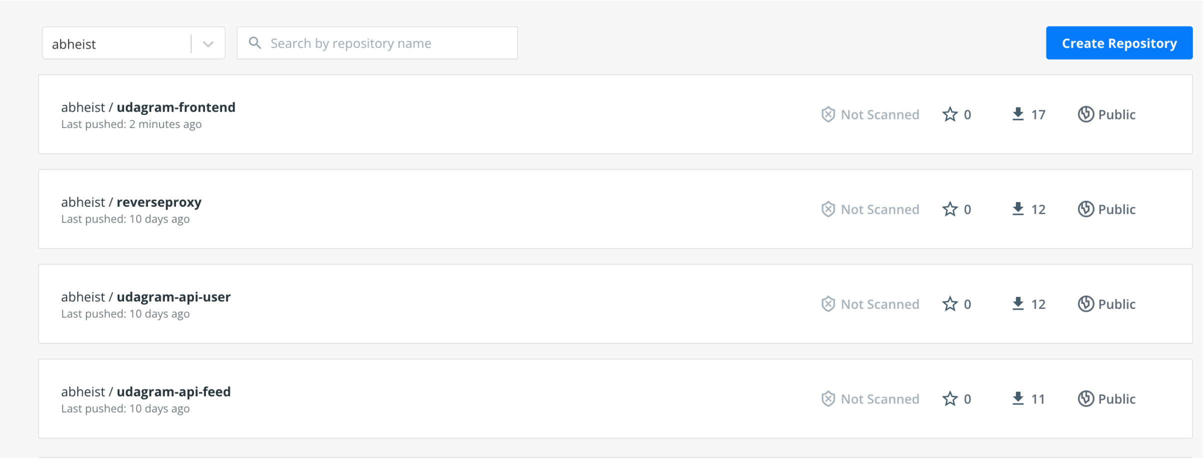
Refactor Udagram App into Microservices and Deploy

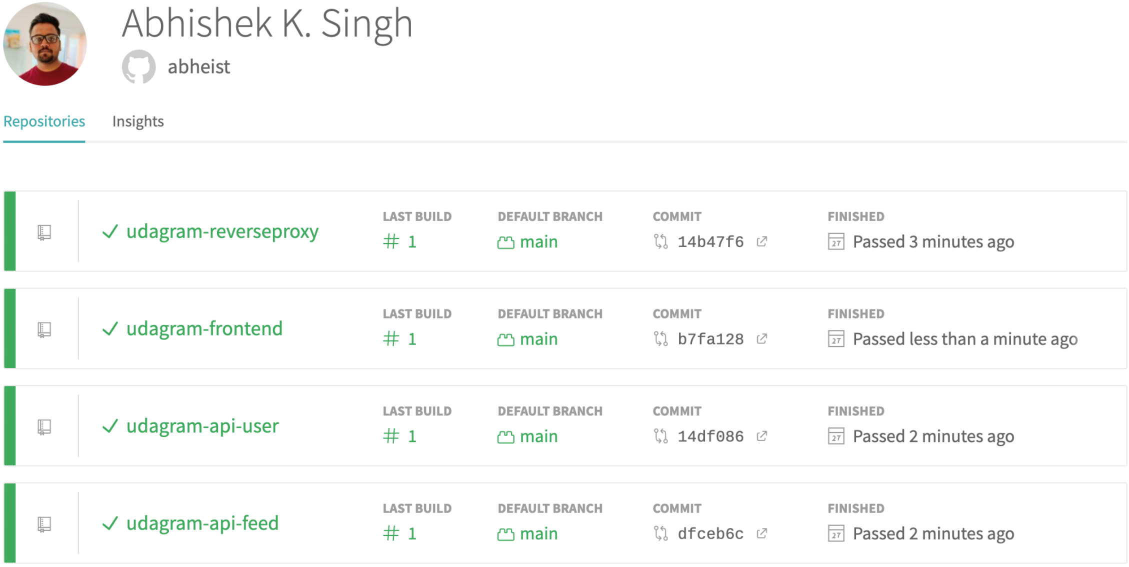
Tags

Screenshot of DockerHub shows the images



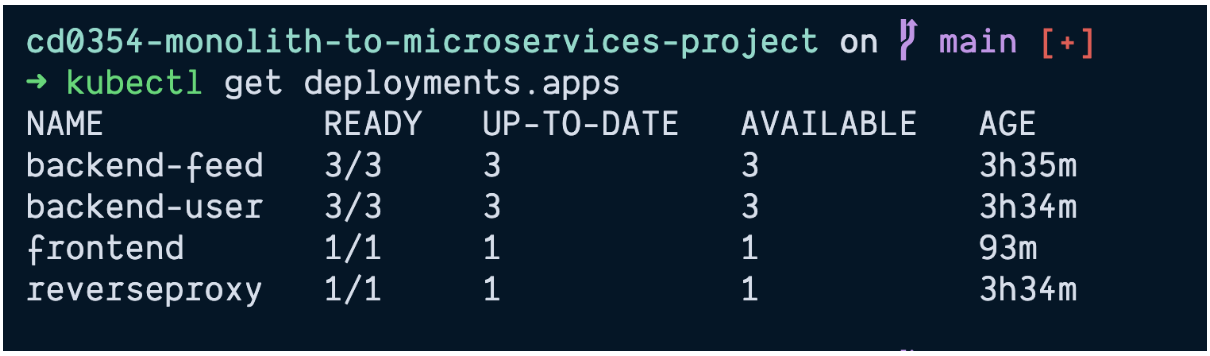
abheist / **udagram-frontend** Last pushed: 2 minutes ago	Not Scanned	☆ 0	17	Public
abheist / **reverseproxy** Last pushed: 10 days ago	Not Scanned	☆ 0	12	Public
abheist / **udagram-api-user** Last pushed: 10 days ago	Not Scanned	☆ 0	12	Public
abheist / **udagram-api-feed** Last pushed: 10 days ago	Not Scanned	☆ 0	11	Public

Screenshot of the Travis CI interface shows a successful build and deploy job

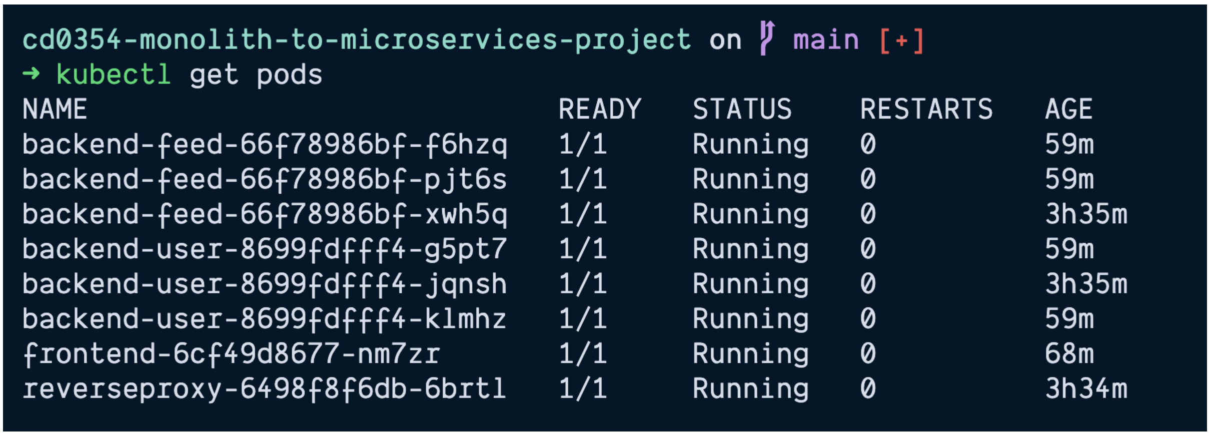


	LAST BUILD	DEFAULT BRANCH	COMMIT	FINISHED
✓ udagram-reverseproxy	# 1	main	14b47f6	Passed 3 minutes ago
✓ udagram-frontend	# 1	main	b7fa128	Passed less than a minute ago
✓ udagram-api-user	# 1	main	14df086	Passed 2 minutes ago
✓ udagram-api-feed	# 1	main	dfceb6c	Passed 2 minutes ago

A screenshots of `kubectl` commands show the Frontend and API projects deployed in Kubernetes.



```
cd0354-monolith-to-microservices-project on  main [+]
➜ kubectl get deployments.apps
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
backend-feed   3/3     3            3           3h35m
backend-user   3/3     3            3           3h34m
frontend       1/1     1            1           93m
reverseproxy   1/1     1            1           3h34m
```

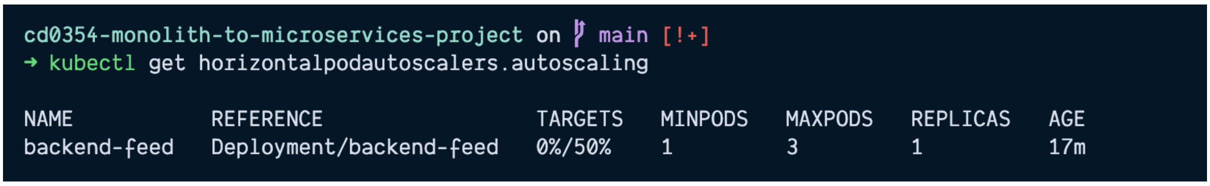


```
cd0354-monolith-to-microservices-project on  main [+]
➜ kubectl get pods
NAME                            READY   STATUS    RESTARTS   AGE
backend-feed-66f78986bf-f6hzq   1/1     Running   0          59m
backend-feed-66f78986bf-pjt6s   1/1     Running   0          59m
backend-feed-66f78986bf-xwh5q   1/1     Running   0          3h35m
backend-user-8699fdfff4-g5pt7   1/1     Running   0          59m
backend-user-8699fdfff4-jqnsh   1/1     Running   0          3h35m
backend-user-8699fdfff4-klmhz   1/1     Running   0          59m
frontend-6cf49d8677-nm7zr       1/1     Running   0          68m
reverseproxy-6498f8f6db-6brtl   1/1     Running   0          3h34m
```

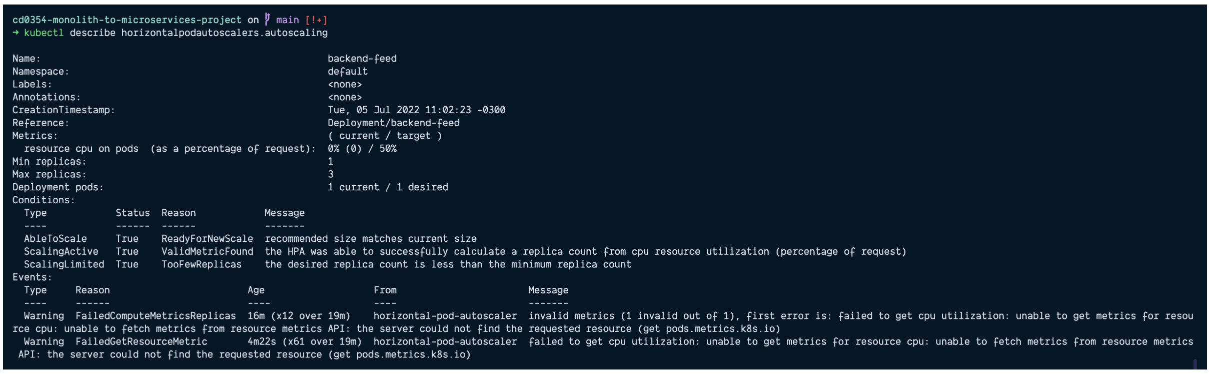
```
cd0354-monolith-to-microservices-project on  main [+]
➜ kubectl describe services
Name:              backend-feed
Namespace:         default
Labels:            service=backend-feed
Annotations:       <none>
Selector:          service=backend-feed
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                10.100.35.146
IPs:               10.100.35.146
Port:              8080  8080/TCP
TargetPort:        8080/TCP
Endpoints:         172.31.23.247:8080,172.31.85.100:8080,172.31.92.186:8080
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:                10.100.127.119
IPs:               10.100.127.119
Port:              8080  8080/TCP
TargetPort:        8080/TCP
Endpoints:         172.31.18.219:8080,172.31.84.82:8080,172.31.93.85:8080
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:                10.100.50.0
IPs:               10.100.50.0
Port:              8100  8100/TCP
TargetPort:        80/TCP
Endpoints:         172.31.27.79:80
Session Affinity:  None
Events:            <none>


Name:              kubernetes
Namespace:         default
Labels:            component=apiserver
                   provider=kubernetes
Annotations:       <none>
Selector:          <none>
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                10.100.0.1
IPs:               10.100.0.1
Port:              https  443/TCP
TargetPort:        443/TCP
Endpoints:         172.31.3.231:443,172.31.86.139:443
Session Affinity:  None
Events:            <none>


Name:                     publicfrontend
Namespace:                default
Labels:                   service=frontend
Annotations:              <none>
Selector:                 service=frontend
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.100.73.65
IPs:                      10.100.73.65
LoadBalancer Ingress:     af6cd9566ed7e44489ffeef8c2f2c391-762354092.us-east-1.elb.ama
zonaws.com
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
NodePort:                 <unset>  30859/TCP
Endpoints:                172.31.27.79:80
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>


Name:                     publicreverseproxy
Namespace:                default
Labels:                   service=reverseproxy
Annotations:              <none>
Selector:                 service=reverseproxy
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.100.97.220
IPs:                      10.100.97.220
LoadBalancer Ingress:     a2eadaae92d784276ad3cd7dddae569d-1975764867.us-east-1.elb.am
azonaws.com
Port:                     <unset>  8080/TCP
TargetPort:               8080/TCP
NodePort:                 <unset>  31720/TCP
Endpoints:                172.31.87.109:8080
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:                10.100.220.100
IPs:               10.100.220.100
Port:              8080  8080/TCP
TargetPort:        8080/TCP
Endpoints:         172.31.87.109:8080
Session Affinity:  None
Events:            <none>
```

HPA - replicas

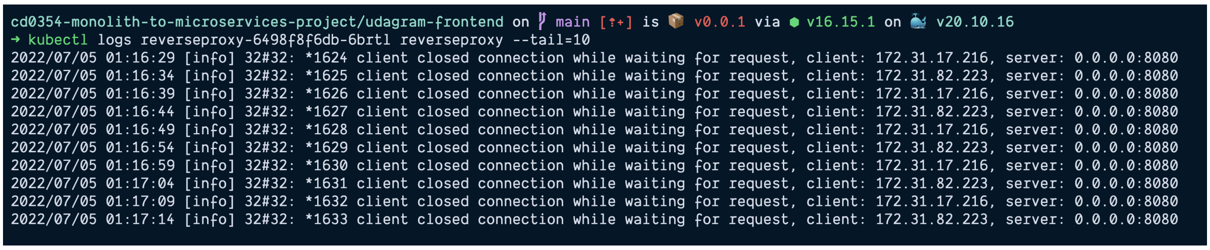


```
cd0354-monolith-to-microservices-project on  main [!+]
➜ kubectl get horizontalpodautoscalers.autoscaling

NAME           REFERENCE                 TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
backend-feed   Deployment/backend-feed   0%/50%    1         3         1          17m
```



```
cd0354-monolith-to-microservices-project on  main [!+]
➜ kubectl describe horizontalpodautoscalers.autoscaling

Name:                                                  backend-feed
Namespace:                                             default
Labels:                                                <none>
Annotations:                                           <none>
CreationTimestamp:                                     Tue, 05 Jul 2022 11:02:23 -0300
Reference:                                             Deployment/backend-feed
Metrics:                                               ( current / target )
  resource cpu on pods  (as a percentage of request):  0% (0) / 50%
Min replicas:                                          1
Max replicas:                                          3
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
  ----     ------                        ----                ----                       -------
  Warning  FailedComputeMetricsReplicas  16m (x12 over 19m)    horizontal-pod-autoscaler  invalid metrics (1 invalid out of 1), first error is: failed to get cpu utilization: unable to get metrics for resou
rce cpu: unable to fetch metrics from resource metrics API: the server could not find the requested resource (get pods.metrics.k8s.io)
  Warning  FailedGetResourceMetric       4m22s (x61 over 19m)  horizontal-pod-autoscaler  failed to get cpu utilization: unable to get metrics for resource cpu: unable to fetch metrics from resource metrics
 API: the server could not find the requested resource (get pods.metrics.k8s.io)
```

Screenshot of one of the backend API pod logs



```
cd0354-monolith-to-microservices-project/udagram-frontend on  main [⇡+] is 📦 v0.0.1 via ● v16.15.1 on 🐳 v20.10.16
➜ kubectl logs reverseproxy-6498f8f6db-6brtl reverseproxy --tail=10
2022/07/05 01:16:29 [info] 32#32: *1624 client closed connection while waiting for request, client: 172.31.17.216, server: 0.0.0.0:8080
2022/07/05 01:16:34 [info] 32#32: *1625 client closed connection while waiting for request, client: 172.31.82.223, server: 0.0.0.0:8080
2022/07/05 01:16:39 [info] 32#32: *1626 client closed connection while waiting for request, client: 172.31.17.216, server: 0.0.0.0:8080
2022/07/05 01:16:44 [info] 32#32: *1627 client closed connection while waiting for request, client: 172.31.82.223, server: 0.0.0.0:8080
2022/07/05 01:16:49 [info] 32#32: *1628 client closed connection while waiting for request, client: 172.31.17.216, server: 0.0.0.0:8080
2022/07/05 01:16:54 [info] 32#32: *1629 client closed connection while waiting for request, client: 172.31.82.223, server: 0.0.0.0:8080
2022/07/05 01:16:59 [info] 32#32: *1630 client closed connection while waiting for request, client: 172.31.17.216, server: 0.0.0.0:8080
2022/07/05 01:17:04 [info] 32#32: *1631 client closed connection while waiting for request, client: 172.31.82.223, server: 0.0.0.0:8080
2022/07/05 01:17:09 [info] 32#32: *1632 client closed connection while waiting for request, client: 172.31.17.216, server: 0.0.0.0:8080
2022/07/05 01:17:14 [info] 32#32: *1633 client closed connection while waiting for request, client: 172.31.82.223, server: 0.0.0.0:8080
```