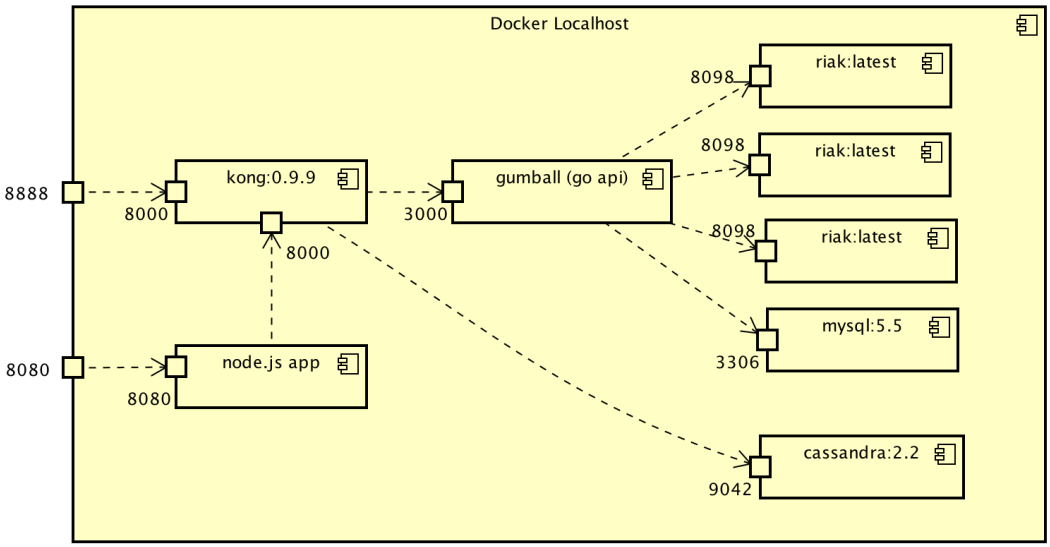
CMPE 281 - LAB #10 - Docker & Kubernetes (Part 2)

Lab Files:

- https://github.com/paulnguyen/cmpe281/tree/master/labs/lab10

Part 1 - Docker Compose (Single Network)



- In the **go-gumball-riak1** lab folder, deploy using **Docker Compose the Nodejs, Kong, Gumball, MySQL and Riak Cluster Stack**. Use the ***single network mode***l as shown in the UML diagram below.

- Make modifications to the **Node.js Compose Yaml to** change the deployment of Node.js App to use Environment variables (instead of hard coding the config an keys). Include the following:
  - ***gumball_endpoint***
  - ***order_endpoint***
  - ***apikey***
  - ***secretKey***

Observer the following:

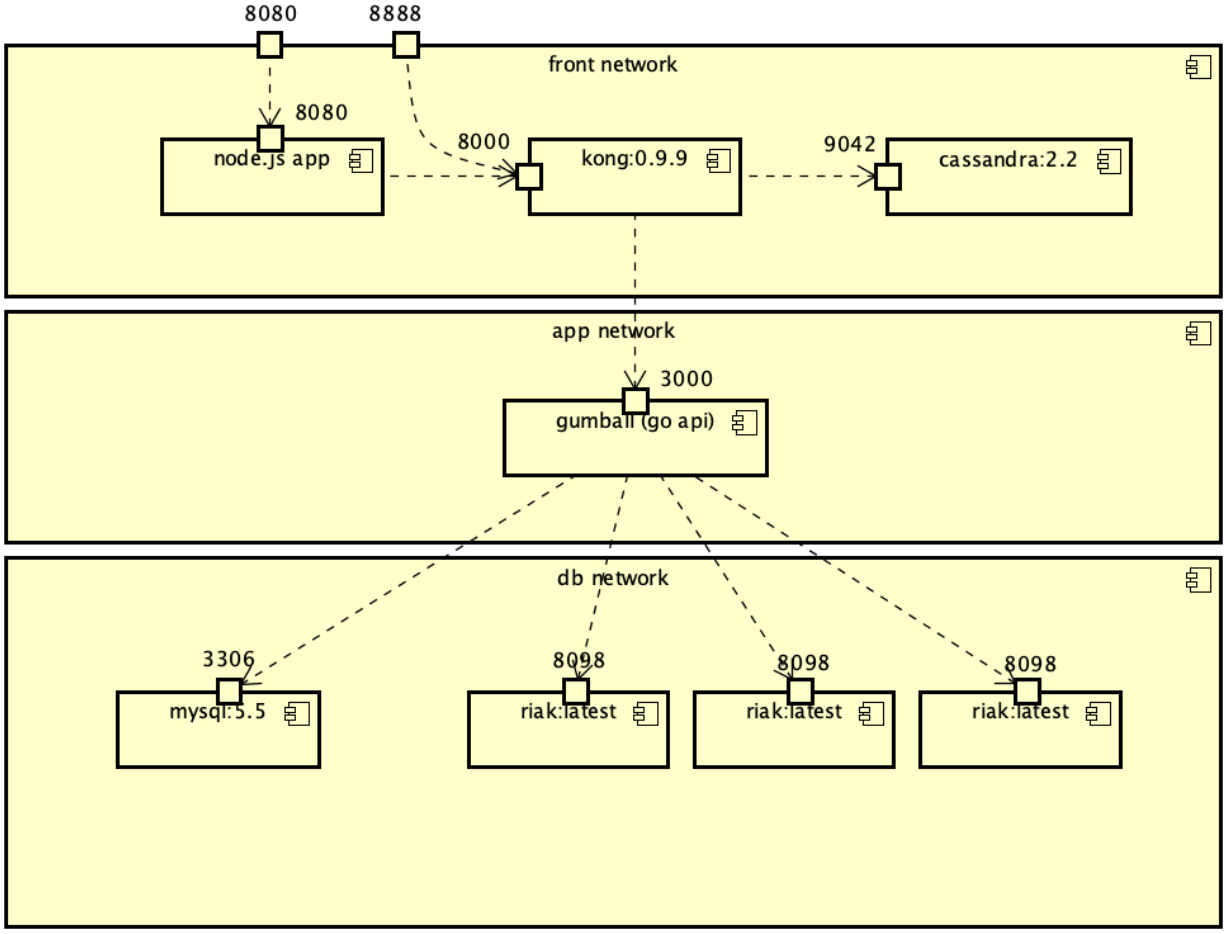
- **Kong API Gateway Test results (from Makefile). Make modifications to the Makefile if needed.**

- **Node.js Web App running on the Browser after placing one Order.**

I.E. Use the following Makefile Rules:

```
kong-test-ping:
curl -X GET \
http://localhost:8888/goapi/ping \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-inventory:
curl -X GET \
http://localhost:8888/goapi/gumball \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-place-order:
curl -X POST \
http://localhost:8888/goapi/order \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-get-order:
curl -X GET \
http://localhost:8888/goapi/order/$(id) \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-process-order:
curl -X POST \
http://localhost:8888/goapi/order/$(id) \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'
```

Part 2 - Docker Compose (Multi-Network Full Stack)



- In the **go-gumball-riak1** lab folder, deploy using **Docker Compose the Nodejs, Kong, Gumball, MySQL and Riak Cluster Stack**. Use the ***mullti-network model*** as shown in the UML diagram below. The Makefile and Docker Compose files are available in the **go-gumball-riak1/stack** lab folder.

- Use the same modifications you made to the **Node.js Compose Yaml to** change the deployment of Node.js App to use Environment variables (instead of hard coding the config an keys). Include the following:
  - ***gumball_endpoint***
  - ***order_endpoint***
  - ***apikey***
  - ***secretKey***

- **Observer Kong API Gateway Test results (from Makefile). Make modifications to the Makefile if needed.**

- **View Node.js Web App running on the Browser after placing one Order.**

I.E. Use the following Makefile Rules:

```
kong-test-ping:
curl -X GET \
http://localhost:8888/goapi/ping \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-inventory:
curl -X GET \
http://localhost:8888/goapi/gumball \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-place-order:
curl -X POST \
http://localhost:8888/goapi/order \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-get-order:
curl -X GET \
http://localhost:8888/goapi/order/$(id) \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'

kong-test-process-order:
curl -X POST \
http://localhost:8888/goapi/order/$(id) \
-H 'Content-Type: application/json' \
-H 'apikey: $(key)'
```

Part 3 - Kubernetes Gumball Stack

- In the **gumball_kubernetes** folder, follow the steps in the Markdown file and Makefile to deploy Pods, Deployments, Services for the ***Gumball API Stack***.

- Keep a "**Detailed Journal**" of **Screenshots** and **Command Line Terminal Output of each Step**. Make sure to also show the results of the Deployments or Deletions from the Kubernetes Dashboard.

- **Steps:**
    1. If you have not already, make sure to create the ***gumball namespace*** in Kubernetes. All the K8S Objects in this part of the lab are deployed to the gumball namespace.
    2. Use the ***Deployments*** and ***Service*** Yaml files (i.e. don't use the Pod Yaml files)
    3. Deploy the "**gumball-mongo**" and "**gumball-rabbitmq**" Services first
    4. Configure the Data in these two services (before deployment the "**gumball-service**")
    5. ***Note: you can use the Kubernetes Dashboard to get access to a Shell for the Mongo & Rabbitmq Containers***
    6. Build and Deploy your Gumball Service (Go API) to Docker Hub. Make sure to update the Gumball Deployment Yaml to point at your Docker Image.
    7. Deploy the "gumball-service" to Kubernetes. You will need to setup a "Jumpbox" in Kubernetes to test the API. Use the Jumpbox Pod Yaml to deploy the Jumpbox.
    8. Test the API from your Jump Box. The tests are available in the top level Makefile. These tests are also shown below.

```
jumpbox-ping:
curl http://gumball-service:9000/ping

jumpbox-get-inventory:
curl http://gumball-service:9000/gumball

jumpbox-update-inventory:
curl -X PUT \
http://gumball-service:9000/gumball \
-H 'Content-Type: application/json' \
-d '{ \
"CountGumballs": 1000 }'

jumpbox-place-order:
curl -X POST \
http://gumball-service:9000/order \
-H 'Content-Type: application/json'

jumpbox-order-status:
curl -X GET \
http://gumball-service:9000/order \
-H 'Content-Type: application/json'

jumpbox-process-order:
curl -X POST \
http://gumball-service:9000/orders \
-H 'Content-Type: application/json'
```