Assignment 2

CSCI 5410 (Serverless Data Processing)
Date Given: May 30, 2022
Due Date: Jun 10, 2022 at 11:59 pm
Late Submissions are not accepted.
A deduction of 10% per day will be applied for late submission.

To avoid any additional charges for resource consumption - Delete the GCP resources, and AWS services after fulfilling the assignment submission requirements.

Objective:

This assignment covers concepts of containerization and Serverless components of cloud computing. The primary objective of this assignment is to introduce you to the cloud computing containerization application using Docker and creation of a chatbot using Lex.

Plagiarism Policy:

- This assignment is an individual task. Collaboration of any type amounts to a violation of the academic integrity policy and will be reported to the AIO.
- Content cannot be copied verbatim from any source(s). Please understand the concept and write in your own words. In addition, cite the actual source. Failing to do so will be considered as plagiarism and/or cheating.
- The Dalhousie Academic Integrity policy applies to all material submitted as part of this course. Please understand the policy, which is available at: https://www.dal.ca/dept/university_secretariat/academic-integrity.html

Assignment Rubric - based on the discussion board rubric (McKinney, 2018)

	Excellent (25%)	Proficient (15%)	Marginal (5%)	Unacceptable (0%)	Problem # where applied
Completeness including Citation	All required tasks are completed	Submission highlights tasks completion. However, missed some tasks in between, which created a disconnection	Some tasks are completed, which are disjoint in nature.	Incorrect and irrelevant	**Part A** **Part B**
Correctness	All parts of the given tasks are correct	Most of the given tasks are correct However, some portions need minor modifications.	Most of the given tasks are incorrect. The submission requires major modifications.	Incorrect and unacceptable	**Part A** **Part B**
Novelty	The submission contains novel contribution in key segments, which is a clear indication of application knowledge.	The submission lacks novel contributions. There are some evidence of novelty, however, it is not significant	The submission does not contain novel contributions. However, there is an evidence of some effort.	There is no novelty	**Part A** **Part B**

Clarity	The written or graphical materials, and developed applications provide a clear picture of the concept and highlights the clarity.	The written or graphical materials, and developed applications do not show clear picture of the concept. There is room for improvement	The written or graphical materials, and developed applications fail to prove the clarity. Background knowledge is needed.	Failed to prove the clarity. Need proper background knowledge to perform the tasks.	**Part A** **Part B**

Citation:
McKinney, B. (2018). The impact of program-wide discussion board grading rubrics on students' and faculty satisfaction. Online Learning, 22(2), 289-299.

Tasks:

This assignment has 2 parts. Part A is related to coding, and development. Part B is related to exploring a service.

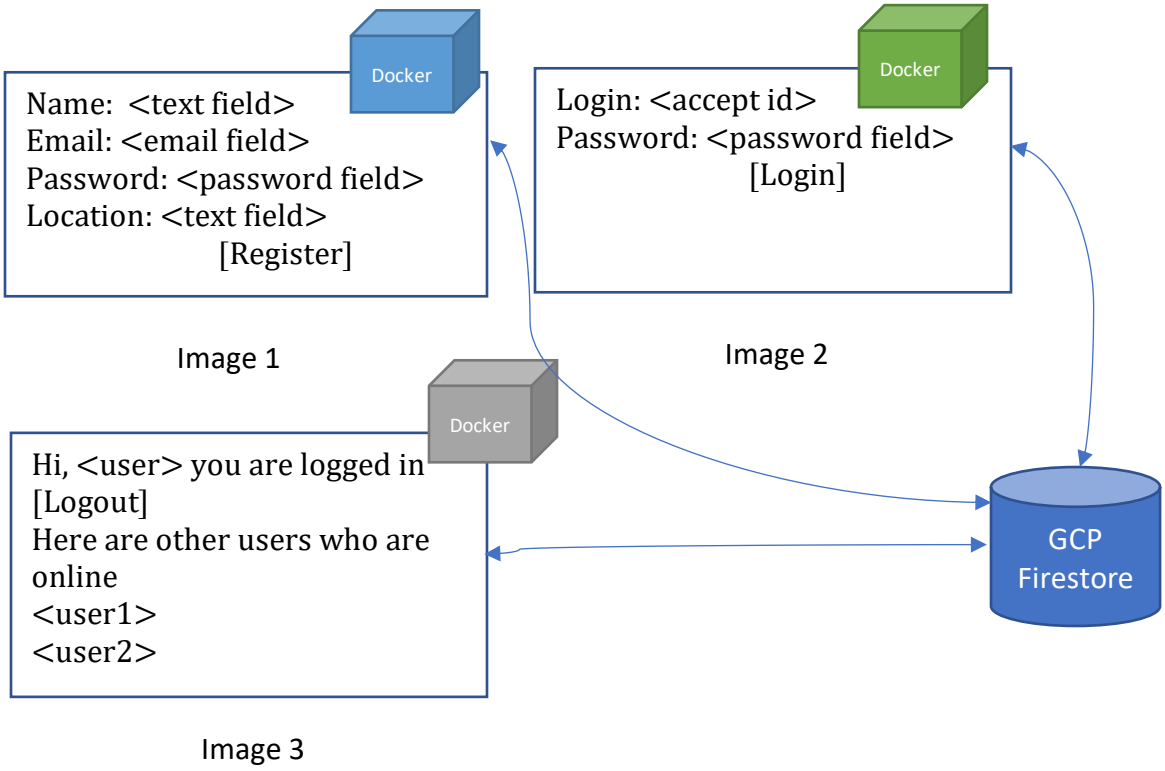
Part A. Build, deploy, and run a Containerized Application using GCP.

Using GCP create and validate an online meeting account.

take screenshots at every step and submit as part of the PDF:

a. Create three microservices and package in three containers, which will be responsible for the backend logic in this application. The database you will be using here is, **Firestore**
b. Code and required dependencies in **Container #1** are responsible for accepting registration details from frontend and store it in backend database. (image 1)
c. Code and required dependencies in **Container #2** are responsible for validating the Login information (image 2)
d. Once a user is logged in – the state of a user changes to online in the FireStore database, and it appears on the front page (image 3)
e. Your database should contain only 2 documents. One document to contain **registration data** (Name, Password, Email, Location), another document to contain user **state** (online, offline, timestamp etc.) information.
f. Code and required dependencies in **Container #3** are responsible for extracting state information from database. E.g. who is online. You need to maintain the session from login to logout. The session must expire after clicking the logout, which should update the **state** item in the Firestore database.
g. Once the docker images are built, you need to push those container images to artifact registry repository (GCR is deprecated now). Once it is done, you need to deploy those in **Cloud Run**.
h. In order complete the tasks, and perform interaction, you need build 3 simple web pages (or 1), using any technology of your choice.
i. Write test case to test your application and perform testing. Provide screenshots as evidence for all steps.

j. You need to study Google Cloud Run, GCR/ Artifact registry, Docker Container documentations, and write a summary within 1 page explaining how you have used these technologies in your application.



Part A - Submission requirement:

For (a to h), submit screenshots of every steps. Please do not exclude any steps.
Submit screenshots of empty and populated Firestore database
For (a to h), submit your program files (Source code on Gitlab) as well.
However, from the source code - add the important methods (pseudocode), or program instructions as part of the pdf. E.g. Login Validation method etc.
For (i), add the test cases and screenshots of tests in the pdf
For (j), add the summary as part of the pdf file

Part B. Building a Chatbot:

Using AWS academy account or AWS account perform the following:
take screenshots at every step

a. Using AWS Lex - Create a chatbot on **RideRequest**
b. Consider it as a "Taxi and Car rental service". (Assumptions: they have 3 types of vehicles –SUV, Sedan, Minivan.
c. The chatbot can accept information on car, such as **self-drive** or **Taxi**
d. If it is Taxi, then customer address, pickup date, and time are accepted
e. If it is self-drive, then assuming same day, it should ask arrival time of customer.

E.g.
Utterances – "I want to request a self-drive ride"
Prompts - "When are you coming to get your vehicle?"
Slots – "I will come around noon"
Prompts – "What do you want today (SUV, Sedan, Minivan)?"
Slots – "SUV"
Prompts – "How many?"
Slots – "1"
Fulfillment –
"You have requested for 1 SUV, and you will be arriving at 12:00 pm"
"Yes"
"Your request has been placed successfully"

Part B - Submission requirement:

A pdf file with the screenshots of AWS Lex chatbot creation, customization, test etc.
Paragraph explaining how the operation is performed.