Computer Science Department, University of Crete

CS 452 – Introduction to the Science and Technology of Services
Winter semester 2022-23

Instructor: Kostas Magoutis

Teaching Assistants: Giorgos Kelantonakis (csdp1224@csd.uoc.gr), Fallia Kourou (csdp1162@csd.uoc.gr), Achilleas Filippidis (csdp1276@csd.uoc.gr)

1st Assignment
Due date: 23/10/2022

General instructions
Write a report demonstrating your results using images, screenshots, code and any other explanation necessary. Include a cover page with your student ID and name. The filename of the deliverable must be in the following form: **ask1_AM** (where AM is your student id number e.g. ask1_1234). Any images/screenshots and code should be included in your report document. In particular, screenshots of the AWS UI showing the implementation and test runs of the serverless applications should be included. The report document must be in pdf format. Submit a single compressed file (zip/rar/gz etc) if your work contains multiple files.

Note that this assignment is to be done **individually**. Cheating in any way, including giving your work to someone else, will result in failing this assignment (a mark of zero will be given).

Submit your assignment **electronically** (online) until 23/10/2022 at 23:59 using the course webpage. Delayed submission is possible with a penalty of 10% for each additional day. You should be ready to demonstrate your developed prototypes in an examination to be scheduled shortly after the submission date.

Overview

In this assignment, you will experiment with Amazon AWS and more specifically you will deploy two event-driven serverless applications using your Amazon Web Services (AWS) Educate Starter accounts. Amazon AWS provides a variety of serverless services with which you can interact. Basic such services are AWS Lambda, Step Functions and Amazon DynamoDB.

Background

We strongly suggest you read or review the following resources:

- Course slides
- Creating and running a Serverless "Hello, World!"
- Core Components of Amazon DynamoDB
- Creating and run a Workflow using Step Functions

Task A (15 points)

In this task you will practice with Step Functions as seen in the lab of 5/10. You will implement a workflow that could be used as an API over a database. In particular, this database will be used to store and retrieve test cases for external services.

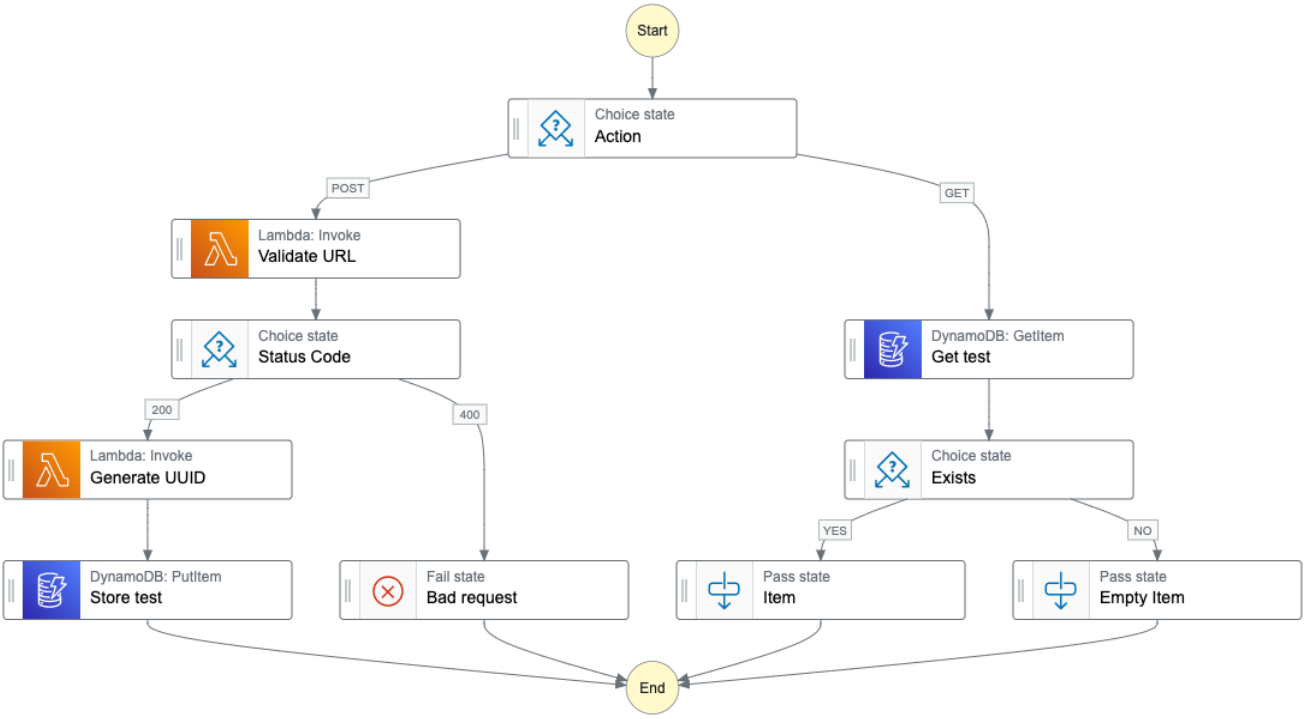
You will use DynamoDB to create a table with the following fields

Field	**Description**	**Example**
Id	Must be unique	uuid
Method	Http method that needs to be used	GET
URL	Endpoint to test	https://www.example.com/api
Expected_Result	Response from calling the above url	

For simplicity, you will focus on 2 basic actions

Action	**Input**	**Result**
POST	• Method • URL • Expected_Result	Status:200 _____OR_____ Bad request
GET	• Id	• Method • URL • Expected_Result _____OR_____ Empty object

Your final workflow will look like this



For the implementation you will need to:

- Create a table in DynamoDB, to store the tests
- Create Lambdas, for validation and Id generation
- Create the workflow using Workflow Studio

Task B (20 points)

In this task, you will build a tool for test automation of online services. In particular, you will use the above AWS services.

AWS Service	**Purpose**	**Icon**
API Gateway	A way to interact with your tool	
Lambdas	Add logic	

Simple Queue Service (SQS)	Intermediate event holders	
Simple Notification Service (SNS)	Forward an event to interested services	
Step Functions	Learn how to integrate with other services	
Email (not a service)		

API Gateway in detail

Endpoint	**Input**	**Result**
POST /test	• Method • URL • Expected_Result	Status:200 _____OR_____ Bad request
POST /tests/id*		Email with result for test with specified id
POST /tests/all		Email with results for all tests

* id is variable

Keep in mind that the color of each endpoint shows which lambda should be triggered.

Lambdas in detail

Number	**Action**
1	Take the id from the URL and put it in the queue
2	Read all ids from the same table as your State Machine (Step function) uses and give each one as argument to Lambda 1
3	Adds the fields that are required by the State machine, in the event
4	1. Get the test from the table using your State Machine 2. Forward it to Lambda 5

5	1. Call the endpoint and get the response 2. Forward the expected result and the actual result to Lambda 6
6	Validate the expected and actual results
7	Store results to a DynamoDB Table

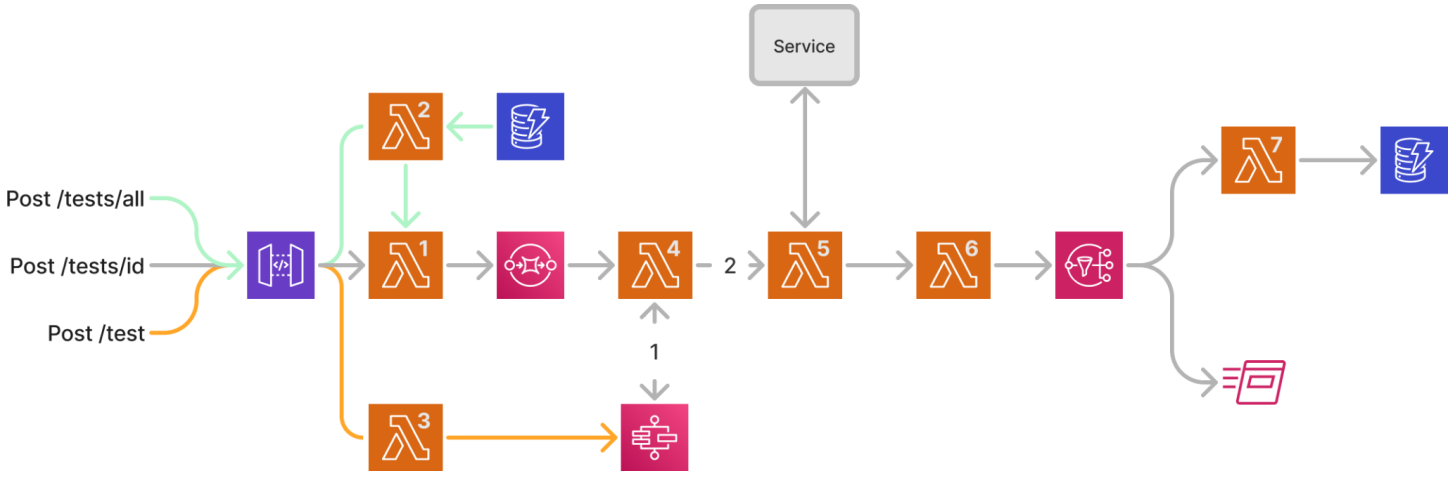
For a service to test, you can use whichever service you want that provides an open API or else you can use the service we made in the lab.

Action	**Path**	**Example**
Base	https://tfr5qh5091.execute-api.eu-central-1.amazonaws.com/beta	
Add	/add?a=x&b=y	a=1, b=2
Subtract	/sub?a=x&b=y	a=1, b=2

In order to interact with any of these services and lambda functions you have to use Amazon Web Services (AWS) SDK for Python (Boto).

Note that actions in case of 'Lambda Function Execution Failures' are indicative and should not be implemented as part of this homework.

Your final tool will look like the above figure.



For the implementation you will need to:

- API Gateway
- Simple Queue Service

- Simple Notification Service