Luck Quiz?

목 차

I. 빌드 및 배포

1. 개발 환경

Server1: AWS EC2 Ubuntu 20.04 LTS

Server2: AWS EC2 Ubuntu 20.04 LTS
(EC2[xlarge] - CPU : 4vCPUs, RAM : 16GB, SSD : 320GB SSD, HDD : 6 TB)

Server3: AWS EC2 Ubuntu 20.04 LTS
(EC2[t2.2xlarge] - CPU : 8vCPUs, RAM : 32GB, SSD : 320GB SSD, HDD : 6 TB)

Visual Studio Code: 1.75.1

IntelliJ IDEA: 2022.3.1 (Ultimate Edition)

JVM : OpenJDK 11

Docker : 23.0.5

Node.js : 18.15.0

TypeScript: 4.9.5

Redis :

Nginx :

Jenkins :

Azure :

Kubernetes:

2. 설정 파일 목록 환경 변수 정보

Frontend:

- Dockerfile: /front
- .env: /front
- front.conf: /front

Backend_Auth:

- Dockerfile: /back/auth
- application.yml: kubernetes secret
- application.yml: (develop) /back/auth/src/main/resources

Backend_Quiz:

- Dockerfile: /back/quiz
- application.yml: kubernetes secret
- application.yml: (develop) /back/quiz/src/main/resources

Backend_QuizRoom:

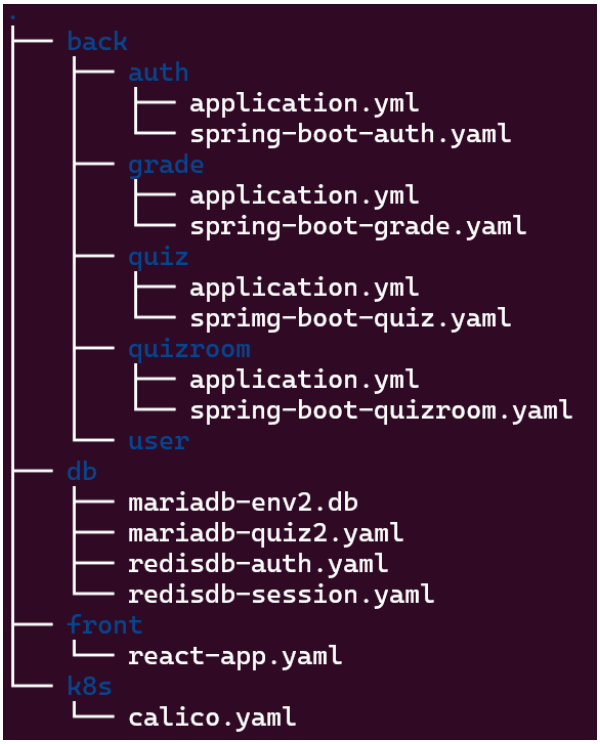
- Dockerfile: /back/quizroom
- application.yml: kubernetes secret
- application.yml: (develop) /back/quizroom/src/main/resources

Backend_Grade:

- Dockerfile: /back/grade
- application.yml: kubernetes secret
- application.yml: (develop) /back/grade/src/main/resources

Kubernetes - Control Plane:

Kubernetes - Work Node:



3. 쿠버네티스 설정

Nginx

Ingress-nginx.yaml:

```
curl -o ingress-nginx.yaml
https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.6.
4/deploy/static/provider/baremetal/deploy.yaml

# NodePort에 80, 443 포트 할당
vi ingress-nginx.yaml
```

```
# Service 부분에 nodePort 추가
~~~
ports:
  - appProtocol: http
    name: http
    port: 80
    protocol: TCP
    targetPort: http
    nodePort: 80
  - appProtocol: https
    name: https
    port: 443
    protocol: TCP
    targetPort: https
    nodePort: 443
~~~

# 마스터노드에 생성하려면 nodeSelector에 다음 내용 추가
~~~
spec:
  nodeSelector:
    node-role.kubernetes.io/control-plane: ""
~~~
```

Mariadb

mariadb-env2.db:

```
MYSQL_HOST=%
MYSQL_PORT=3308
MYSQL_ROOT_PASSWORD=ekdrmsehdrms1111!!
MYSQL_DATABASE=luckquiz
MYSQL_USER=carrot707
MYSQL_PASSWORD=ekdrmsehdrms1111!!
```

mariadb-quiz2.yaml:

```yaml
apiVersion: v1
kind: Service
metadata:
  name: mariadb-quiz2
spec:
  type: NodePort
  ports:
    - protocol: TCP
      name: mariadb-quiz2
      port: 3306
      targetPort: 3306
      nodePort: 3308
  selector:
    app: mariadb-quiz2
---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mariadb-pv2
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/data/mariadb2"
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mariadb-pv-claim2
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
---
```

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mariadb-quiz2
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mariadb-quiz2
  template:
    metadata:
      labels:
        app: mariadb-quiz2
    spec:
      nodeSelector:
       node-role.kubernetes.io/control-plane: "" #create at master node
      containers:
        - image: mariadb:latest
          name: mariadb-quiz2
          envFrom:
            - secretRef:
                name: mariadb-bdg2
          ports:
            - containerPort: 3306
              name: mariadb2
          volumeMounts:
            - name: mariadb-persistent-storage2
              mountPath: /var/lib/mysql
      volumes:
        - name: mariadb-persistent-storage2
          persistentVolumeClaim:
            claimName: mariadb-pv-claim2
```

Redisdb

redisdb-session.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: redisdb-session
spec:
  type: NodePort
  ports:
    - protocol: TCP
      name: redisdb-session
      port: 6379
      targetPort: 6379
      nodePort: 3309
  selector:
    app: redisdb-session
---
```

```yaml
---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: redisdb-pv2
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 2Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/data/redisdb/quiz"
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: redisdb-pv-claim2
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
      #allowVolumeExpansion: true
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: redisdb-session
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redisdb-session
  template:
    metadata:
      labels:
        app: redisdb-session
    spec:
      nodeSelector:
              #node-role.kubernetes.io/control-plane: "" #create at master node
      containers:
        - image: redis
          name: redisdb-session
          args: ["--requirepass", "eodrms1111!"]
          ports:
            - containerPort: 6379
              name: redisdb
          volumeMounts:
            - name: redisdb-persistent-storage2
              mountPath: /var/lib/mysql
      volumes:
        - name: redisdb-persistent-storage2
          persistentVolumeClaim:
            claimName: redisdb-pv-claim2
```

kubectl create secret generic spring-boot-{service}-secret -from-file="application.yml" -n {namespace}

AuthServer

Spring-boot-auth.yaml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: spring-boot-auth
  namespace: spring-boot-v2
spec:
  replicas: 1
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: spring-boot-auth
  template:
    metadata:
      labels:
        app: spring-boot-auth
    spec:
      containers:
      - name: spring-boot-auth
        image: docker.io/carrot707/luckquiz:spring-boot-auth-2
        ports:
        - containerPort: 8080
        volumeMounts:
        - name: secret-volume
          mountPath: /config/secret
          readOnly: true
        env:
        - name: SPRING_CONFIG_LOCATION
          value: "file:/config/secret/application.yml"
      volumes:
      - name: secret-volume
        secret:
          secretName: spring-boot-auth-secret
      imagePullSecrets:
      - name: dockerhub-secret

---
```

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: spring-boot-auth
  namespace: spring-boot-v2
spec:
  selector:
    app: spring-boot-auth
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
  type: ClusterIP
---
apiVersion: v1
kind: Service
metadata:
  name: auth-external-v2
  namespace: default
spec:
  type: ExternalName
  externalName: spring-boot-auth.spring-boot-v2.svc.cluster.local
  ports:
    - name: http
      port: 8080
```

GradeServer

Spring-boot-grade.yaml:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: spring-boot-grade
  namespace: spring-boot-v2
spec:
  replicas: 1
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: spring-boot-grade
  template:
    metadata:
      labels:
        app: spring-boot-grade
    spec:
      containers:
      - name: spring-boot-grade
        image: docker.io/carrot707/luckquiz:spring-boot-grade-4
        ports:
        - containerPort: 8080
        volumeMounts:
        - name: secret-volume
          mountPath: /config/secret
          readOnly: true
        env:
        - name: SPRING_CONFIG_LOCATION
          value: "file:/config/secret/application.yml"
      volumes:
      - name: secret-volume
        secret:
          secretName: spring-boot-grade-secret
      imagePullSecrets:
      - name: dockerhub-secret

---
```

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: spring-boot-grade
  namespace: spring-boot-v2
spec:
  selector:
    app: spring-boot-grade
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
  type: ClusterIP
---
apiVersion: v1
kind: Service
metadata:
  name: grade-external-v2
  namespace: default
spec:
  type: ExternalName
  externalName: spring-boot-grade.spring-boot-v2.svc.cluster.local
  ports:
    - name: http
      port: 8080
```

QuizServer

Spring-boot-quiz.yaml:

```yaml
metadata:
  name: spring-boot-quiz
  namespace: spring-boot-v2
spec:
  replicas: 1
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: spring-boot-quiz
  template:
    metadata:
      labels:
        app: spring-boot-quiz
    spec:
      containers:
      - name: spring-boot-quiz
        image: docker.io/carrot707/luckquiz:spring-boot-quiz-28
        ports:
        - containerPort: 8080
        volumeMounts:
        - name: secret-volume
          mountPath: /config/secret
          readOnly: true
        env:
        - name: SPRING_CONFIG_LOCATION
          value: "file:/config/secret/application.yml"
      volumes:
      - name: secret-volume
        secret:
          secretName: spring-boot-quiz-secret
      imagePullSecrets:
      - name: dockerhub-secret
```

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: spring-boot-quiz
  namespace: spring-boot-v2
spec:
  selector:
    app: spring-boot-quiz
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
  type: ClusterIP
---
apiVersion: v1
kind: Service
metadata:
  name: quiz-external-v2
  namespace: default
spec:
  type: ExternalName
  externalName: spring-boot-quiz.spring-boot-v2.svc.cluster.local
  ports:
    - name: http
      port: 8080
```

QuizRoomServer

Spring-boot-quizroom.yaml:

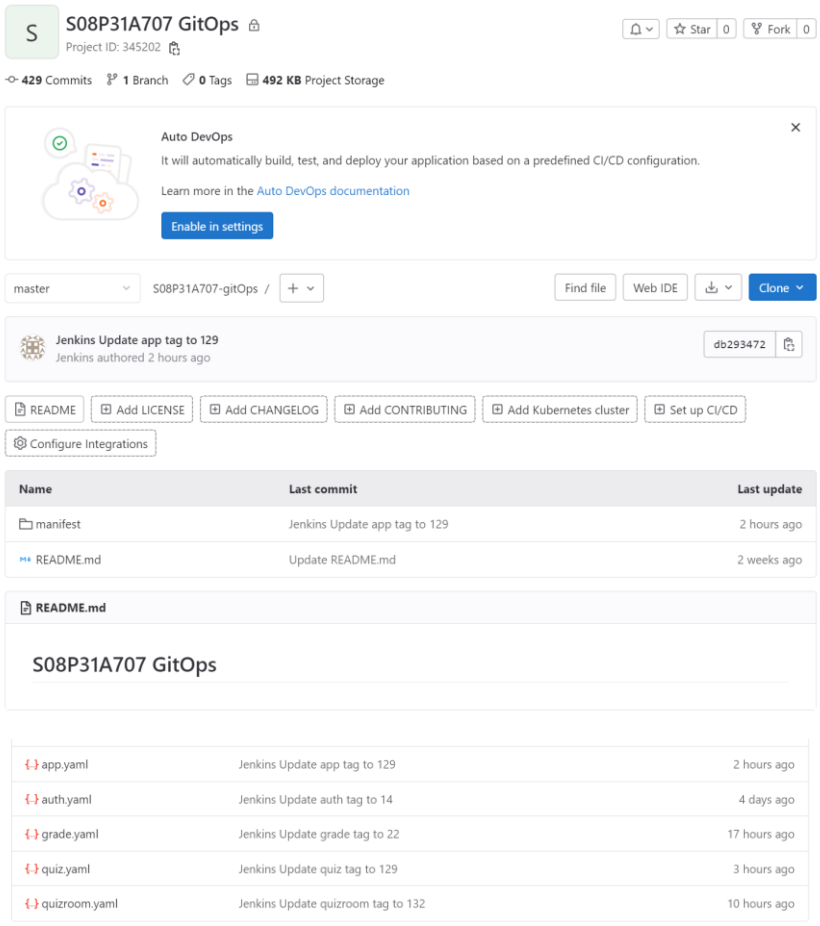
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: spring-boot-quizroom
  namespace: spring-boot-v2
spec:
  replicas: 1
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: spring-boot-quizroom
  template:
    metadata:
      labels:
        app: spring-boot-quizroom
    spec:
      containers:
      - name: spring-boot-quizroom
        image: docker.io/carrot707/luckquiz:spring-boot-quizroom-1
        ports:
        - containerPort: 8080
        volumeMounts:
        - name: secret-volume
          mountPath: /config/secret
          readOnly: true
        env:
        - name: SPRING_CONFIG_LOCATION
          value: "file:/config/secret/application.yml"
      volumes:
      - name: secret-volume
        secret:
          secretName: spring-boot-quizroom-secret
      imagePullSecrets:
      - name: dockerhub-secret

---
```

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: spring-boot-quizroom
  namespace: spring-boot-v2
spec:
  selector:
    app: spring-boot-quizroom
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
  type: ClusterIP
---
apiVersion: v1
kind: Service
metadata:
  name: quizroom-external-v2
  namespace: default
spec:
  type: ExternalName
  externalName: spring-boot-quizroom.spring-boot-v2.svc.cluster.local
  ports:
    - name: http
      port: 8080
```


3. GitOps 및 ArgoCD

GitOps



3. GitOps 및 ArgoCD

app.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: react-app
  namespace: react-v2
spec:
  replicas: 1
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: react-app
  template:
    metadata:
      labels:
        app: react-app
    spec:
      containers:
      - name: react-app
        image: docker.io/carrot707/luckquiz:react-app-129
        ports:
        - containerPort: 3000
      imagePullSecrets:
      - name: dockerhub-secret
```

auth.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: spring-boot-auth
  namespace: spring-boot-v2
spec:
  replicas: 1
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: spring-boot-auth
  template:
    metadata:
      labels:
        app: spring-boot-auth
    spec:
      containers:
      - name: spring-boot-auth
        image: docker.io/carrot707/luckquiz:spring-boot-auth-14
        ports:
        - containerPort: 8080
        volumeMounts:
        - name: secret-volume
          mountPath: /config/secret
          readOnly: true
        env:
        - name: SPRING_CONFIG_LOCATION
          value: "file:/config/secret/application.yml"
      volumes:
      - name: secret-volume
        secret:
          secretName: spring-boot-auth-secret
      imagePullSecrets:
      - name: dockerhub-secret
```


3. 카프카 설정

Apache Kafa 다운로드
압축 해제

kafka_2.13-3.4.0/config/zookeeper.properties

kafka_2.13-3.4.0/config/server.properties

주키퍼 백그라운드 실행

카프카 백그라운드 실행
kafdrop 설치 및 실행

4. Frontend 배포

Apache Kafa 다운로드
압축 해제

kafka_2.13-3.4.0/config/zookeeper.properties

kafka_2.13-3.4.0/config/server.properties

주키퍼 백그라운드 실행

카프카 백그라운드 실행
kafdrop 설치 및 실행

4. Backend 배포

Apache Kafa 다운로드
압축 해제

kafka_2.13-3.4.0/config/zookeeper.properties

kafka_2.13-3.4.0/config/server.properties

주키퍼 백그라운드 실행

카프카 백그라운드 실행
kafdrop 설치 및 실행