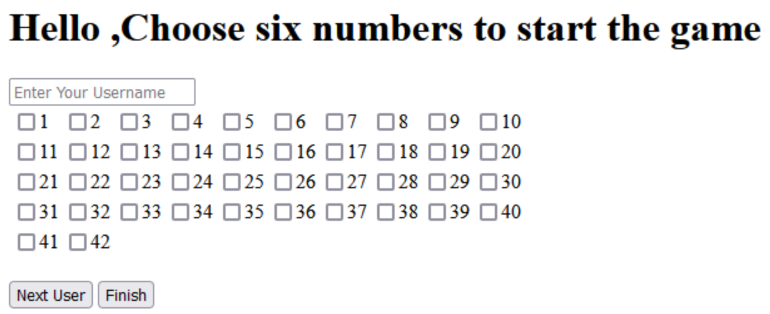
Kubernetes - project

Lottery - Webapp

Lottery -webapp

I have Developed a simple web application composed of 2 screens one for saving new tickets(of 6 numbers) with the username (unique username) and another screen to display results of draw by date.

1



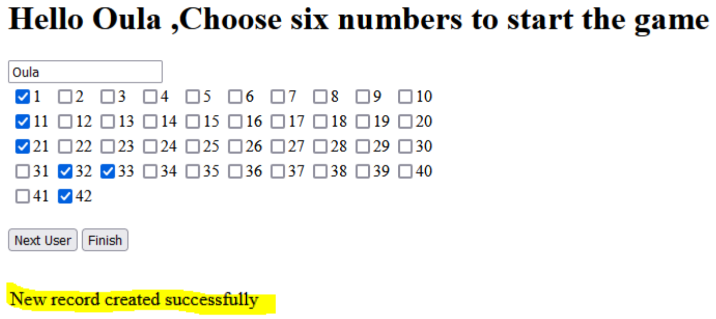
Hello ,Choose six numbers to start the game

Enter Your Username

☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10
☐11 ☐12 ☐13 ☐14 ☐15 ☐16 ☐17 ☐18 ☐19 ☐20
☐21 ☐22 ☐23 ☐24 ☐25 ☐26 ☐27 ☐28 ☐29 ☐30
☐31 ☐32 ☐33 ☐34 ☐35 ☐36 ☐37 ☐38 ☐39 ☐40
☐41 ☐42

Next User | Finish

2



Hello Oula ,Choose six numbers to start the game

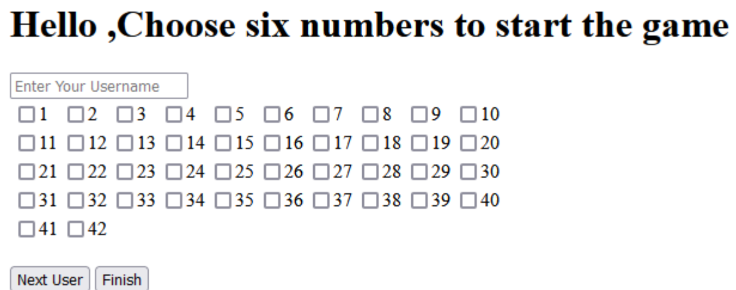
Oula

☑1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10
☑11 ☐12 ☐13 ☐14 ☐15 ☐16 ☐17 ☐18 ☐19 ☐20
☑21 ☐22 ☐23 ☐24 ☐25 ☐26 ☐27 ☐28 ☐29 ☐30
☐31 ☑32 ☑33 ☐34 ☐35 ☐36 ☐37 ☐38 ☐39 ☐40
☐41 ☑42

Next User | Finish

New record created successfully

3



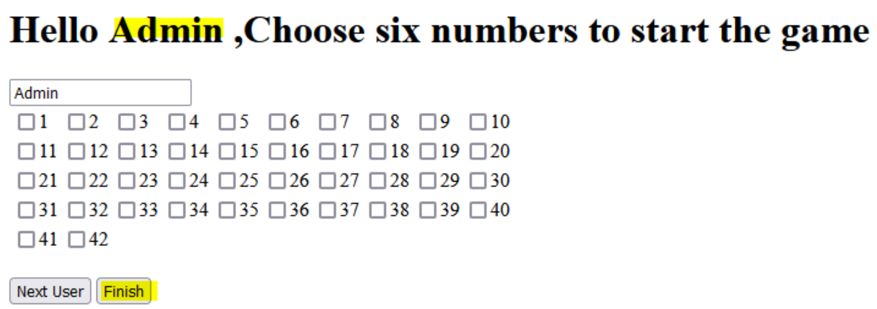
Hello ,Choose six numbers to start the game

Enter Your Username

☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10
☐11 ☐12 ☐13 ☐14 ☐15 ☐16 ☐17 ☐18 ☐19 ☐20
☐21 ☐22 ☐23 ☐24 ☐25 ☐26 ☐27 ☐28 ☐29 ☐30
☐31 ☐32 ☐33 ☐34 ☐35 ☐36 ☐37 ☐38 ☐39 ☐40
☐41 ☐42

Next User | Finish

4



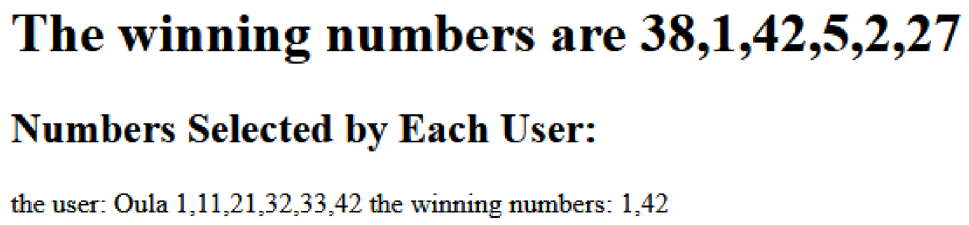
Hello Admin ,Choose six numbers to start the game

Admin

☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10
☐11 ☐12 ☐13 ☐14 ☐15 ☐16 ☐17 ☐18 ☐19 ☐20
☐21 ☐22 ☐23 ☐24 ☐25 ☐26 ☐27 ☐28 ☐29 ☐30
☐31 ☐32 ☐33 ☐34 ☐35 ☐36 ☐37 ☐38 ☐39 ☐40
☐41 ☐42

Next User | Finish

5



The winning numbers are 38,1,42,5,2,27

Numbers Selected by Each User:

the user: Oula 1,11,21,32,33,42 the winning numbers: 1,42

Lottery

I have used aws machine (EC2) and prepared the Ubuntu machine environment by intalling docker container, Jenkins,mysql , apache & PHP, minikube and kubernetes container (kubectl, kubeadm, kubelet).

Using my github account I have push my application PHP code and pull it to my machine from my repository "oulahn/Lottery"

```
git clone https://github.com/oulahn/Lottery
cd Lottery
git pull  (in case of modification)
```

Docker compose Yaml file

I have created the docker compose yaml file file composed of the deployment of my app and the mysql db to create the corresponding images :

- Webapp (using service)
- Database_server (using ip pod)

```
version: '3.9'
services:
  webapp:
    build: '.'
    ports:
      - 80:80
    networks:
      - devops
  database_server:
    image: mysql
    environment:
      MYSQL_ROOT_PASSWORD: "password"
      MYSQL_DATABASE: "LottoDB"
      MYSQL_USER: "oulahn"
      MYSQL_PASSWORD: "Password@123#"
    networks:
      - devops
networks:
  devops:
    driver: bridge
```

Enable nginx

enable nginx from kubernetes:

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.1.0/deploy/static/provider/baremetal/deploy.yaml
```

```
oulahn@ip-172-31-5-56:~$ kubectl get pods -n ingress-nginx
```

Create webapp-deployment

Notes:

- imagePullPolicy: "Always“ : to be able to re-build the image in case of any change (in code)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: oulahn-webapp
spec:
  replicas: 1
  selector:
    matchLabels:
      app: oulahn-webapp
  template:
    metadata:
      labels:
        app: oulahn-webapp
    spec:
      containers:
        - name: oulahn-webapp-1
          image: oulahn/dlottery:lotto
          imagePullPolicy: "Always"
          ports:
            - containerPort: 80
```

Create webapp-service.yaml

```
kinapiVersion: v1
d: Service
metadata:
  name: webapp-service
spec:
  selector:
    app: oulahn-webapp
  ports:
    - name: http
      port: 80
      targetPort: 80
      protocol: TCP
      #nodePort: 31500
  type: LoadBalancer
```

Create webapp-ingress.yaml

Enable ingress in minikube :

```
minikube addons enable ingress
```

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: webapp-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  defaultBackend:
    service:
      name: webapp-service
      port:
        number: 80
  rules:
    - host: oulahn-lotto.com
    - http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: webapp-service
                port:
                  number: 80
```

Deploy the app

```
kubectl apply -f webapp-deployment.yaml

kubectl apply -f webapp-service.yaml

kubectl apply -f webapp-ingress.yaml
```

Deploy the db-deployment.yaml

- The db is persistent and mounted to PV
- Use secret and configmap files

Mysql-secret.yaml

```
apiVersion: v1
kind: Secret
metadata:
  name: mysql-secret
type: Opaque
data:
  MYSQL_ROOT_PASSWORD: cGFzc3dvcmQ=
  MYSQL_PASSWORD: UGFzc3dvcmRAMTIzIw==
```

Configmap.yaml

```
apiVersion: v1
kind: ConfigMap
metadata:
   name: webapp-config
data:
  DB_HOST : database-server
  DB_PORT : "3306"
  DB_DATABASE : LottoDB
  DB_USER : oulahn
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: db
  labels:
    app: db
spec:
  replicas: 1
  selector:
    matchLabels:
      app: db
  template:
    metadata:
      labels:
        app: db
    spec:
      containers:
      - name: db
        image: mysql
        ports:
          - containerPort: 3306
        env:
          - name:
MYSQL_ROOT_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mysql-secret
                key:
MYSQL_ROOT_PASSWORD
          - name: MYSQL_USER
            valueFrom:
secretKeyRef:
                name: mysql-secret
                key: MYSQL_ROOT_PASSWORD
          - name: MYSQL_USER
            valueFrom:
              configMapKeyRef:
                name: webapp-config
                key: DB_USER
          - name: MYSQL_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mysql-secret
                key: MYSQL_PASSWORD
          - name: MYSQL_DATABASE
            valueFrom:
              configMapKeyRef:
                name: webapp-config
                key: DB_DATABASE
        volumeMounts:
        - name: mysql-pv-volume
          mountPath: /var/lib/mysql
      volumes:
      - name: mysql-pv-volume
        hostPath:
          path: /database
```

Deploy the db-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: database-server
spec:
  selector:
    app: db
  ports:
    - protocol: TCP
      port: 3306
      targetPort: 3306
```

```
kubectl apply -f db-deployment.yaml

kubectl apply -f db-service.yaml
```

Deploy the application under multiple pods
multi-pod.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: oulahn-webapp
spec:
  replicas: 3
  selector:
    matchLabels:
      app: oulahn-webapp
  template:
    metadata:
      labels:
        app: oulahn-webapp
    spec:
      containers:
      - name: oulahn-webapp-container
        image: oulahn/dlottery:lotto
        ports:
        - containerPort: 80
```

Docker image build to docker hub

```
docker build -t oulahn-webapp .
docker tag oulahn-webapp:latest oulahn/dlottery:lotto
docker push oulahn/dlottery:lotto
```

docker compose up -d(to run the docker image in background)

Get deployments

kubectl get deployments

```
NAME            READY   UP-TO-DATE   AVAILABLE   AGE
database-server   1/1     1            1           8d
oulahn-webapp     1/1     1            1           8d
```

kubectl get ingress

```
NAME            CLASS   HOSTS              ADDRESS        PORTS   AGE
oulahn-webapp   nginx   oulahn-lotto.com   192.168.58.2   80      8d
```

kubectl get service

```
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
database-server   ClusterIP   10.105.119.89   <none>        3306/TCP   8d
kubernetes        ClusterIP   10.96.0.1       <none>        443/TCP    11d
oulahn-webapp     ClusterIP   10.109.35.222   <none>        80/TCP     8d
```

docker images

```
REPOSITORY        TAG      IMAGE ID       CREATED       SIZE
oulahn/dlottery   lotto    420d43dc955c   8 days ago    460MB
mysql             latest   412b8cc72e4a   3 weeks ago   531MB
```

kubectl get pods

```
NAME                               READY   STATUS    RESTARTS   AGE
database-server-649c979ccd-ggfcn   1/1     Running   0          8d
oulahn-webapp-5749685dcd-6p4qs     1/1     Running   0          8d
```

helm chart

1. Create helm chart repo named dlottery:

```
helm create dlottery
```
(or helm)

2. Under templates I have placed the corresponding helm chart configuration including configmap and secrets file .

```
~/dlottery/templates$ ls
 NOTES.txt     configmap.yaml      hpa.yaml      secrets.yaml  serviceaccount.yaml  webapp-deployment.yaml
_helpers.tpl  db-deployment.yaml  ingress.yaml  service.yaml  tests
```

3. Under values.yaml I have configured the webapp with the database where the file I will deploy .

4. install helm chart app

```
helm install oulahn-webapp dlottery/ --values dlottery/values.yaml
```
Or from
```
helm install oulahn-webapp helm/ --values helm/values.yaml
```

<u>to uninstall helm chart app:</u>

```
helm uninstall oulahn-webapp -n default
```

Export the Pod Node Port and IP Address for helm chart
===
```
export NODE_PORT=$(kubectl get --namespace default -o jsonpath="{.spec.ports[0].nodePort}" services oulahn-
webapp)
```

output of install should be :

```
-----------------
NAME: oulahn-webapp-o
LAST DEPLOYED: Wed Apr 19
22:07:10 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running
these commands:
  http://oulahn-lotto.com/
```

Load test

```
wget https://downloads.apache.org/jmeter/binaries/apache-jmeter-5.5.tgz
```

```
tar -xzf apache-jmeter-5.5.tgz
```

```
cd apache-jmeter-5.5/
```

after configur the jmx file "mytestplan.jmx" , we run the below command to **start testing :**

```
jmeter -n -t ~/mytestplan.jmx -l ~/testresult.csv
```

result will be saved in csv file : testresult.csv

Autoscaling

I have used CPU usage as the metric to trigger the autoscaling . We will scale up when CPU usage exceeds 70% and scale down when it drops below 50%.I have run the below command to create a HPA resource :

```
kubectl autoscale deployment oulahn-webapp --cpu-percent=70 --min=1 --max=10
```

I have amended the webapp-deployment.yaml file by adding the resource

```
 resources:
          # requests:
          #   cpu: 50m
          limits:
            cpu: 50m
```

Same for the deployment file in template inside the helm chart directory by adding :

```
{{- if not .Values.autoscaling.enabled }}
  replicas: {{ .Values.replicaCount }}
  {{- end }}
```

To enable autoscaling

Db access

```
access db
===========
kubectl get svc (to get the server name)
kubectl get pods (to get db pod)
kubectl exec -it database-server-649c979ccd-2gkn7 -- bash

mysql -h database-server -u oulahn -pPassword@123#
use LottoDB;
Select * from draw; (to show the persistent data in draw table)
show tables;
open app : http://oulahn-lotto.com/ on firefox
oulahn-lotto.com
```

Open app using firefox

```
ssh -i devops.pem ubuntu@43.207.231.151 -D1234
```

App URL: http://oulahn-lotto.com/