

## Deployment:

برای این قسمت ابتدا سه فایل deployment را برای سه ایمیج که داریم می نویسیم نکته مهم این است که ما فایل config map مربوطه را نیز مینویسیم و بعد فایل های deployment هامون رو متناسب با آن تغییر می دهیم

elasticsearch:

```

Terminal Help
elasticsearch.yaml x
docker_1 > kuber > deployment > elasticsearch.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: elasticsearch
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: elasticsearch
10   template:
11     metadata:
12       labels:
13         app: elasticsearch
14     spec:
15       containers:
16       - name: elasticsearch
17         image: docker.elastic.co/elasticsearch/elasticsearch:7.4.0
18         envFrom:
19         - configMapRef:
20           name: app-config
21         ports:
22         - containerPort: 9200
23         - containerPort: 9300
24
25

```

redis:

```
Terminal Help
serviceapi.yaml  redis.yaml X
docker_1 > kuber > deployment > redis.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: redis-deployment
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: redis
10   template:
11     metadata:
12       labels:
13         app: redis
14     spec:
15       containers:
16       - name: redis
17         image: redis:latest
18         ports:
19         - containerPort: 6379
20         envFrom:
21         - configMapRef:
22           name: app-config
23
24
25
```

movie app:

```
serviceapi.yaml x
docker_1 > kuber > deployment > serviceapi.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: movieapp-deployment
5    labels:
6      app: movie-app
7  spec:
8    replicas: 3
9    selector:
10     matchLabels:
11       app: movie-app
12   template:
13     metadata:
14       labels:
15         app: movie-app
16     spec:
17       containers:
18       - name: movie-app
19         image: movie_image
20         ports:
21         - containerPort: 8000
22         envFrom:
23         - configMapRef:
24           name: app-config
25
```

config map

```
Terminal Help
app-config.yaml x
docker_1 > kuber > deployment > app-config.yaml
1  apiVersion: v1
2  kind: ConfigMap
3  metadata:
4    name: app-config
5  data:
6    elasticsearch.yml: |
7      xpack.security.enabled: false
8      xpack.security.transport.ssl.enabled: false
9      xpack.security.http.ssl.enabled: false
10     discovery.type: single-node
11     ELASTIC_PASSWORD: "123456"
12   redis.conf: |
13
14   movie-app-config.properties: |
15     SERVER_PORT=8000
16
17
```

افزودن فایل ها

```
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl apply -f .
configmap/app-config created
deployment.apps/elasticsearch created
deployment.apps/redis-deployment created
deployment.apps/movieapp-deployment created
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
elasticsearch-78c9f4d94d-72md2      0/1     ContainerCreating   0           7s
elasticsearch-78c9f4d94d-h769p      0/1     ContainerCreating   0           7s
elasticsearch-78c9f4d94d-hfm7k      0/1     ContainerCreating   0           7s
movieapp-deployment-54698fcc5f-8bl7h 0/1     ContainerCreating   0           7s
movieapp-deployment-54698fcc5f-9qq7q 0/1     ContainerCreating   0           7s
movieapp-deployment-54698fcc5f-pn8ff 0/1     ContainerCreating   0           7s
redis-deployment-8457ffffbbb-84zdd   0/1     ContainerCreating   0           7s
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get deployment
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
elasticsearch       0/3     3             0           30s
movieapp-deployment 0/3     3             0           30s
redis-deployment    0/1     1             0           30s
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get configmap
NAME      DATA   AGE
app-config 3        43s
kube-root-ca.crt 1       14d
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$
```

```
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
elasticsearch-fbfd4d5d-58vg4        1/1     Running   0           5s
movieapp-deployment-79c89648d9-d4q58 1/1     Running   0           5s
movieapp-deployment-79c89648d9-q8p42 1/1     Running   0           5s
movieapp-deployment-79c89648d9-r6jhc 1/1     Running   0           5s
redis-deployment-8457ffffbbb-84zdd    1/1     Running   1 (31m ago) 23h
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$
```

نام پاد ها بر اساس کدام فیلد deployment است؟

نام انها بر اساس فیلد metadata.name است و در ادامه آن هاش شده به آن وصل میشود.

نشان دهید که آدرس IP پاد ها متفاوت است و احتیاج به تعریف سرویس داریم.

برای اینکار ابتدا یکی از پاد ها را دستی پاک میکنیم و سپس بعد از ساخته شدن آن میبینم که ای

پی جدیدی به آن اختصاص داده میشود.

```
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get pods -o wide
NAME                                READY   STATUS              RESTARTS   AGE   IP           NODE       NOMINATED NODE   READINESS GATES
elasticsearch-fbfd4d5d-58vg4        0/1     CrashLoopBackOff    6 (106s ago) 8m26s 10.1.0.103   docker-desktop <none>          <none>
movieapp-deployment-79c89648d9-d4q58 1/1     Running             0           8m26s 10.1.0.100   docker-desktop <none>          <none>
movieapp-deployment-79c89648d9-q8p42 1/1     Running             0           8m26s 10.1.0.101   docker-desktop <none>          <none>
movieapp-deployment-79c89648d9-r6jhc 1/1     Running             0           8m26s 10.1.0.102   docker-desktop <none>          <none>
redis-deployment-8457ffffbbb-84zdd    1/1     Running             1 (39m ago) 23h    10.1.0.92    docker-desktop <none>          <none>
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl delete pods elasticsearch-fbfd4d5d-58vg4
pod "elasticsearch-fbfd4d5d-58vg4" deleted
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP           NODE       NOMINATED NODE   READINESS GATES
elasticsearch-fbfd4d5d-wb8fw         1/1     Running   0           4s    10.1.0.104   docker-desktop <none>          <none>
movieapp-deployment-79c89648d9-d4q58 1/1     Running   0           10m   10.1.0.100   docker-desktop <none>          <none>
movieapp-deployment-79c89648d9-q8p42 1/1     Running   0           10m   10.1.0.101   docker-desktop <none>          <none>
movieapp-deployment-79c89648d9-r6jhc 1/1     Running   0           10m   10.1.0.102   docker-desktop <none>          <none>
redis-deployment-8457ffffbbb-84zdd    1/1     Running   1 (41m ago) 23h    10.1.0.92    docker-desktop <none>          <none>
yasaman@yasaman-VlvoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$
```

همانطور که در این عکس میبینیم پاد مربوط به الستیک در ابتدا روی ای پی 10.1.0.103 وصل شده بود که بعد با کامندی این پاد را پاک کردیم. سپس دوباره بعد از ساخته شدن آن توسط کوبر میبینیم که پاد ساخته شده دارای ای پی 10.1.0.104 است.

همچنین پاد های که مربوط به یک کانتینر هستند مانند سه تا پاد ساخته شده از کانتینر movieapp دارای ای پی های متفاوت است در نتیجه برای اتصال این کانتینر ها به هم نیاز به تعریف service داریم. تا بتوانیم از یک لایه میانی برای اتصال استفاده کنیم.

در Kubernetes، پادها ممکن است به طور دینامیک تولید و از بین بروند، اما از طرف دیگر، سرویس ها یک لایه میانی بین پادها و دیگر اجزای سیستم هستند و این امکان را فراهم می کنند که به طور ثابت به یک گروه از پادها متصل شوید، بدون نگرانی از تغییرات در IP آنها.

برای این کار سه تا سرویس زیر را تعریف میکنیم

elasticsearch

```
locker_1 > kuber > service > 📄 elasticsearch.yaml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: elasticsearch-service
5  spec:
6    selector:
7      app: elasticsearch
8    ports:
9      - protocol: TCP
10        port: 9200
11        targetPort: 9200
12    type: ClusterIP
13
14
```

redis

```
elasticsearch.yaml  redis.yaml X
docker_1 > kuber > service > redis.yaml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: redis-service
5  spec:
6    selector:
7      app: redis
8    ports:
9      - protocol: TCP
10        port: 6379
11        targetPort: 6379
12    type: ClusterIP
13
```

movieapp

```
elasticsearch.yaml  serviceapi.yaml X
docker_1 > kuber > service > serviceapi.yaml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: movie-app-service
5  spec:
6    selector:
7      app: movie-app
8    ports:
9      - protocol: TCP
10        port: 8000
11        targetPort: 8000
12    type: ClusterIP
13
```

با دستور زیر سرویس ها رو به کوبر وصل میکنیم

```

yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/service$ kubectl apply -f .
service/elasticsearch-service created
service/redis-service created
service/movie-app-service created
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/service$ kubectl get services
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
elasticsearch-service ClusterIP    10.101.246.98    <none>           9200/TCP         14s
kubernetes           ClusterIP    10.96.0.1        <none>           443/TCP          15d
movie-app-service    ClusterIP    10.111.62.236    <none>           8000/TCP         14s
redis-service        ClusterIP    10.97.145.110    <none>           6379/TCP         14s
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/service$

```

در Kubernetes، ارتباط بین Deployment و Service از طریق برچسب‌ها (labels) برقرار می‌شود. یکی از مزایای service تعیین یک IP ثابت و یکتا برای دسترسی به پاد های یک برنامه است. این کار توسط کدام یک از service type ها انجام می پذیرد؟

این کار از طریق سرویس‌های با نوع ClusterIP انجام می‌شود. وقتی یک سرویس با نوع ClusterIP ایجاد می‌شود، Kubernetes یک IP ثابت و یکتا برای آن سرویس تعیین می‌کند که به صورت داخلی درون کلاستر قابل دسترس است. در واقع سرویس‌های با نوع Cluster IP یک آدرس IP داخلی ثابت درون کلاستر Kubernetes را به گروهی از پادها اختصاص می‌دهند. این آدرس IP می‌تواند برای دسترسی به پادهای مرتبط با سرویس از داخل کلاستر استفاده شود.

چرا الزامی به مشخص کردن این سرویس در فایل service نیست؟  
Kubernetes به طور خودکار یک ClusterIP برای هر سرویس ایجاد می‌کند، حتی اگر این مقدار در تعریف Service مشخص نشده باشد. به عبارت دیگر، Kubernetes به صورت پیش‌فرض برای هر سرویسی که ایجاد می‌شود، یک Cluster IP اختصاص می‌دهد.

آدرس port و target port سرویس که بر روی کلاستر دیپلوی شده را نشان دهید.

با دستور زیر میتوانیم اطلاعات کامل یک سرویس را مشاهده کنیم

`kubectl describe service <service_name>`

```
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl describe service elasticsearch-service
Name: elasticsearch-service
Namespace: default
Labels: <none>
Annotations: <none>
Selector: app=elasticsearch
Type: ClusterIP
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.101.246.98
IPs: 10.101.246.98
Port: <unset> 9200/TCP
TargetPort: 9200/TCP
Endpoints:
Session Affinity: None
Events: <none>
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl describe service movie-app-service
Name: movie-app-service
Namespace: default
Labels: <none>
Annotations: <none>
Selector: app=movie-app
Type: ClusterIP
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.111.62.236
IPs: 10.111.62.236
Port: <unset> 8000/TCP
TargetPort: 8000/TCP
Endpoints: 10.1.0.100:8000,10.1.0.101:8000,10.1.0.102:8000
Session Affinity: None
Events: <none>
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl describe service redis-service
Name: redis-service
Namespace: default
Labels: <none>
Annotations: <none>
Selector: app=redis
Type: ClusterIP
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.97.145.110
IPs: 10.97.145.110
Port: <unset> 6379/TCP
TargetPort: 6379/TCP
Endpoints: 10.1.0.92:6379
Session Affinity: None
Events: <none>
```

## بخش دوم

ابتدا باید نشان دهیم که با پاک شدن پاد اطلاعات کش شده در ردیس هم پاک میشود. با توجه به دو عکس زیر میبینیم که دو تا رکورد در ردیس قرار دارد

Databases > 127.0.0.1:6379 db0 ⓘ ⓘ

🔍 0.21% | 🔄 0 | 📄 1 MB | 📄 2 | ⌚ 4 | 🔍 Insights

🔍 📄 All Key Types Filter by Key Name or Pattern 🔍

Total: 2 Last refresh: <1 min 🔄 ⌵ ⌵

🔍 STRING movies\_redis:lana 59 min 2 KB

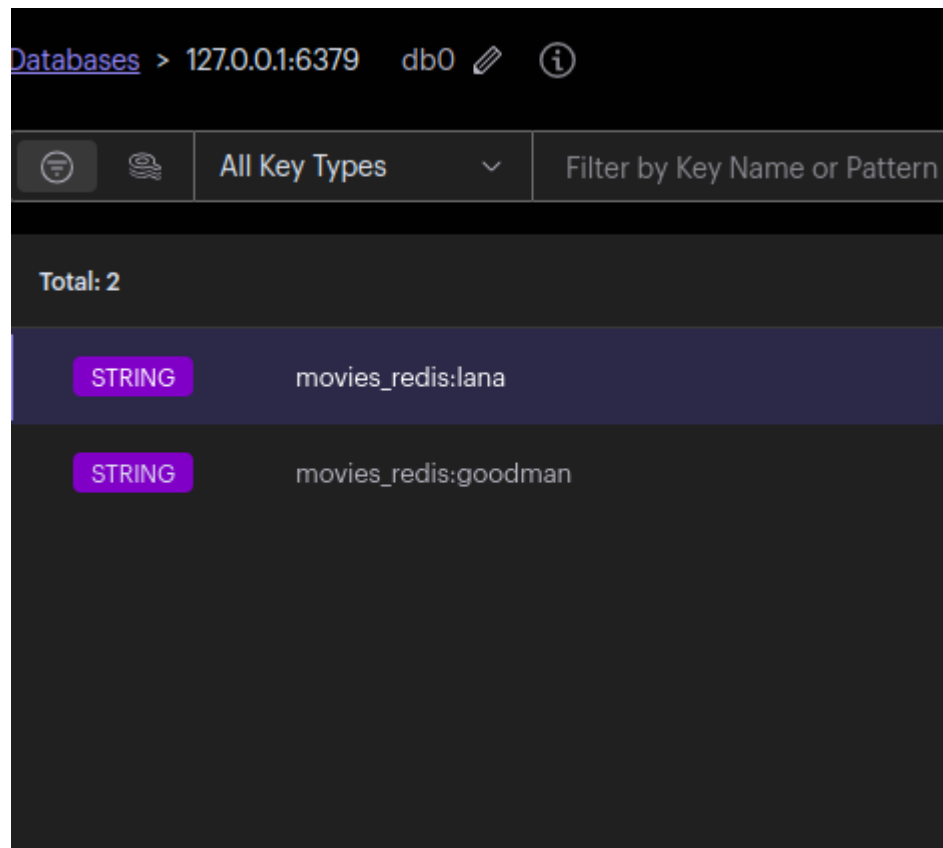
🔍 STRING movies\_redis:goodman 59 min 960 B

🔍 STRING movies\_redis:lana

Key Size: 2 KB Length: 1533 TTL: 3561 Last refresh: <1 min 🔄 ⌵ ⌵ Unicode 📄 🗑️

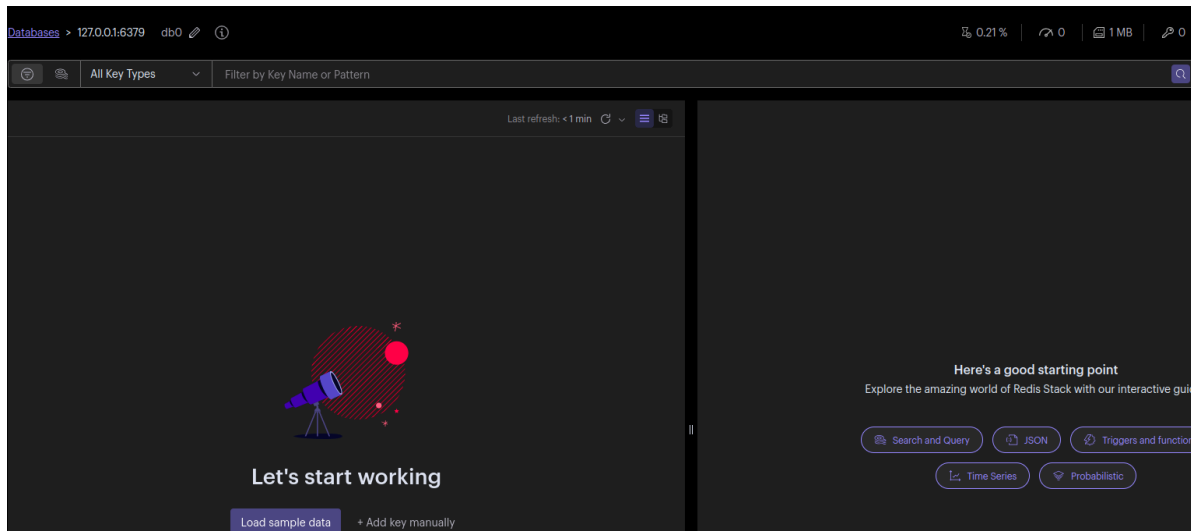
🔍 🔍 🔍 X🔍 {"title": "Explanation for Everything", "description": "It\u0027s 2019s summer in Budapest, high school student Abel is struggling to focus on his final exams, whilst coming to the realisation that he is hopelessly in love with his best friend Janka. The studios Janka has her own unrequited love with married history teacher Jakab\u0027s who had a previous confrontation with Abel\u0027s conservative father. The tensions of a polarised society come unexpectedly to the surface when Abel\u0027s history graduation exam turns into a national scandal.", "year": "2023", "release\_date": "2023-10-05", "imdb\_rating": "8.5", "vote\_count": "2", "popularity": "0.811", "rated": "15", "runtime": 151, "genres": ["Drama"], "stars": ["Adonyi-Walsh G\u0027s", "Istv\u0027s Znamen\u0027s", "Andr\u0027s Ruszn\u0027s", "Rebeka Hath\u0027s", "Lilla Kizlinger", "Eliza Sodr\u0027s", "D\u0027s Kir\u0027s", "Gergely Kocsis", "Andr\u0027s Hath\u0027s", "Juli Jakab", "Krisztina Urbanovits", "Tam\u0027s Fodor", "Pl\u0027s Tari\u0027s", "Zsuzsanna Sz\u0027s", "Peter T\u0027s", "Bacs\u0027s T\u0027s", "Er\u0027s Mara", "P\u0027s Szeiler", "Heinczinger Bence", "Demeter Bendeg\u0027s", "B\u0027s", "M\u0027s Fodor", "Nagy Istv\u0027s", "Fechete Ivett", "Salamon Johanna", "Roland Luk\u0027s", "Szab\u0027s Mesi", "Salamon Te\u0027s", "K\u0027s", "L\u0027s", "Valuska", "Sodr\u0027s Eliza"], "directors": ["G\u0027s Reisz"], "countries": ["Hungary", "Slovakia"], "language": ["Magyar"]}





حالا یکبار پاد مربوط به ردیس را پاک میکنیم و دوباره آن را ران می شود داریم:

```
novieapp-deployment-79c89648d9-d4q58 1/1 Running 0 42m 10.1.0.100 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-q8p42 1/1 Running 0 42m 10.1.0.101 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-r6jhc 1/1 Running 0 42m 10.1.0.102 docker-desktop <none> <none>
redis-deployment-8457ffffbbb-84zdd 1/1 Running 1 (73m ago) 24h 10.1.0.92 docker-desktop <none> <none>
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl delete pods redis-deployment-8457ffffbbb-84zdd
pod "redis-deployment-8457ffffbbb-84zdd" deleted
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get pods -o wide
NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES
elasticsearch-fbfd4d5d-ccvtp 0/1 CrashLoopBackOff 10 (67s ago) 29m 10.1.0.105 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-d4q58 1/1 Running 0 47m 10.1.0.100 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-q8p42 1/1 Running 0 47m 10.1.0.101 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-r6jhc 1/1 Running 0 47m 10.1.0.102 docker-desktop <none> <none>
redis-deployment-8457ffffbbb-vvbrn 0/1 ContainerCreating 0 4s <none> docker-desktop <none> <none>
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$ kubectl get pods -o wide
NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES
elasticsearch-fbfd4d5d-ccvtp 0/1 CrashLoopBackOff 10 (74s ago) 29m 10.1.0.105 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-d4q58 1/1 Running 0 47m 10.1.0.100 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-q8p42 1/1 Running 0 47m 10.1.0.101 docker-desktop <none> <none>
novieapp-deployment-79c89648d9-r6jhc 1/1 Running 0 47m 10.1.0.102 docker-desktop <none> <none>
redis-deployment-8457ffffbbb-vvbrn 1/1 Running 0 11s 10.1.0.106 docker-desktop <none> <none>
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/deployment$
```



حالا فایل های pv, pvc را به صورت زیر مینویسیم

redis pv

```
redis_pv.yaml x
docker_1 > kuber > pvc,pv > redis_pv.yaml
1  apiVersion: v1
2  kind: PersistentVolume
3  metadata:
4    name: redis-pv
5    labels:
6      type: local
7  spec:
8    storageClassName: manual
9    capacity:
10     storage: 3Gi
11    accessModes:
12     - ReadWriteOnce
13    hostPath:
14     path: "/data"
15
16
17
```

redis pvc

```
docker_1 > kuber > pvc,pv > redis_pvc.yaml
1  apiVersion: v1
2  kind: PersistentVolumeClaim
3  metadata:
4    name: redis-pvc
5  spec:
6    storageClassName: manual
7    accessModes:
8     - ReadWriteOnce
9    resources:
10     requests:
11       storage: 3Gi
12
```



در اینجا با افزودن لیبِل hostPath در واقع داریم بیان میکنیم که این pvc باید به این pv وصل شود. برای اتصال پاد ردیس به این pvc نیاز داریم تا فایل دولوپمنت آن را تغییر دهیم




```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: redis-deployment
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: redis
10   template:
11     metadata:
12       labels:
13         app: redis
14     spec:
15       containers:
16       - name: redis
17         image: redis:latest
18         ports:
19         - containerPort: 6379
20         envFrom:
21         - configMapRef:
22             name: app-config
23         volumeMounts:
24         - name: redis-data
25           mountPath: /data
26       volumes:
27       - name: redis-data
28         persistentVolumeClaim:
29           claimName: redis-pvc
```





افزودن فایل ها

```
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/pvc,pv$ kubectl apply -f .
persistentvolume/redis-pv created
persistentvolumeclaim/redis-pvc created
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/pvc,pv$ kubectl get pv
NAME      CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM                STORAGECLASS  VOLUMEATTRIBUTESCLASS  REASON  AGE
redis-pv  3Gi       RWO           Retain          Bound   default/redis-pvc    manual         <unset>                 13s
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/pvc,pv$ kubectl get pvc
NAME      STATUS  VOLUME  CAPACITY  ACCESS MODES  STORAGECLASS  VOLUMEATTRIBUTESCLASS  AGE
redis-pvc Bound   redis-pv  3Gi       RWO           manual         <unset>                 18s
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/pvc,pv$
```

حالا اینبار دوباره کش ردیس را تست میکنیم و میبینیم که پاک نمی شود



Databases > 127.0.0.1:6379 db0  




  All Key Types  Filter by Key Name or Pattern





Total: 1 Last refresh: < 1 min    

STRING	movies_redis:goodman	59 min	960 B
--------	----------------------	--------	-------

```
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/pvc,pv$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
elasticsearch-fbfd4d5d-s5nmg        1/1     Running   0           6s
movieapp-deployment-79c89648d9-d4q58 1/1     Running   1 (79m ago) 2d
movieapp-deployment-79c89648d9-q8p42 1/1     Running   1 (79m ago) 2d
movieapp-deployment-79c89648d9-r6jhc 1/1     Running   1 (79m ago) 2d
redis-deployment-858c6d7699-fbl9g    1/1     Running   0           46h
```

Databases > 127.0.0.1:6379 db0  

  All Key Types  Filter by Key Name or Pattern

Total: 1 Last refresh: < 1 min    

STRING	movies_redis:goodman	58 min	960 B
--------	----------------------	--------	-------

بخش سوم

```
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/docker_project/docker_1/kuber/pvc,pv$ kubectl port-forward redis-deployment-858c6d7699-fbl9g 8080:6379
Forwarding from 127.0.0.1:8080 -> 6379
Forwarding from [::1]:8080 -> 6379
```