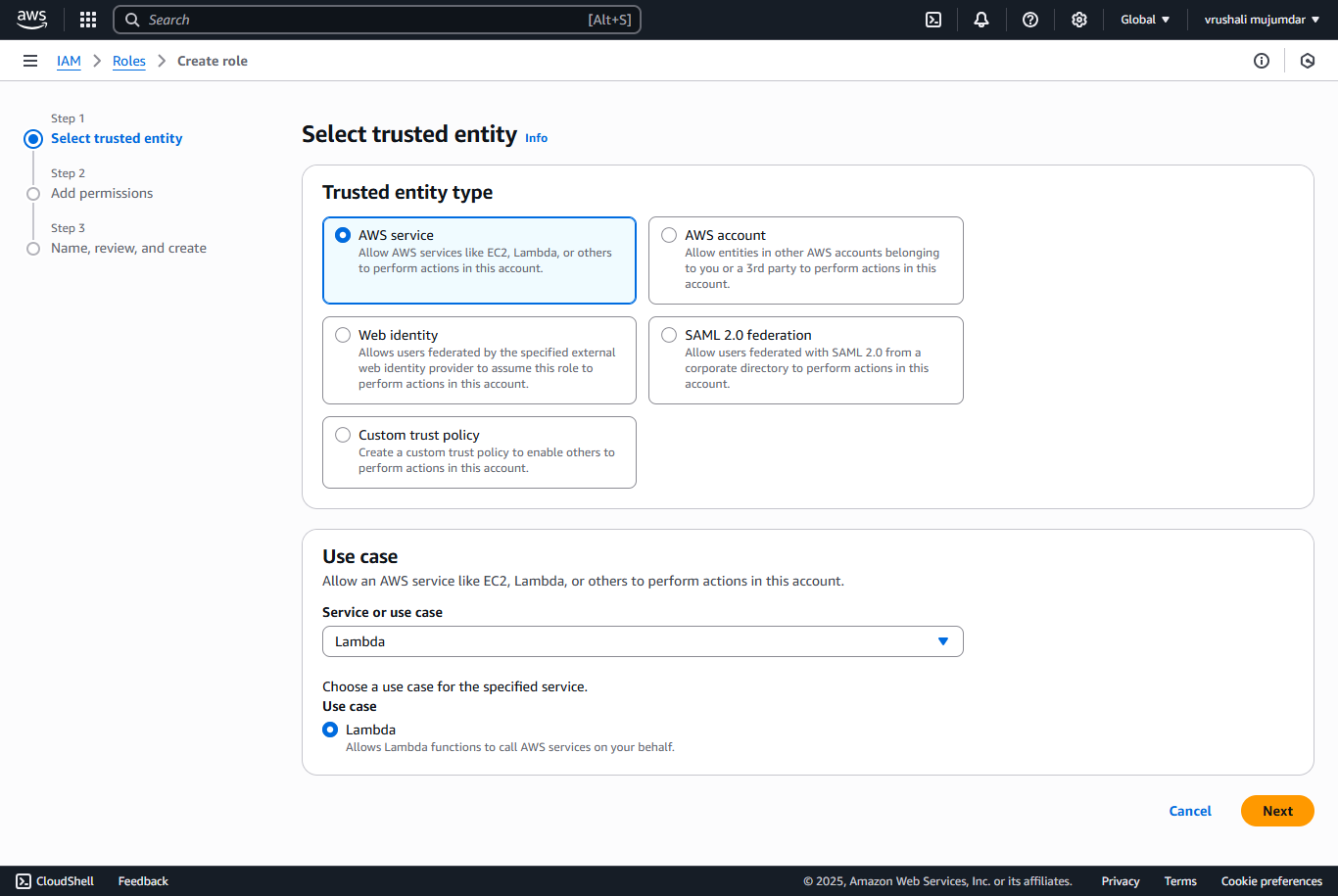
**4. Attach Permissions Policies**

* In the **Permissions policies** list, search for and **select the following policies**:
  + AmazonPollyFullAccess
  + AmazonS3FullAccess
  + AWSLambdaBasicExecutionRole
* Click **Next**.

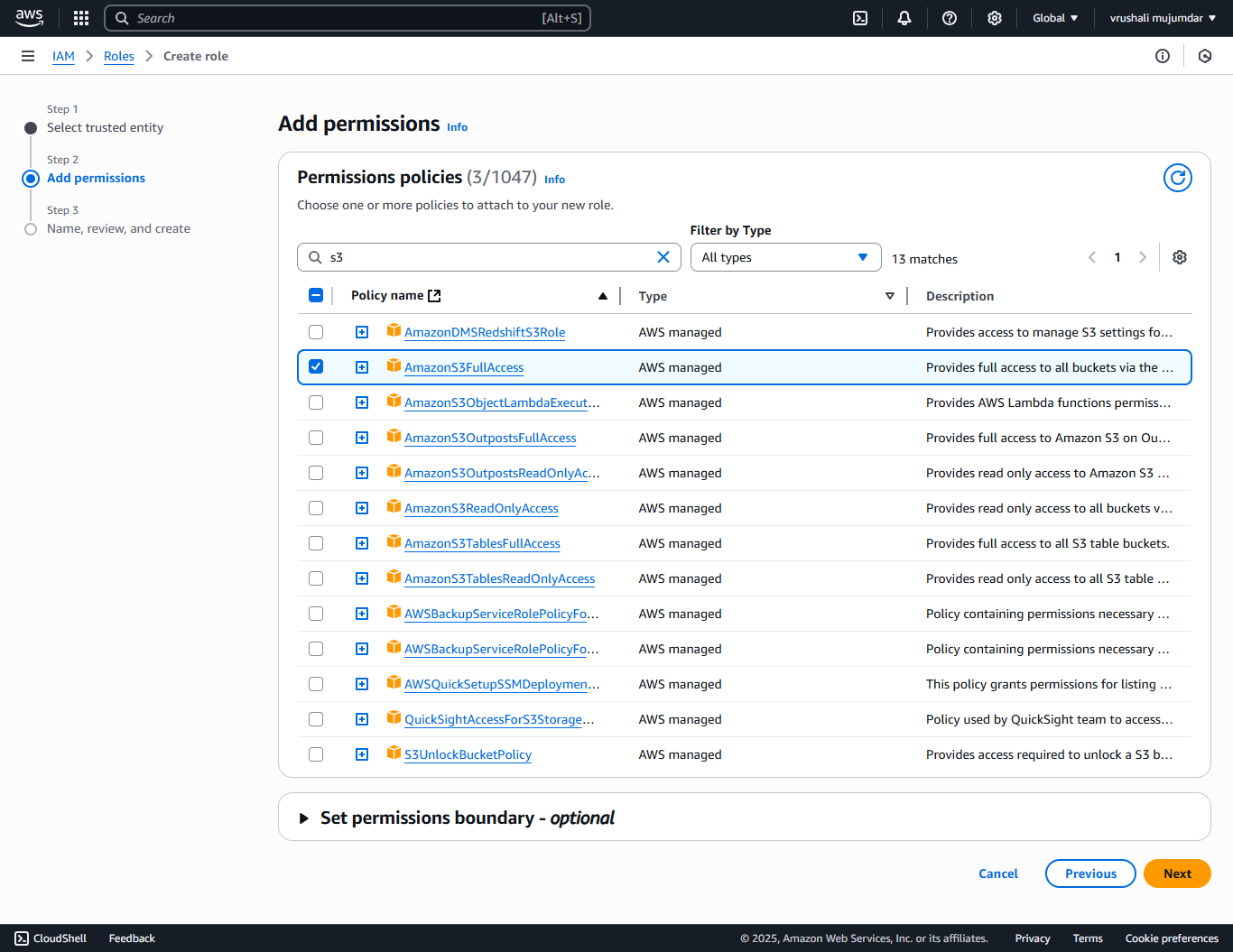
**5. Name, Review, and Create the Role**

* Enter a **Role name**, such as:  
  Lambda\_Polly\_S3\_Role
* (Optional) Enter a **description**.
* Review the permissions and click **Create role**.

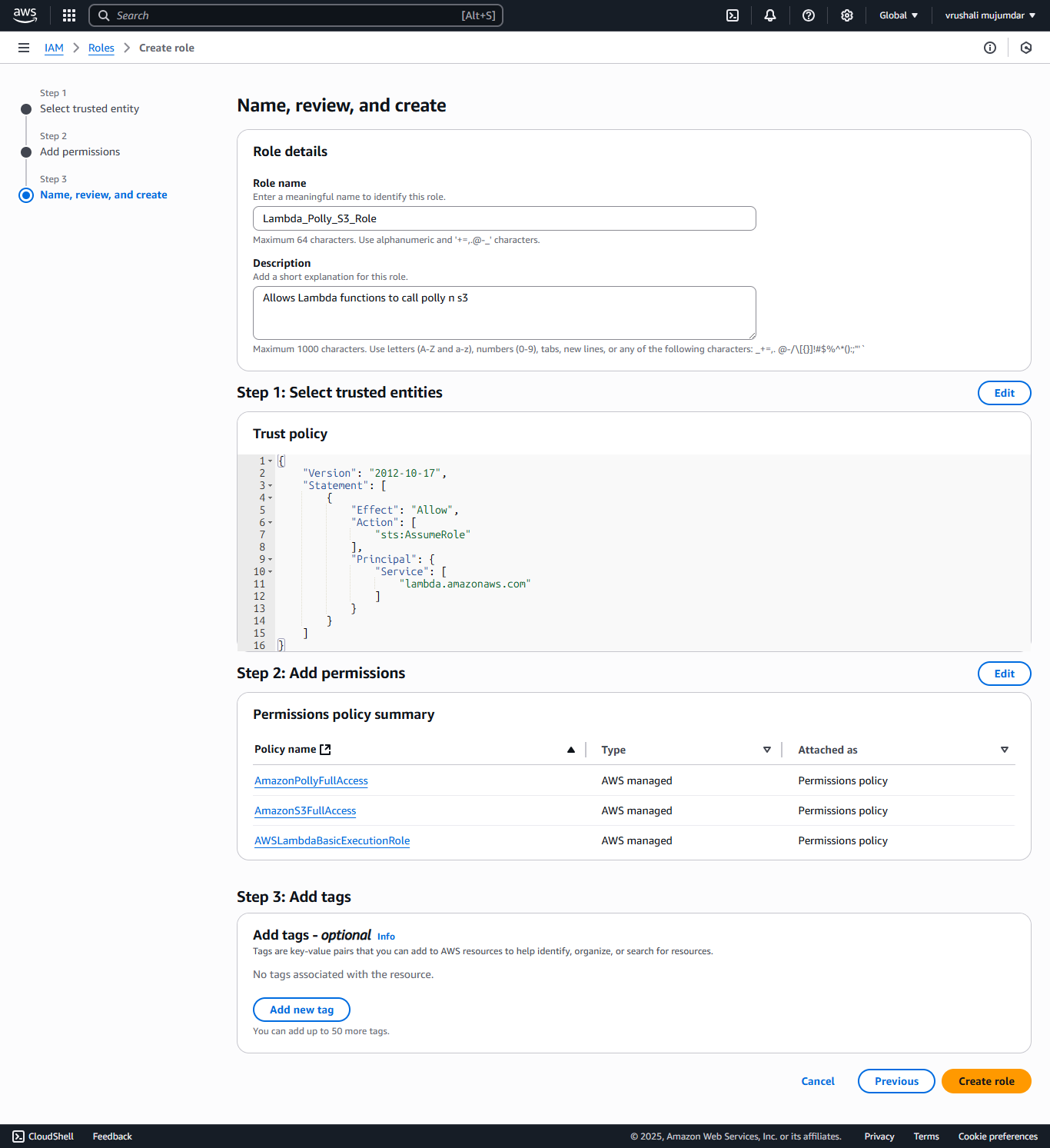
Step:1

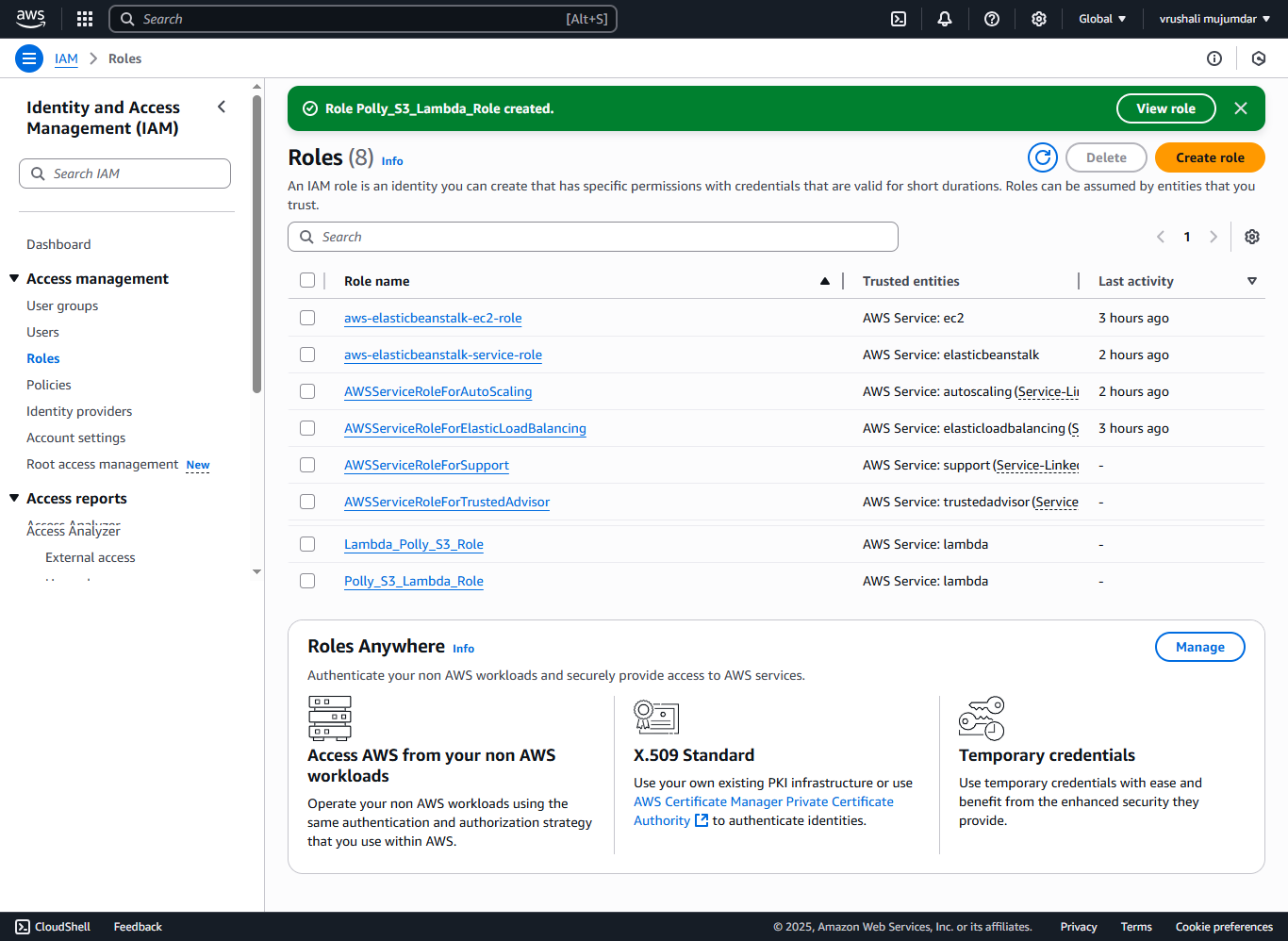


Click Next



Click Next:





**2. Creating an S3 Bucket**

* Navigate to S3 in the AWS console.
* Create a new bucket for storing audio files.

**1. In AWS Console**

* In the **search bar**, type S3 and select **S3** under "Services."

**2. Create a New Bucket**

* Click the **“Create bucket”** button at the top right.

**3. Configure the Bucket**

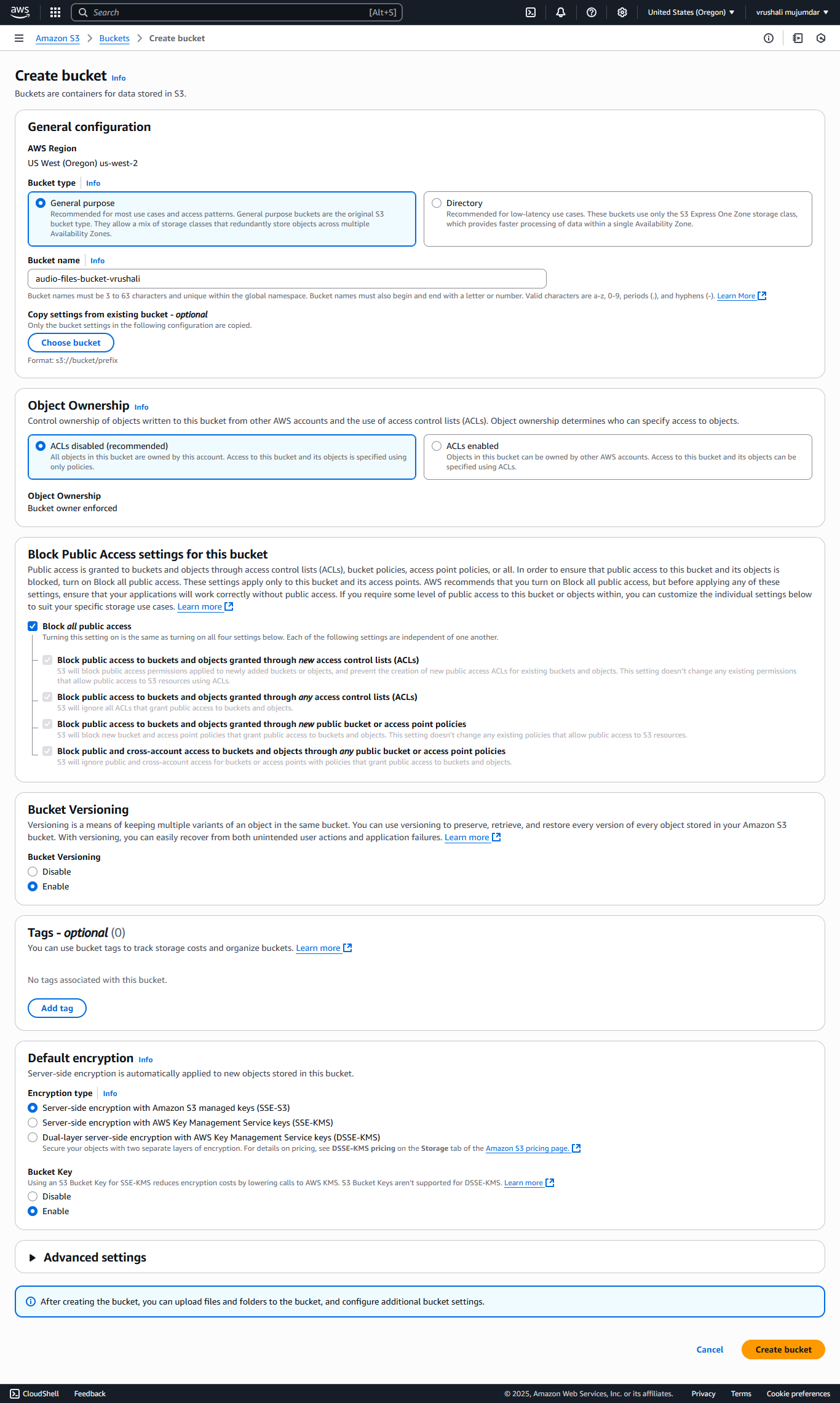
* **Bucket name**:  
  Enter a unique name (e.g., audio-files-bucket-vrushali).  
  Bucket names must be globally unique and follow naming rules (lowercase, no spaces, etc.).
* **AWS Region**:  
  Select the AWS Region closest to your user base or where your Lambda function is hosted (e.g., Asia Pacific (Mumbai)).

**4. Set Bucket Options**

* **Block Public Access settings**:  
  Keep all options **checked** to block public access (recommended for private audio storage).
* **Versioning**:  
  You can **enable** versioning if needed (optional).
* **Encryption**:  
  Choose **"Enable"** if you want the files to be encrypted automatically (recommended).

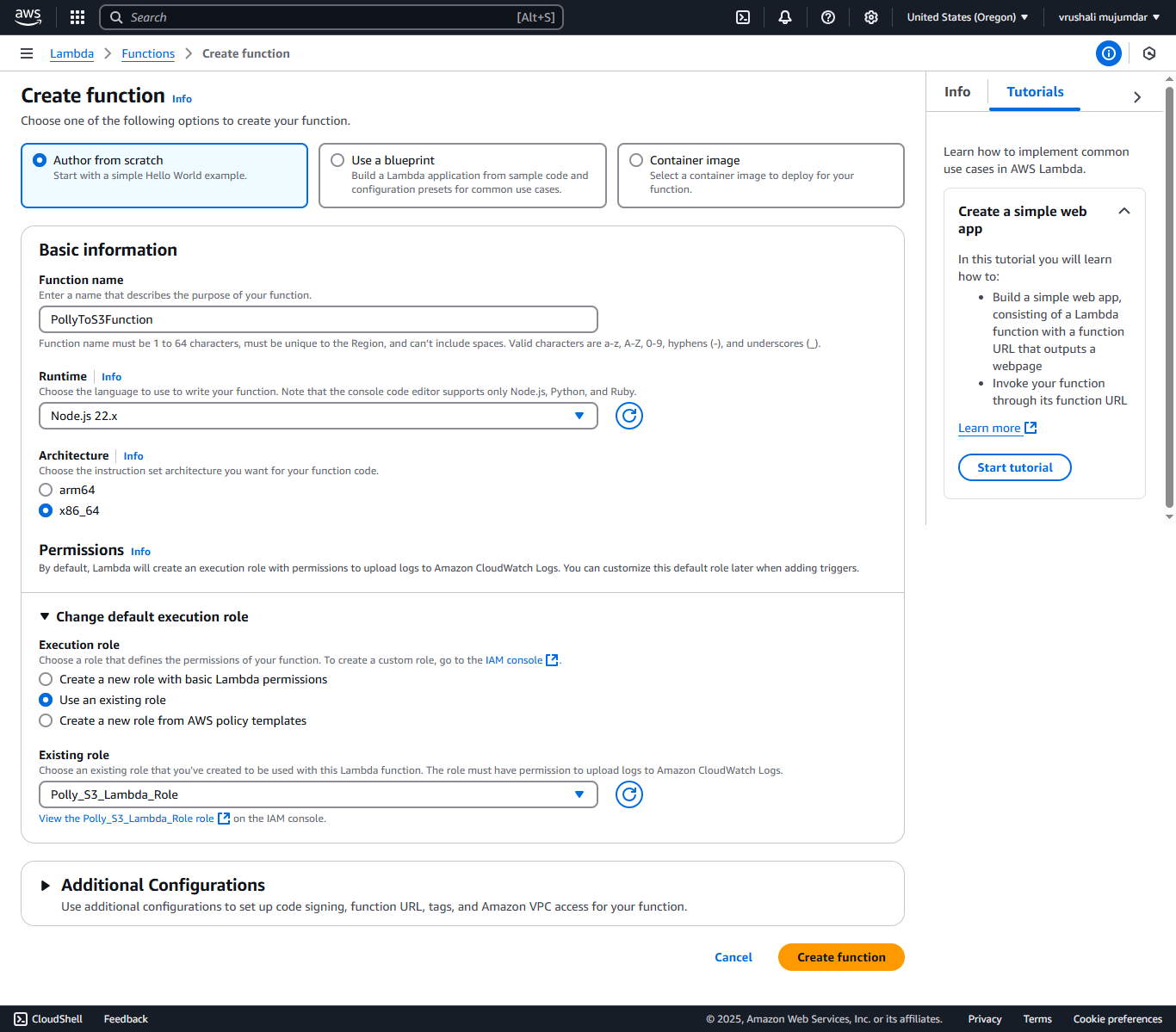
**5. Review and Create**

* Skip the remaining options unless you need logging, tags, etc.
* Click **"Create bucket"** at the bottom.



**3. Creating a Lambda Function**

* Navigate to Lambda in the AWS console.
* Create a new Lambda function with Node.js 18.x runtime.
* Use the existing IAM role created earlier.
* Configure the Lambda function to interact with Amazon Polly and S3.



**1. Install the Required AWS SDK Modules**

AWS Lambda **Node.js 18.x** runtime does **not** include the AWS SDK v2 (aws-sdk) by default. You must package and upload the required modules.

**✔️ On your local machine:**

1. **Create a project folder**, e.g., polly-lambda
2. Inside it, run:

npm init -y

npm install @aws-sdk/client-polly @aws-sdk/client-s3

1. **Add Lambda function code** in a file called index.mjs (you can also use .js, but then configure it properly to use ES Modules).
2. The folder structure will look like:

polly-lambda/

├── node\_modules/

├── package.json

├── package-lock.json

└── index.mjs

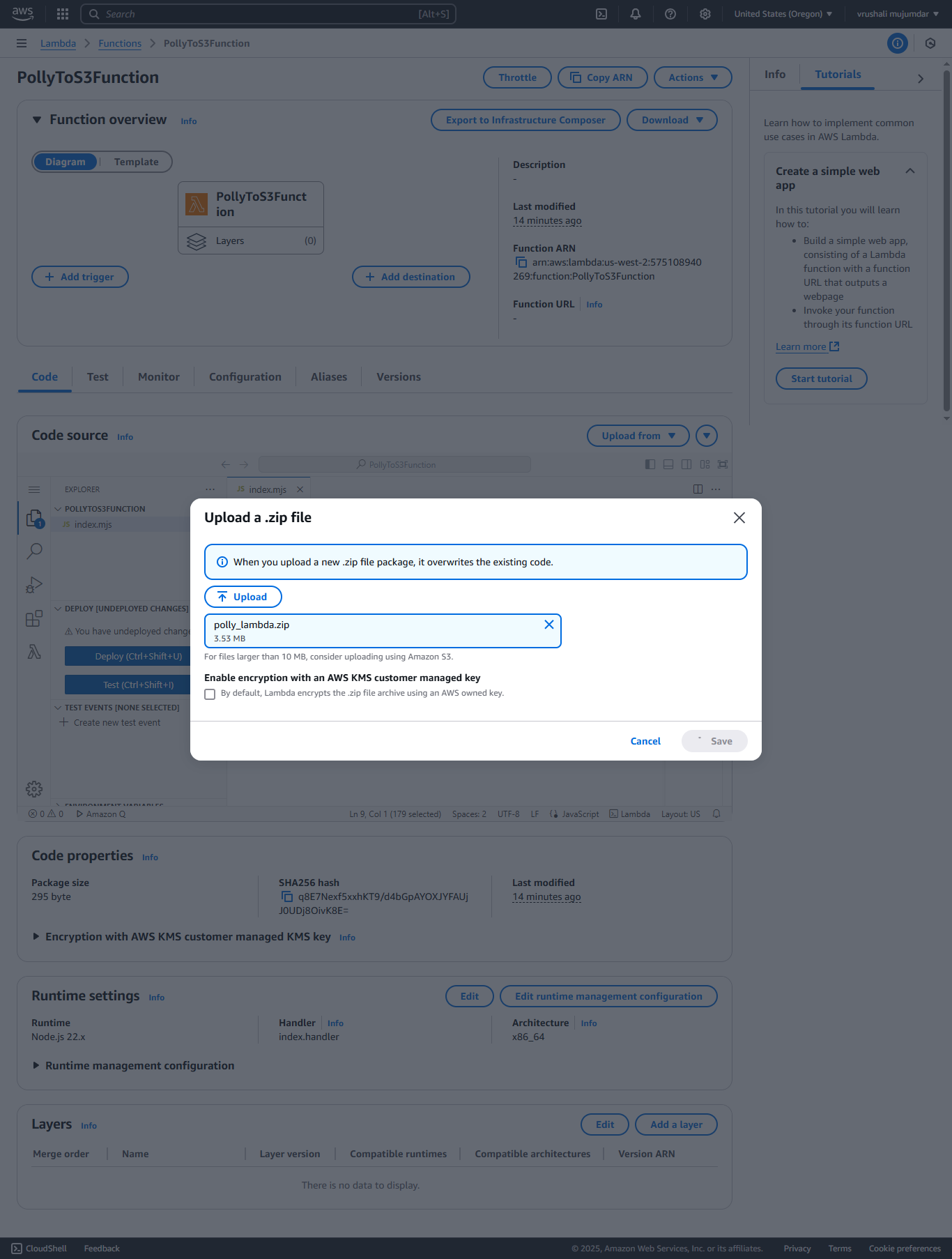
**2. Zip the Folder for Deployment**

* Select all contents **inside** the polly-lambda folder (not the folder itself)
* Compress them into a .zip file (e.g., polly-lambda.zip)

**3. Upload the Code to Lambda**

Back in the **AWS Console > Lambda**:

1. In function page, under the **Code** section:
   * Click **“Upload from”** → **“.zip file”**
   * Choose polly-lambda.zip file
   * Click **Save / Deploy**



**4. Test the Function**

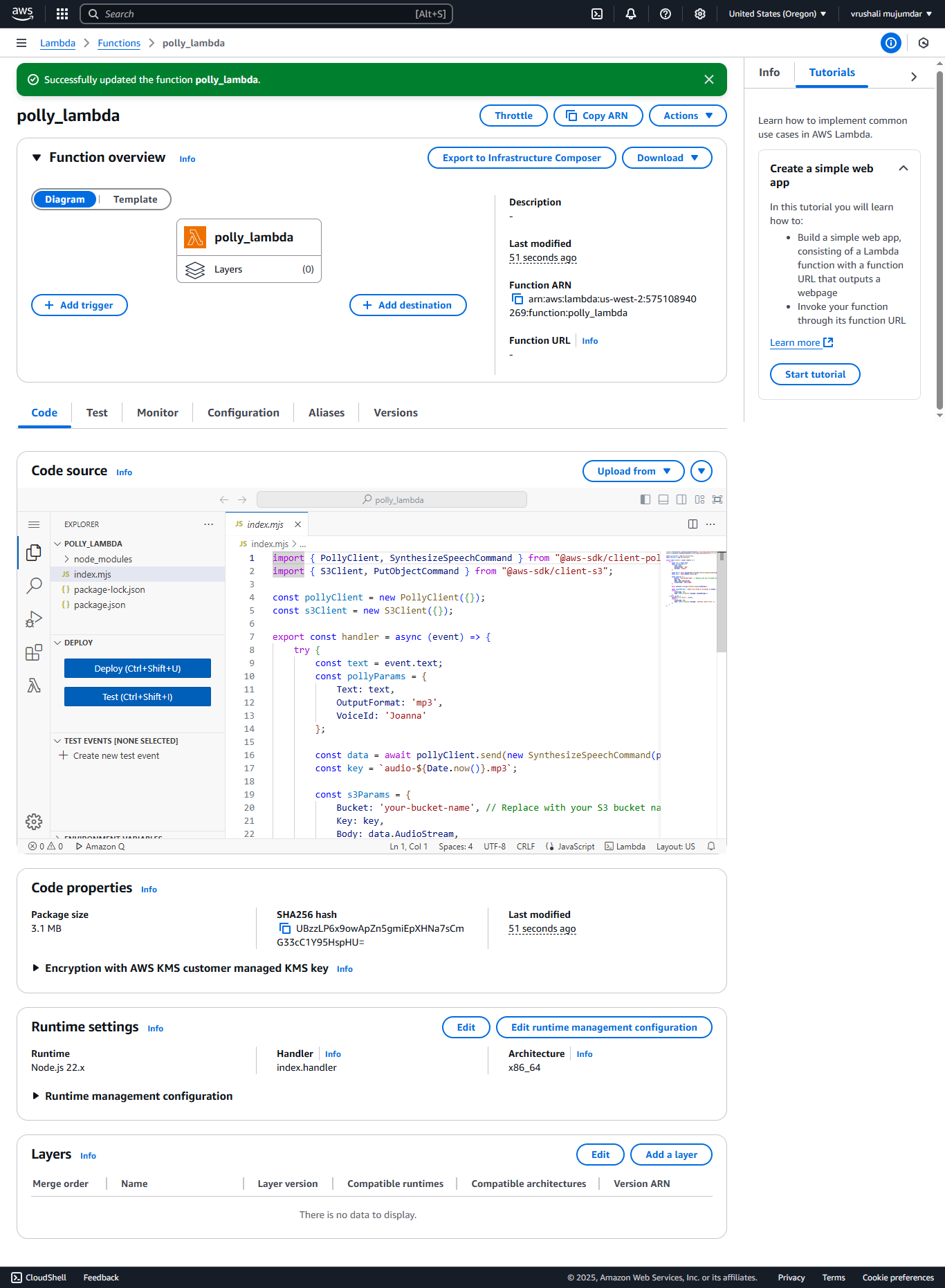
1. Click **Test**
2. Create a test event with the following JSON:

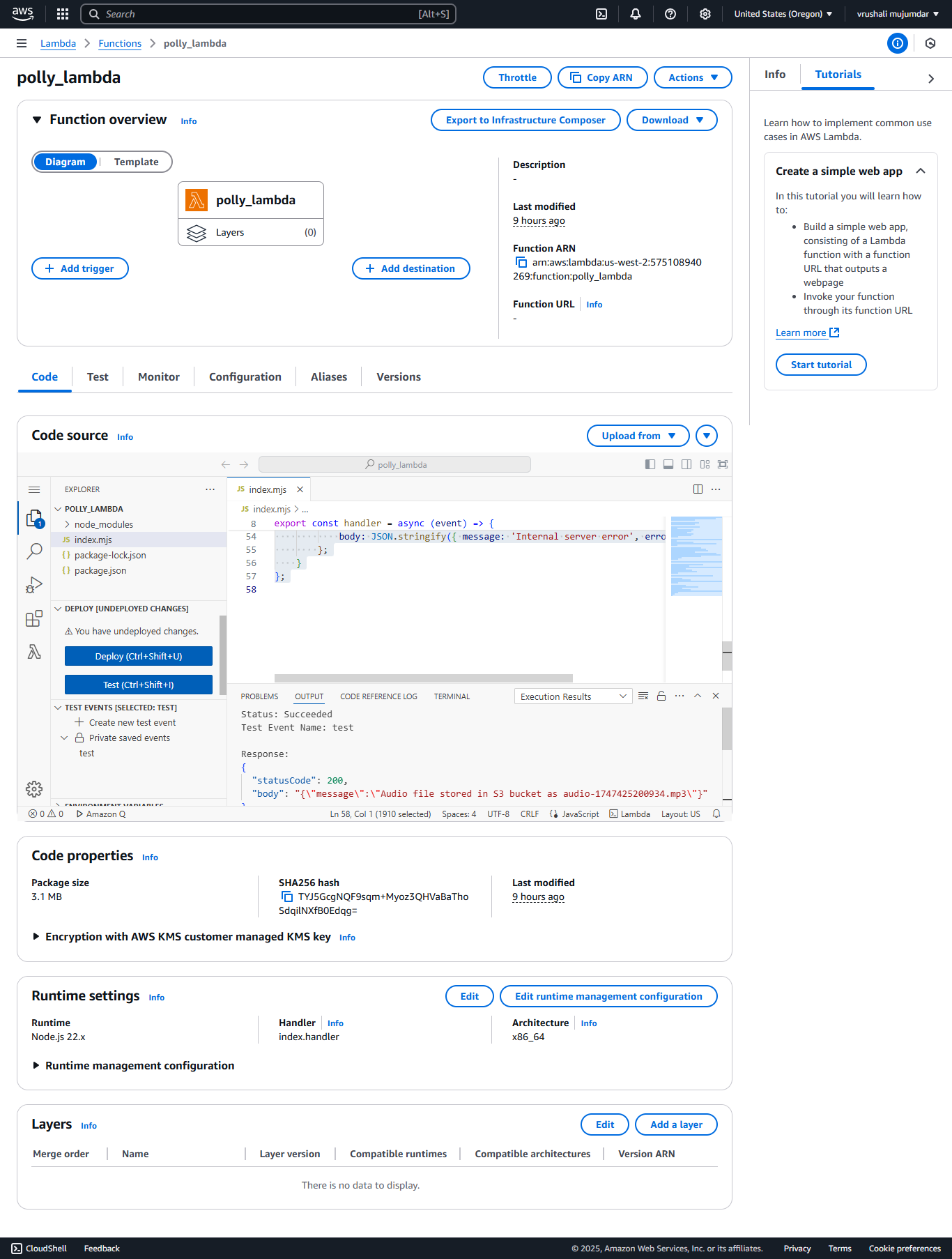
{

"text": "Hello from Node.js Lambda!"

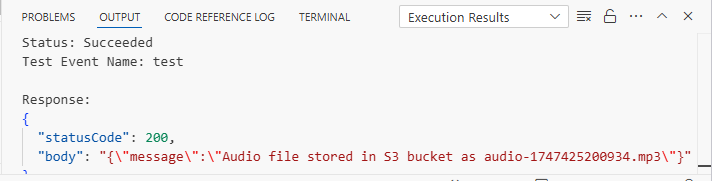
}

1. Click **Test** again.





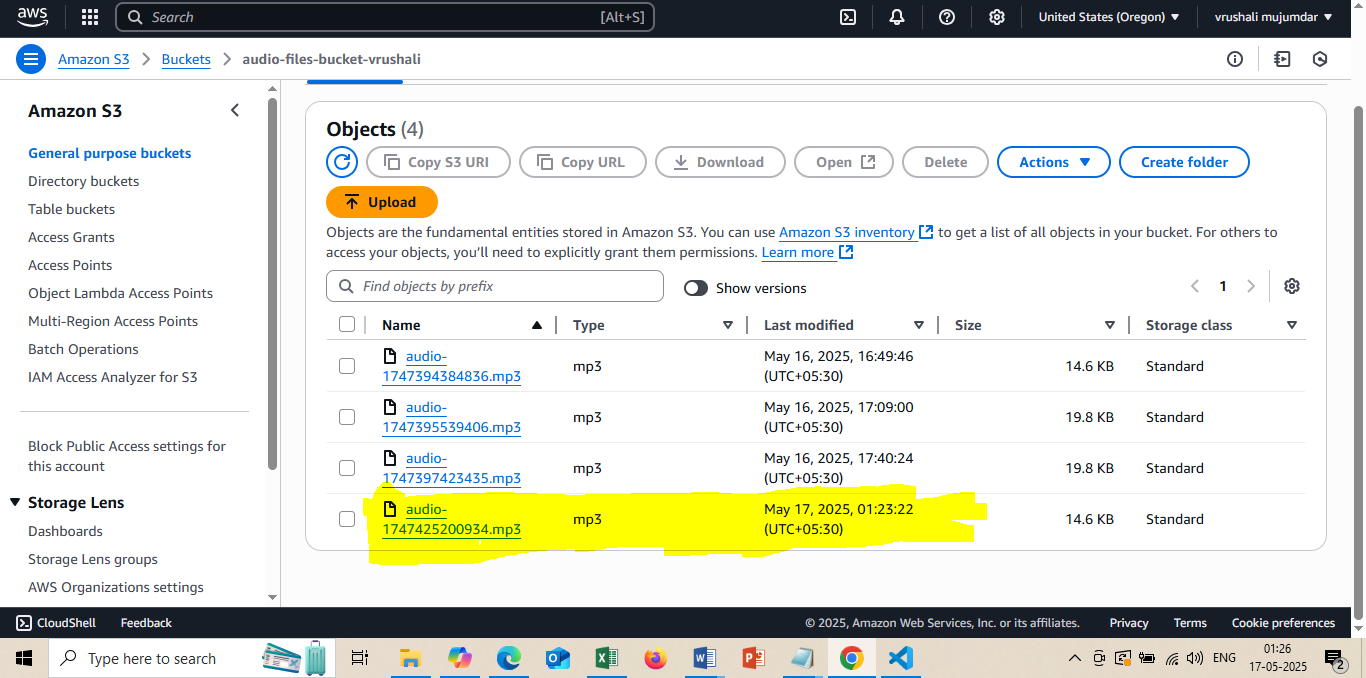
Result:



If all is correct, the function will:

* Use Amazon Polly to convert the text to speech
* Save the audio file to your S3 bucket
* Return a success message with the filename

Here is the audio file that was saved in S3 after the Lambda function successfully executed.



## **Future Enhancements**

Several enhancements can further elevate this project.

1)we plan to add real-time text-to-speech streaming, allowing users to hear speech instantly without waiting for file processing.

2)Support for multiple languages and voices will enable a more personalized and inclusive experience, catering to diverse user needs.

3)We also aim to implement SSML – Speech Synthesis Markup Language – for advanced voice control, such as adjusting pitch, speaking rate, and emphasis.

4)Integration with Amazon Transcribe can enable a complete voice pipeline – converting speech to text and back again.

5)we plan to provide user authentication and history tracking, so users can store, retrieve, and manage their audio files securely.

6)Finally, we envision deploying the project in a mobile app, making voice services available on the go.

These future enhancements will make the application more intelligent, user-friendly, and production-ready."

## **Conclusion**

To sum it up, this project shows how we can turn written text into speech using Amazon Polly and AWS Lambda. I used Lambda to run my code without needing to manage any servers, and Polly to create realistic voice output from text. The audio files are saved in Amazon S3, so they can be played or downloaded anytime. This setup is simple, fast, and useful for things like learning apps, announcements, or helping people who prefer listening over reading. In the future, I can add more voices, languages, and even make it work on mobile phones.

Access the source code via the GitHub link provided.

<https://github.com/vrush119/text-narrator>

Thank you for listening!