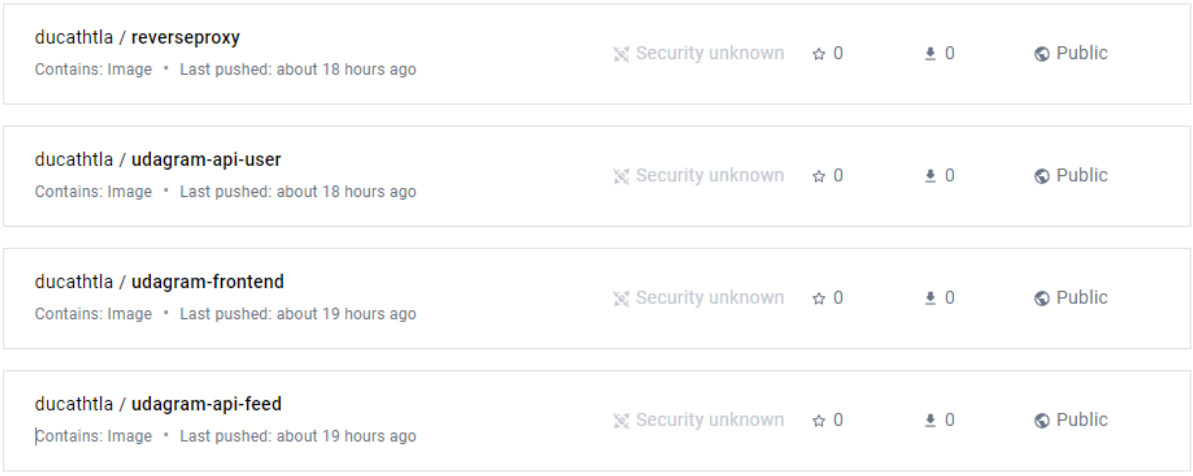
Project 3: Refactor Monolith to Microservices and Deploy

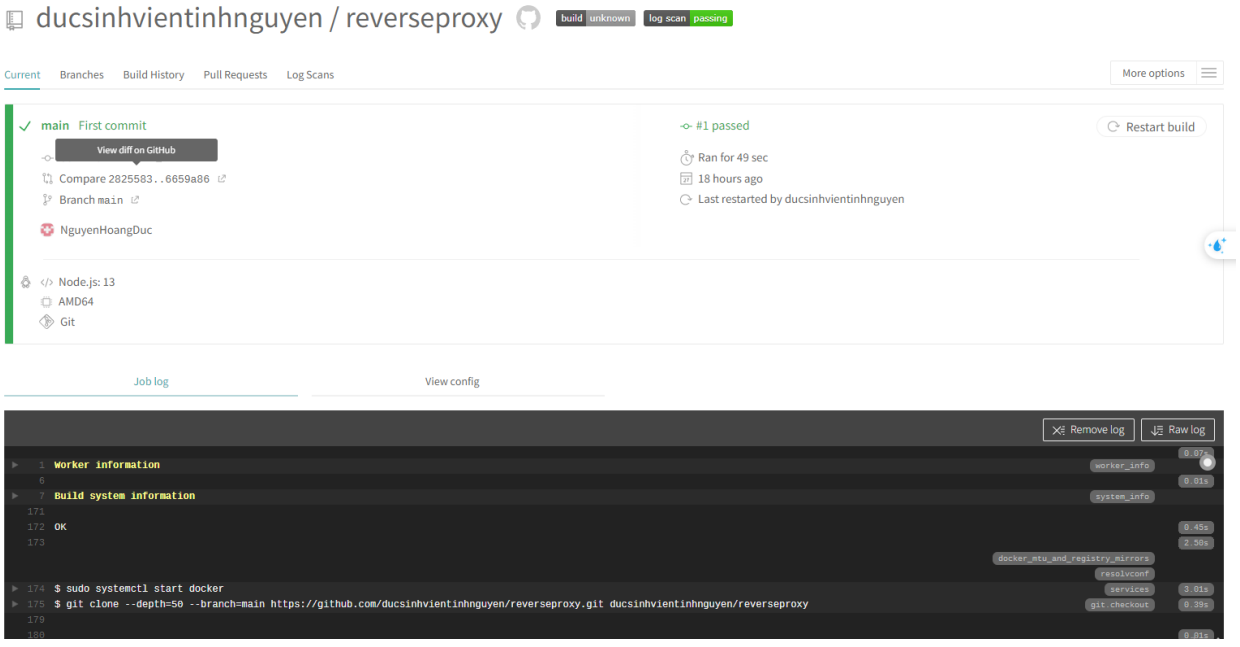
Project Rubric:

I. Dockerhub: I have push docker images in my repository.

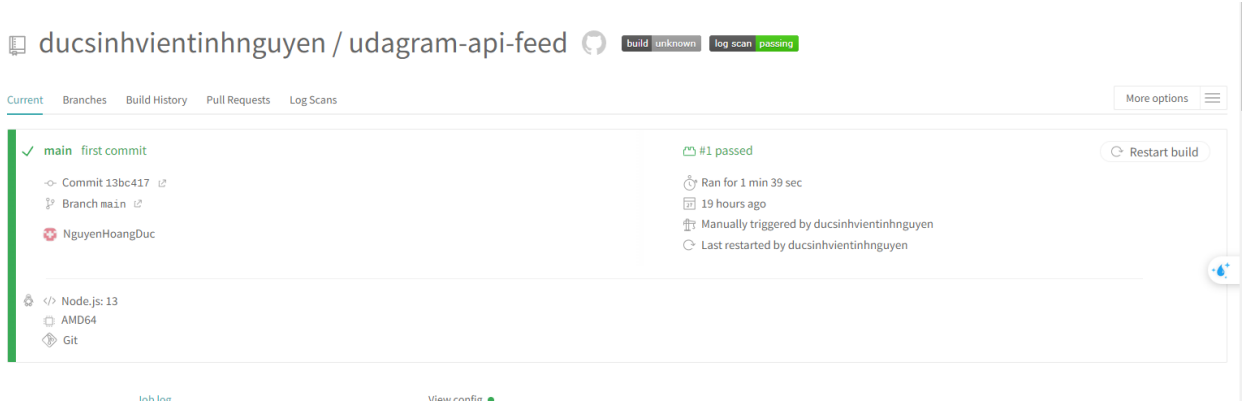


II. Travis

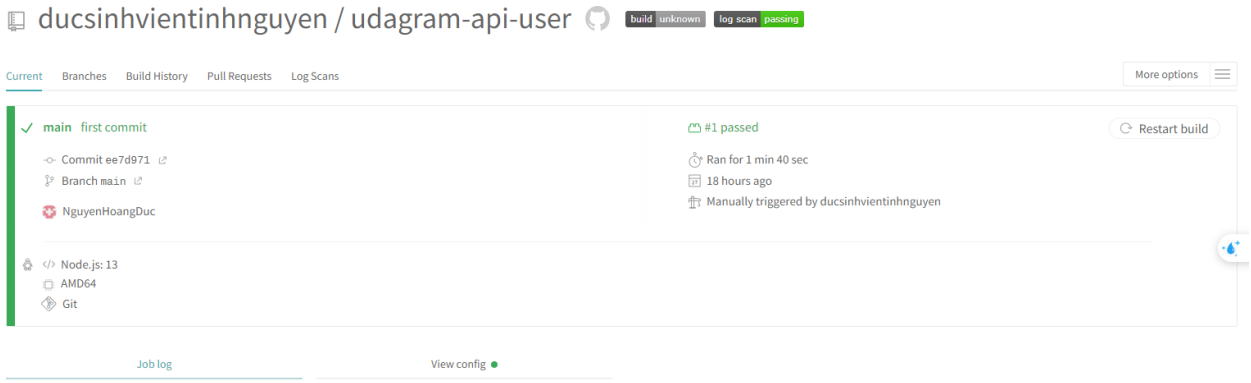
1. Reverseproxy



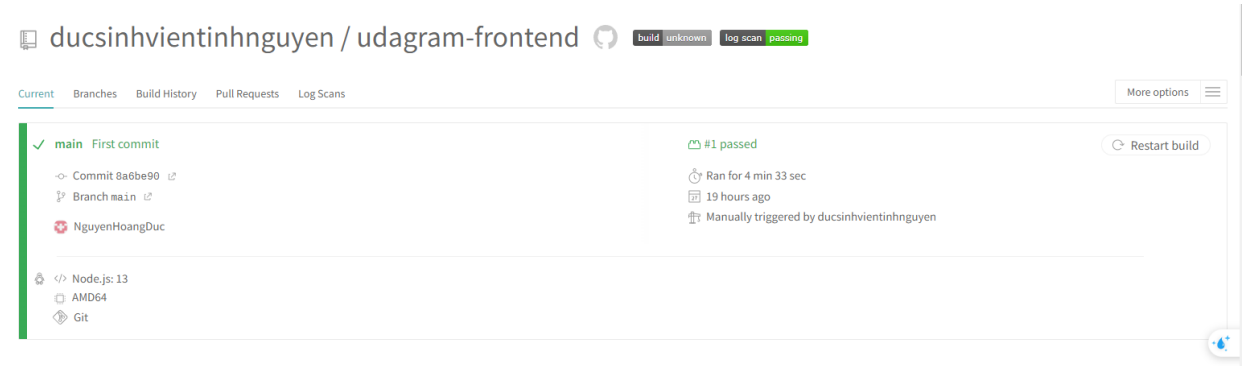
2. Udagram-api-feed



3. Udagram-api-user



4. Udagram-frontend



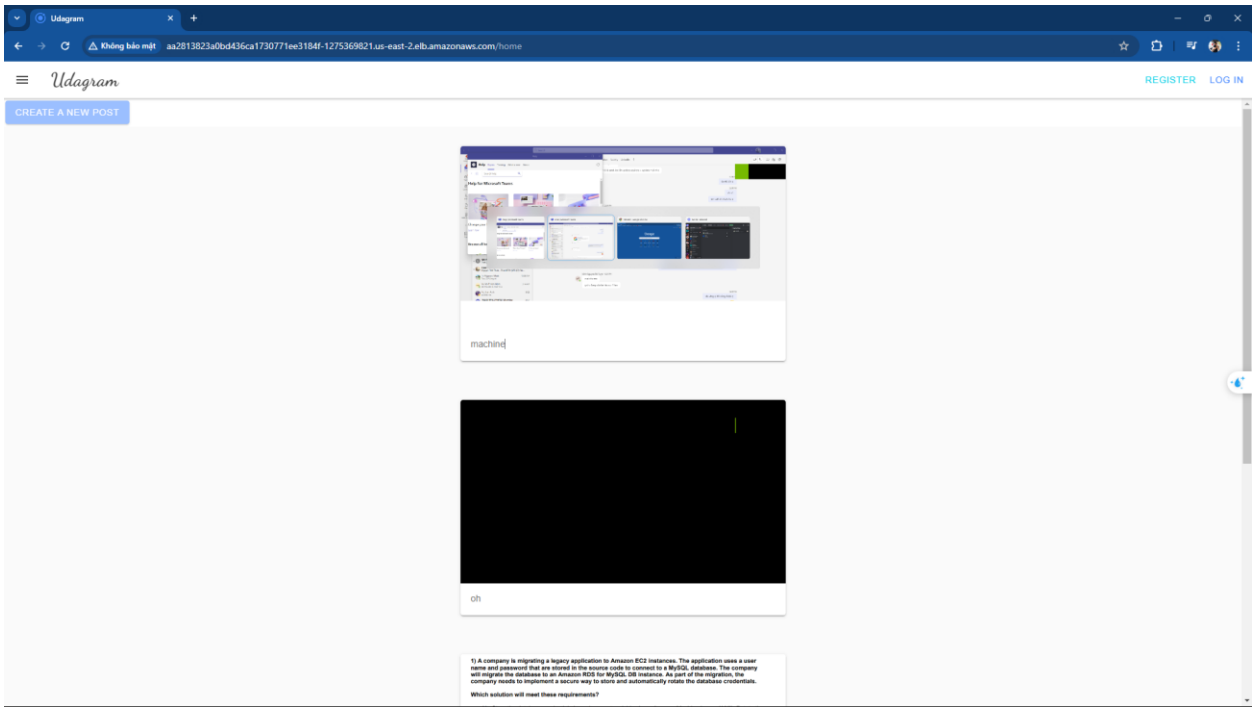
III. Kubernetes kubectl get pods output

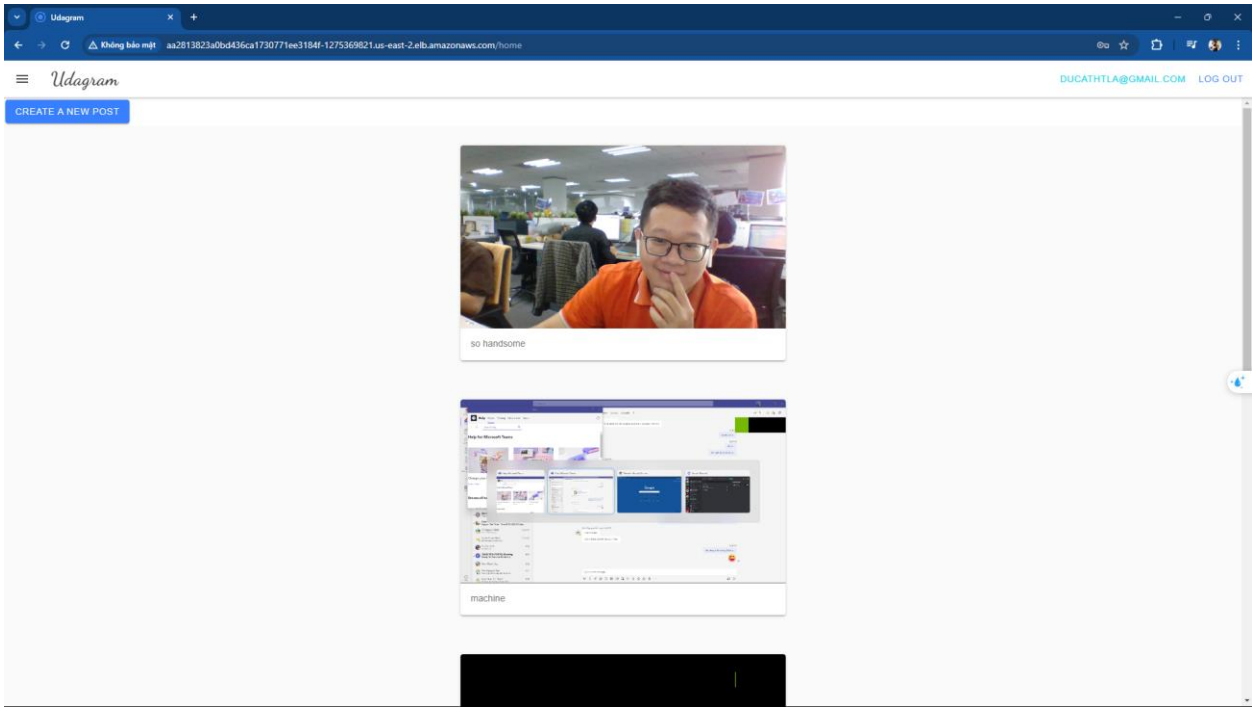
```
ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl get deployment
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
backend-feed   2/2     2            2           73m
backend-user   2/2     2            2           73m
frontend       1/1     1            1           72m
reverseproxy   1/1     1            1           73m

ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl get pods
NAME                            READY   STATUS    RESTARTS       AGE
backend-feed-9b7586f5-tdkbx     1/1     Running   11 (37m ago)   73m
backend-feed-9b7586f5-vbd8h     1/1     Running   0              27m
backend-user-64fc79d649-h7wdx   1/1     Running   11 (37m ago)   73m
backend-user-64fc79d649-s59vs   1/1     Running   0              25m
frontend-7c776675c4-kxdxw       1/1     Running   0              30m
reverseproxy-7f8c66c8bb-6gfsx   1/1     Running   0              73m

ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl get services
NAME             TYPE           CLUSTER-IP       EXTERNAL-IP                                                                PORT(S)          AGE
backend-feed     ClusterIP      172.20.185.87    <none>                                                                     8080/TCP         73m
backend-user     ClusterIP      172.20.70.113    <none>                                                                     8080/TCP         73m
frontend         ClusterIP      172.20.135.255   <none>                                                                     8100/TCP         72m
frontend-ep      LoadBalancer   172.20.223.78    a882327a793a548588d8324e301aa510-1286218394.us-east-2.elb.amazonaws.com    80:30298/TCP     70m
kubernetes       ClusterIP      172.20.0.1       <none>                                                                     443/TCP          89m
reverseproxy     ClusterIP      172.20.254.51    <none>                                                                     8080/TCP         73m
reverseproxy-ep  LoadBalancer   172.20.226.157   ae4e9841143ec4dd39bf48d1a9a9bf4d-935476678.us-east-2.elb.amazonaws.com      8080:32110/TCP   71m
```

My Udagram: I haved try to create an account and upload some image.





IV. Kubernetes kubectl describe services output

```
$ kubectl describe services
Name:              backend-feed
Namespace:         default
Labels:            service=backend-feed
Annotations:       <none>
Selector:          service=backend-feed
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                172.20.185.87
IPs:               172.20.185.87
Port:              8080  8080/TCP
TargetPort:        8080/TCP
Endpoints:         10.0.0.94:8080,10.0.19.205:8080
Session Affinity:  None
Events:            <none>


Name:              backend-user
Namespace:         default
Labels:            service=backend-user
Annotations:       <none>
Selector:          service=backend-user
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                172.20.70.113
IPs:               172.20.70.113
Port:              8080  8080/TCP
TargetPort:        8080/TCP
Endpoints:         10.0.11.131:8080,10.0.26.3:8080
Session Affinity:  None
Events:            <none>
```

```
Name:              frontend
Namespace:         default
Labels:            service=frontend
Annotations:       <none>
Selector:          service=frontend
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                172.20.135.255
IPs:               172.20.135.255
Port:              8100  8100/TCP
TargetPort:        80/TCP
Endpoints:         10.0.28.106:80
Session Affinity:  None
Events:            <none>


Name:                     frontend-ep
Namespace:                default
Labels:                   service=frontend
Annotations:              <none>
Selector:                 service=frontend
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       172.20.223.78
IPs:                      172.20.223.78
LoadBalancer Ingress:     a882327a793a548588d8324e301aa510-1286218394.us-east-2.elb.amazonaws.com
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
NodePort:                 <unset>  30298/TCP
Endpoints:                10.0.28.106:80
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```

```
Name:              reverseproxy
Namespace:         default
Labels:            service=reverseproxy
Annotations:       <none>
Selector:          service=reverseproxy
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                172.20.254.51
IPs:               172.20.254.51
Port:              8080  8080/TCP
TargetPort:        8080/TCP
Endpoints:         10.0.31.75:8080
Session Affinity:  None
Events:            <none>


Name:                     reverseproxy-ep
Namespace:                default
Labels:                   service=reverseproxy
Annotations:              <none>
Selector:                 service=reverseproxy
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       172.20.226.157
IPs:                      172.20.226.157
LoadBalancer Ingress:     ae4e9841143ec4dd39bf48d1a9a9bf4d-935476678.us-east-2.elb.amazonaws.com
Port:                     <unset>  8080/TCP
TargetPort:               8080/TCP
NodePort:                 <unset>  32110/TCP
Endpoints:                10.0.31.75:8080
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```

V. Kubernetes kubectl describe hpa output

```
ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl describe hpa
Name:                                                  backend-user
Namespace:                                             default
Labels:                                                <none>
Annotations:                                           <none>
CreationTimestamp:                                     Sat, 20 Apr 2024 19:01:11 +0700
Reference:                                             Deployment/backend-user
Metrics:                                               ( current / target )
  resource cpu on pods  (as a percentage of request):  0% (0) / 70%
Min replicas:                                          1
Max replicas:                                          2
Deployment pods:                                       1 current / 1 desired
Conditions:
  Type            Status  Reason            Message
  ----            ------  ------            -------
  AbleToScale     True    ReadyForNewScale  recommended size matches current size
  ScalingActive   True    ValidMetricFound  the HPA was able to successfully calculate a replica count from cpu resource utilization (percentage
of request)
  ScalingLimited  True    TooFewReplicas    the desired replica count is less than the minimum replica count
Events:
  Type    Reason             Age   From                       Message
  ----    ------             ----  ----                       -------
  Normal  SuccessfulRescale  66s   horizontal-pod-autoscaler  New size: 1; reason: All metrics below target
```

VI. Kubernetes kubectl logs <your pod name> output

```
ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl get pods
NAME                          READY   STATUS    RESTARTS       AGE
backend-feed-9b7586f5-tdkbx   1/1     Running   11 (42m ago)   79m
backend-feed-9b7586f5-vbd8h   1/1     Running   0              33m
backend-user-64fc79d649-s59vs   1/1     Running   0              31m
frontend-7c776675c4-kxdxw     1/1     Running   0              36m
reverseproxy-7f8c66c8bb-6gfsx   1/1     Running   0              79m

ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl logs backend-feed-9b7586f5-tdkbx

> udagram-api@2.0.0 prod /usr/src/app
> tsc && node ./www/server.js

Initialize database connection...
Executing (default): CREATE TABLE IF NOT EXISTS "FeedItem" ("id"   SERIAL , "caption" VARCHAR(255), "url" VARCHAR(255), "createdAt" TIMESTAMP WIT
H TIME ZONE, "updatedAt" TIMESTAMP WITH TIME ZONE, PRIMARY KEY ("id"));
Executing (default): SELECT i.relname AS name, ix.indisprimary AS primary, ix.indisunique AS unique, ix.indkey AS indkey, array_agg(a.attnum) as
column_indexes, array_agg(a.attname) AS column_names, pg_get_indexdef(ix.indexrelid) AS definition FROM pg_class t, pg_class i, pg_index ix, pg_a
ttribute a WHERE t.oid = ix.indrelid AND i.oid = ix.indexrelid AND a.attrelid = t.oid AND t.relkind = 'r' and t.relname = 'FeedItem' GROUP BY i.r
elname, ix.indexrelid, ix.indisprimary, ix.indisunique, ix.indkey ORDER BY i.relname;
server running http://localhost:8100
press CTRL+C to stop server
Executing (default): INSERT INTO "FeedItem" ("id","caption","url","createdAt","updatedAt") VALUES (DEFAULT,$1,$2,$3,$4) RETURNING *;
```

```
ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl logs backend-user-64fc79d649-s59vs

> udagram-api@2.0.0 prod /usr/src/app
> tsc && node ./www/server.js

Initialize database connection...
Executing (default): CREATE TABLE IF NOT EXISTS "User" ("email" VARCHAR(255) , "passwordHash" VARCHAR(255), "createdAt" TIMESTAMP WITH TIME ZONE,
 "updatedAt" TIMESTAMP WITH TIME ZONE, PRIMARY KEY ("email"));
Executing (default): SELECT i.relname AS name, ix.indisprimary AS primary, ix.indisunique AS unique, ix.indkey AS indkey, array_agg(a.attnum) as
column_indexes, array_agg(a.attname) AS column_names, pg_get_indexdef(ix.indexrelid) AS definition FROM pg_class t, pg_class i, pg_index ix, pg_a
ttribute a WHERE t.oid = ix.indrelid AND i.oid = ix.indexrelid AND a.attrelid = t.oid AND t.relkind = 'r' and t.relname = 'User' GROUP BY i.relna
me, ix.indexrelid, ix.indisprimary, ix.indisunique, ix.indkey ORDER BY i.relname;
server running http://localhost:8100
press CTRL+C to stop server
Executing (default): SELECT "email", "passwordHash", "createdAt", "updatedAt" FROM "User" AS "User" WHERE "User"."email" = 'Duc123@gmail.com';
```

```
ADMIN@LAPTOP-GG86CK8D MINGW64 ~/Documents/cd0354-monolith-to-microservices-project/deployment (main)
$ kubectl get svc
NAME             TYPE           CLUSTER-IP       EXTERNAL-IP                                                               PORT(S)          AGE
backend-feed     ClusterIP      172.20.185.87    <none>                                                                    8080/TCP         81m
backend-user     ClusterIP      172.20.70.113    <none>                                                                    8080/TCP         81m
frontend         ClusterIP      172.20.135.255   <none>                                                                    8100/TCP         80m
frontend-ep      LoadBalancer   172.20.223.78    a882327a793a548588d8324e301aa510-1286218394.us-east-2.elb.amazonaws.com   80:30298/TCP     78m
kubernetes       ClusterIP      172.20.0.1       <none>                                                                    443/TCP          97m
reverseproxy     ClusterIP      172.20.254.51    <none>                                                                    8080/TCP         81m
reverseproxy-ep  LoadBalancer   172.20.226.157   ae4e9841143ec4dd39bf48d1a9a9bf4d-935476678.us-east-2.elb.amazonaws.com   8080:32110/TCP   79m
```