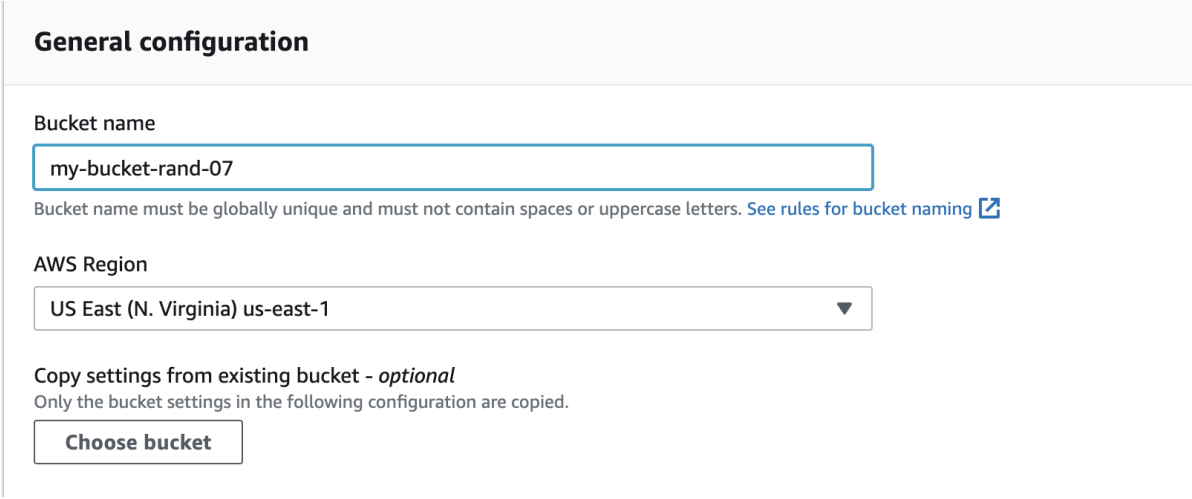
Create an API endpoint through API Gateway which calls an AWS lambda that will pull JSON data from an S3 bucket

1. Create an S3 Bucket:

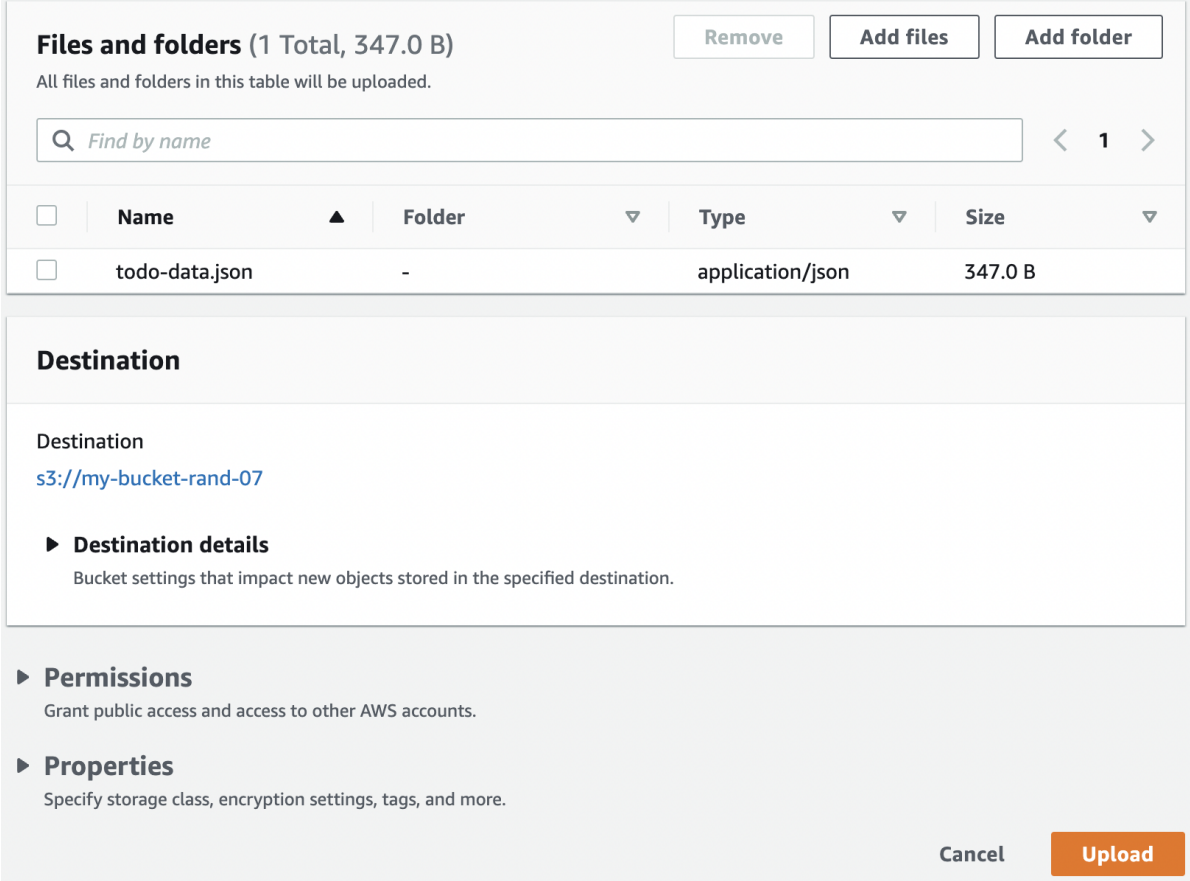
- Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
- On the S3 page, click on the "Create bucket" button.
- Provide a unique name for the bucket, select a region, and leave the other settings as default.
- Click "Create bucket" to complete the process.
- This bucket will be used to store your JSON data.



2. Upload JSON Data:

- After you have created the S3 bucket, click on the bucket to open it.

- Then, click on the "**Upload**" button.
- Select the JSON file you want to upload.
- Click on the “**Upload**” button at the bottom to start the upload.

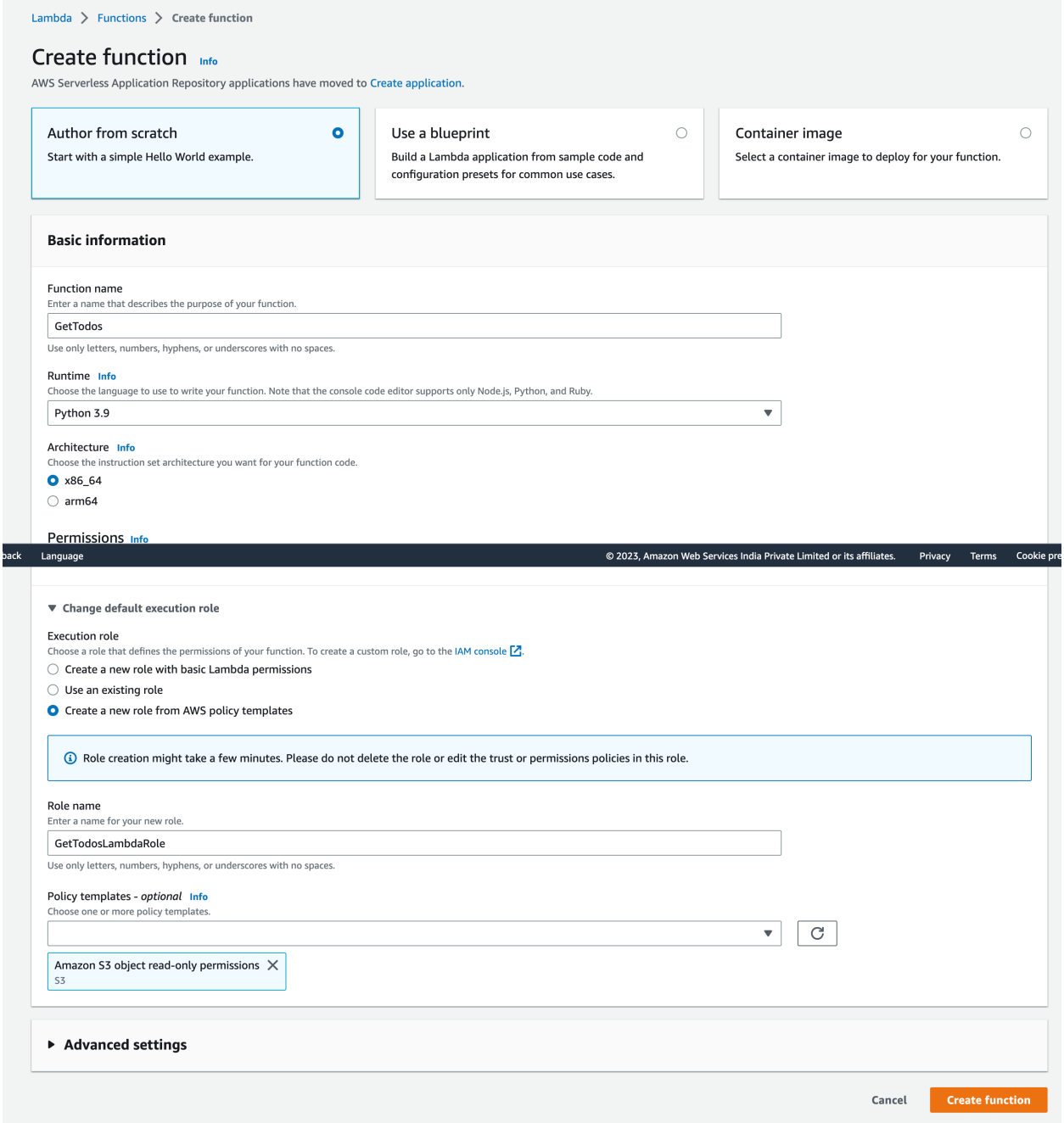


- Once the file is uploaded you will see a success screen. Press the “**Close**” button at the top right
- You can now see the JSON file in your S3 bucket.

3. Create a Lambda Function:

- Open the Lambda console
- On the Lambda page, click on the "Create function" button.
- Select the "Author from scratch" option.

- Give your function a name (eg. GetTodos), select a runtime (e.g. Python)
- Expand the `Change Default Execution` dropdown, Select **Create a new role from AWS policy templates**, Assign a role name (eg. GetTodosLambdaRole) and from the policy templates select (search for) **Amazon S3 object read-only permissions.**
- Click on **Create Function**



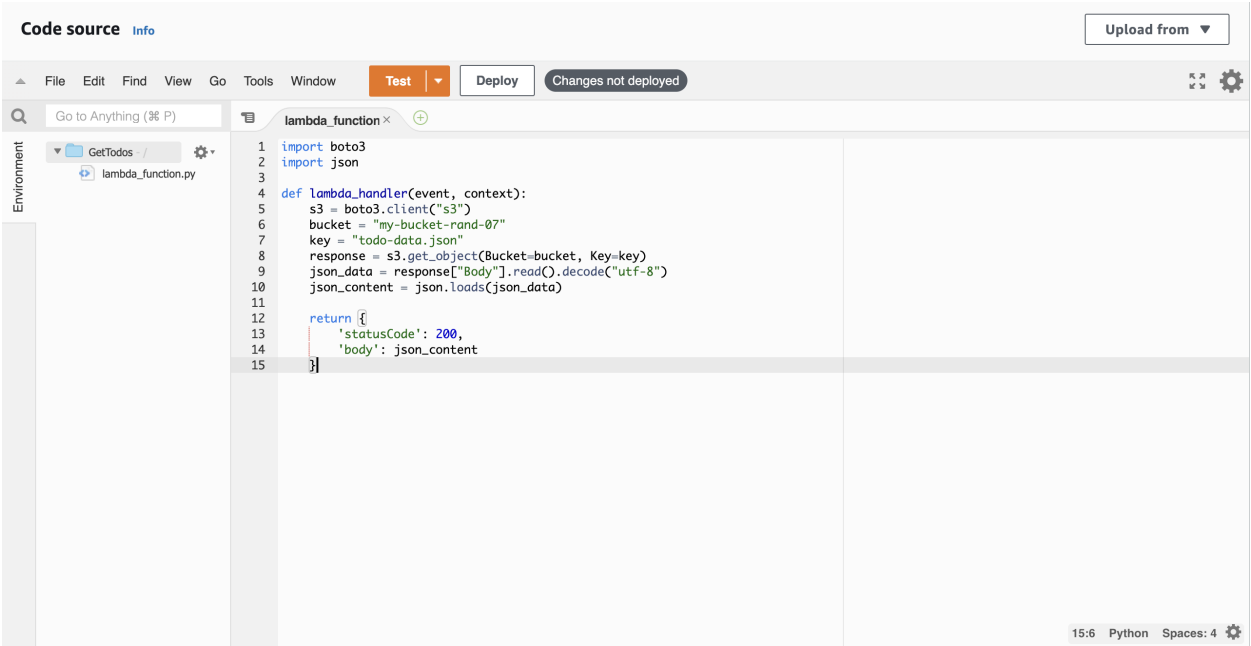
- This function will be used to retrieve the JSON data from the S3 bucket and return it to the API endpoint.

4. Add Code to the Lambda Function:

- In the Lambda function editor, scroll to the bottom where you will find a code window with a file with a name like `lambda_function.py` open
- paste the following code, which will retrieve the JSON data from the S3 bucket:

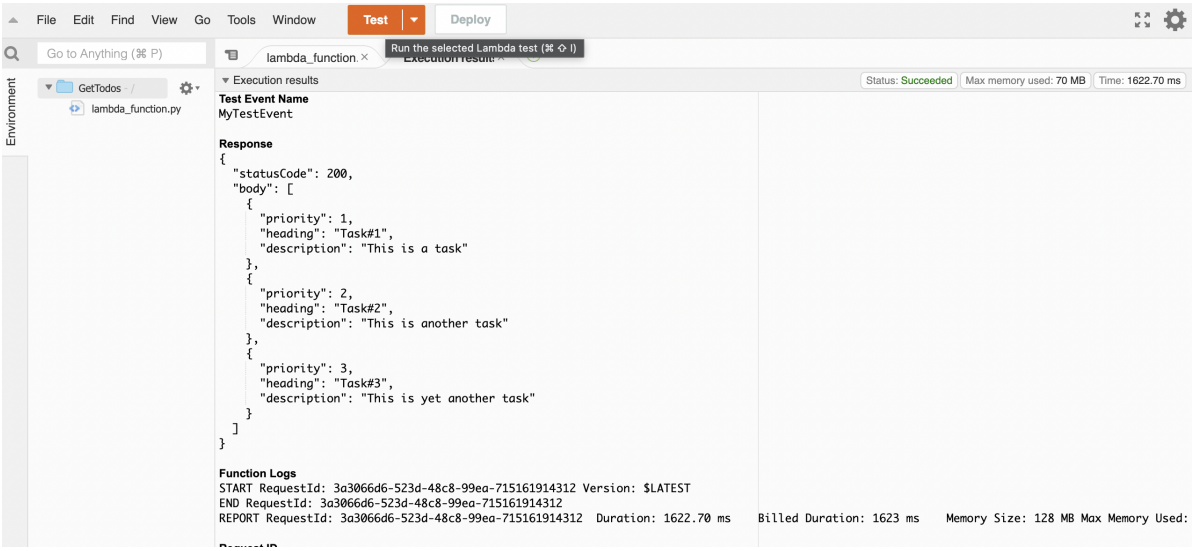
```
import boto3
import json

def lambda_handler(event, context):
    s3 = boto3.client("s3")
    bucket = "your-bucket-name"
    key = "your-json-file-name.json"
    response = s3.get_object(Bucket=bucket, Key=key)
    json_data = response["Body"].read().decode("utf-8")
    json_content = json.loads(json_data)

    return {
        'statusCode': 200,
        'body': json_content
    }
```

- Replace "your-bucket-name" with the actual name of your S3 bucket and "your-json-file-name.json" with the actual name of the JSON file you uploaded.
- Save the changes by clicking on the **“File”** → **"Save"** button.
- Finally, click on “**Deploy**”



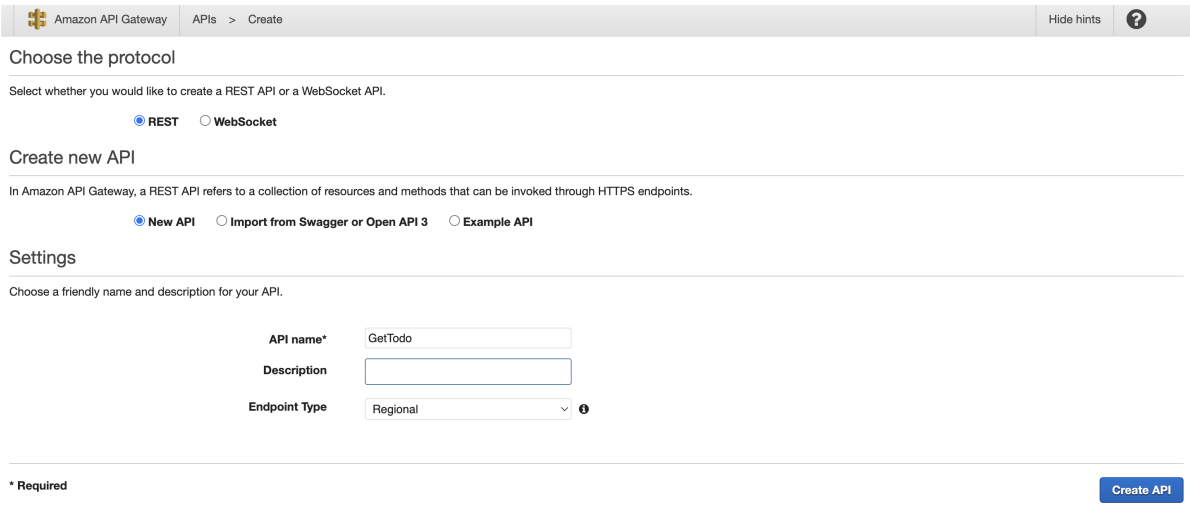
4. Test the Lambda Function:

- Click on the **Test** button,
- give a name to the test event (e.g. MyTestEvent), leave others setting as default and save the test event
- Click on the **Test** button again, and select the event you just created.
- The result will be shown in the execution tab, near your code window. If the status says succeeded, your lambda function works as expected.



5. Create an API Gateway:

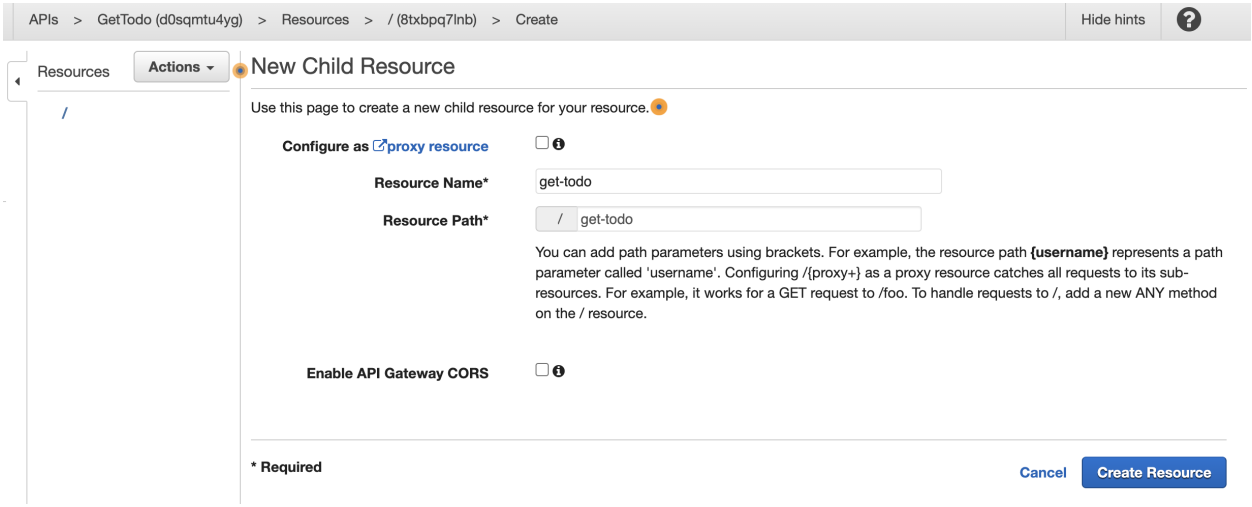
- Sign in to the API Gateway console
- If this is your first time using API Gateway, you see a page that introduces you to the features of the service. Under **REST API**, choose **Build**. When the **Create Example API** popup appears, choose **OK**.
- If this is not your first time using API Gateway, choose **Create API**. Under **REST API**, choose **Build**.
- Create an empty API as follows:
  - Under **Create new API**, choose **New API**.
  - Under **Settings**:
    - Enter the **API name**, If desired, enter a description in the **Description** field; otherwise, leave it empty.
    - Leave **Endpoint Type** set to **Regional**.
  - Choose **Create API**.



6. Create a Resource:

- Choose the root resource (***/***) in the **Resources** tree.

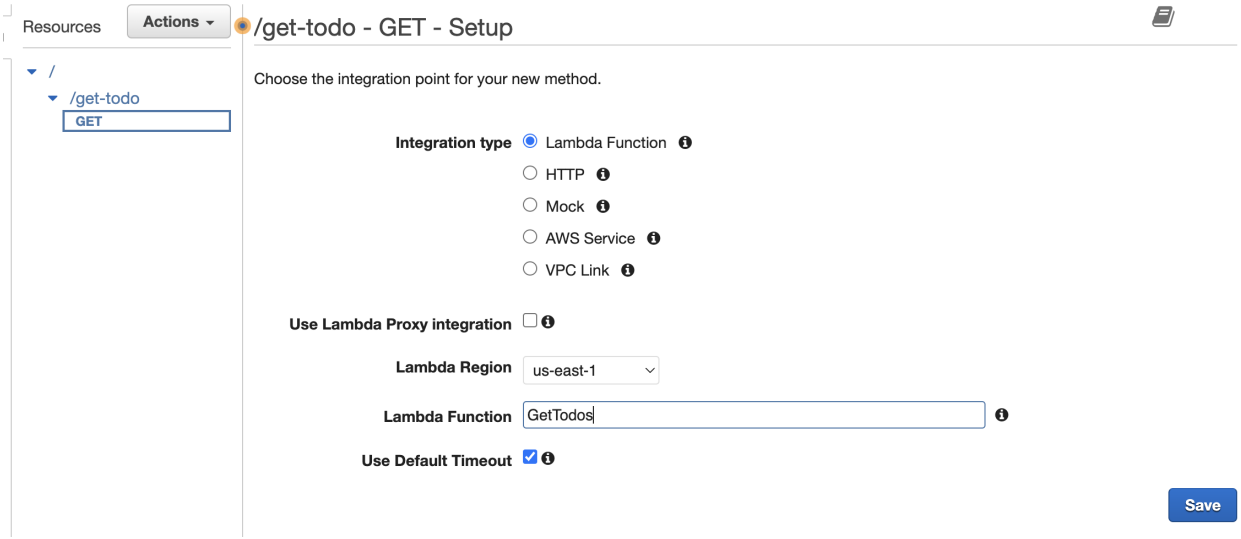
- Choose **Create Resource** from the **Actions** dropdown menu.
- Leave **Configure as proxy resource** unchecked.
- Set **Resource Name (**eg `get-todo`) and **Resource Path** (eg **/get-todo**)
- Leave **Enable API Gateway CORS** unchecked.
- Click on **Create Resource**



7. Create A method:

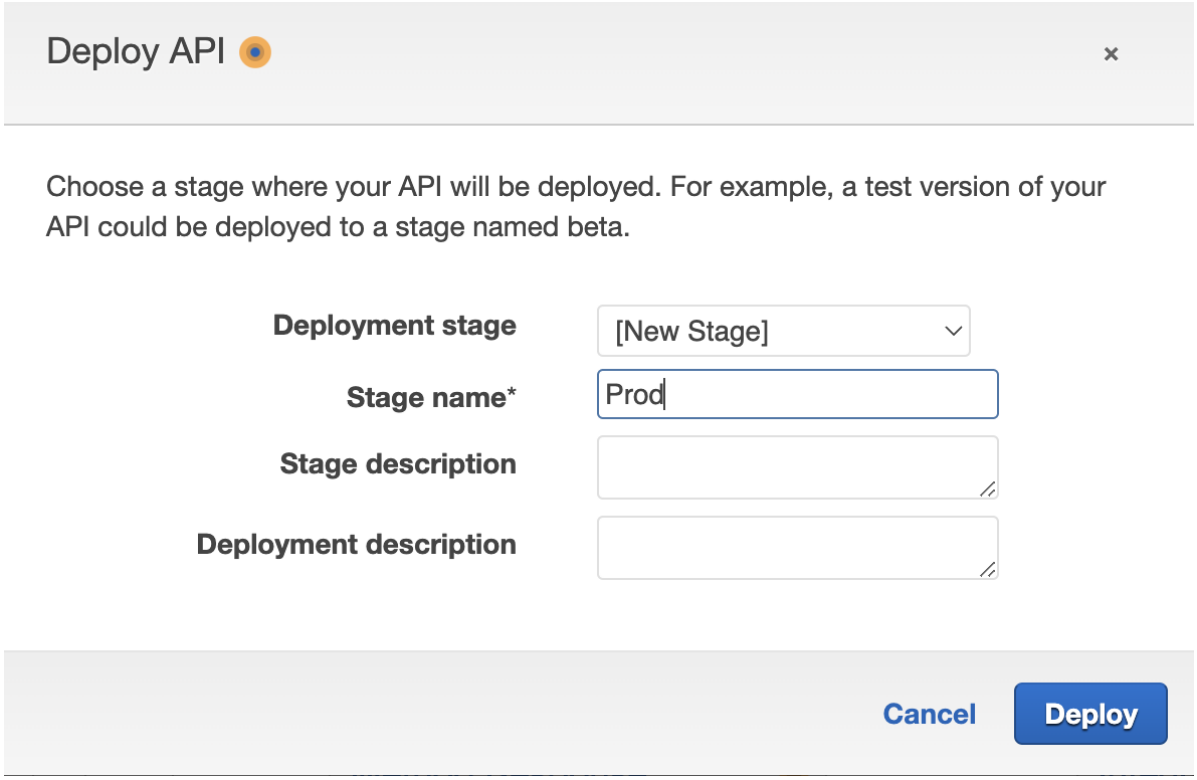
- In the **Resources** list, choose the resource you just created (eg **/get-todo**)
- In the **Actions** menu, choose **Create method**.
- Choose **GET** from the dropdown menu, and choose the checkmark icon

- Leave the **Integration type** set to **Lambda Function**.
- Choose **Use Lambda Proxy integration**.
- From the **Lambda Region** dropdown menu, choose the region where you created the Lambda function.
- In the **Lambda Function** field, type any character and choose from the dropdown menu.
- Leave **Use Default Timeout** checked.
- Choose **Save**.
- Choose **OK** when prompted with **Add Permission to Lambda Function**.



8. Deploy the API:

- After you have created the resource and method, you'll need to deploy the API.
- Click on the "Actions" menu, and then select "Deploy API".
- Choose a deployment stage (e.g. "prod"), and then click on the "Deploy" button.
- You'll now see the "Invoke URL" for your API, which you can use to access the JSON data.



9. Test the API:

- Open a web browser, and paste the "Invoke URL" + “ResourceName” for your API into the address bar.
  - Eg: If Invoke URL is `https://d0sqmtu4yg.execute-api.us-east-1.amazonaws.com/prod` and Resource name is `/get-todo`, paste `https://d0sqmtu4yg.execute-api.us-east-1.amazonaws.com/prod/get-todo`
- You should see the JSON response similar to what we saw during the lambda testing