Computer Science Department, University of Crete

CS 452 – Introduction to the Science and Technology of Services

Winter semester 2023-2024

Responsible Instructor: Marina Bitsaki (marina@csd.uoc.gr)

Teaching Assistants: ***Αποστόλου Ηρόδοτος, csd4437@csd.uoc.gr***
Παπαγεωργίου Ευθύμιος, csd4340@csd.uoc.gr

Assignment 3

Due date: 18-1-2024

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: **proj_AM** (where AM is your student id number e.g. proj_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. Regarding the programming language we recommend you to use Python. 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 18-1-2024 at 23:59 using the course webpage. Delayed submission is permitted with a penalty of 10% for each additional day.

You should be ready to demonstrate your developed prototypes in an examination that may be scheduled shortly after the submission date.

Overview

In this assignment, you will experiment with Amazon AWS and more specifically you will deploy one event-driven serverless application using your Amazon Web Services (AWS) Educate Starter accounts. Amazon AWS provides a variety of serverless services with which you can interact. Basic services are AWS Lambda, Amazon SNS, Amazon SQS, Amazon DynamoDB, and Amazon EC2. You can also interact with various amazon APIs.

CASE STUDY: Drone Delivery: An efficient way to reduce delivery time and traffic congestion

Brief description

Drone delivery involves using flying drones as a means of delivering packages from retailers to customers. Drones are small or medium-sized unmanned aerial vehicles that can drive remotely and autonomously, and maintain a consistent level of flight.

Drone delivery offers the potential for significantly faster delivery times compared with traditional methods. Drones can operate autonomously and are not subject to traffic congestion or other logistical challenges that traditional delivery vehicles may face. In addition, retailers can potentially save on costs by reducing labor costs.

Drones can transport packages from a local distribution center or retailer directly to a customer's doorstep. In the case of last-mile deliveries, drones are typically used for delivering small or medium-sized parcels and can cover relatively short distances.

In this project we are interested in developing a service system for drone delivery in an efficient way. It involves a retailer, a drone delivery provider for last-mile deliveries, and clients.

Suppose that clients make requests for last-mile services to providers that own a limited number of identical resources (drones). We aim at defining a scheduling algorithm for serving tasks (requests) so that efficiency is achieved.

Consider that time is partitioned into rounds and at the beginning of each round, the provider receives a fixed number of requests. In each round an auction is performed determining the winners (requests to be served) and the price (cost of delivery). The losers participate again in the next round.

The service system consists of the following entities:

- **Product retailer:** Manages service requests
    - Pricing Policy
    - Decision: Which requests to serve

    - Selection criteria: Bids
    - Goal: efficiency (total sum of clients utility is maximized)
- **Drone delivery provider:** Intermediate that undertakes the delivery of the product
    - Decision: what is the delivery order of the winning requests
    - Goal: efficiency (total time is minimized)
- **Clients:** People buying products/services on the retailer's website
    - Decision: what bid to submit

PHASE A (15 units)

Design a Petri net that represents the drone delivery service system for two consecutive bidding rounds. Consider the interactions between clients, the product retailer and the drone delivery provider. Your Petri net should include at least 2 drones and 4 customers. You could make any other assumption for the allocation of the resources in a realistic scenario. For the possible states of the petri net we assume the following:

- Initial State (before starting the auction):
    - All drones are available.
    - No bids have been submitted.
    - The retailer is ready to manage requests.
- Middle State (After the termination of the first round):
    - Some drones are allocated to winning bids.
    - Loser bids are waiting for the next round.
    - The retailer is processing the outcomes of the first round.
- Final State (when both bidding rounds have finished):
    - Allocation of all drones based on the outcomes of both rounds.
    - Completion of deliveries or preparation for the same.
    - Retailer and clients are in a state reflecting the end of the bidding process.

Also design the reachability graph of your petri net. Illustrate how the system transitions from the initial state, through the middle state, to the final state. Show all possible states and transitions, considering the limitations and actions of each entity.

Deliverable: 1) A detailed Petri net diagram representing the described system over two bidding rounds 2) A reachability graph showing all possible transitions and states from the initial to the final state. These deliverable can be either handwritten or using a software (e.g. Powerpoint).

PHASE B (30 units)

Consider two basic scenaria, as follows:

1. All drones are identical in terms of maximum speed and maximum weight of package to deliver.
2. There are two types of drones: Type A (low speed, heavy packages) and Type B (high speed, light packages.

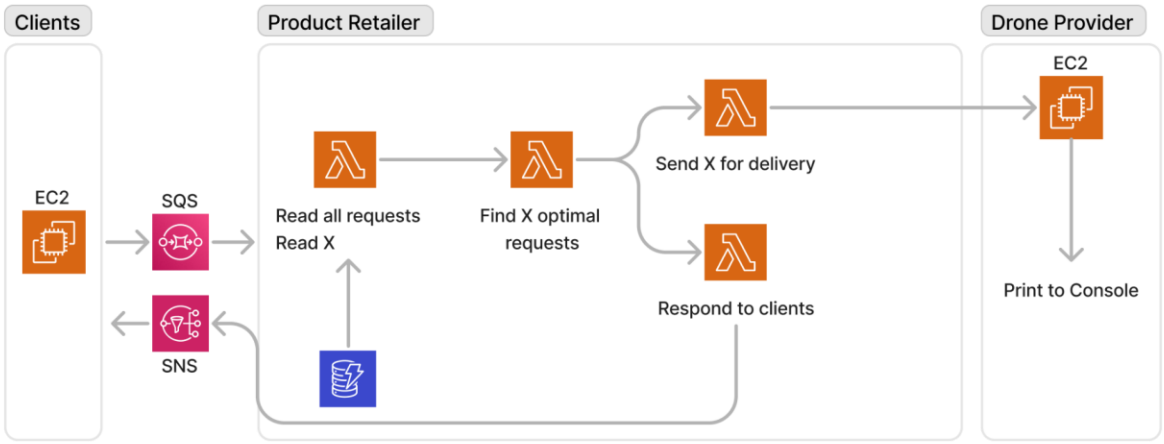
Tasks for both scenaria:

1. For Retailers:
    a) Define auction type
    b) Determine winners and prices based on clients' bids
2. For Drone Providers:
    a. Define scheduling algorithm
    b. Determine delivery order based on clients' profile
3. For Clients:
    a. Define bidding strategy
    b. Define profile (utility, time constraints, type of product)

PHASE C (55 units)

Implementation

Implement the entities described above and their interactions using serverless technologies. A possible implementation is shown in the following figure:



In particular, you should implement the following considering that the game consists of 20 clients and continues until there is no client sending requests:

C1. Implement the Product Retailer as a serverless web application with lambda functions that perform the following specific tasks using SNS, SQS and DynamoDB.

Task	Description
1	Get all the bid messages from a SQS queue
2	Read the number of available drones (X) from the DynamoDB table
3	Forward all the requests and the number of drones to the second Lambda function and calculate the optimal X bids to maximize the retailer's utility.
4a	Send the optimal X bids to Drone provider for delivery
4b	Respond to clients. Respond with 'delivered' to clients whose bids were chosen and with 'declined' to all other clients.

Tip: To read all bids from clients together, set the batch size for the first Lambda function, equal to the number of clients, at the beginning.

C2. Implement the Drone Provider using an EC2 instance that performs the following tasks.

Tasks	Description
1	Accept a list with the orders, using a REST endpoint
2	Schedule orders in the order that maximizes its utility
3	Print the result to the console

C3. Implement the Clients using an EC2 instance with a fixed number of processes describing the number of clients that perform the following tasks.

Tasks	Description
1	Calculate and send a bid for their order
2	Subscribe to an SNS topic and wait for the response from the Product Retailer.
3a	If the response is ‘delivered’, he can exit (happy)

3b	If the response is ‘declined’ , he can either: 1. Try again and send a bid 2. Has surpassed his waiting time and exits (sad)

C4. Implement both scenarios described in Phase B and provide the following three graphs for each:

1. A graph that shows the bids of all clients vs. rounds
2. A graph that shows the profit of Product Retailer vs. rounds
3. A graph that shows the revenues of the Drone Provider vs. rounds

Study the results and make your observations.

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.