Final Project
Azure Container Service

Imtiaz, Shaquille

Deep Azure@McKesson
Dr. Zoran B. Djordjević

Motivation for On-Demand Build & Integration Environment

- With the adoption of Agile development, build and test environment provisioning has grown ever challenging to manage. Even large enterprises with large number of environment pools face challenges in re-use and orchestration of multi-version-line concurrent and staggered development lifecycle. Imagine the need of every commit to be able to be fully integration tested in a fully functional environment before allowing to be merged into the release line.

Azure Container Service (AKS) for On-Demand Build & Integration Environment

- Use Microsoft Azure Container Service (AKS), to create, configure, run and manage **containerized** code build (managing all build/run-time dependencies) and deployment (provisioning all runtime infrastructures) into an on-demand integration testing environment

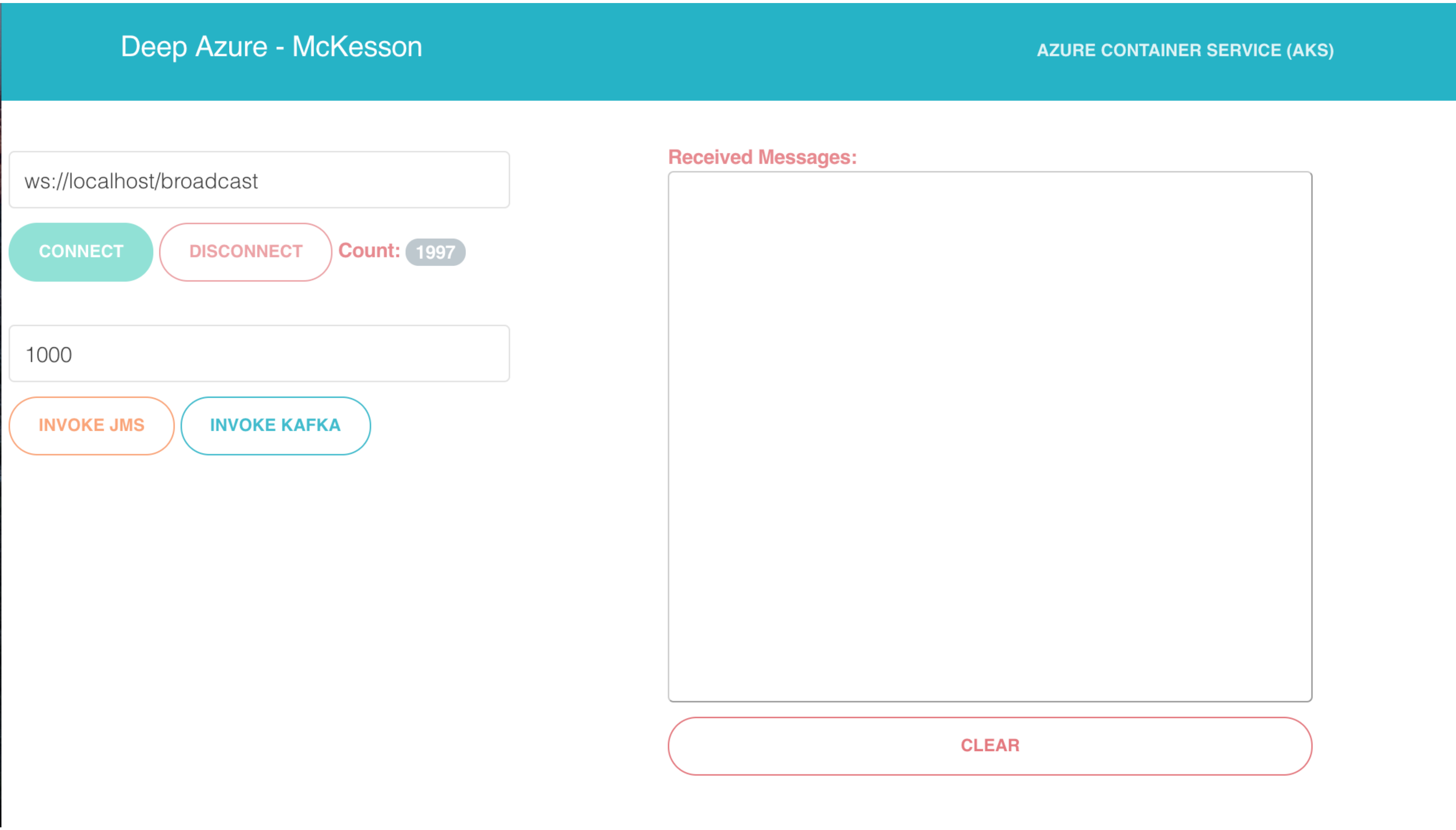
Azure Container Service (AKS)

- Azure Container Service (AKS) is a **free** managed **Kubernetes** container orchestration service that simplifies the deployment, management, and operations of containerized applications using **Docker** image format.
  - You do pay for the VMs and other resources, *the service is free*
- Currently available in public preview
  - us east, us central
- AKS facilitates managing containers without container expertise
  - Automatic binpacking
  - Self-healing
  - Horizontal scaling
  - Service discovery and load balancing
  - Automated rollouts and rollbacks
  - Secret and configuration management
  - Storage orchestration
  - Batch execution

The Sample App

- The sample app is a message broadcast application where a web based client subscribes to receive real-time messages from it’s service layer. The service layer, which in turn consumes messages via Kafka and Apache ActiveMQ and broadcasts the messages to connected clients.
- Uses maven to build
- Has a quite a number of **build time and runtime dependencies**
- And off course, it has Apache Kafka and ActiveMQ runtime **infrastructure dependencies**

The Sample App



ws://localhost/broadcast

Count: 1997

1000

Received Messages:

Solution

- Dockerized the actual maven build process and the bootstrap invocation of the application (*Dockfile yml*)
- *…Tested locally*
- Used Docker-Compose tool to create a multi container application stack *(docker-compose yml)*
    - SP-KAFKA
    - SP-ActiveMQ
    - SP-WEB
    - A maven repository shared volume, so that the maven artifacts are not downloaded every time
- *…Tested locally*
- Registered the container stack into Azure Container Registry (ACR)
- Used Kubernetes Deployment file *(sp-web-all-in-one yml)* to deploy the registered stack into Azure Container Service (AKS)
- *…Trouble-shoot authentication issues for AKS to talk to ACR*
- ***Yay it works!***

DEMO

- *Hopefully it works!*
  - *'Good grief' if it does not*
  - *'Very good' if it does*

- *THANKS*
  - *Dr. Zoran and the TA team*
  - *Fellow classmates*
  - *Fellow McKesson-ians that made the course possible*
  - *https://azure.microsoft.com/en-us/services/container-service/*
  - *https://docs.docker.com/*
  - *https://kubernetes.io/docs*
  - *https://kafka.apache.org/*
  - *http://activemq.apache.org/*

Appendix A (AKS related commands)

- *az group create --name shaqsRg --location eastus*
- *az acr create --resource-group shaqsRg --name shaqsAcr --sku Basic*
- *az acr login --name shaqsAcr*
- *docker tag sp-web shaqsacr.azurecr.io/sp-web:v1*
- *docker push shaqsacr.azurecr.io/sp-web:v1*
- *az acr list --resource-group shaqsRg --query "[].{acrLoginServer:loginServer}" --output table*
- *az acr repository show-tags --name shaqsAcr --repository sp-web --output table*
- *az aks create --resource-group shaqsRg --name shaqsAKSCluster --node-count 1 --generate-ssh-keys*
- *az aks get-credentials --resource-group=shaqsRg --name=shaqsAKSCluster*
- *kubectl get nodes*
- *~/bin/kubeAcrAuth2*
- *kubectl create -f sp-web-all-in-one.yml*
- *kubectl get service sp-web —watch*

YouTube URLs, GitHub URL, Last Page

- Two minute (short):
  - https://youtu.be/NFy2QVOU43w
- 15 minutes (long):
  - https://youtu.be/jAwrXA96bx0
- GitHub Repository with all artifacts:
  - https://github.com/java-stack/sp