😊 Amazon Bedrock (Image Generation)

Amazon Bedrock is a fully managed service by AWS designed to help developers build, customize, and deploy generative AI applications using foundation models (FMs). These are pre-trained large models that can perform a wide range of tasks, including text generation, summarization, question answering, image generation, and more. Bedrock allows developers to access and use these powerful models through an easy-to-use API without needing to worry about the infrastructure or managing the underlying machine learning models.

Key Features:

1. **Access to Foundation Models**: Bedrock provides a variety of foundation models from leading AI companies, such as AI21 Labs, Anthropic, Stability AI, and Amazon's own Titan models. This gives developers access to a broad range of capabilities for different types of tasks, including natural language processing (NLP), image generation, and other generative AI use cases.
2. **Customization**: With Bedrock, users can customize these foundation models using their own data. This customization is done using a process known as fine-tuning, where developers adjust the model to better suit specific business needs or use cases, without requiring deep AI or machine learning expertise.
3. **Zero Infrastructure Management**: Since Bedrock is fully managed, developers don't have to worry about the underlying hardware or model updates. AWS handles scaling, performance, and infrastructure, allowing users to focus on building their applications.
4. **Seamless Integration with AWS**: Bedrock integrates with other AWS services, making it easier to incorporate AI capabilities into existing workflows and applications. This includes services like Amazon S3, Lambda, SageMaker, and more.
5. **Pay-as-you-go**: As with most AWS services, Bedrock follows a pay-as-you-go model, allowing developers to pay for the usage of the models without large upfront costs.

Use Cases:

- **Chatbots**: Building AI-powered chatbots that can engage in human-like conversations.
- **Content Generation**: Creating text or image-based content such as blogs, social media posts, or marketing materials.
- **Data Summarization**: Using foundation models to summarize lengthy documents or articles.
- **Code Assistance**: Generating code snippets, documentation, or debugging help.

Target Audience:

- **Developers**: Looking for an easy way to integrate generative AI models into their applications.
- **Businesses**: Seeking to enhance customer experience, automate processes, or generate content using AI without deep expertise in machine learning.

1. Foundation Models on Bedrock:

Amazon Bedrock provides access to a wide range of foundation models (FMs) from leading AI providers. Some of the most notable models include:

- **Amazon Titan**: These are Amazon's own generative models designed for common use cases like text generation, summarization, and personalization. They focus on high scalability and reliability for enterprise use cases.
- **Jurassic-2 from AI21 Labs**: A powerful NLP model capable of generating and comprehending long text inputs. It's suitable for tasks like creating high-quality text or supporting conversational AI.
- **Claude from Anthropic**: An AI system focused on safe and trustworthy conversational models. Claude can be fine-tuned for customer support, virtual assistants, and other applications where safety and fairness are important.
- **Stable Diffusion from Stability AI**: A widely known generative model for image generation that can create high-quality, photorealistic images from text descriptions.

Bedrock enables developers to choose the best-suited foundation model based on their specific task, whether it's for text, images, or other forms of generative content.

2. Customization and Fine-Tuning:

One of the significant advantages of Amazon Bedrock is the ability to **customize foundation models**. This is typically achieved through fine-tuning, which allows developers to adapt models to their own data without needing to train a model from scratch.

Fine-tuning is done in a way that even non-experts can adjust these pre-trained models for unique datasets, making them more accurate for business-specific tasks such as:

- Creating personalized customer interactions.
- Optimizing content based on a company's specific tone or style.
- Training models for industry-specific language or jargon.

The customization process is seamless and can be done through the Bedrock interface or API without needing in-depth ML knowledge.

3. Integration with AWS Ecosystem:

Bedrock is natively integrated with other **AWS services**, which simplifies deploying and scaling generative AI models within your infrastructure. Some of the common integrations include:

- **Amazon S3**: Storing and retrieving large datasets for training and fine-tuning foundation models.
- **AWS Lambda**: Automating workflows where you can trigger AI actions like generating summaries or answering queries.
- **Amazon SageMaker**: For those who need more control over model customization and deployment, SageMaker can also be used alongside Bedrock.

- **Amazon CloudWatch**: To monitor the performance and usage of Bedrock applications in real time.

This tight integration with AWS's vast ecosystem allows organizations to easily incorporate generative AI models into existing workflows, pipelines, and applications.

4. Key Benefits of Amazon Bedrock:

- **No AI/ML Expertise Required**: Bedrock lowers the barrier to entry by providing pre-trained models, reducing the need for deep AI knowledge. This makes it easy for any developer to integrate cutting-edge AI into applications without building models from scratch.
- **Cost-Effective and Scalable**: Bedrock operates on a **pay-as-you-go model**, meaning users only pay for what they use. It also automatically scales based on demand, so whether you're serving a few users or thousands, the service adjusts accordingly.
- **Model Experimentation**: With Bedrock, users can easily test and switch between different models provided by various AI partners (AI21 Labs, Anthropic, Stability AI) to find the most effective one for their use case. This avoids vendor lock-in and offers flexibility in model selection.
- **Automatic Infrastructure Management**: AWS handles the underlying infrastructure, model updates, and scaling, which frees up developers to focus on creating high-value applications without worrying about model maintenance or hardware.
- **Data Privacy and Security**: As a fully managed AWS service, Bedrock adheres to **AWS's high standards for data security and compliance**, including encryption of data both at rest and in transit. This is essential for companies in regulated industries like healthcare or finance.

5. Typical Use Cases for Bedrock:

- **Customer Support Automation**: Using conversational models like Claude from Anthropic to power intelligent, context-aware chatbots or virtual assistants that can handle complex queries with natural language.
- **Content Personalization**: Leveraging Titan or Jurassic-2 for tailored content generation, such as personalized recommendations, blog posts, or social media content based on user preferences.
- **Image and Video Generation**: Utilizing Stable Diffusion from Stability AI to create custom images, visual assets, or even prototypes from text descriptions, applicable in marketing, e-commerce, and design.
- **Document Summarization**: Simplifying document-heavy workflows with models that can read and generate concise summaries or extract relevant information from long-form text, useful in legal or academic research.
- **Code Generation**: Assisting developers by generating or suggesting code snippets, documentation, or bug fixes using generative AI models specialized for programming tasks.

6. Comparison with Other AI Services:

- **OpenAI**: Similar to Amazon Bedrock, OpenAI offers access to pre-trained large language models (like GPT-4) for text generation and other tasks. However, Bedrock differentiates itself by offering multiple models from various providers and deeper integration into the AWS ecosystem.
- **Google Vertex AI**: Google's AI platform also provides pre-trained models and tools for customization but focuses heavily on its own infrastructure. Bedrock offers more flexibility in model choices and seamless integration into a broader range of AWS services.
- **Microsoft Azure OpenAI Service**: Azure's counterpart allows access to OpenAI's models like GPT, but Bedrock provides access to models from a broader range of providers, giving developers more choices.

7. When to Use Amazon Bedrock:

- **For quick AI integration without infrastructure management**: If your team wants to incorporate AI quickly without building models or maintaining servers, Bedrock is ideal.
- **For businesses that want to customize AI models**: Bedrock is a good option if you need to adapt AI models to your specific domain, such as personalized customer interactions or industry-specific applications.
- **For scaling AI across cloud-native applications**: Companies with existing AWS workflows can seamlessly integrate Bedrock's generative AI capabilities into their apps.

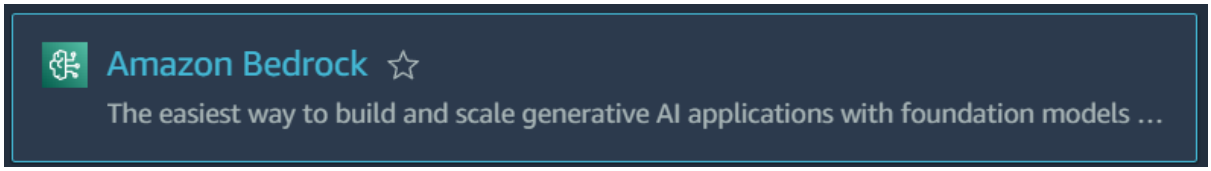
Conclusion:

Amazon Bedrock is a game-changer for developers and organizations looking to leverage the power of generative AI models without the complexity of managing the infrastructure or developing models from scratch. By offering access to a variety of high-performing foundation models, allowing easy customization, and providing seamless integration with AWS services, Bedrock enables users to innovate and build AI-powered applications faster, more securely, and at scale.

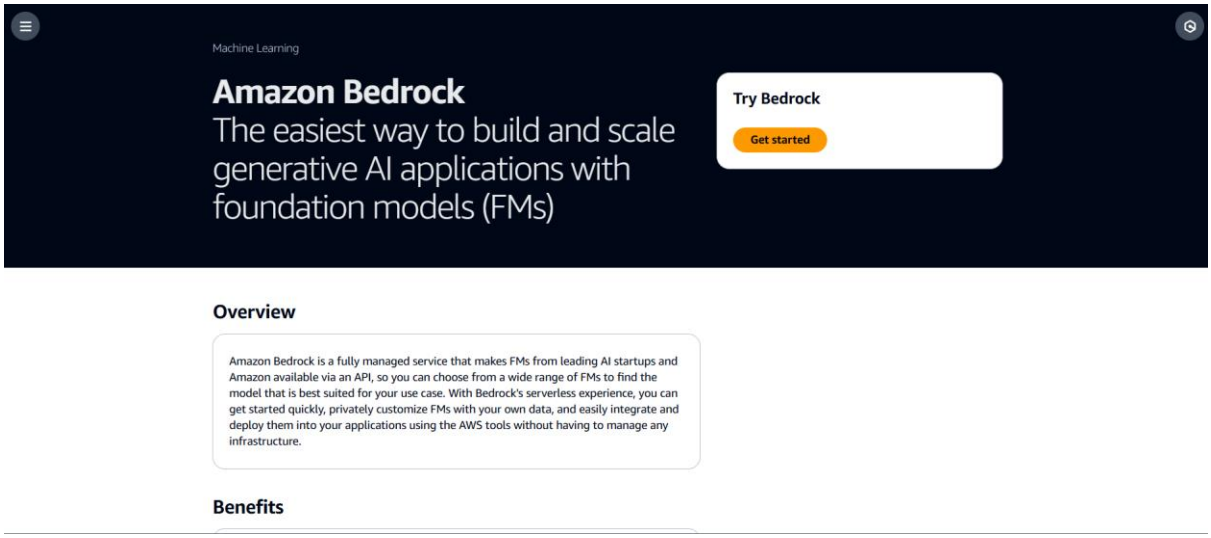
In this lab we will be using Amazon Bedrock to generate images using Lambda and API gateway then those generated images will be stored in an S3 bucket.

😄 To begin with the Lab:

1. In your AWS Console search for Amazon Bedrock. Choose this service accordingly. The region used in this lab is North Virginia, make sure you are also in the same region.

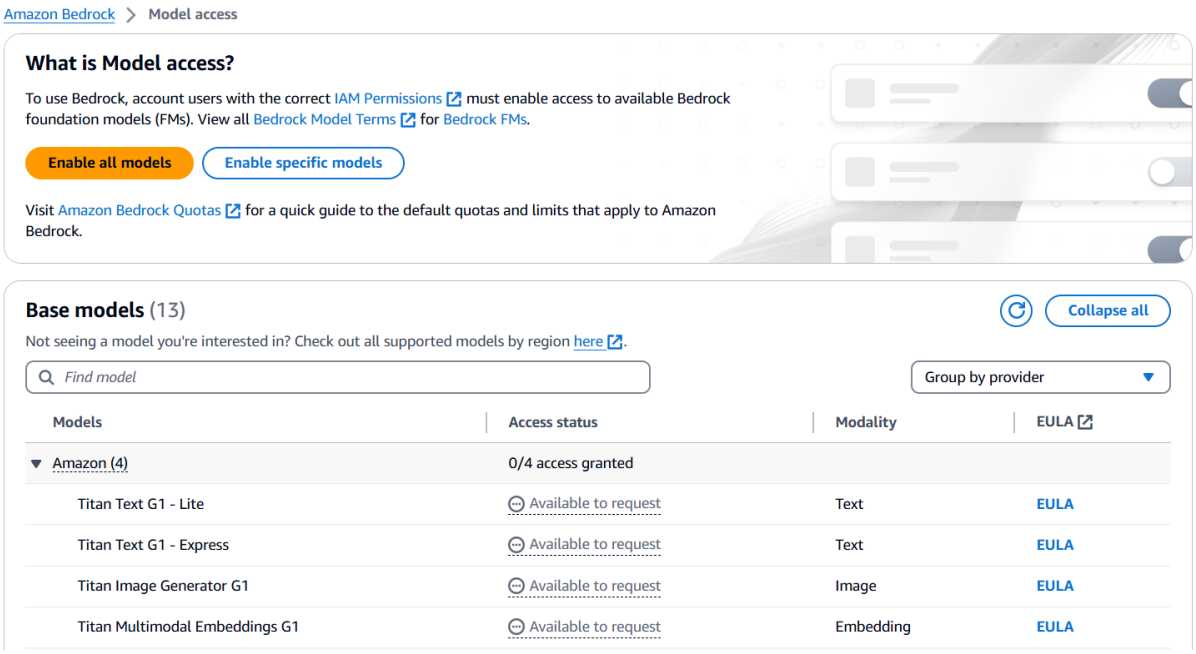


2. This is the landing page Amazon Bedrock.

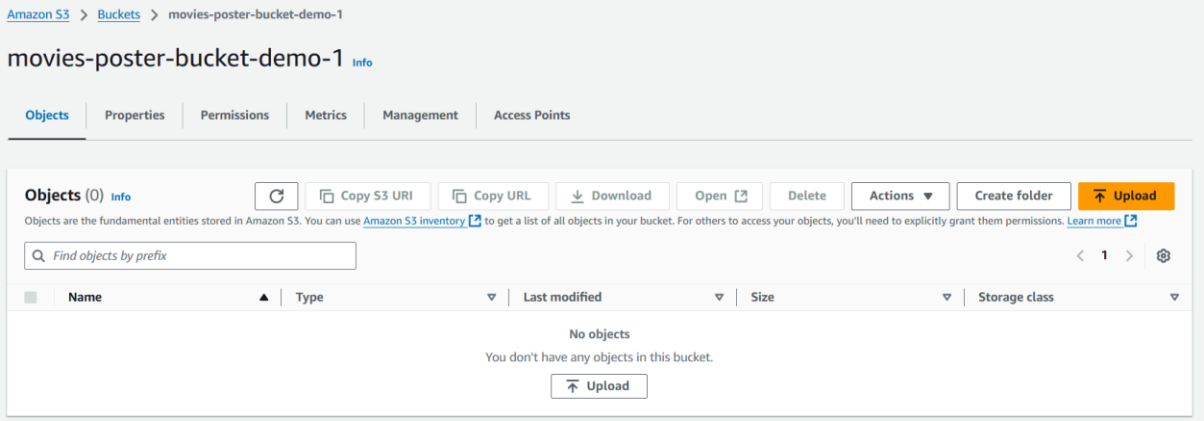


3. Click on the hamburger icon and open the left pane then scroll down to the bottom and choose model access from bedrock configurations.

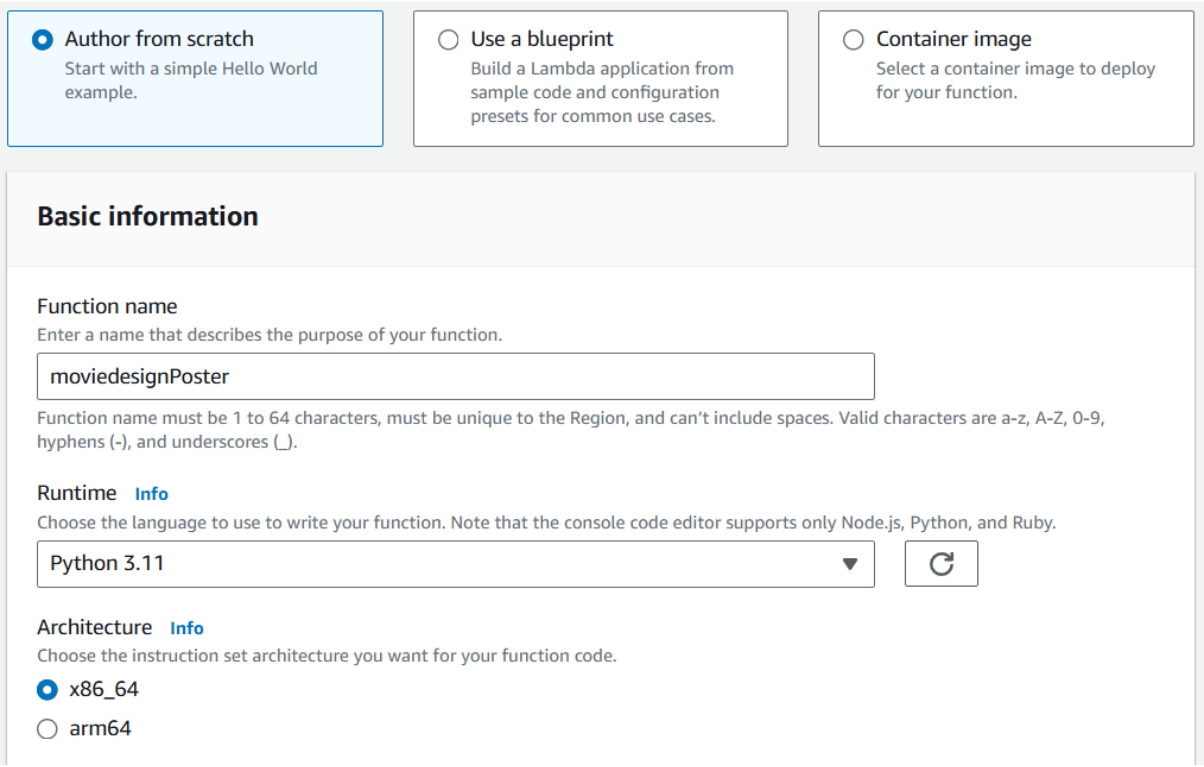
4. Then you will see that there is total 13 models which you need to enable for this lab and they might take at least 1 hour to get activated. Now click on Enable all models and choose to enable each and every one of them.



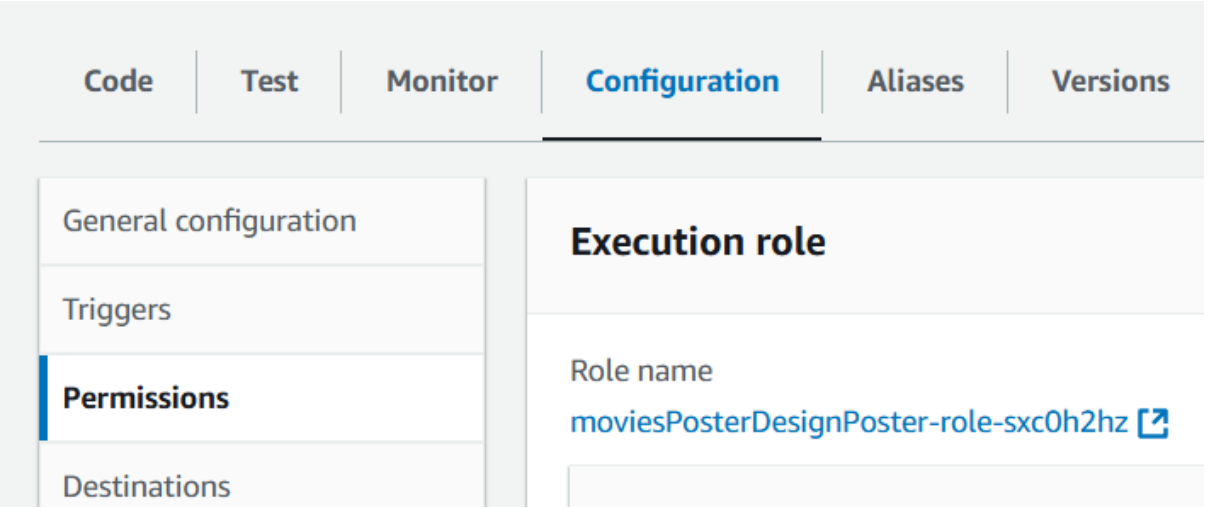
5. After all that first we need to search S3 and create a bucket. Below you can see that our bucket has been created successfully.



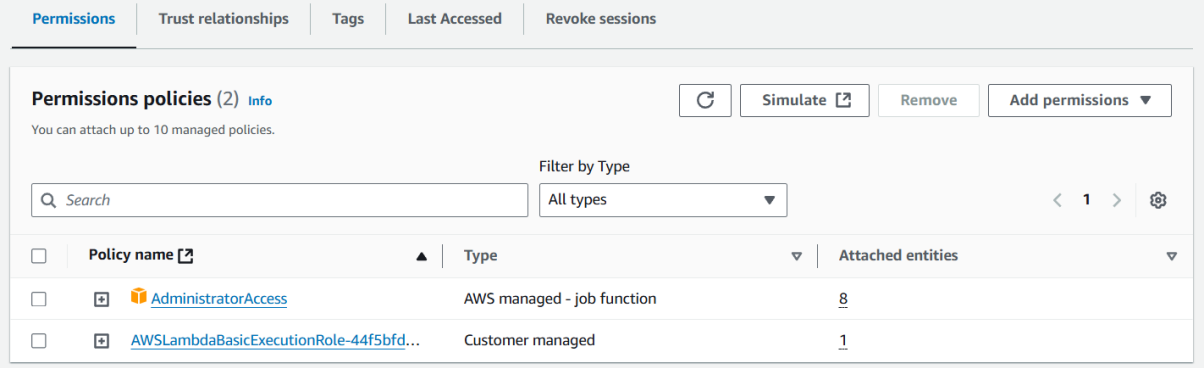
6. Then we need to create a Lambda function, search and navigate to lambda. Here you just need to choose Author from scratch and give it a name then choose Python 3.11 as your environment and keep rest of the settings to default and create your lambda function.



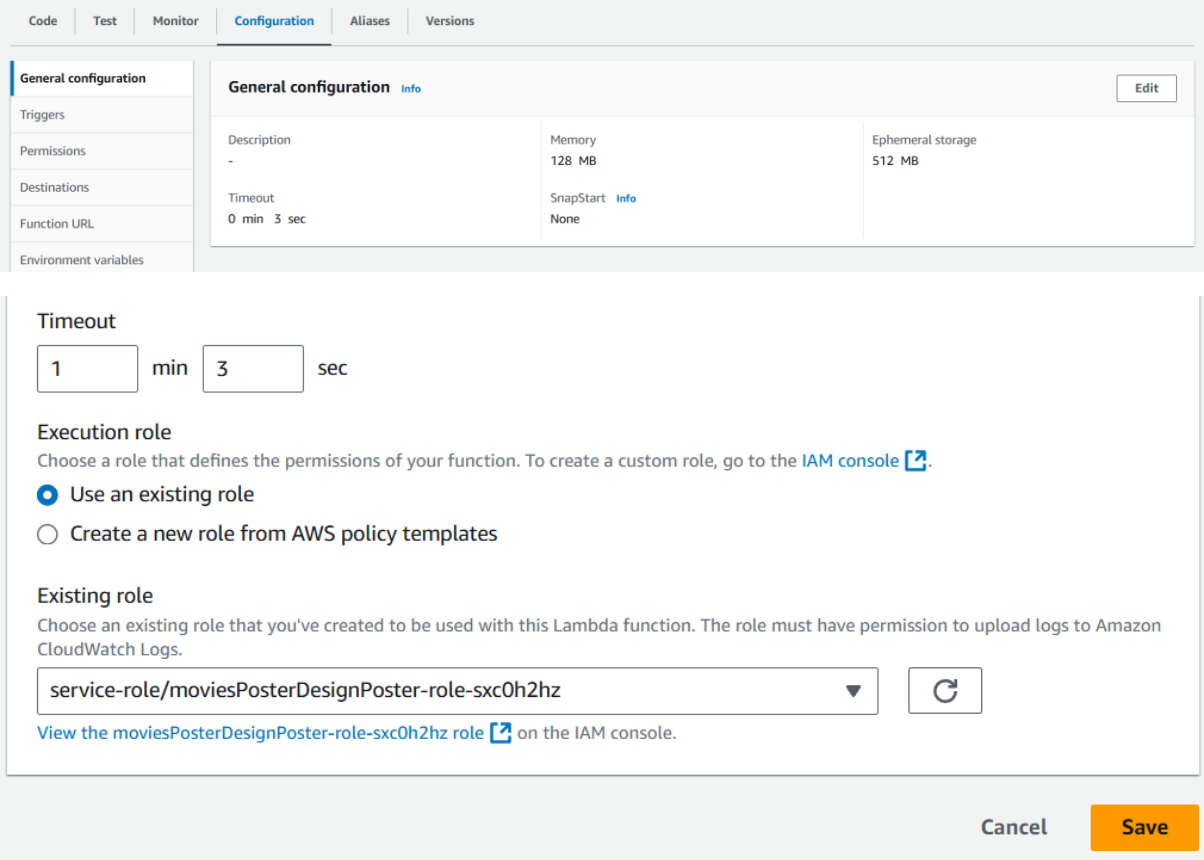
7. Since your lambda function has been created, you need to change some of its properties. For that, you need to go to the configuration tab, then to Permission, and click on the IAM role attached to your lambda function.



8. To this IAM role, for the simplicity of this lab attach administrative access to your IAM role and come back to lambda function. Now IAM role has access to everything.



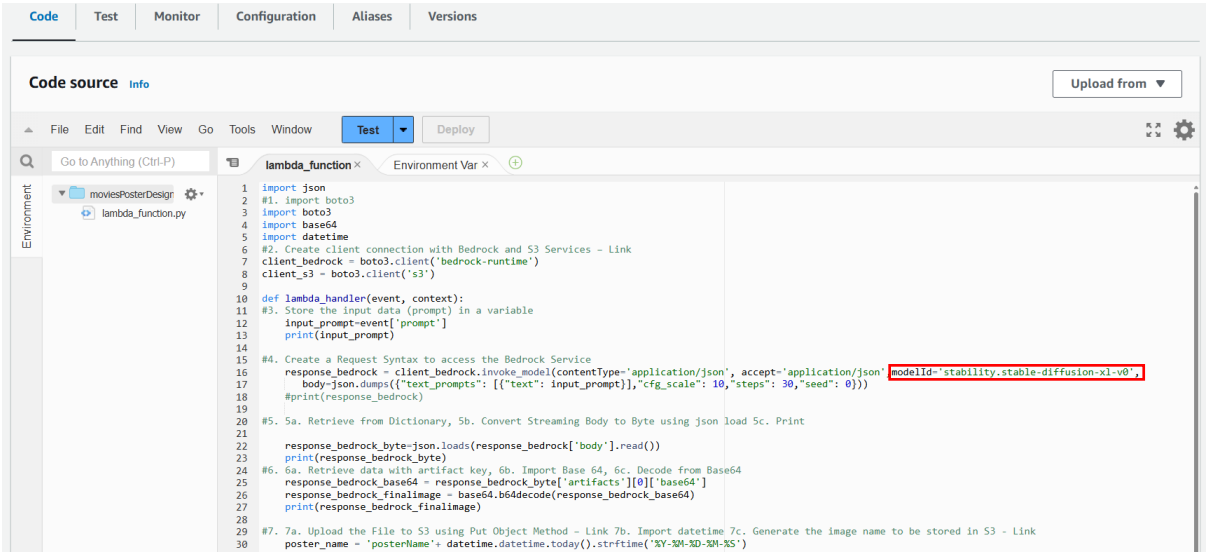
9. Then come to the general configuration and click on edit. Here we are going to increase the timeout limit to 1 minute from just 3 seconds. Click on save to save your changes.



10. After that you need to come back to the code tab and paste the code shown below. You will also get a separate file for this code. Then click on deploy to deploy your code.

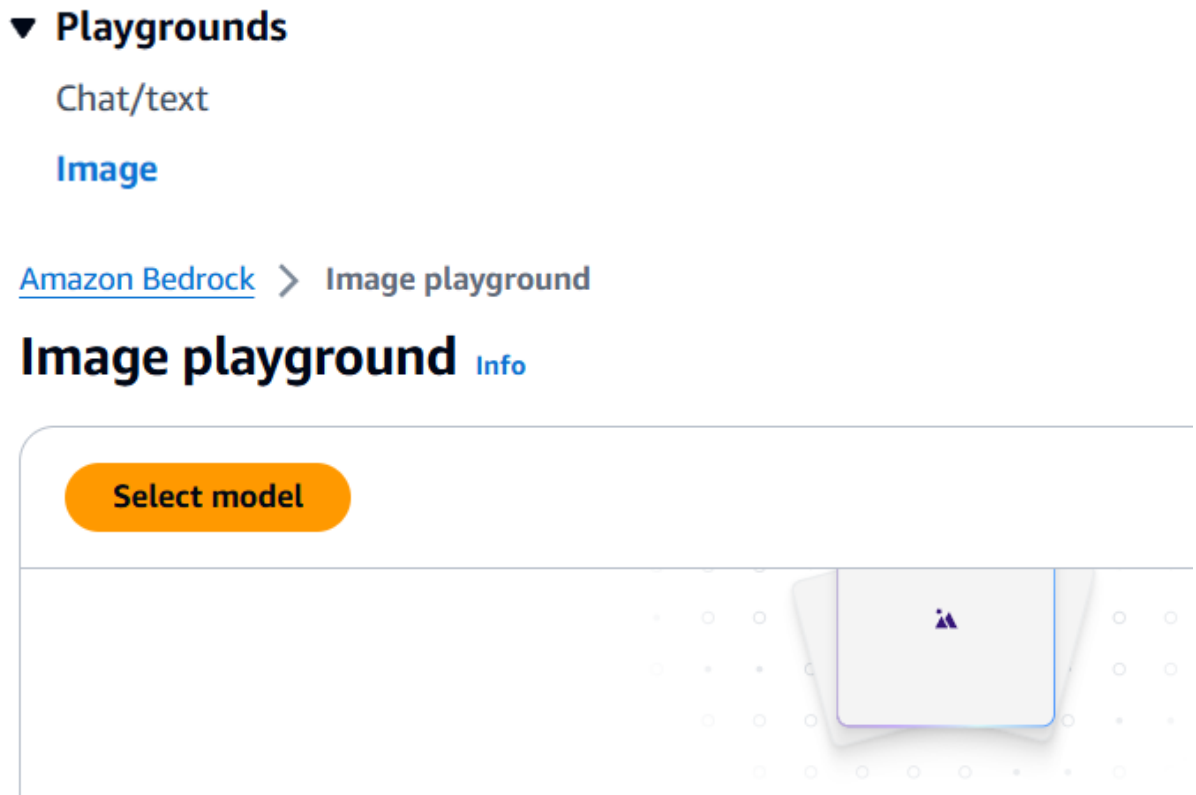
```
import json
#1. import boto3
import boto3
import base64
import datetime
#2. Create client connection with Bedrock and S3 Services – Link
client_bedrock = boto3.client('bedrock-runtime')
client_s3 = boto3.client('s3')
```

```
def lambda_handler(event, context):
#3. Store the input data (prompt) in a variable
   input_prompt=event['prompt']
   print(input_prompt)

#4. Create a Request Syntax to access the Bedrock Service
                                                response_bedrock                    =
client_bedrock.invoke_model(contentType='application/json',
accept='application/json',modelId='stability.stable-diffusion-xl-v0',
       body=json.dumps({"text_prompts": [{"text": input_prompt}],"cfg_scale":
10,"steps": 30,"seed": 0}))
   #print(response_bedrock)

#5. 5a. Retrieve from Dictionary, 5b. Convert Streaming Body to Byte using json load
5c. Print

   response_bedrock_byte=json.loads(response_bedrock['body'].read())
   print(response_bedrock_byte)
#6. 6a. Retrieve data with artifact key, 6b. Import Base 64, 6c. Decode from Base64
   response_bedrock_base64 = response_bedrock_byte['artifacts'][0]['base64']
   response_bedrock_finalimage = base64.b64decode(response_bedrock_base64)
   print(response_bedrock_finalimage)

#7. 7a. Upload the File to S3 using Put Object Method – Link 7b. Import datetime 7c.
Generate the image name to be stored in S3 - Link
    poster_name = 'posterName'+ datetime.datetime.today().strftime('%Y-%M-
%D-%M-%S')

   response_s3=client_s3.put_object(
      Bucket='movieposterdesign01',
      Body=response_bedrock_finalimage,
      Key=poster_name)

#8. Generate Pre-Signed URL
       generate_presigned_url   =   client_s3.generate_presigned_url('get_object',
Params={'Bucket':'movieposterdesign01','Key':poster_name}, ExpiresIn=3600)
   print(generate_presigned_url)
   return {
      'statusCode': 200,
      'body': generate_presigned_url
   }
```

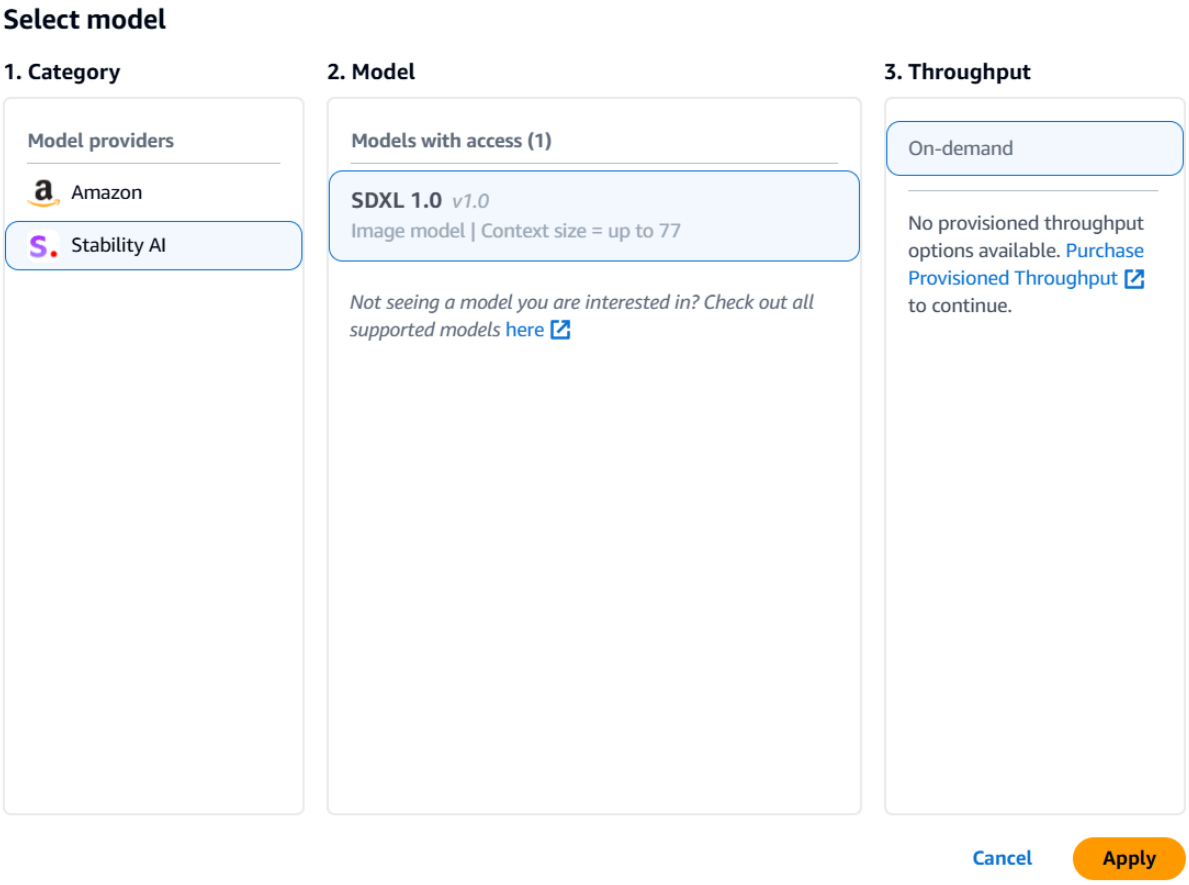


11. In this code if you look at line number 16, we need a model ID and we can get this on the Bedrock console. So, navigate to it.

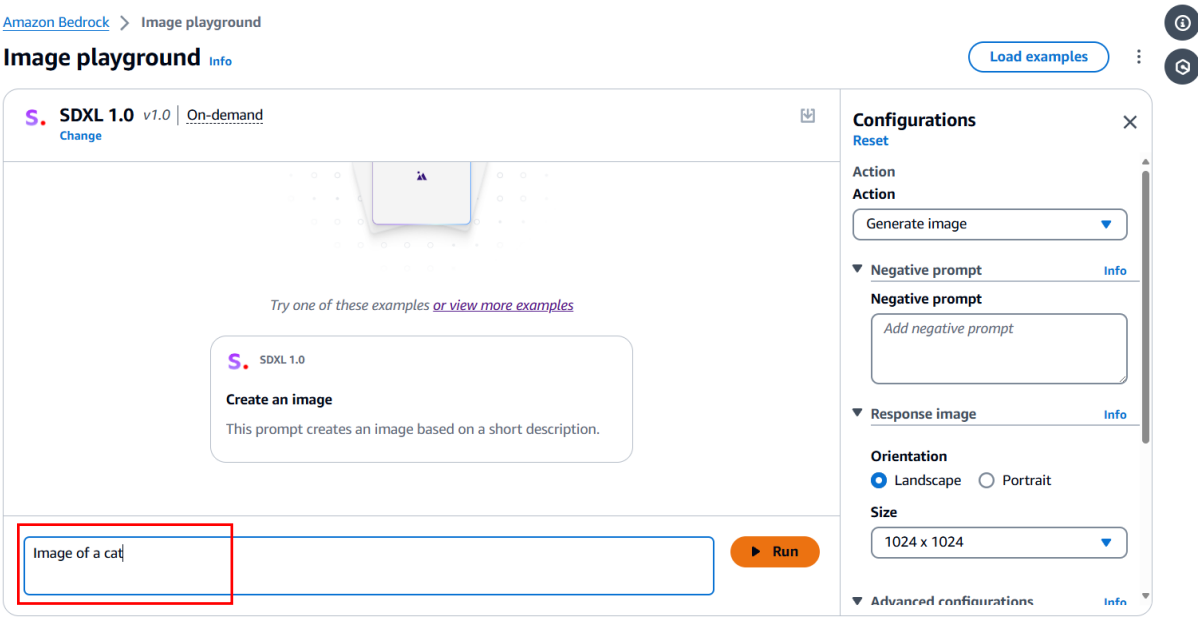
12. Then from the left pane choose image from Playgrounds. Click on Select model.



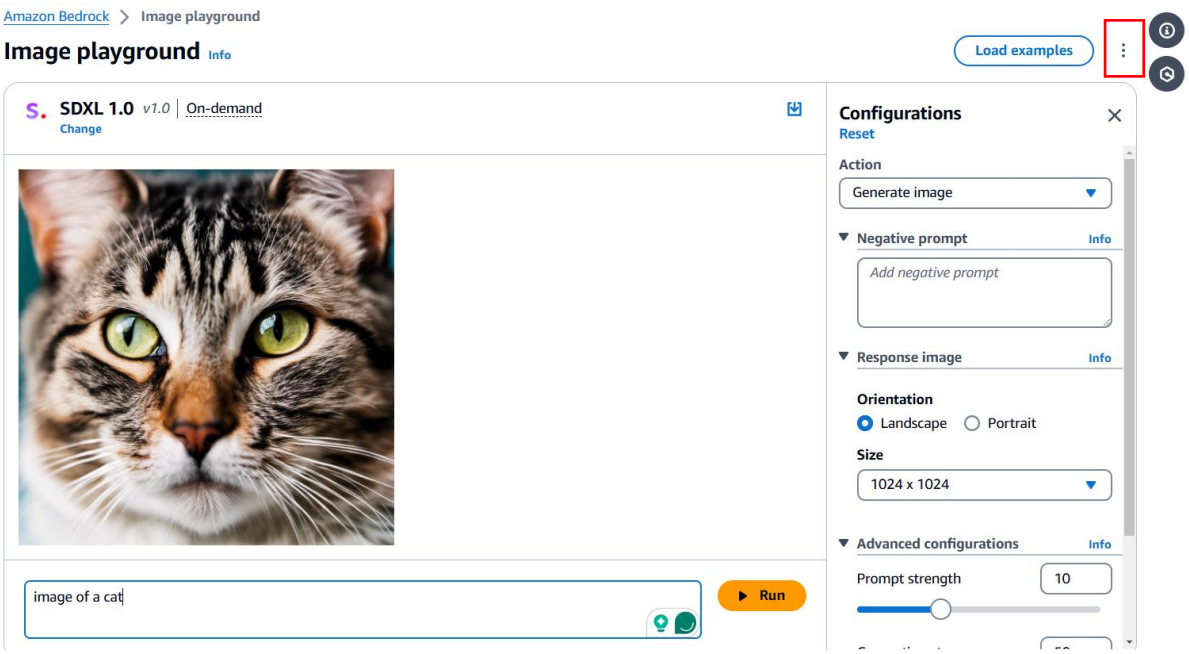
13. Choose Stability AI and the Model then click on Apply.



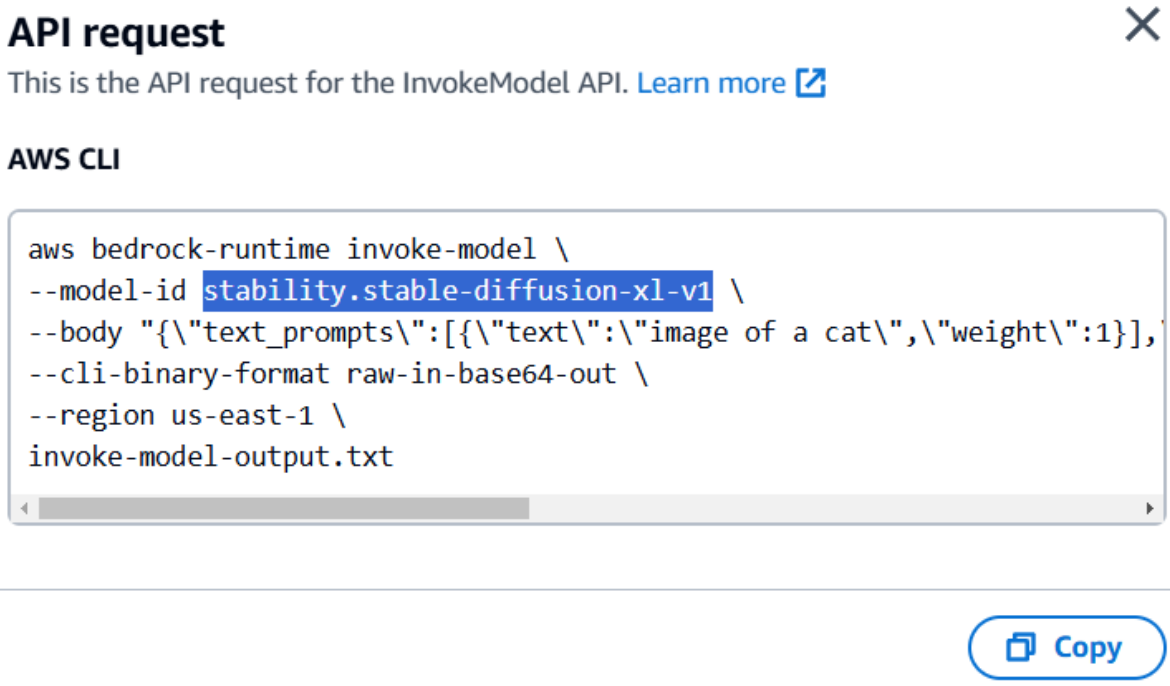
14. Then in the input text field write a text to generate an image and click on run.



15. Below you can see that we have an image of a cat generated. Now if you click on the 3 dots highlighted in the snapshot and click on view API request.



16. From there you need to copy the Model ID and paste it in your lambda function.



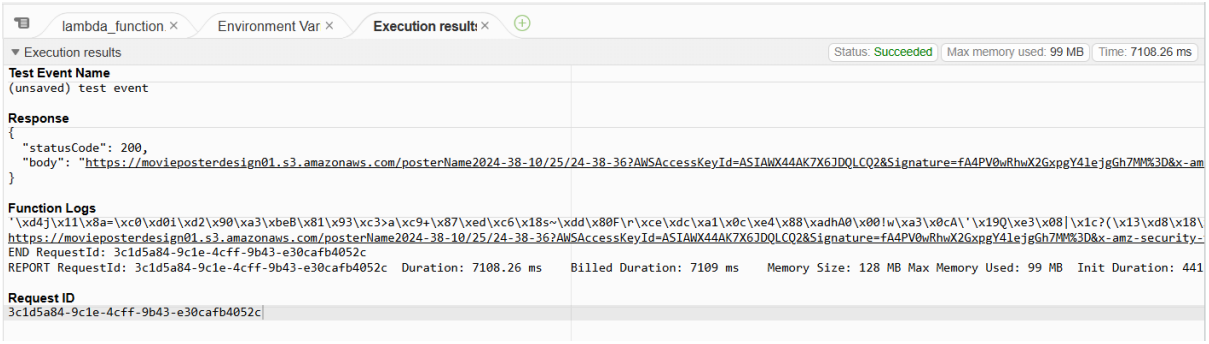
17. Then in line number 33 and 38 you need to write your bucket name here and click on deploy code.

```
31
32      response_s3=client_s3.put_object(
33          Bucket='movies-poster-design-demo-1',
34          Body=response_bedrock_finalimage,
35          Key=poster_name)
```

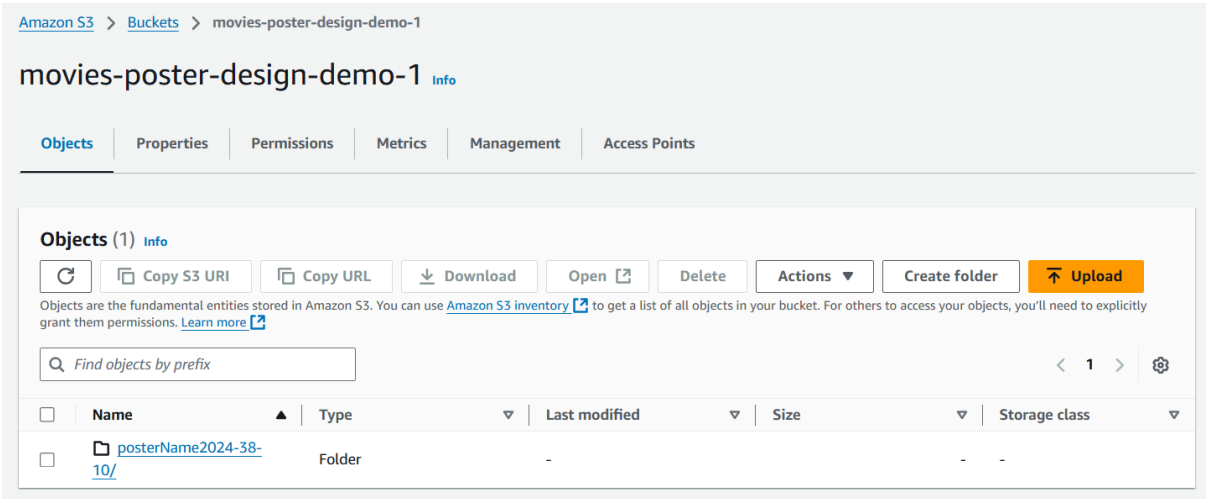
18. After that you need to create a new test event and in the event JSON area you need to write this. Click on save to save your test then click on test.

```
{
  "prompt": "image of a dog"
}
```

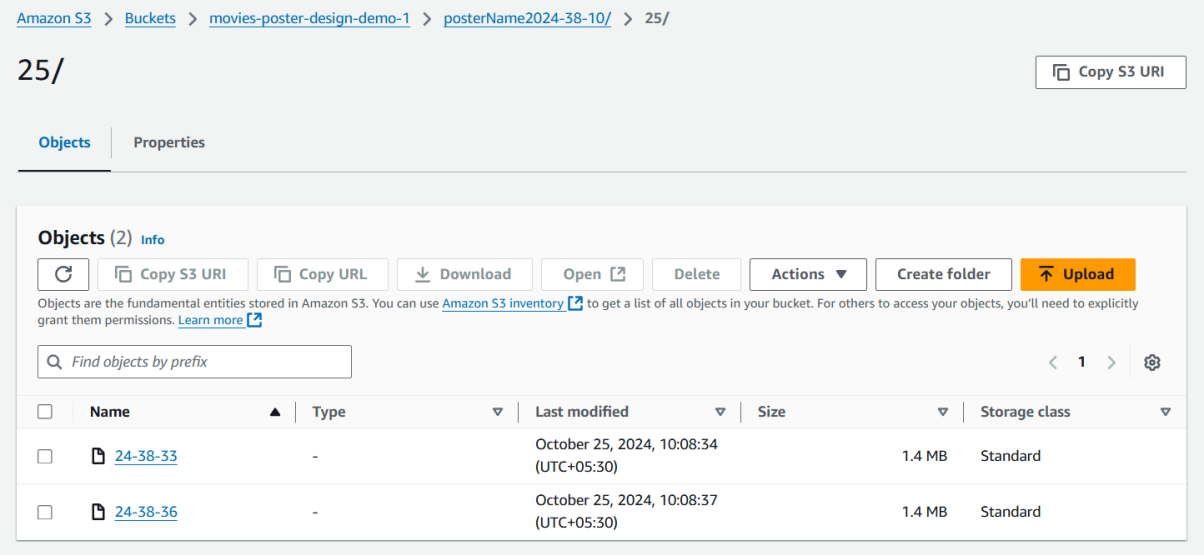
19. Here you can see that we got status code 200 which means that our image has been generated and it has been stored in our S3 bucket.



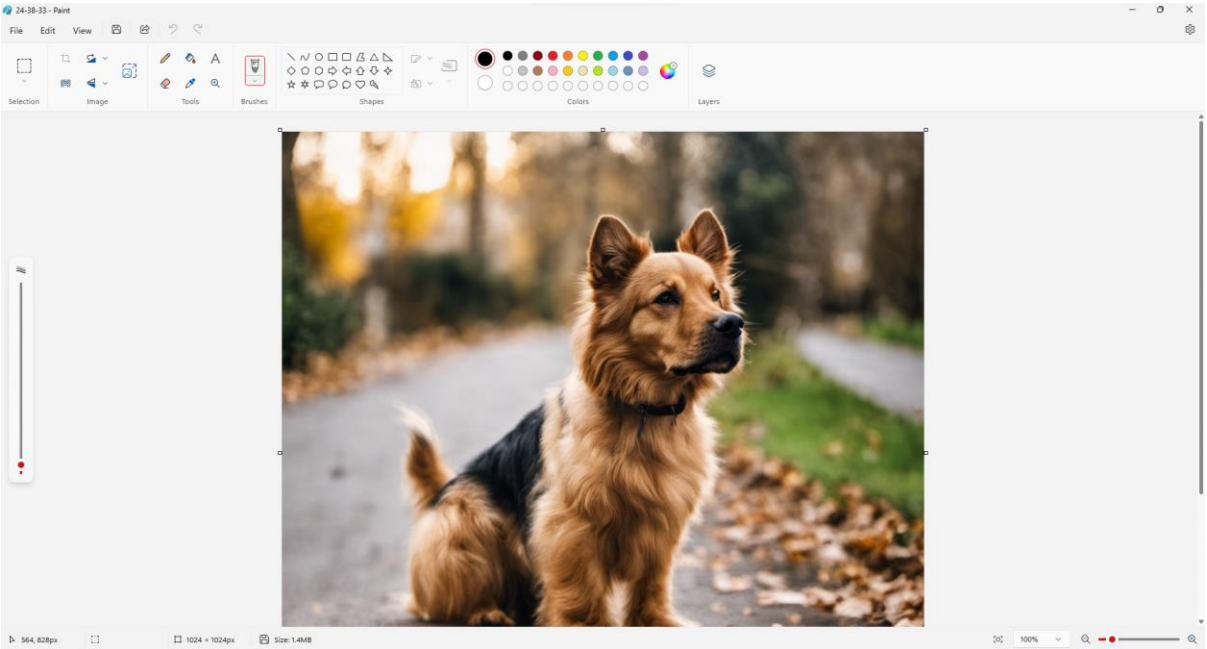
20. Now if go back to my S3 bucket and there I will see a folder.



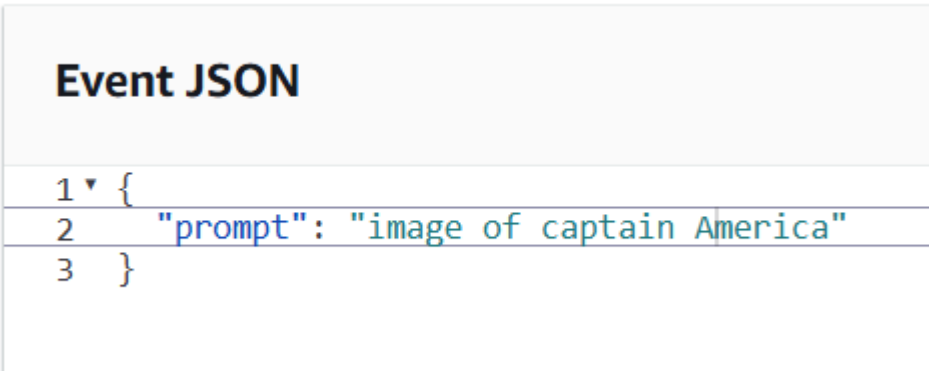
21. Now if we go inside this folder, we will see some objects download one of the object.



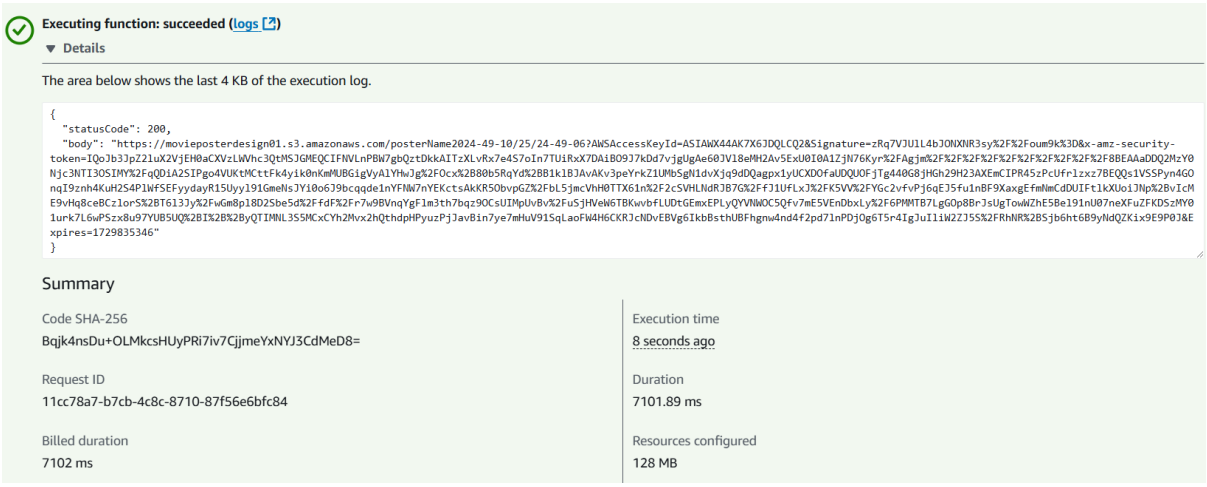
22. Now what you need to do is, open this downloaded object using Paint on your laptop and you will see an image of a dog.
23. This was the prompt we had given while saving our test in the lambda function.
24. You can also try out a different prompt and test it out using your lambda function.



25. Also, in our code we have defined it to generate a pre-signed URL. So, we will change the prompt and try to test it again.
26. As you can see we have changed the prompt and saved it then clicked on test.

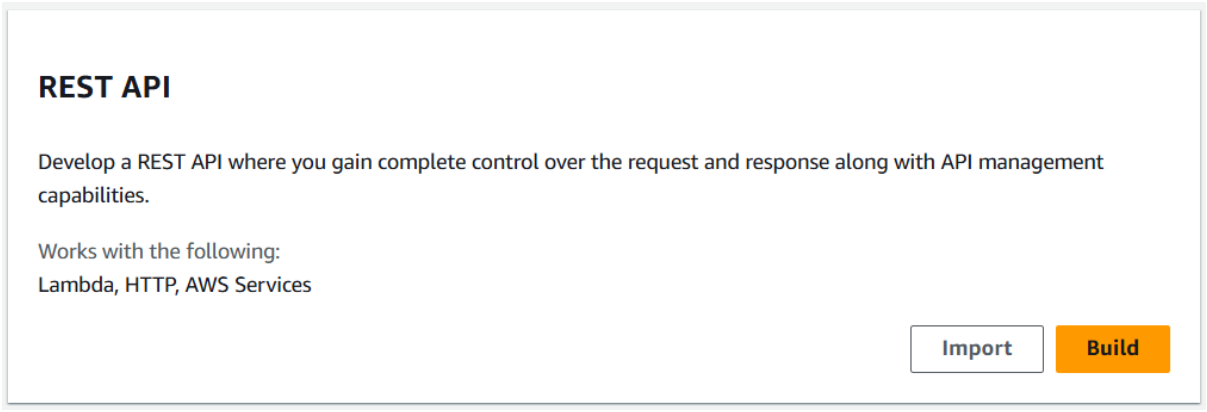


27. Below you can see that our request was successful and from the body if you copy the URL and paste it in a new tab then a file will be downloaded.

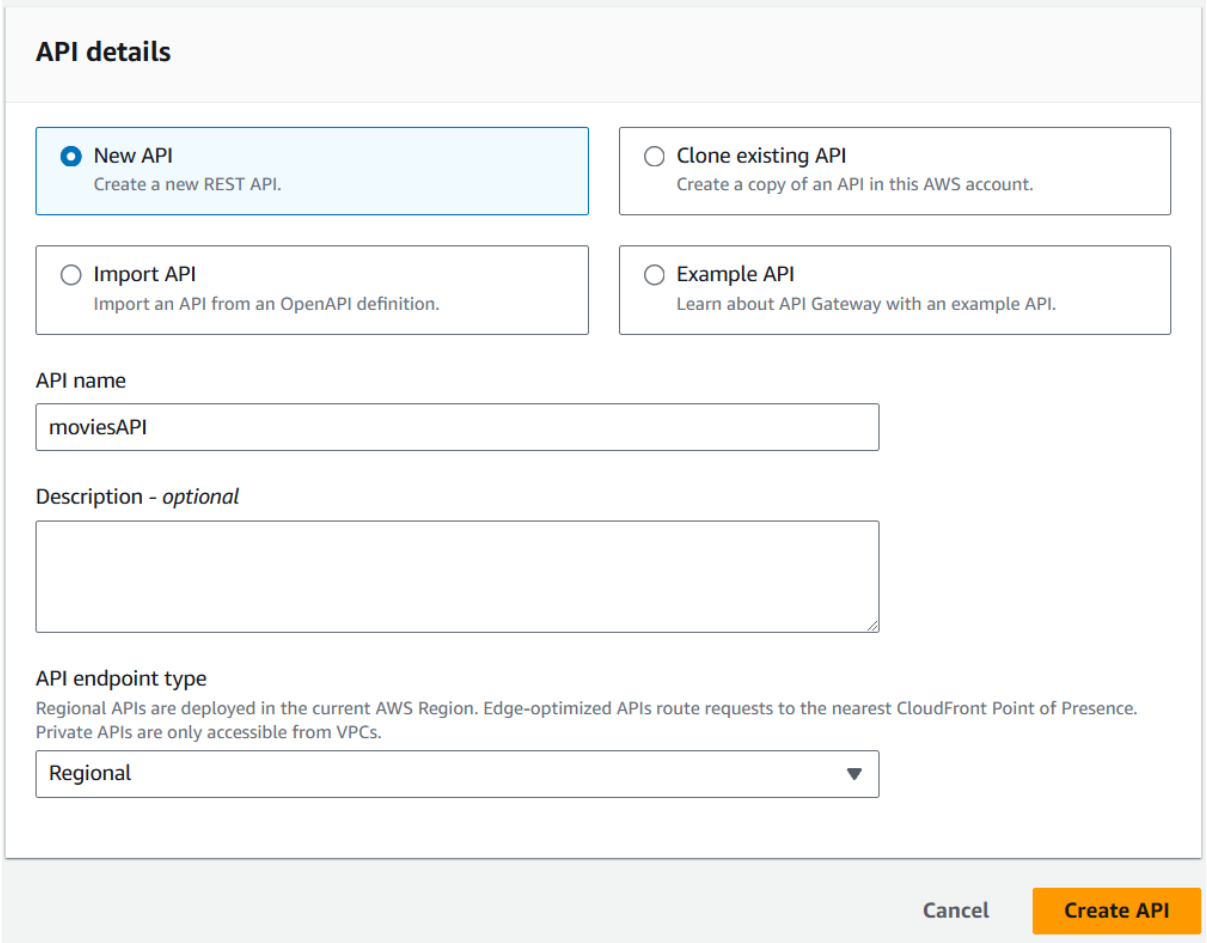


28. If you open that file in your Paint then you will see an image.

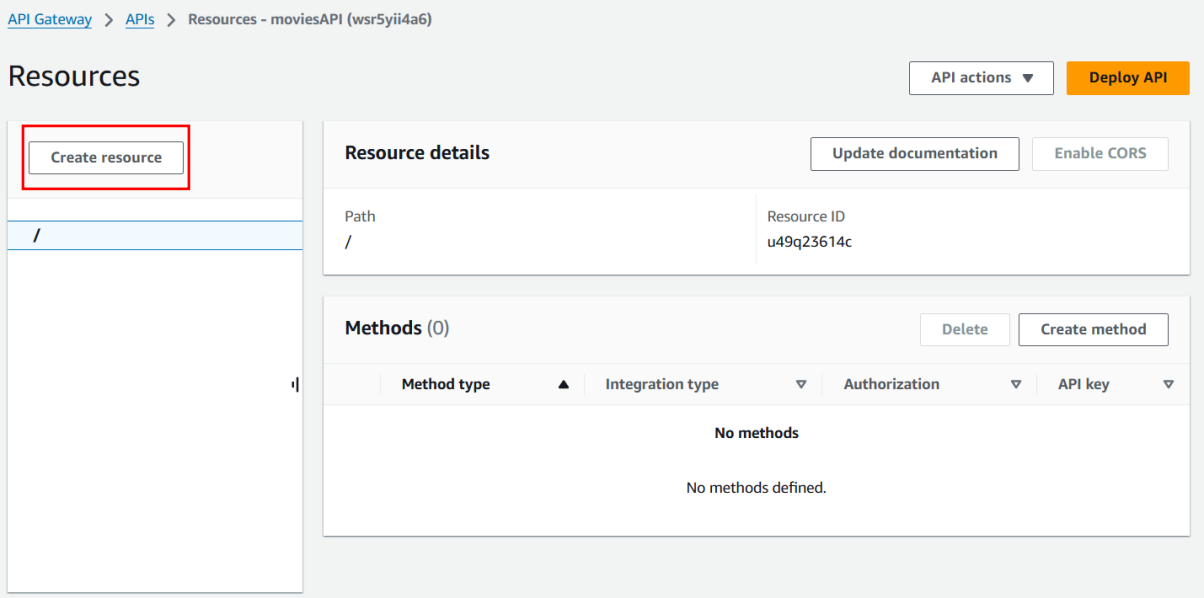
29. Now you need to search for API gateway in your console and navigate to it. Then you need to click on build.

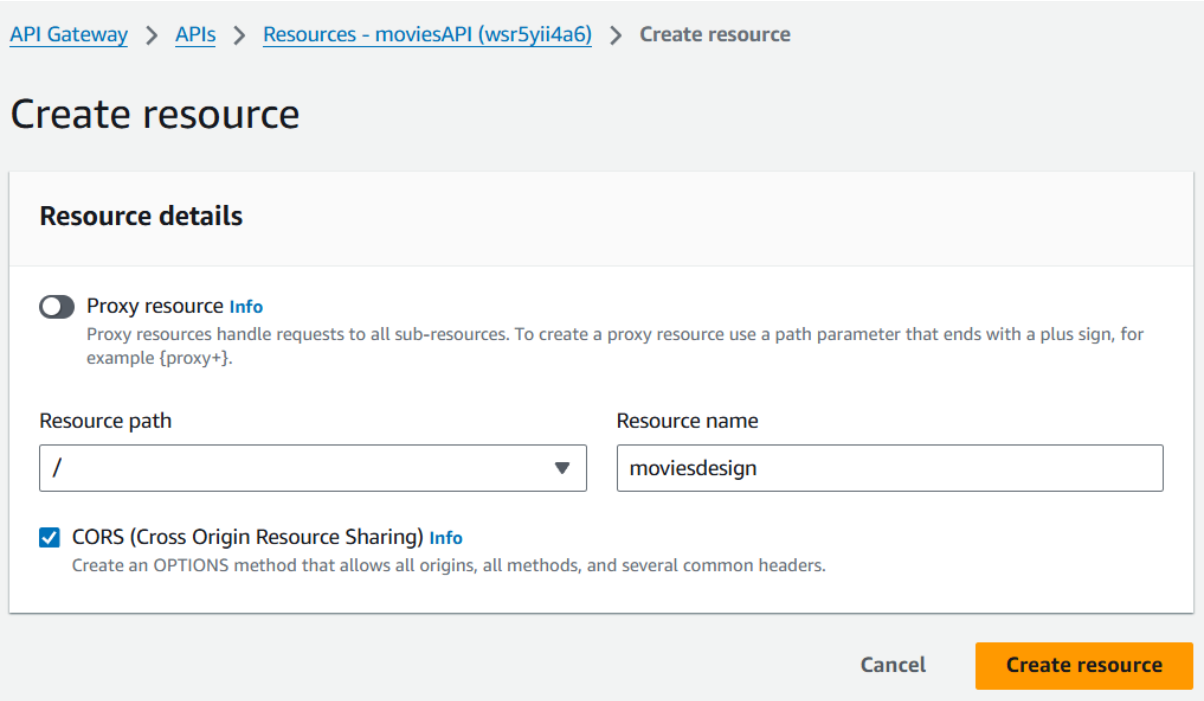


30. Here you just need to give it a name and click on Create API.

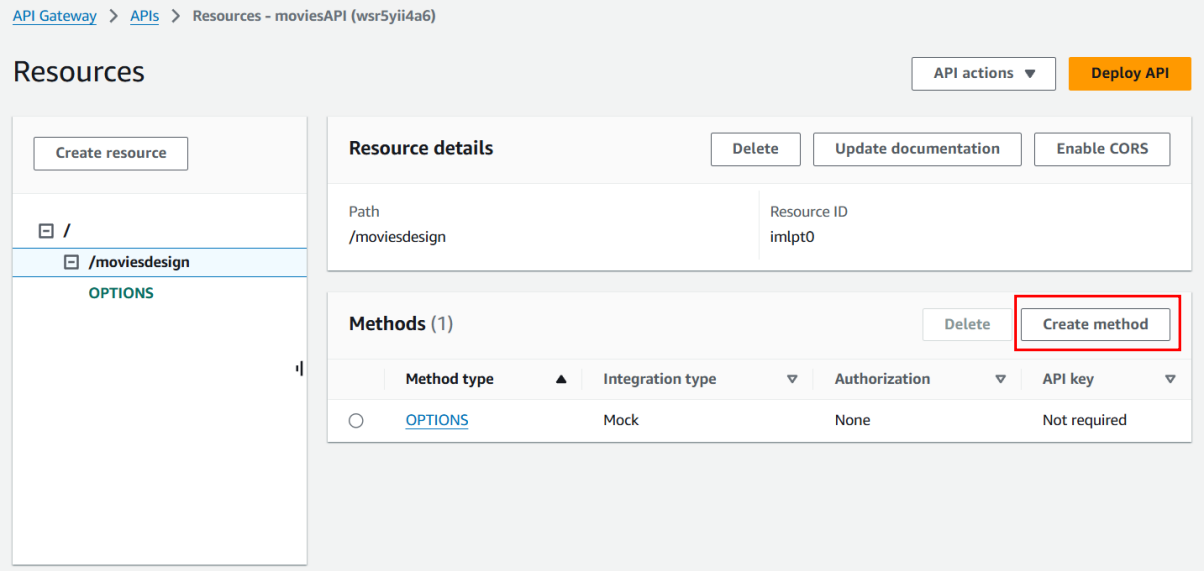


31. First, we need to create a resource. Give a name to your resource and enable CORS.

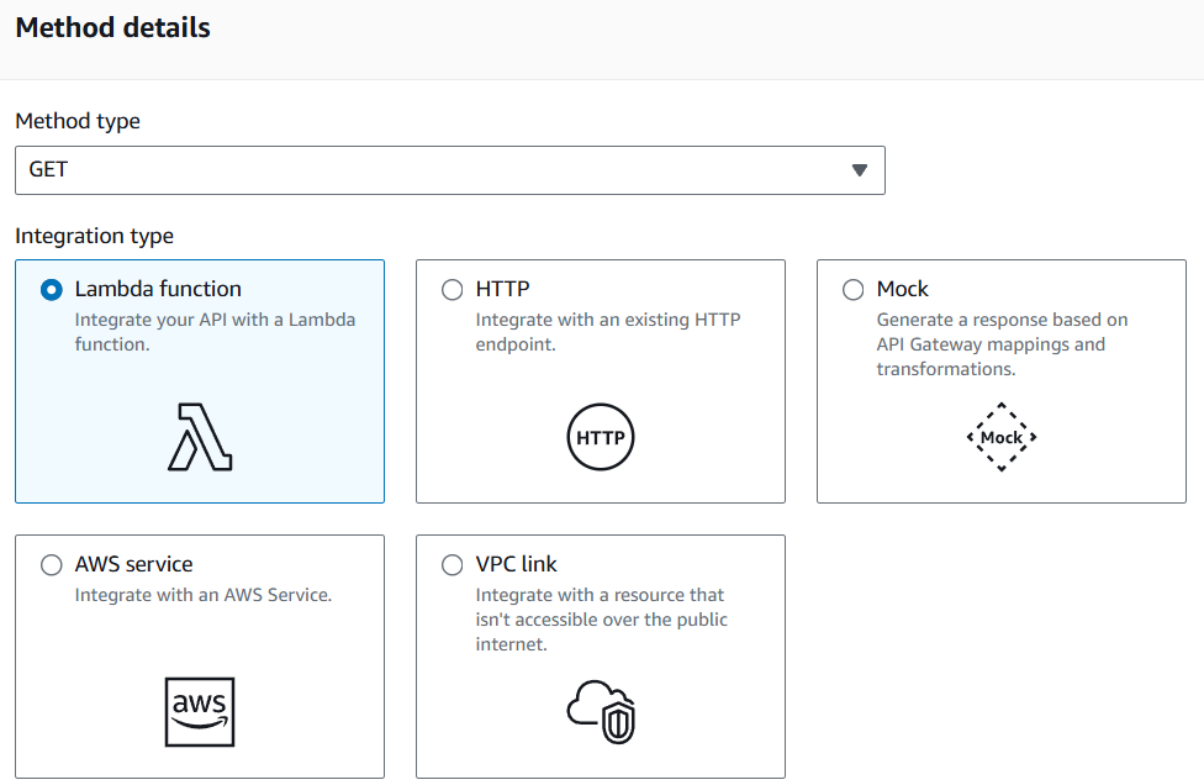




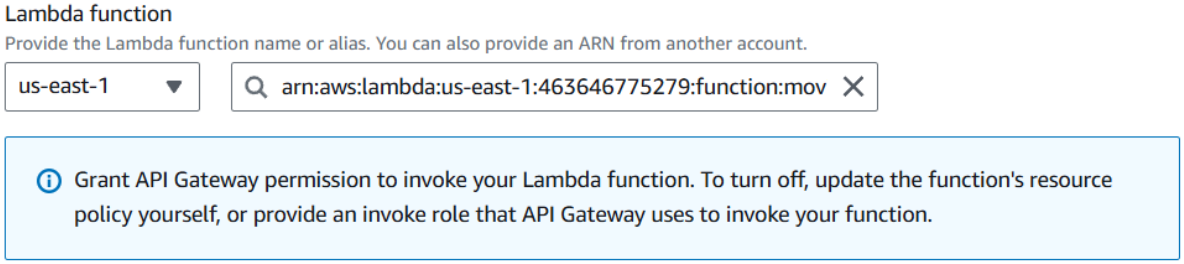
32. Then you need to select your resource and you need to click on create method.



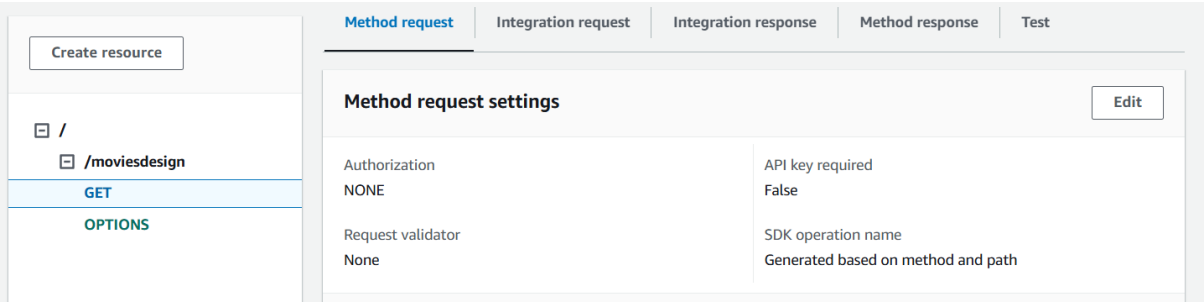
33. Choose GET method to integrate it with your lambda function.



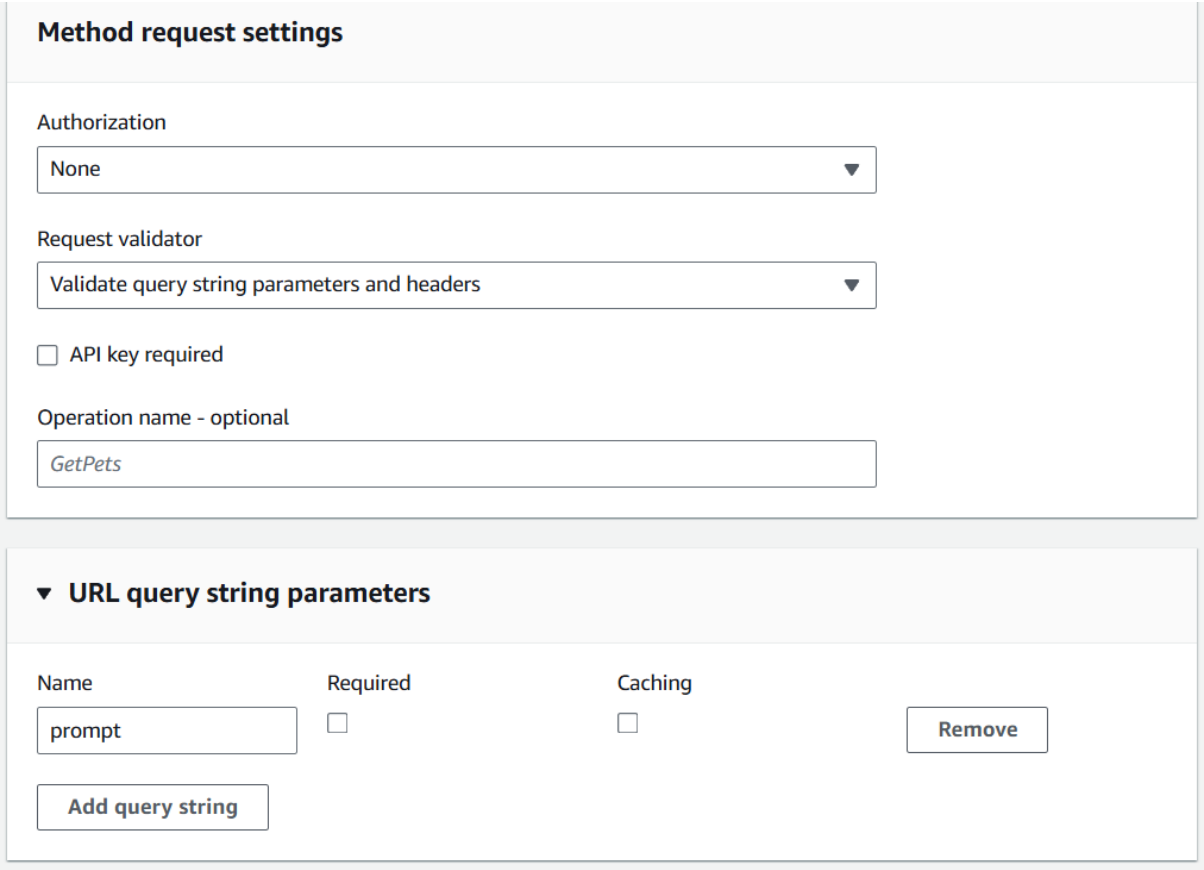
34. And choose your lambda function and click on create method.



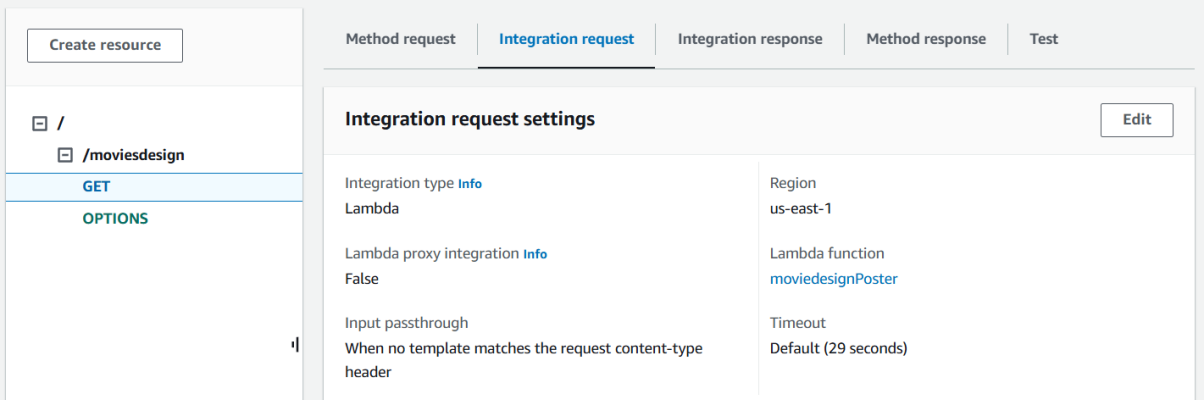
35. In your GET method go to method request and click on edit.



36. You need to change the request validator to what shown in the snapshot and expand the URL query string parameters and in the name write prompt as shown below then click on save.

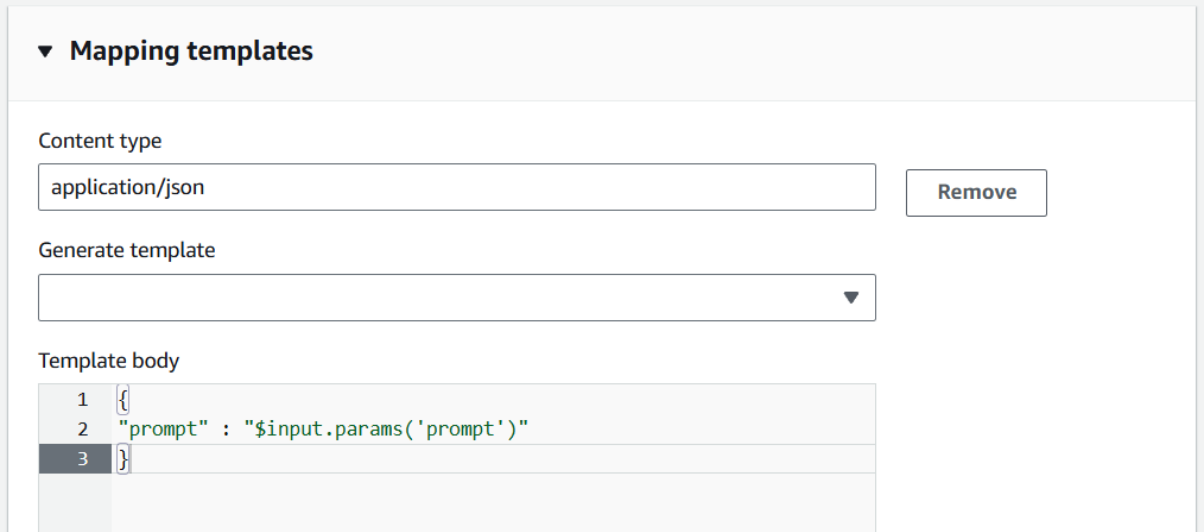


37. Then go to integration request in your GET method and click on edit.

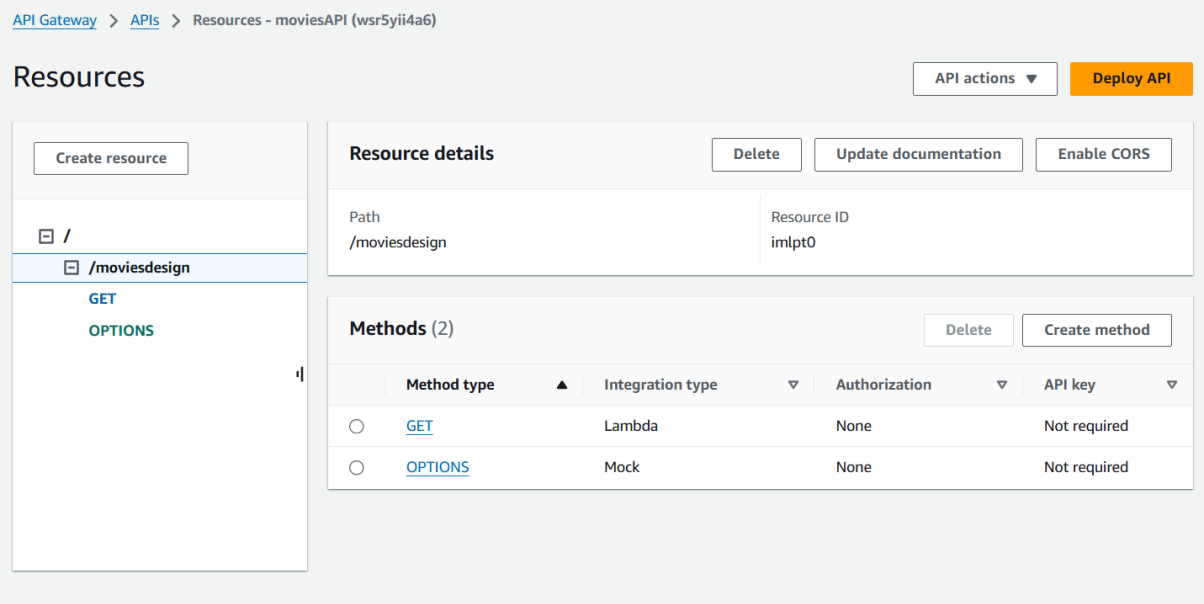


38. Scroll down to the bottom and expand Mapping templates then choose to add a template. In the content type and template body write the same. Click on save.

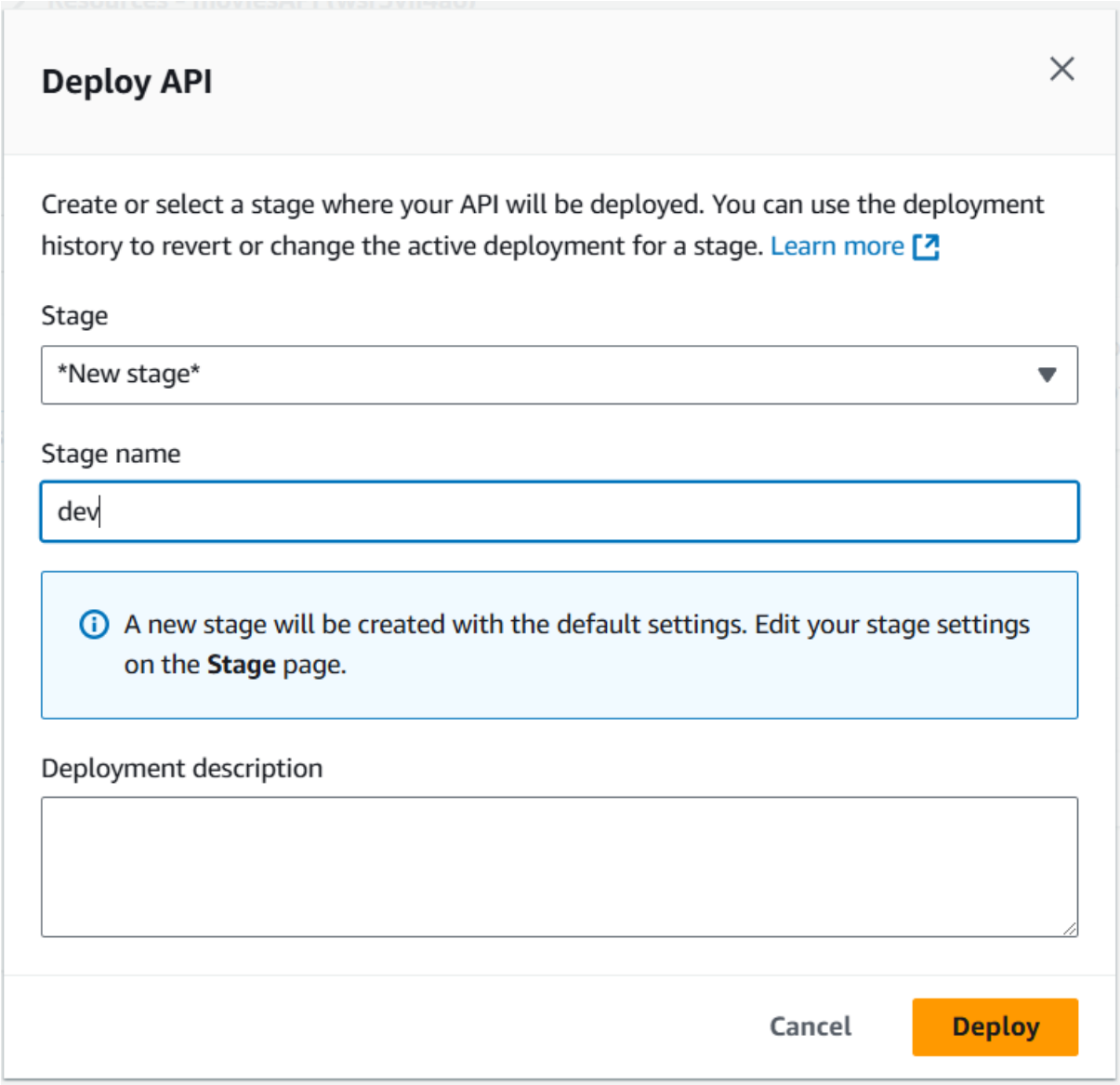
    {
    "prompt" : "$input.params('prompt')"
    }



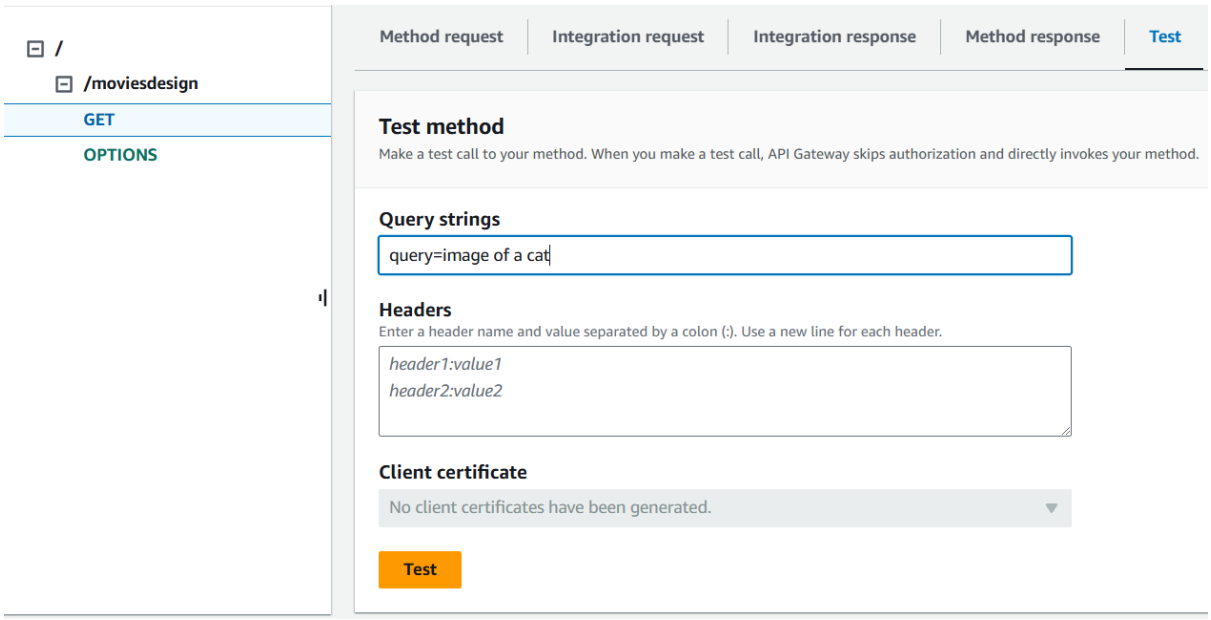
39. After that you need to deploy your API, for that you need to click on resource and from here choose to deploy you API.



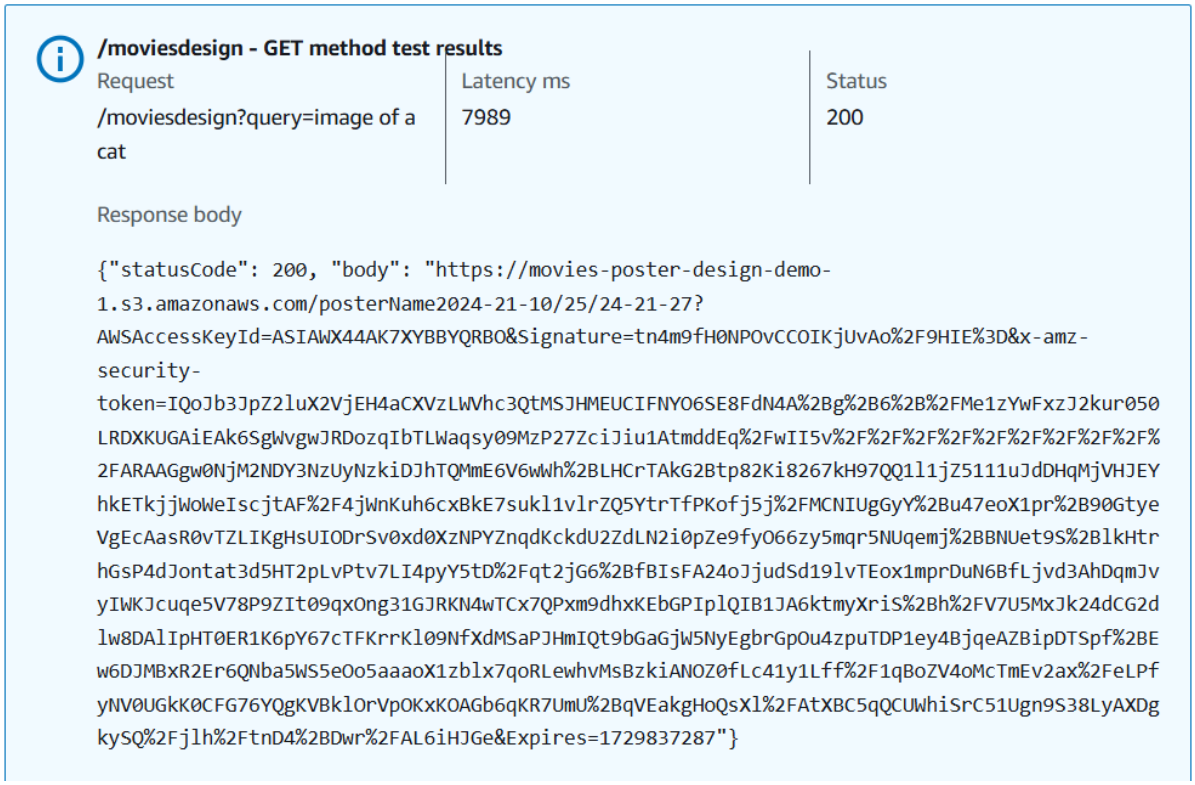
40. Choose a new stage give it a name and then click on deploy.



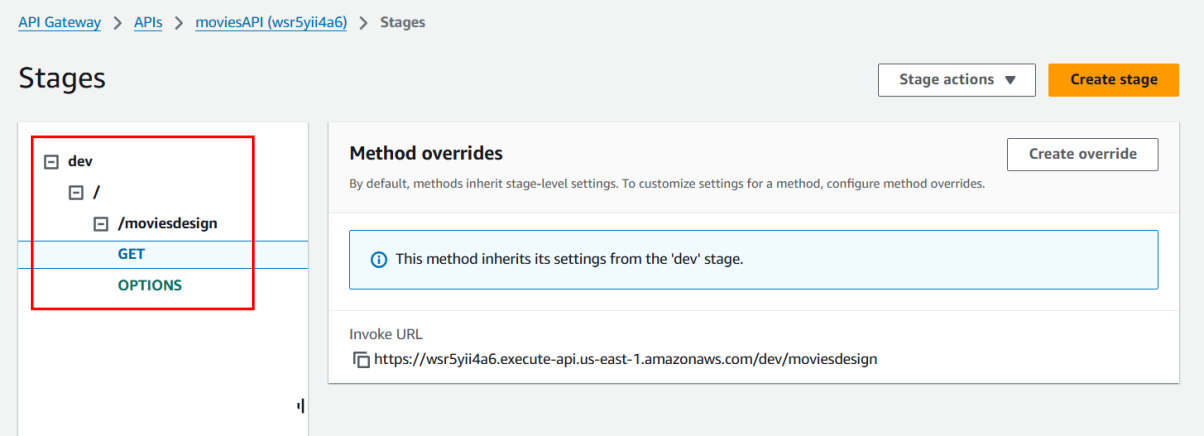
41. Once it is deployed now, we are going to test it out, so go to the GET method and this time choose Test and, in the query, strings give the same query as you can see below. Then click on test.



42. Here you can see that we get the response with status code 200 which means that it succeeded. Also, if you copy the URL and paste it in a new tab a file will be downloaded. Open this file in paint you will see an image of a cat.

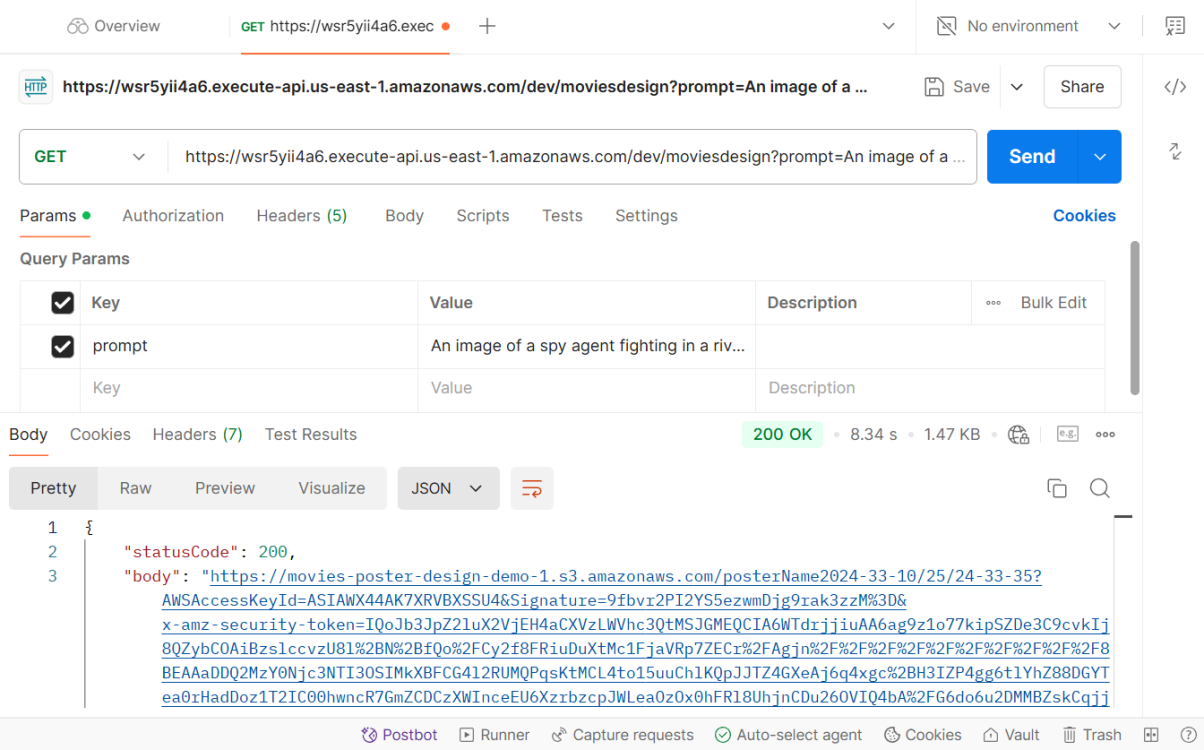


43. Now in your API Gateway from the left pane choose stages. Then you need to expand your dev stage and go to GET method then copy the invoke URL.

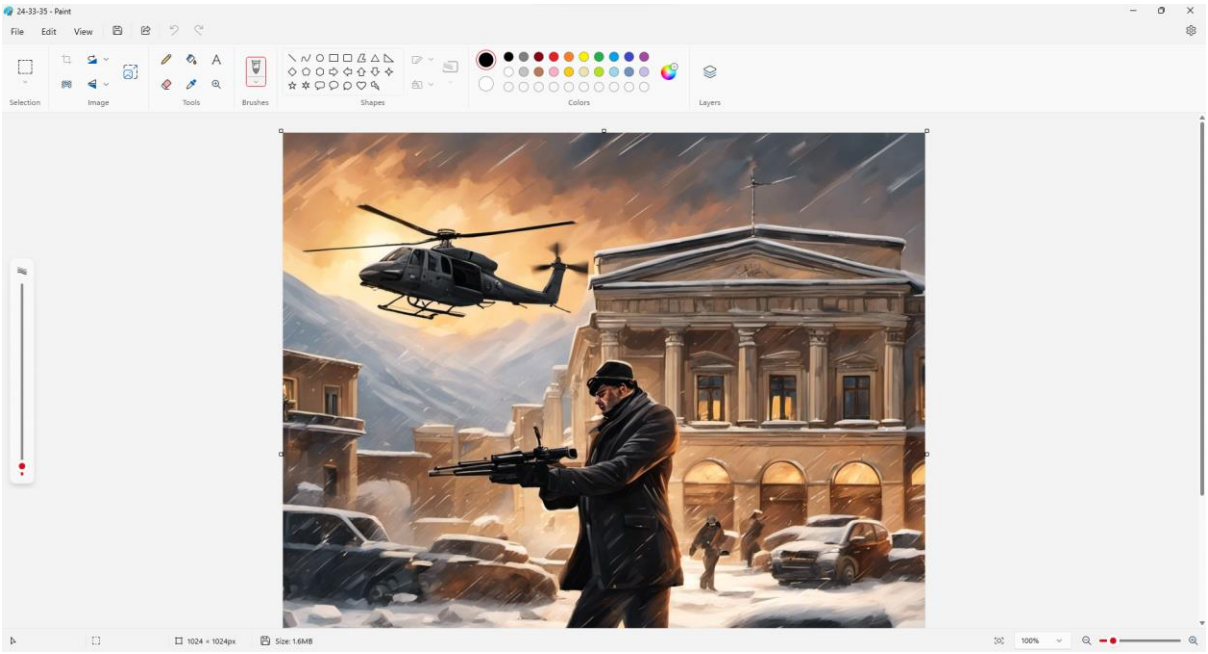


44. In your laptop you need to open the Post Man tool. In the tool create a workspace and choose GET method, then paste the invoke URL you copied from API gateway here.
45. After that in the key write prompt and, in the prompt, you need write this sentence given below. Click on Send.
46. Then you will get the status code 200 and the URL. Copy the URL and paste it in the browser then a file will be downloaded.
47. Open that file in Paint app in your laptop.

"An image of a spy agent fighting in a rival country with guns and helicopters with backdrop of a shopping complex with heavy snow and old Greek architecture building late in the evening with sunsetting behind mountains. The iamge should be a photograph with Aaron Jasinski style"



48. In the paint app you will see the image.


24-33-35 - Paint
File
Edit
View
Selection
Image
Tools
Brushes
Shapes
Colors
Layers
1024 × 1024px
Size: 1.6MB
100%