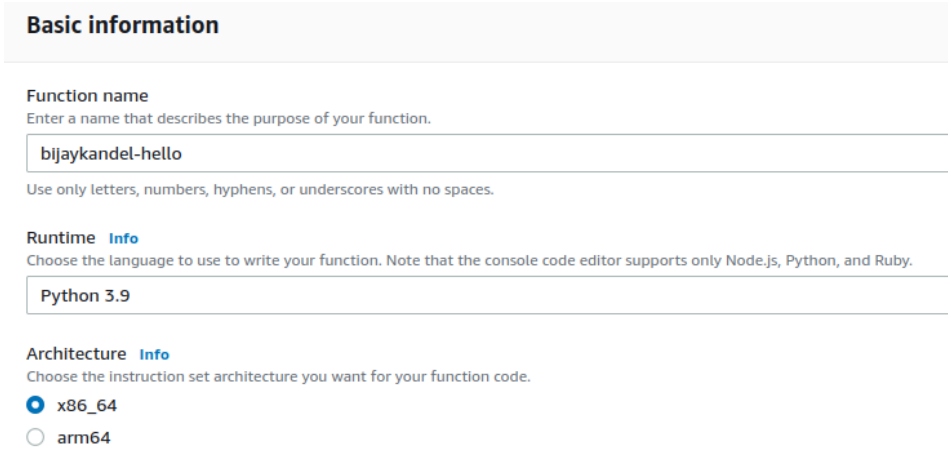
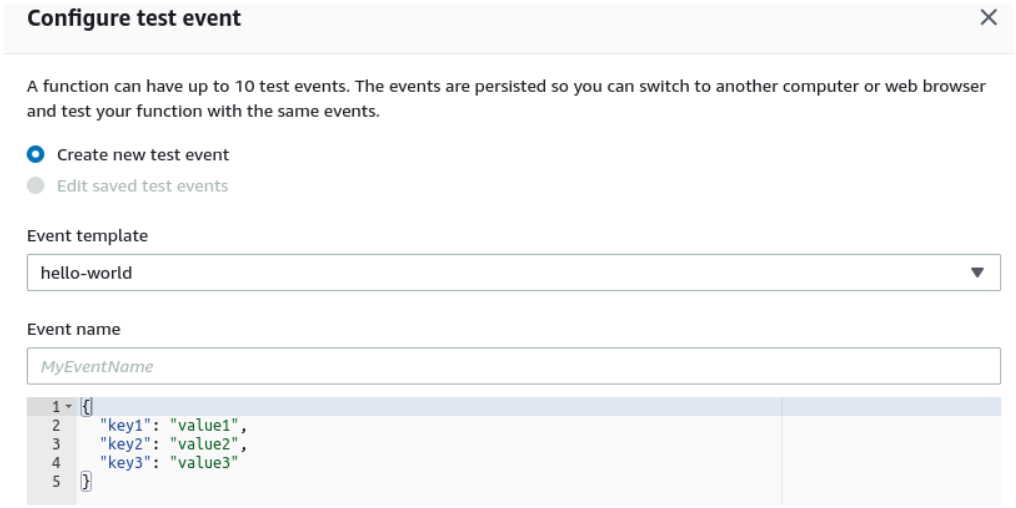
- Create two Lambda Functions
- First Lambda function returns 200 Response as {“Hello”: “Default”}
- Second Lambda function returns 200 Response as {“Hello”: “{Dynamic route name}”}
- Configure API Gateway with that hits first lambda function on / and the second lambda function on /*

Two lambda functions were created for this assignment:

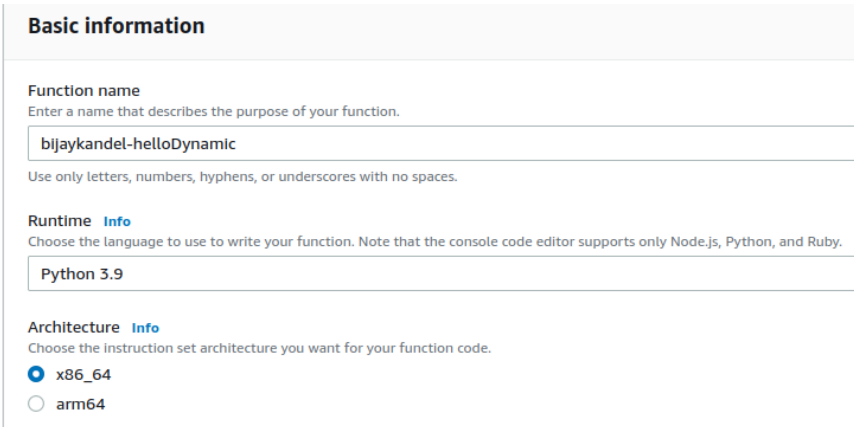
First one is bijaykandel-hello



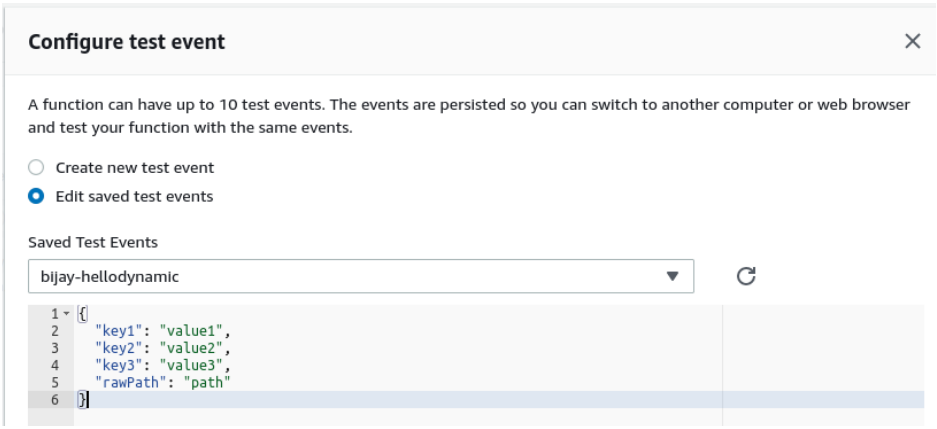
And its test event is configured as default.



The second lambda function is named as bijaykandel-helloDynamic



And its test event is configured as follows: (“rawPath” option is added in the existing default document)



First lambda function is written as:

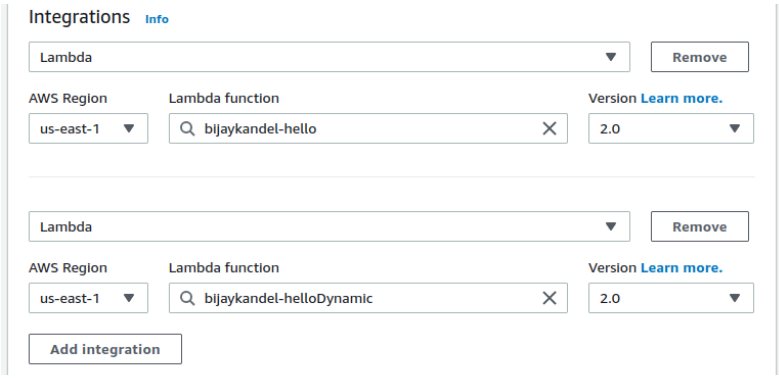
```python
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps({'Hello': 'General'})
    }
```

And the second lambda function is written as:

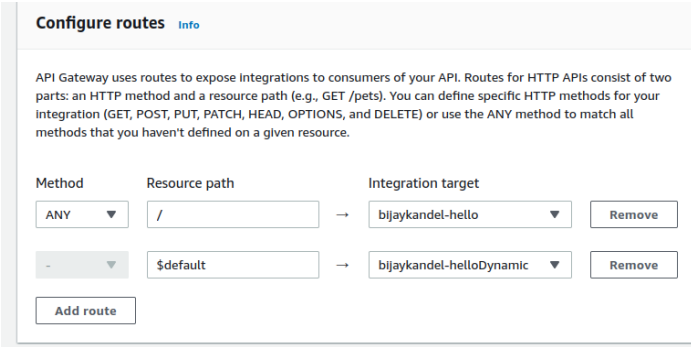
```python
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps(event['rawPath'])
    }
```

Now API gateway is created: **API Gateway >> Create API gateway:**

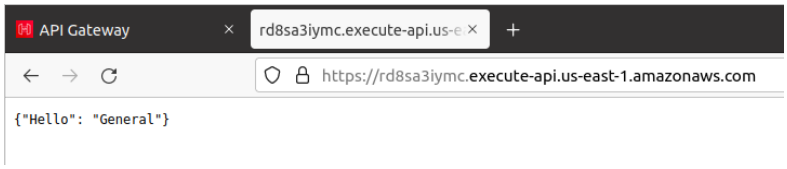
We added two integrations for two lambda functions as follows:



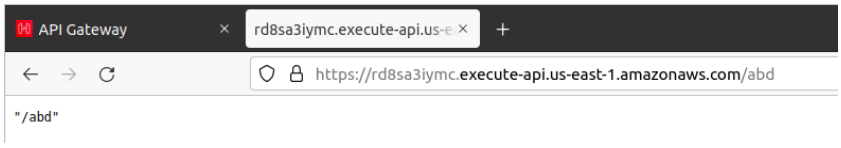
And Routes are configured so that accessing the URL triggers first lambda function and accessing URL/* triggers the second lambda function.



After creating the API gateway, we can click on the link inside our API gateway. With no path specified, the url returns the following value:



And if path is provided with url, it returns the given path as shown below:



- Create a bash script to deploy your lambda functions
- Create a bash script to deploy your react app to S3
- Integrate both these scripts with one of Jenkins, Github Actions, CircleCI or TravisCI

To deploy react app to s3, following bash script file is created:

Sudo nano cicd.sh

```
 GNU nano 4.8                                                cicd.sh
#npx create-react-app reactapp
#cd reactapp/

echo 'inside reactapp building static files'

npm run build

echo 'Build completed'


echo 'uploading static files of build folder to aws via cli'
aws s3 cp build s3://intern-bijaykandel37 --recursive

echo 'uploading static files completed'


echo 'Now you can go to s3 bucket and see the static url link in index.js object'

echo 'OR'

echo 'click on the link below to see the hosted app'
aws s3 presign s3://intern-bijaykandel37/index.html
```

Making it executable: ***sudo chmod +x cicd.sh***

Running the script file with command: ***./cicd.sh***

```
The build folder is ready to be deployed.
You may serve it with a static server:

  npm install -g serve
  serve -s build

Find out more about deployment here:

  https://cra.link/deployment

Build completed
uploading static files of build folder to aws via cli
upload: build/static/css/main.a617e044.chunk.css.map to s3://intern-bijaykandel37/static/css/main.a617e044.chunk.css.map
```

And we can get the link of index.html file as shown:

```
upload: build/static/js/2.6bdd5a89.chunk.js.map to s3://intern-bijaykandel37/static/js/2.6bdd5a89.chunk.js.map
uploading static files completed
Now you can go to s3 bucket and see the static url link in index.js object
OR
click on the link below to see the hosted app
https://intern-bijaykandel37.s3.us-east-1.amazonaws.com/index.html?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIA52BEGI3BCCP6PEYC%2F20211215%2
Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20211215T182738Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=dda4a875bee1dad71388cf51e0c86d5
94b07c1e2928fafd9e7b4e1f11a807d51
bj@batman:~/react/react-bijay$
```

When we go to the link, we can see our page being hosted statically from s3.