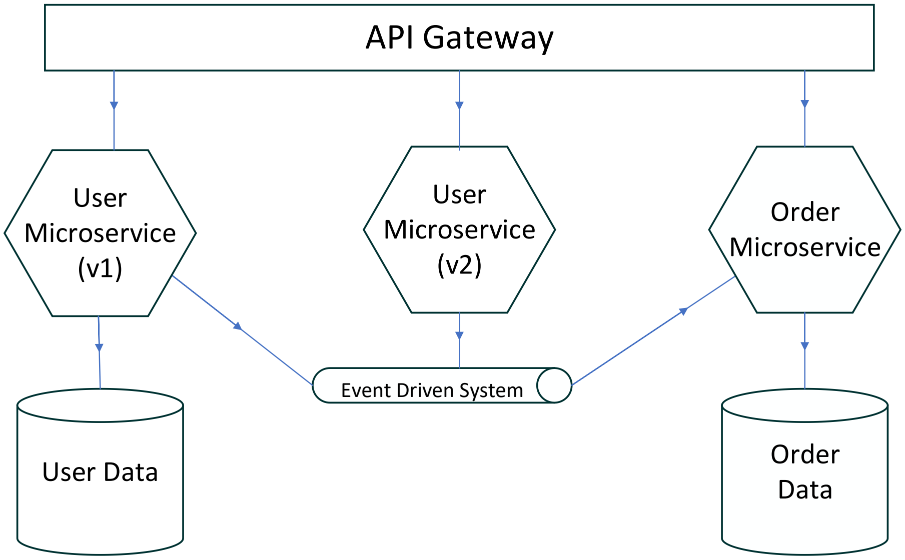
COEN 424/6313 Assignment Two Fall 2024

Individual or Group of 2 or 3 Assignment due by November 30th 23:59

This assignment is originally developed by Yan Liu @ Concordia University. This assignment is only for the course teaching and education purpose. Any distribution of this document to the Internet that involves any profit-making purpose is not given the consent from the author.

This assignment is designed to practise microservices and CI/CD deployment and MogoDB database. Program microservices and deploy the data synchronization solution. The context of the architecture is depicted below with requirements of the solutions.



(1) The user database contains a user's information including at least the user account id, emails and delivery address. The order database contains a set of items ordered, the user emails and delivery address, unique order id and order status ("under process","shipping" and "delivered"). The email and delivery address are synchronized across two databases.

(2) The API gateway routes the HTTP requests to microservices. The user microservice has a REST API or OpenAPI to handle POST requests to create a new user; PUT requests to update the email address or delivery address. The order microservice has a REST API or OpenAPI to handle GET requests to retrieve orders with a state; POST requests to create a new order with items and email address and delivery address; PUT requests to update the status("under process","shipping" and "delivered"); PUT requests to update emails or delivery address.

(3) When the email or delivery address is updated by a PUT request of the user microservice, the synchronization between the user database and the order database is triggered by an event in the event driven system.

(4) Introduce a new version (v2) of the user microservice. Some changes are refactored to the user microservice (v1).

(5) Develop the strangler pattern in the API gateway by routing P percentage of user requests to v1 and (1-P) percentage of user requests to v2. P can be specified in a configuration file associated with the API gateway. P should not be hardcoded in the API gateway.

(6) The user database and the order database can use MongoDB or Redis cloud service. The event driven system can use RabbitMQ.

(7) Choose monolith or microservice deployment using docker solutions. In the monolith deployment, API gateway, two microservices and the event driven system can be packed in one deployment configuration. In microservice development, API gateway, microservices and event driven system each has its own deployment configuration.

(8) Launch the docker deployment on a cloud VM and enable the APIs with cloud endpoints in (2)-(3) to accept HTTP requests.

(9) Bonus (optional) – Configure the github account to integrate with the docker deployment so that any commitment will trigger the auto-deployment and service launch in the cloud.

(10) Write up the solution report following the Template IEEE - Manuscript Templates for Conference Proceedings   https://www.ieee.org/conferences/publishing/templates.html. The report should provide sections for each solution of the above requirements (1) – (8), (9) optional. The report should provide necessary details for data models, API design and definition, stranger pattern implementation, and docker deployment solutions, cloud endpoint and token authentication configuration with diagrams and/or screenshots.

Member contribution list of each member's contribution according to the checklist below.

Name (SID) and Signature	Task List		Contribution Role and Percentage (X %) (If Y for the task list, write the contribution role and percentage counted for the completeness of this task)
Member 1	Rq (1)	Y / N	
	Rq (2)	Y / N	
	Rq (3)	Y / N	
	Rq (4)	Y / N	
	Rq (5)	Y / N	
	Rq (6)	Y / N	
	Rq (7)	Y / N	
	Rq (8)	Y / N	
	Rq (9)	Y / N	

Create **[SID_1]_[SID_2]_[SID_3]A2.zip** (.gz, .tar, .zip are accepted) file contains a folder named with your student ID. This folder contains all the source code + report.pdf for the assignment. One submission for the group is sufficient.