Task 1: API Implementation

Source code and `README file` see in the zip file:

GermanPhoneNumberProjects

How to Build and Run docker file

1. Open a command prompt (all docker commands pasted in cmd)
2. Pull the docker image from the docker hub
   docker pull sshamnatvm/phonefilter
3. Build the docker image
   docker build --rm -t sshamnatvm/phonefilter .
4. Run the docker image
   docker run -p 5000:5000 sshamnatvm/phonefilter
5. Open Swagger UI
   http://localhost:5000/swagger/index.html

Swagger UI running from code and docker image

http://localhost:5000/swagger/index.html

Task 2: Testing

Added unit test as well

- Install Postman/Use web Postman site
- Run the docker image created in previous
- Start Postman.
- Create a new request

1. For adding data to the testing environment use below POST method
   - Set the HTTP method to POST. (This post method is only for testing purposes, not visible from Swagger UI)

     http://localhost:5000/testPost

   - Select the Body tab.
   - Select raw.
   - Set the type to JSON.
   - In the request body enters the below line and Click Send button:

{

"phoneId": 10,

"phoneResults": "+4915201365263, +4915201365264, +4915201365269"

}

- Enter the below line and Click Send button (this is for the second entry)

{

"phoneId": 11,

"phoneResults": "+4915201365276, +4915201365289, +4915201365299"

}

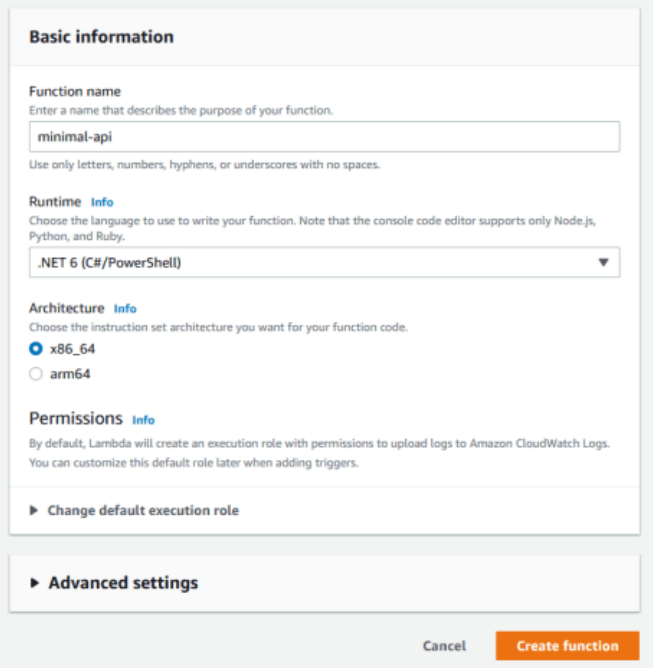
2. For getting all phone details and id use the below instruction
   - Set the HTTP method to GET and use the below link and click Send button.
   - http://localhost:5000/getAllPhoneDetails

3. Get Single detail and id use the below instruction
   - Set the HTTP method to GET and use the below link and click Send button.
   - http://localhost:5000/getPhoneDetail/11
   - http://localhost:5000/getPhoneDetail/10

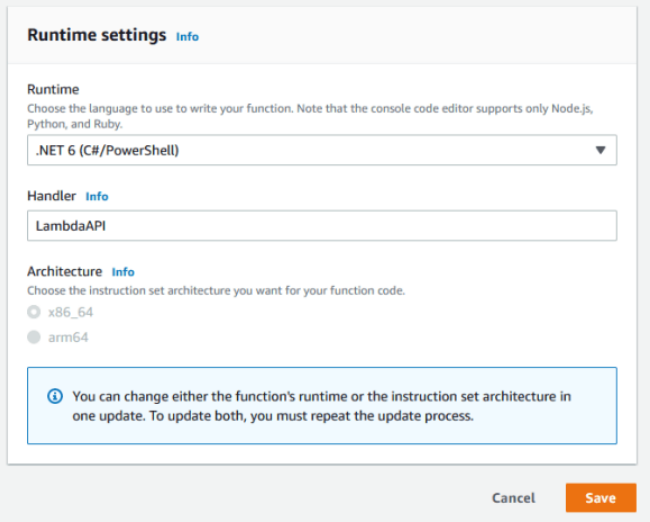
4. For deleting phone number use the below instruction
   - Set the HTTP method to DELETE and use the below link and click Send button.
   - **http://localhost:5000/deletePhoneDetail/10**

Task 3: Deployment using AWS

- Navigate to the AWS Console and go to Lambda.
- Click Create function, then Author from scratch, give it a name and select .NET 6 for the runtime, and then click Create function.



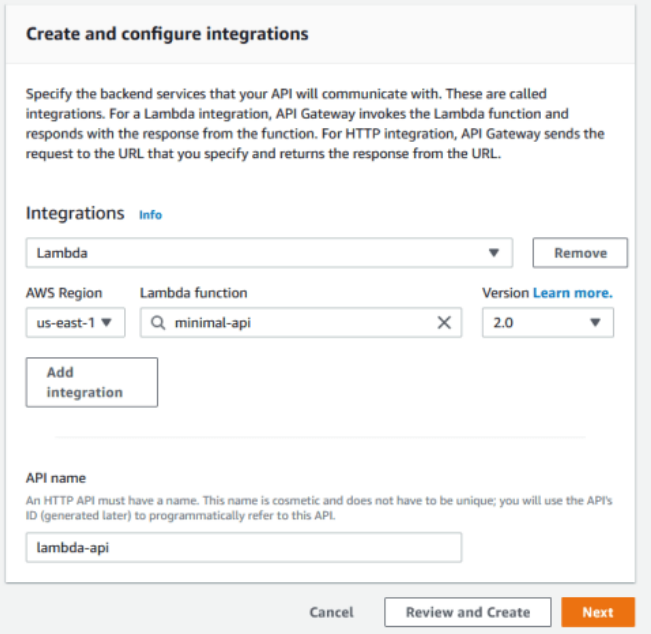
- Now under Runtime settings click Edit and change the handler to the name of the assembly, “LambdaAPI” if you’re following along.



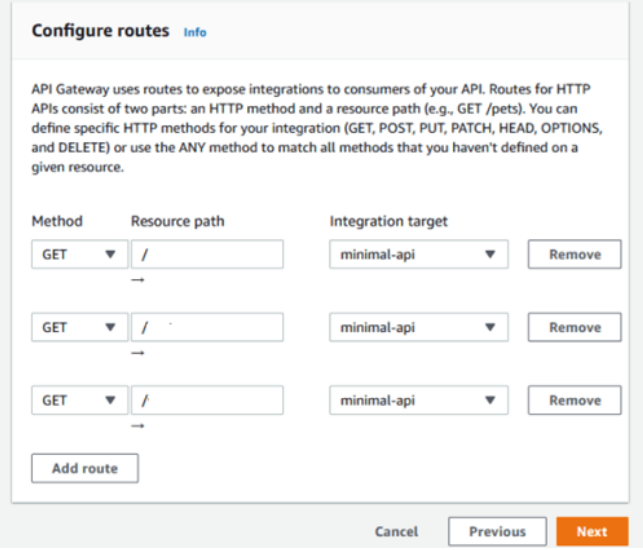
- Lastly for the Lambda, we need to package the assemblies and upload them to AWS.
- Go back to your project directory and run below comment from project directory:

```
dotnet publish -c Release --self-contained false -r linux-x64 -o publish
```

- Zip up the contents of the /publish directory.
- Then back on the AWS Console, under Code source click Upload from and upload your .zip file.
- Now navigate to API Gateway and create a new HTTP API. Configure the integrations to use the Lambda we just created and give it a name.



- Set up the routes manually. Test all 4 API routes we have configured.



- Leave the stages configured as they are and click Next, click Create
- It should be deployed and functioning.
- Take the provided Invoke URL from amazon and try it out.