

ENGR-E 534 BIG DATA APPLICATIONS

Assignment 8

Shalini Kothuru (University ID 2001096463)

Note : I have attached dataset, notebook file - html and ipynb, index.html file and lambda function – python file. Index.html file and Lambda function varies from the screenshot shown in this document as I have updated while working on web app but forgot to take screenshot.

Architecture Design:

Data source: I have downloaded data from **Kaggle**. Dataset has 1025 records with 14 columns including target variable(0 = no disease and 1 = disease) for heart disease.

Data storage: I have uploaded the dataset to **S3**. S3 also store the models once trained.

Data preparation: I have created **notebook instance in sagemaker** to explore, preprocess, prepare data and train model.

Model building and training: Sagemaker has built-in algorithms. I have used **xgboost and linear learner built-in model** for my binary classification problem. The sagemaker training jobs use the dataset stored in S3 to train the model and the trained model back in S3 again.

Model deployment: the trained models both xgboost and linear learner are deployed to **sagemaker endpoints**. These deployed models will be used by lambda function to predict the target variable

API and application layer: **AWS Lambda** hosts the backend code in python file which makes call to sagemaker endpoints. It processes the incoming requests from app and sends them to model in sagemaker. **API gateway** is used to trigger lambda function on http requests.

Frontend web application: **S3** is used to the index.html file.

Workflow:

The user enters details basically feature values in web application hosted on S3. The web application sends a post request to API gateway with the details entered which routes the request to lambda function. Now lambda function processes the request and invokes sagemaker endpoint and sends the prediction result back to API gateway which in turn returns to web app. The web application displays the result.

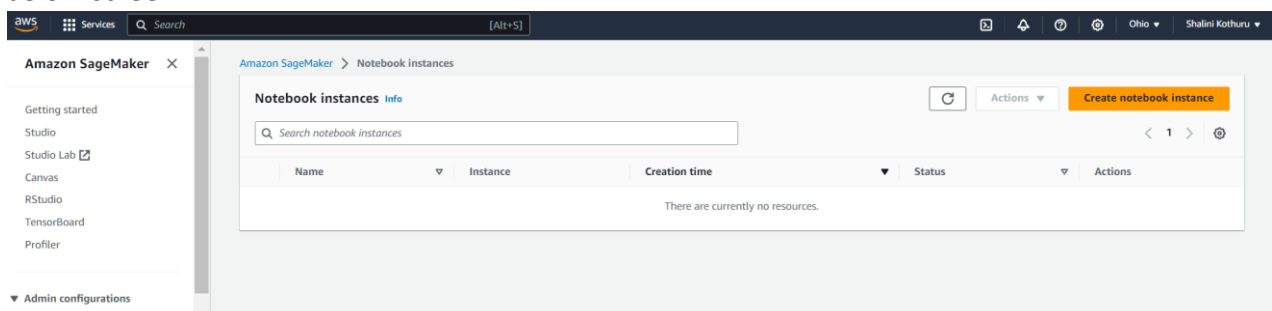
Configuration Details:

I have explained configuration details step by step using screenshots.

AWS Sagemaker for ML project:

Login in to AWS management console and search for sagemaker. Open Sagemaker service.

In left side menu, you will find notebook, click on notebook instance under notebook which takes you to below screen



Click on create notebook instance. Enter details as below

Services

Search

[Alt+S]

Amazon SageMaker

>

Notebook instances

>

Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. [Learn more](#)

Notebook instance settings

Notebook instance name

Assignment8

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type

ml.t2.medium

Elastic Inference

[Learn more](#)

none

Platform identifier

[Learn more](#)

Amazon Linux 2, Jupyter Lab 3

Additional configuration

Next, click on create role.

Create an IAM role

X

Passing an IAM role gives Amazon SageMaker permission to perform actions in other AWS services on your behalf. Creating a role here will grant permissions described by the [AmazonSageMakerFullAccess](#) IAM policy to the role you create.

The IAM role you create will provide access to:

☒ S3 buckets you specify - optional

☒ Any S3 bucket

Allow users that have access to your notebook instance access to any bucket and its contents in your account.

☐ Specific S3 buckets

Example: bucket-name-1, bucket-name-2, bi

Comma delimited. ARNs, ** and */ are not supported.

☐ None

☒ Any S3 bucket with "sagemaker" in the name

☒ Any S3 object with "sagemaker" in the name

☒ Any S3 object with the tag "sagemaker" and value "true"

[See Object tagging](#)

☒ S3 bucket with a Bucket Policy allowing access to SageMaker

[See S3 bucket policies](#)

Cancel

Create role

Next click on create notebook instance.

Permissions and encryption

IAM role
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the [AmazonSageMakerFullAccess](#) IAM policy attached.

AmazonSageMaker-ExecutionRole-20231203T204129

Success! You created an IAM role.
[AmazonSageMaker-ExecutionRole-20231203T204129](#)

Create role using the role creation wizard

Root access - optional

☒ Enable - Give users root access to the notebook

☐ Disable - Don't give users root access to the notebook
Lifecycle configurations always have root access

Encryption key - optional
Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

Network - optional

Git repositories - optional

Tags - optional

Cancel Create notebook instance

Notebook instance is created successfully

Success! Your notebook instance is being created.
Open the notebook instance when status is InService and open a template notebook to get started. [View details](#)

Amazon SageMaker > Notebook instances

Notebook instances info [Refresh](#) [Actions](#) [Create notebook instance](#)

Search notebook instances

Name	Instance	Creation time	Status	Actions
Assignment8	ml.t2.medium	12/3/2023, 8:44:09 PM	Pending	-

One status is in service, click on open jupyter/jupyterlab. I opened Jupyter lab

Amazon SageMaker > Notebook instances

Notebook instances info [Refresh](#) [Actions](#) [Create notebook instance](#)

Search notebook instances

Name	Instance	Creation time	Status	Actions
Assignment8	ml.t2.medium	12/3/2023, 8:44:09 PM	InService	Open Jupyter Open JupyterLab

On clicking it, it will redirect to notebook UI. Select kernel as Python. I renamed my file to heart.ipynb

assignment8-ttfj.notebook.us-east-2.sagemaker.aws/lab/tree/heart.ipynb

File Edit View Run Kernel Git Tabs Settings Help

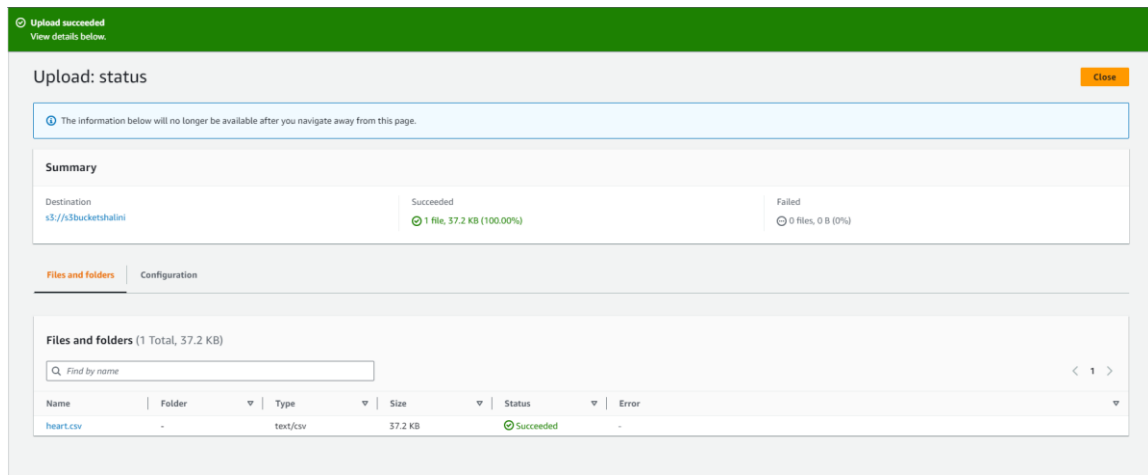
Filter files by name

Name	Last Modified
heart.ipynb	a minute ago

heart.ipynb

conda_python3

I have uploaded the heart.csv file to S3 bucket



Notebook:

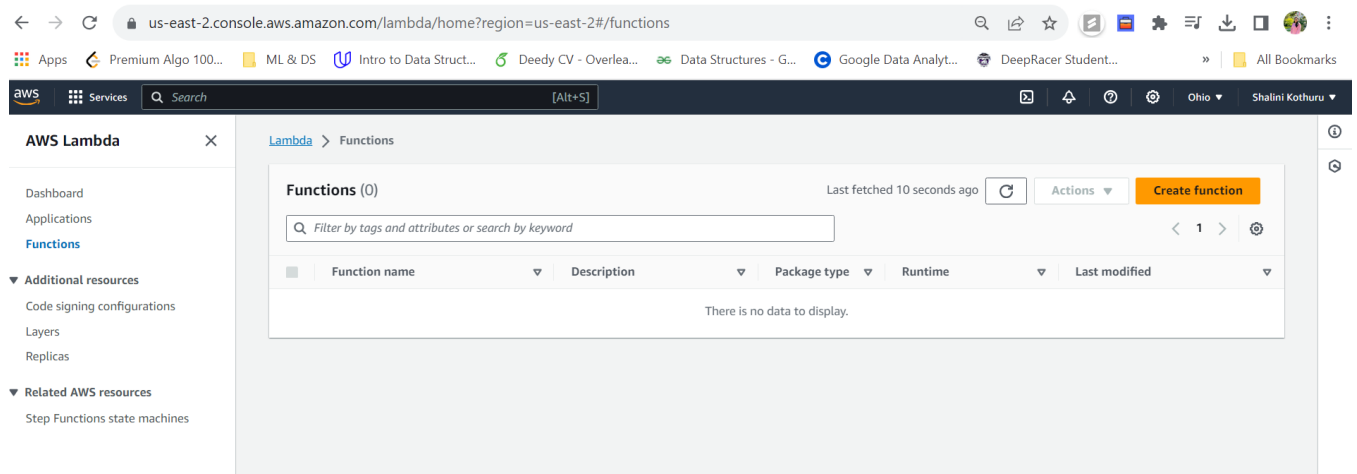
I have uploaded the notebook in canvas where I did data exploration, processing, feature selection, building, training and deploying the model.

Results :

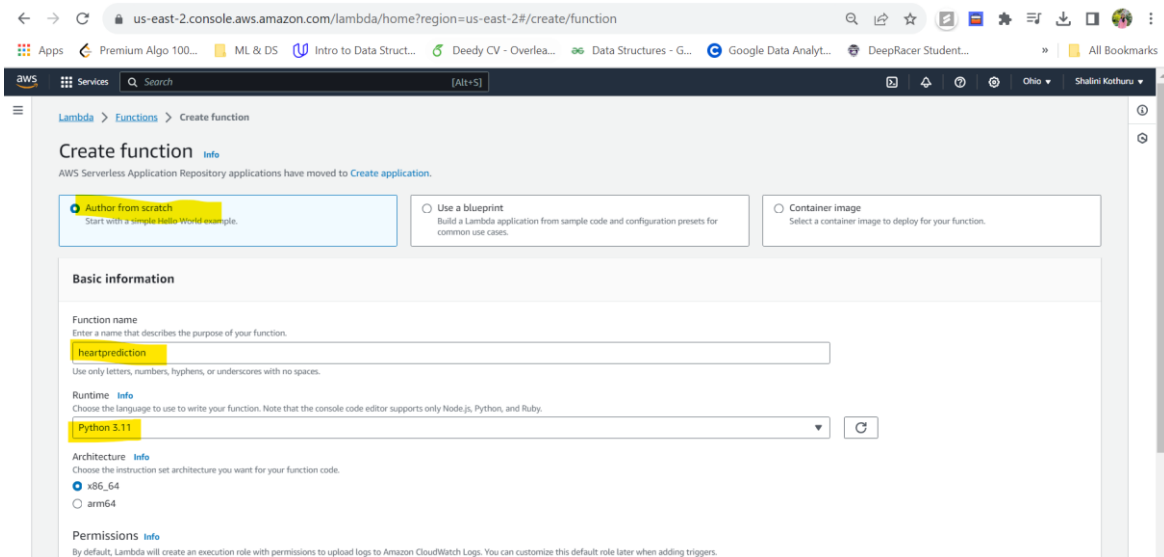
My XGBoost model resulted in 85.7% while Linear learner model resulted in 81.4 accuracy. Since XGBoost model performed better than linear learner model, I used XGBoost model to predict results based on inputs from user. I deleted endpoints after taking all the screenshots. Below is the process to create web app that takes inputs from user and predict the target variable indicating presence of heart disease.

Web application:

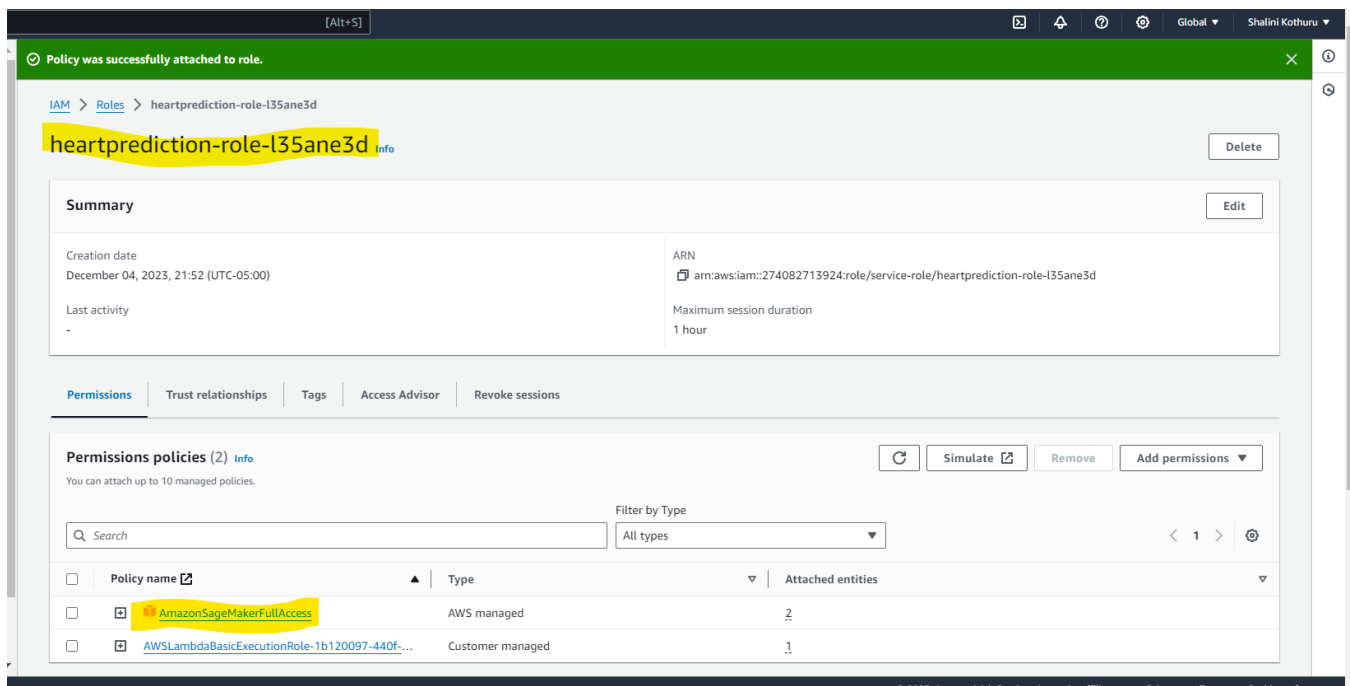
Go to AWS Management console and search for lambda and then click on create function



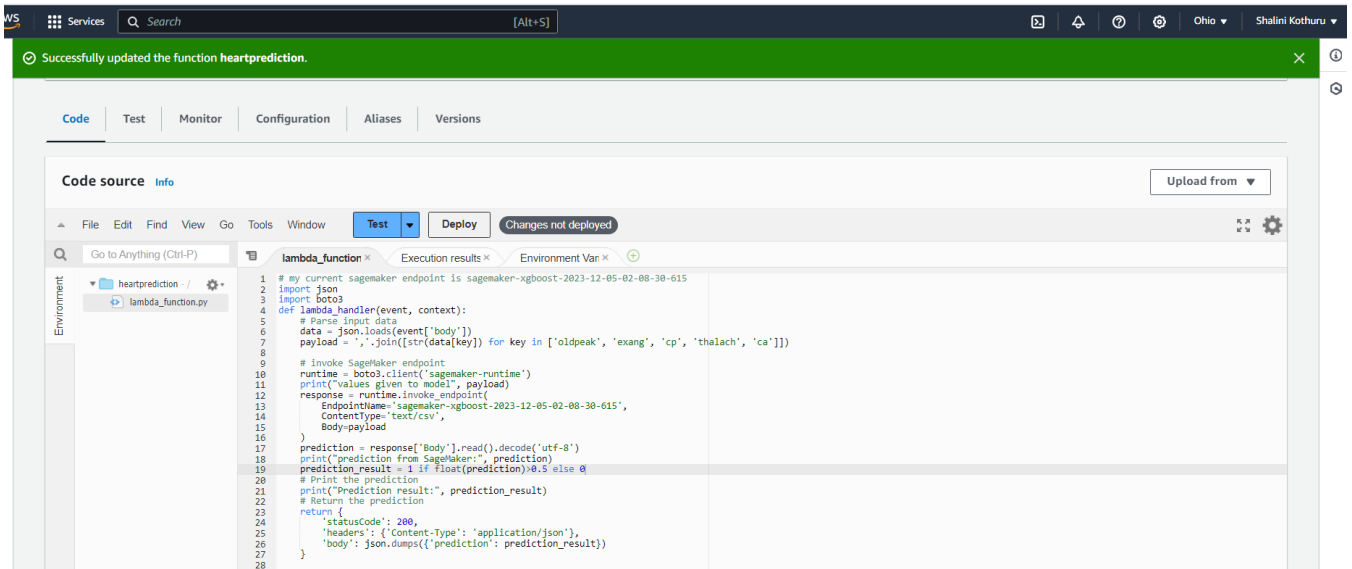
Add details as below and click on create function at bottom right corner of web page which creates function



The function is created successfully. There is default role created when this function was created. I went to IAM services to update policy for that role. I have attached SAGEMAKERFULLACCESS for that role.



Now add code as below in code tab. My xgb performed better so using xgb endpoint for prediction.



Now click on test next to code tab and create a test event. I gave random values. Now click on test

Test event info

Save Test

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

Create new event Edit saved event

Event name

testing

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Private

This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

Shareable

This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

hello-world

Event JSON

Format JSON

```

1 {
2   "body": "{\"oldpeak\": 2.3, \"exang\": 1, \"cp\": 3, \"thalach\": 150, \"ca\": 0}"
3 }
4

```

You can see logs in same page. It predicted 0 for this values

Executing function: succeeded (logs)

Details

The area below shows the last 4 KB of the execution log.

```

{
  "statusCode": 200,
  "headers": {
    "Content-Type": "application/json"
  },
  "body": "{\"prediction\": 0}"
}

```

Summary

Code SHA-256	Execution time
CIVRQWruSzpqhBLY+ARAU2KeiSemq4/9yHhoVS+mgI=	12 seconds ago (December 4, 2023 at 10:34 PM EST)
Request ID	Function version
62892108-b9e1-4617-8b66-2d9dd98e233b	\$LATEST
Init duration	Duration
273.66 ms	1285.96 ms
Billed duration	Resources configured
1286 ms	128 MB
Max memory used	
73 MB	

Log output

The section below shows the logging calls in your code. [Click here](#) to view the corresponding CloudWatch log group.

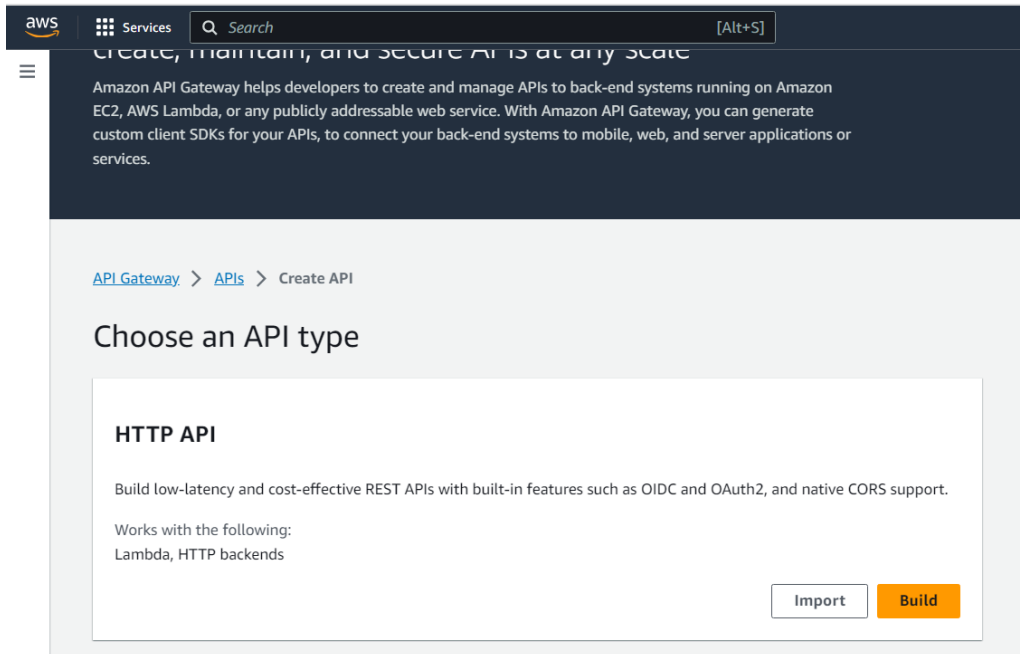
```

START RequestId: 62892108-b9e1-4617-8b66-2d9dd98e233b Version: $LATEST
values given to model 2.3,1,3,150,0

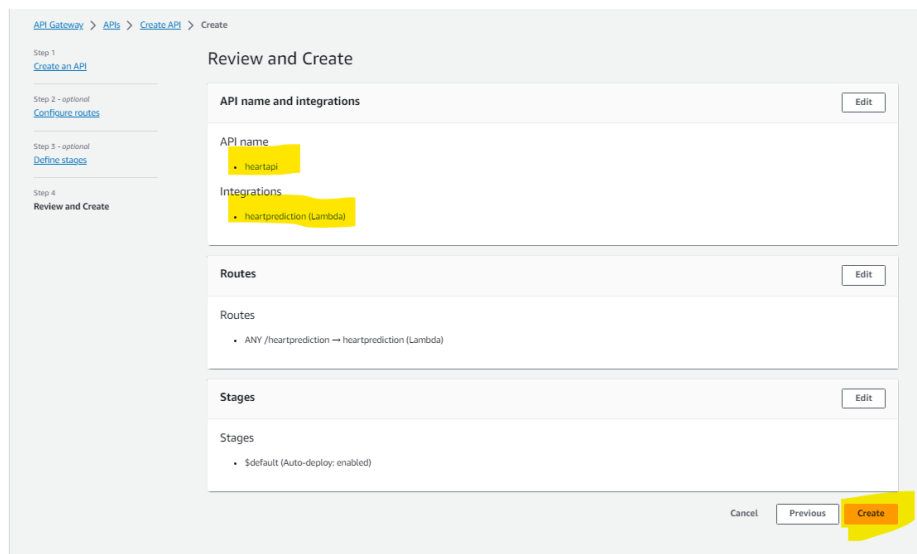
```

So my lambda function is working properly.

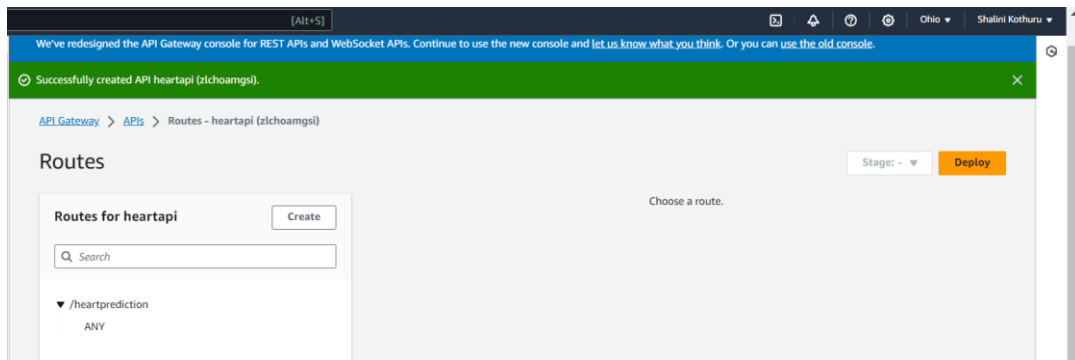
Now go to API gateway in AWS management console and click on HTTP API build



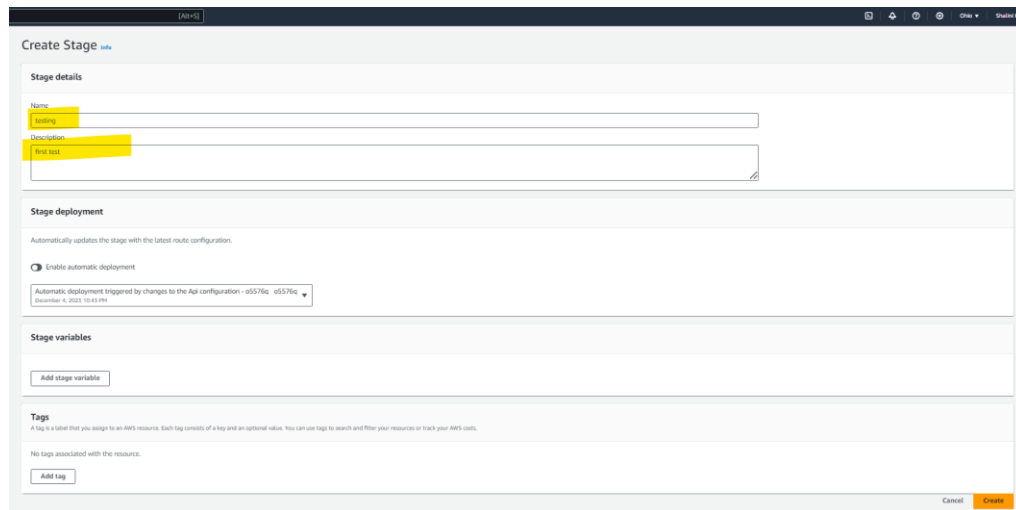
Now enter details as below and create API. You need choose lamda in step 1 and choose lambda function that you have created above.



Click on deploy API

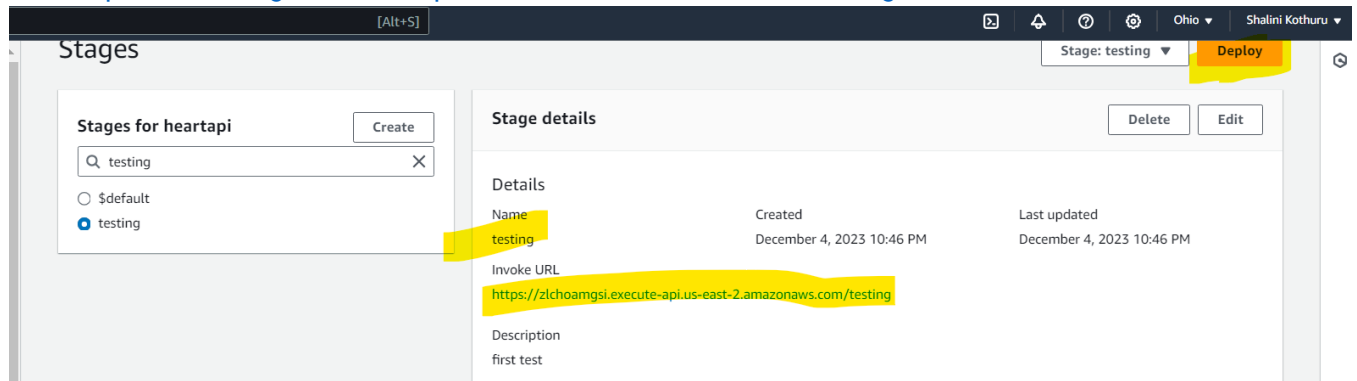


Once you click deploy, it asks for stage. I created stage



Once I created stage. It gave me one url. This url will be used when developing frontend. Notedown the url and click deploy

URL: <https://zlchoamgsi.execute-api.us-east-2.amazonaws.com/testing>



Select the stage you just created and hit deploy to stage.

Create deployment and attach to stage

A deployment is a snapshot of your API's configuration that can be associated with a Stage. Each Stage has an invoke URL and the behavior of this invoke URL is determined by the Stage settings and which deployment is attached to the Stage. (Auto-deploy enabled stages can't be deployed to manually.)
[To create a stage, click here.](#)

Select a stage

Q testing

Describe the changes for this deployment

Description (optional)

Cancel

Deploy to stage

[Alt+S]

Ohio

Shalini Kothuru

Successfully updated "testing" stage with a new deployment ID (2czu4s)

Successfully created API heartapi (zlchoamgsi).

0

0

2

0

0

API Gateway > APIs > heartapi (zlchoamgsi) > Stages

Stages

Stage: testing

Deploy

Stages for heartapi

Create

Q testing

☐ \$default

☒ testing

Stage details

Delete

Edit

Details

Name	Created	Last updated
testing	December 4, 2023 10:46 PM	December 4, 2023 10:46 PM
Invoke URL	https://zlchoamgsi.execute-api.us-east-2.amazonaws.com/testing	
Description	first test	
Attached deployment	Automatic Deployment	
	Disabled	

Update access control allow credentials for this API and then deploy it again

API Gateway > APIs > heartapi (zlchoamgsi) > CORS

Cross-Origin Resource Sharing

Stage: -

Deploy

Configure CORS

Configure

Clear

CORS allows resources from different domains to be loaded by browsers. If you configure CORS for an API, API Gateway ignores CORS headers returned from your backend integration. See our [CORS documentation](#) for more details.

Access-Control-Allow-Origin

https://sagemaker-us-east-2-274082713924.s3.us-east-2.amazonaws.com

Access-Control-Allow-Methods

GET

POST

OPTIONS

PATCH

PUT

HEAD

*

Access-Control-Allow-Credentials

YES

Access-Control-Allow-Headers

content-type

x-amz-date

authorization

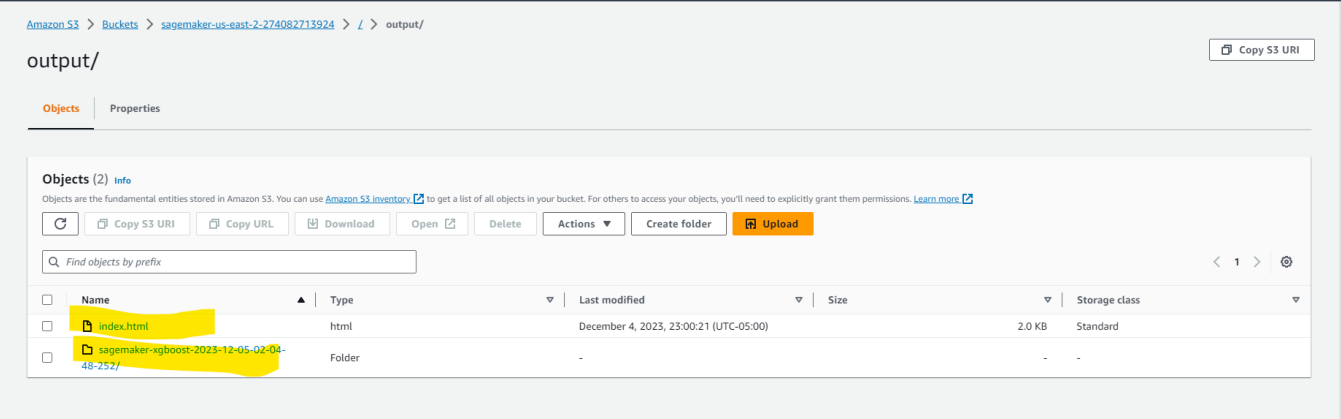
x-api-key

x-amz-security-token

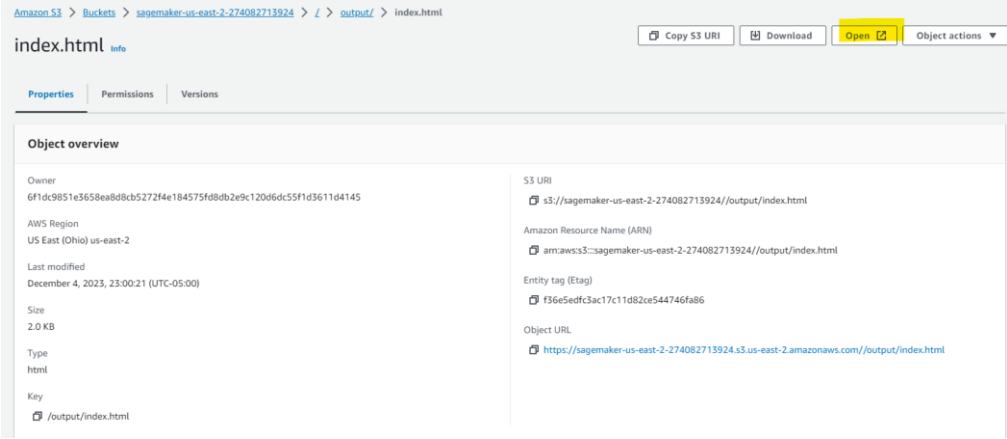
Access-Control-Expose-Headers

No Expose Headers are allowed

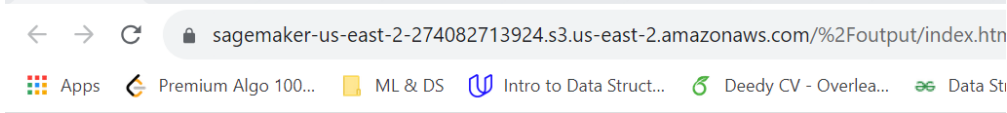
Now go to S3 to store the html file. I have store html file at the same location where I stored sagemaker endpoint



Now click on index.html. It will give details of index.html. Now click on static website URL to access your web application. You can do it by clicking on open button highlighted in below screenshot



It will redirect to following web page. As the goal of this assignment is not about the developing the web application with good UI, I did basic web page. Enter details and click on predict



Heart Disease Prediction Form

Oldpeak (0-10):

Exang (0/1):

CP (0-3):

Thalach (50-200):

CA (0-4):

On clicking predict, it gives the presence of heart disease basically the target variable(0 = no disease or 1 = disease) as show in screenshot.