Serverless Workshop Exercises

Exercise 1 – Deploying a base service

Deploy service already available in base serverless.yml.

Steps:

1) Go to ghc-serverless/workshop-usecases
2) Open serverless.yml file and change s3_bucket from “ghcdemo-manisha” to “ghcdemo-<yourname>”
3) Run the command to deploy your service in serverless.yml
   ```
   sls deploy
   ```
4) Populate DynamoDb with Course/Teacher tables for the purpose of the workshop
   ```
   ./populateDydb <name> <email_id>
   ```
5) Check the current status of deployment
   ```
   sls info
   ./dumpDydb Course/Teacher
   ```

Exercise 2 – Attaching S3 upload trigger to lambda

Trigger the lambda (provided) that publishes SNS notification when a student sends an assignment. The ‘notifier’ lambda subscriber (provided) sends a notification email to Teacher.

Steps:

1) To the lambda ‘demoassignment’ add a S3 event trigger so that the lambda gets called when any .py file is added to the bucket
   ```
   events:
       - s3:
         bucket: ${self:custom.s3_bucket}
         event: s3:ObjectCreated:*
         rules:
          - suffix: .py
   ```
2) Deploy the changes in the service
   ```
   sls deploy
   ```
3) Upload a python file to S3 bucket
   ```
   ./uploadFile <bucket> <file>
   ```
4) Check notification email in your inbox

Exercise 3 – Adding a new SNS subscriber

Add a lambda that updates Teacher table with a new ToDo in DynamoDb. The lambda gets triggered when a SNS notification is published.

Steps:

1) Copy the ‘notifier’ lambda and change the name to ‘updatedb’ and ‘handler’ to ‘handler.updatedb’
```
updatedb:
  handler: handler.updatedb
  events:
   - sns:
      arn:
        Fn::Join:
          - ""
          - - "arn:aws:sns:"
            - Ref: "AWS::Region"
            - ":"
            - Ref: "AWS::AccountId"
            - ":${self:custom.topic_name}"
      topicName: ${self:custom.topic_name}
```
2) Add a log to the handler
3) Deploy the changes in the service
```
sls deploy
```
4) Check if new lambda is deployed
```
sls info
```
5) Upload a python file to S3 bucket
```
./uploadFile <bucket> <file>
```
6) Check the logs for the log you added
```
sls logs
```
7) Check the DynamoDb for updates in ToDo field of the Teacher table
```
./dumpDydb Teacher
```

Exercise 4 – Adding API Gateway events for REST APIs

Add a lambda that gets triggered as a HTTP GET endpoint. The GET is for a particular Teacher’s ToDo and the result is in JSON format

Steps:

1) Add a lambda ‘list_teacher_todos’ as a simple HTTP endpoint
```
list_teacher_todos:
  handler: handler.list_teacher_todos
  events:
    - http: GET list_teacher_todos
```
2) Deploy the changes in the service
```
sls deploy
```
3) Check the current status and get the endpoint
```
sls info
```
Sample output snippet:
```
endpoints:
  GET - https://cexp3zsy3g.execute-api.ap-south-1.amazonaws.com/dev/list_teacher_todos
```
4) Get teacher_id from DynamoDb
```
./dumpDydb Teacher
```
5) Use curl or POSTMAN to make a HTTP GET request for the Teacher’s ToDo based on teacher_id. Use the endpoint from Step 3)

E.g.:
```
curl https://cexp3zsy3g.execute-api.ap-south-1.amazonaws.com/dev/list_teacher_todos?teacher_id=300
```