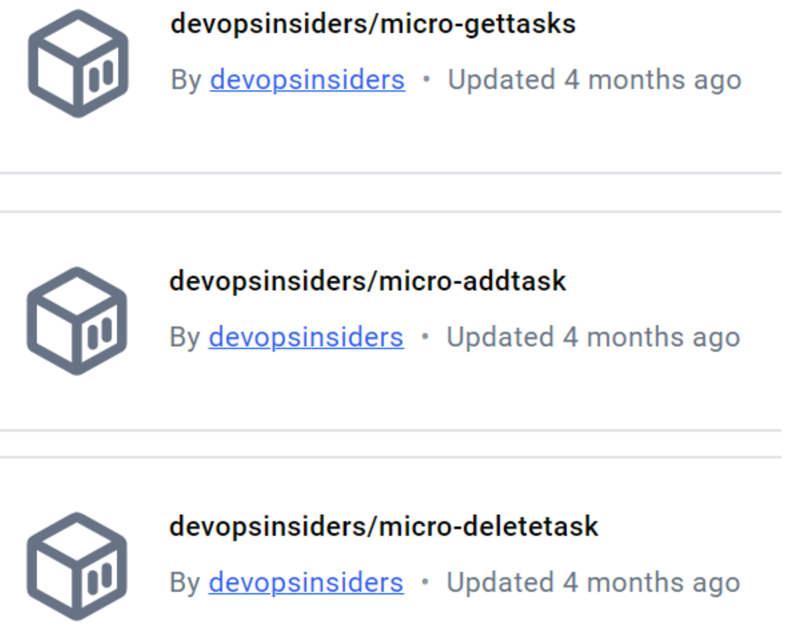
Three tier application Microservices Deployment

Prerequisite:

1. One Databased Creation Needed
2. For Best Practices, Need to store Database connection string in key vault
3. One Key Vault Creation Needed
4. Frontend, Backend and Database
5. Need one Docker System for build Frontend image
6. AKS creation needed
7. Need docker repositories details for backend deployment for docker images.

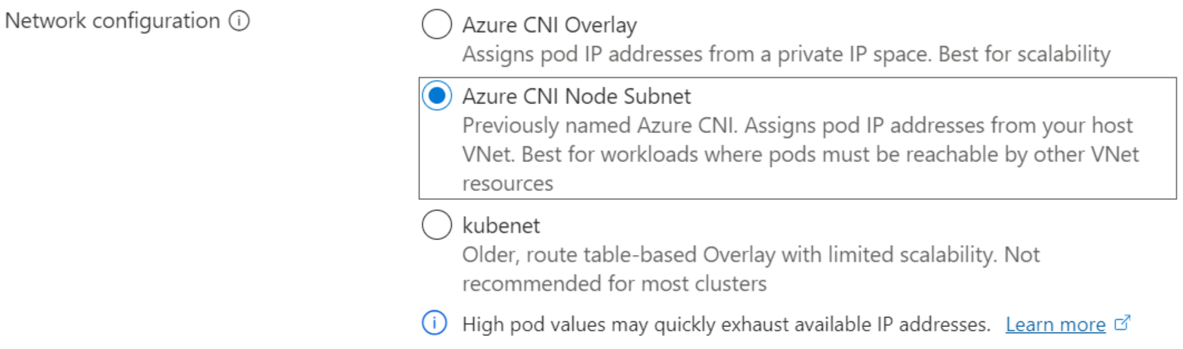


devopsinsiders/micro-gettasks
By devopsinsiders • Updated 4 months ago

devopsinsiders/micro-addtask
By devopsinsiders • Updated 4 months ago

devopsinsiders/micro-deletetask
By devopsinsiders • Updated 4 months ago

8. Need git repositories details for frontend deployment
   https://github.com/satishranjan7183/MicroTodoUI.git
9. Need to enable Azure CNI Node Subnet in network configuration



Network configuration ⓘ

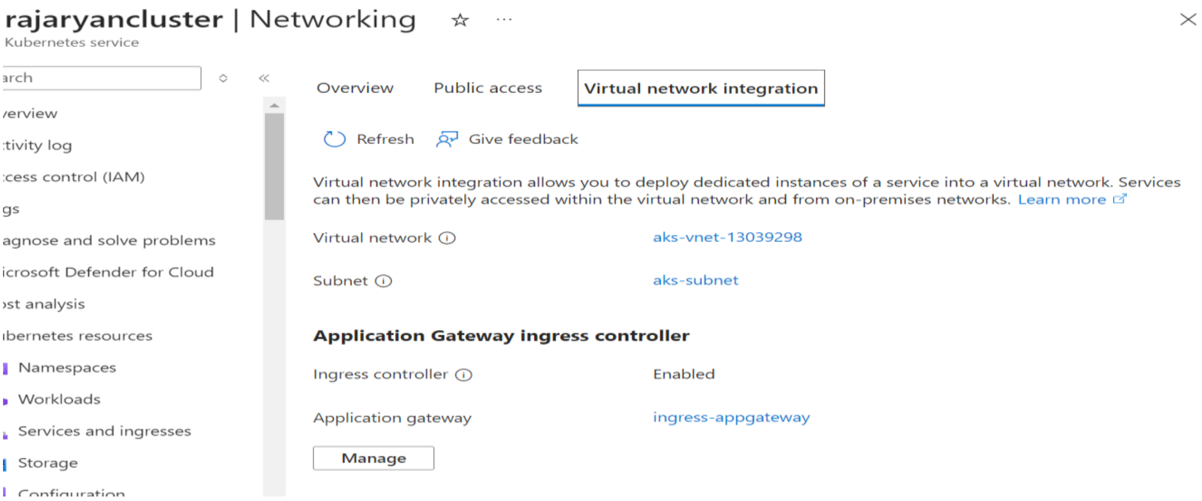
○ Azure CNI Overlay
Assigns pod IP addresses from a private IP space. Best for scalability

● Azure CNI Node Subnet
Previously named Azure CNI. Assigns pod IP addresses from your host VNet. Best for workloads where pods must be reachable by other VNet resources

○ kubenet
Older, route table-based Overlay with limited scalability. Not recommended for most clusters

ⓘ High pod values may quickly exhaust available IP addresses. Learn more

10. After Creating AKS cluster, need to enable Ingress Controller



rajaryancluster | Networking ☆ ···

Kubernetes service

Overview | Public access | Virtual network integration

Refresh | Give feedback

Virtual network integration allows you to deploy dedicated instances of a service into a virtual network. Services can then be privately accessed within the virtual network and from on-premises networks. Learn more

Virtual network ⓘ: aks-vnet-13039298

Subnet ⓘ: aks-subnet

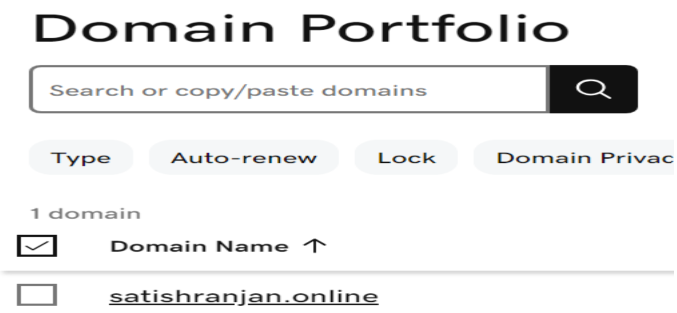
Application Gateway ingress controller

Ingress controller ⓘ: Enabled

Application gateway: ingress-appgateway

Manage

11. DNS needed for this from godaddy.



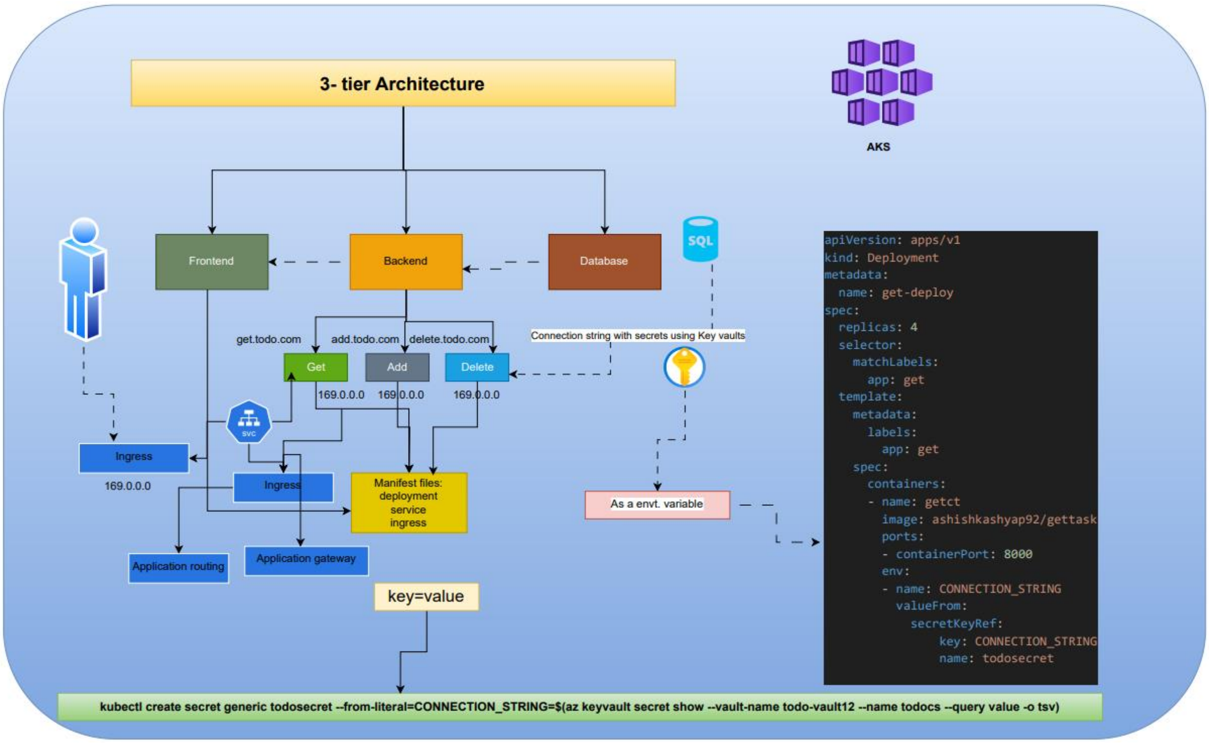
Domain Portfolio

Search or copy/paste domains

1 domain

☑ Domain Name ↑

□ satishranjan.online



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: get-deploy
spec:
  replicas: 4
  selector:
    matchLabels:
      app: get
  template:
    metadata:
      labels:
        app: get
    spec:
      containers:
      - name: getct
        image: ashishkashyap92/gettask
        ports:
        - containerPort: 8000
        env:
        - name: CONNECTION_STRING
          valueFrom:
            secretKeyRef:
                key: CONNECTION_STRING
                name: todosecret
```

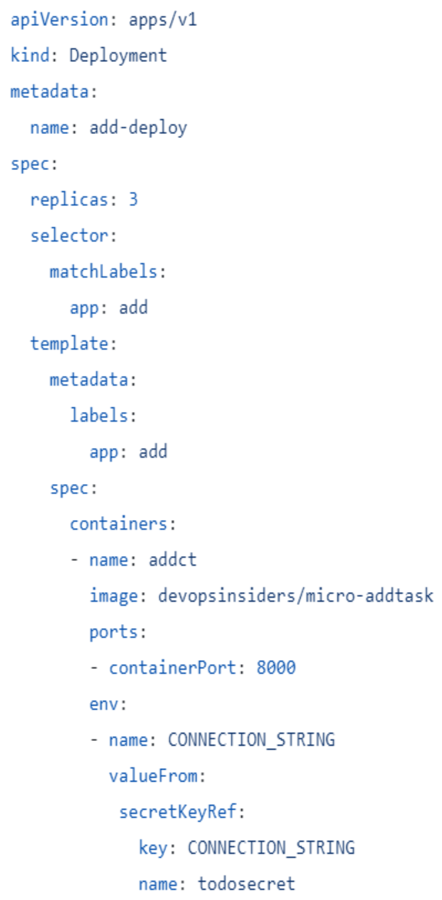
```
kubectl create secret generic todosecret --from-literal=CONNECTION_STRING=$(az keyvault secret show --vault-name todo-vault12 --name todocs --query value -o tsv)
```

Deployment Steps:

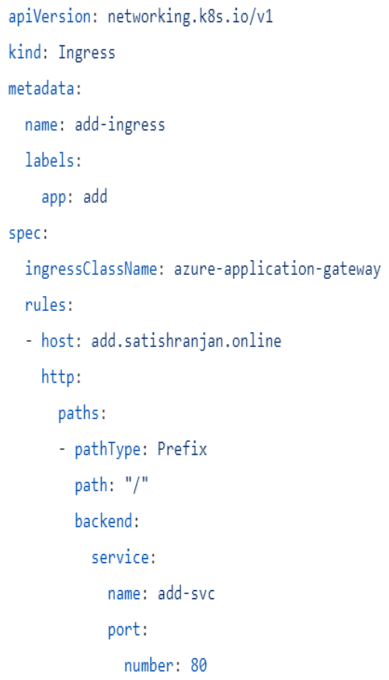
Step1: Need to create 2 folder in our Laptop :

a. Backend
    i. Gettask
        1. Deployment.yaml
        2. Ingress.yaml
        3. Service.yaml
    ii. Addtask
        1. Deployment.yaml
        2. Ingress.yaml
        3. Service.yaml
    iii. Deletetask
        1. Deployment.yaml
        2. Ingress.yaml
        3. Service.yaml

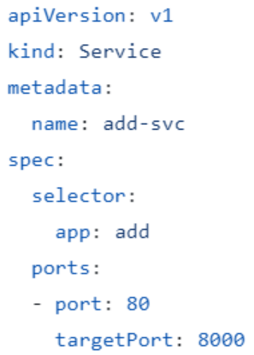
b. frontend
        1. Deployment.yaml
        2. Ingress.yaml
        3. Service.yaml



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: add-deploy
spec:
  replicas: 3
  selector:
    matchLabels:
      app: add
  template:
    metadata:
      labels:
        app: add
    spec:
      containers:
      - name: addct
        image: devopsinsiders/micro-addtask
        ports:
        - containerPort: 8000
        env:
        - name: CONNECTION_STRING
          valueFrom:
           secretKeyRef:
             key: CONNECTION_STRING
             name: todosecret
```



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: add-ingress
  labels:
    app: add
spec:
  ingressClassName: azure-application-gateway
  rules:
  - host: add.satishranjan.online
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: add-svc
            port:
              number: 80
```

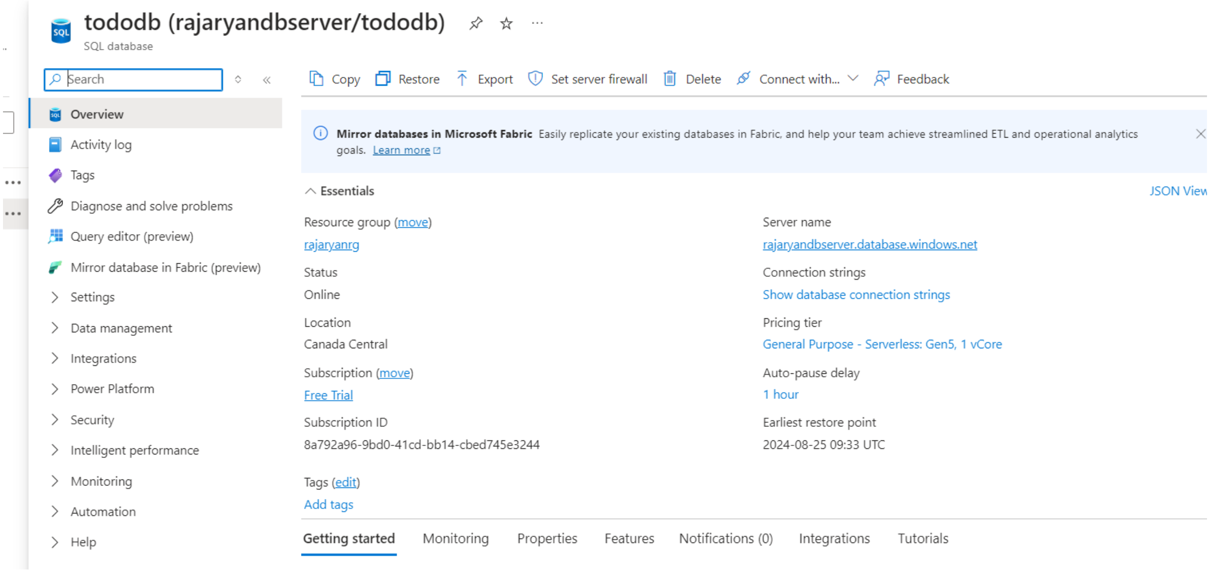


```yaml
apiVersion: v1
kind: Service
metadata:
  name: add-svc
spec:
  selector:
    app: add
  ports:
  - port: 80
    targetPort: 8000
```

Need to write deployment.yaml , ingress.yaml and services.yaml for gettask, addtask and deletetask.

Step2: Create one database for application

- Search sqldatabase in azure portal - > Create database
- while creating database need to enable below parameter :
  Allow Azure services and resources to access this server – Yes
  Add current client IP address – Yes
- Put any name of database



Mirror databases in Microsoft Fabric Easily replicate your existing databases in Fabric, and help your team achieve streamlined ETL and operational analytics goals. Learn more

Resource group (move): rajaryanrg

Status: Online

Location: Canada Central

Subscription (move): Free Trial

Subscription ID: 8a792a96-9bd0-41cd-bb14-cbed745e3244

Tags (edit): Add tags

Server name: rajaryandbserver.database.windows.net

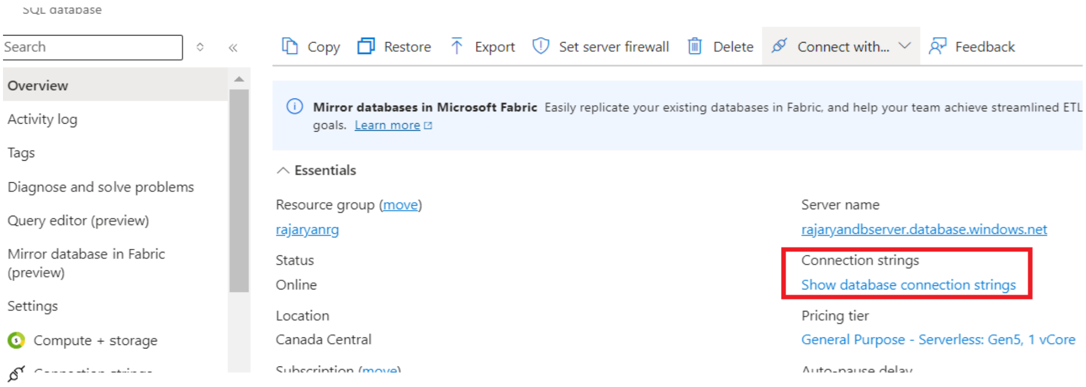
Connection strings: Show database connection strings

Pricing tier: General Purpose - Serverless: Gen5, 1 vCore

Auto-pause delay: 1 hour

Earliest restore point: 2024-08-25 09:33 UTC

Step3: Once database created, then copy connection string



Mirror databases in Microsoft Fabric Easily replicate your existing databases in Fabric, and help your team achieve streamlined ETL goals. Learn more

Resource group (move): rajaryanrg

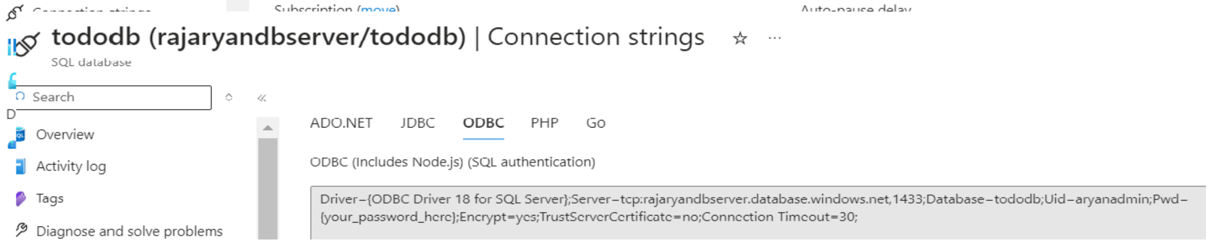
Status: Online

Location: Canada Central

Server name: rajaryandbserver.database.windows.net

Connection strings: Show database connection strings

Pricing tier: General Purpose - Serverless: Gen5, 1 vCore


tododb (rajaryandbserver/tododb) | Connection strings
SQL database
Search
Overview
Activity log
Tags
Diagnose and solve problems
ADO.NET
JDBC
ODBC
PHP
Go

ODBC (Includes Node.js) (SQL authentication)

```
Driver={ODBC Driver 18 for SQL Server};Server=tcp:rajaryandbserver.database.windows.net,1433;Database=tododb;Uid=aryanadmin;Pwd={your_password_here};Encrypt=yes;TrustServerCertificate=no;Connection Timeout=30;
```

Driver={ODBC Driver 18 for SQL Server};Server=tcp:rajaryandbserver.database.windows.net,1433;Database=tododb;Uid=aryanadmin;Pwd={your_password_here};Encrypt=yes;TrustServerCertificate=no;Connection Timeout=30;

Updated ODBC driver version and password and then copy:

Driver={ODBC Driver 17 for SQL Server};Server=tcp:rajaryandbserver.database.windows.net,1433;Database=tododb;Uid=aryanadmin;Pwd=Aryan1215@;Encrypt=yes;TrustServerCertificate=no;Connection Timeout=30;

Step4: Now Need to create Key vault and Secret

Key Vault Creation

- Search Key vault → create Key Vault -> choose Resource group and enter key vault name and then create
  (Put any Key Vault name “todokeyvault11”)

Secret Creation

- Then go inside created keyvault and then got to object → click Secret → Click on Generate/import
- Fill below parameter:
  - Name: <Put any secret name “todocs”>
  - SecretValue : Paste connection string

“Driver={ODBC Driver 17 for SQL Server};Server=tcp:rajaryandbserver.database.windows.net,1433;Database=tododb;Uid=aryanadmin;Pwd=Aryan1215@;Encrypt=yes;TrustServerCertificate=no;Connection Timeout=30;”

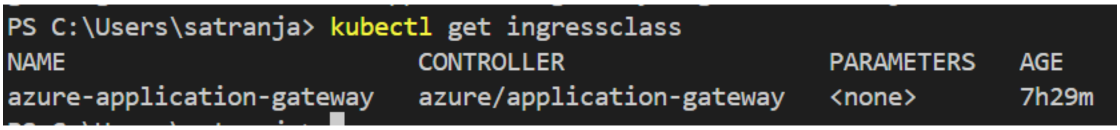
- And then create Secret.

Step5: Now Need to create secret Map in Kubernetes aks

- kubectl create secret generic todosecret --from-literal=CONNECTION_STRING=$(az keyvault secret show --vault-name todovault11 --name todocs --query value -o tsv)
- Kubectl get secret
- Now this secret key and value will use in deployment in environment

Step6: Now Need to update yaml

- Update deployment yaml, ingress yaml and service yaml for all backend

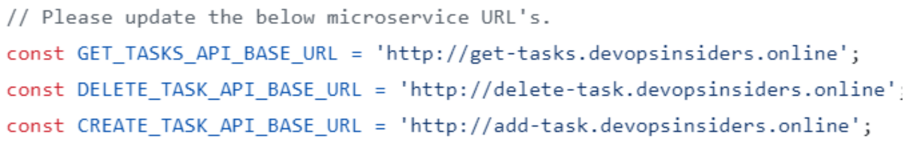


```
PS C:\Users\satranja> kubectl get ingressclass
NAME                        CONTROLLER                  PARAMETERS   AGE
azure-application-gateway   azure/application-gateway   <none>       7h29m
```

- Deploy deployment, ingress and service for all backend application (add task, get task and delete task)
- In deployment yaml update image name which need to take from devopsinsider docker hub repo
- Kubectl apply -f . (Run this command from inside add task , get task and delete task folder where all deployment.yaml, services.yaml and ingress.yaml is present)
- Kubectl get pods
- Kubectl get all

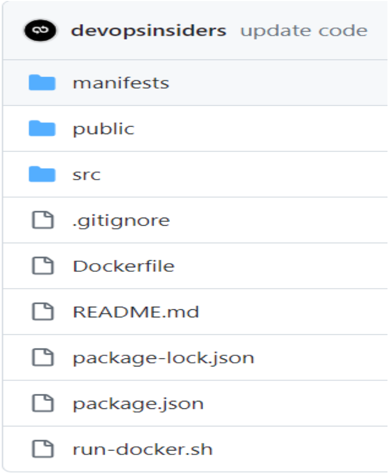
Step7: Now Need to make one folder with name frontend and then git clone the code

- Clone the frontend code from below git hub link:
  https://github.com/satishranjan7183/MicroTodoUI.git
- Once clone successfully, then will go inside src folder and open **TodoApp.js** and update url of microservice:



```
// Please update the below microservice URL's.
const GET_TASKS_API_BASE_URL = 'http://get-tasks.devopsinsiders.online';
const DELETE_TASK_API_BASE_URL = 'http://delete-task.devopsinsiders.online';
const CREATE_TASK_API_BASE_URL = 'http://add-task.devopsinsiders.online';
```

- Now login to docker machine:
  Docker login
- Go inside the MicroTodoUI where all frontend code present, then run docker build command to prepare frontend image:
  Docker build . -t satishranjan7183/frontend



devopsinsiders update code

manifests

public

src

.gitignore

Dockerfile

README.md

package-lock.json

package.json

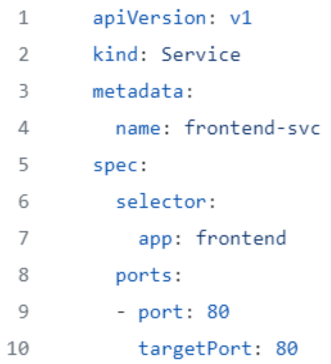
run-docker.sh

Docker images (to check)

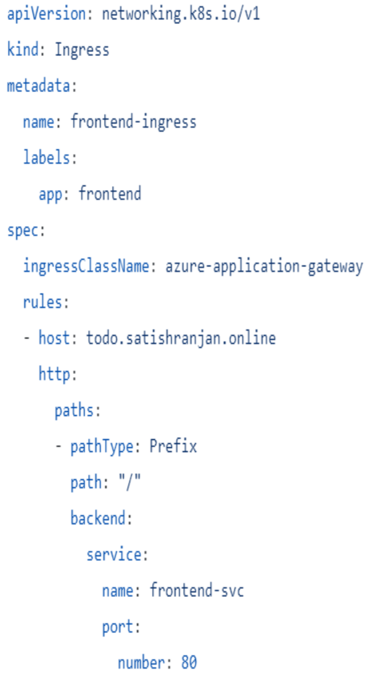
- Once image is ready, then push image to our docker registry
  Docker push satishranjan7183/frontend
- Now go to docker hub and check image pushed in public registry

Step8: Now update the manifest file of frontend

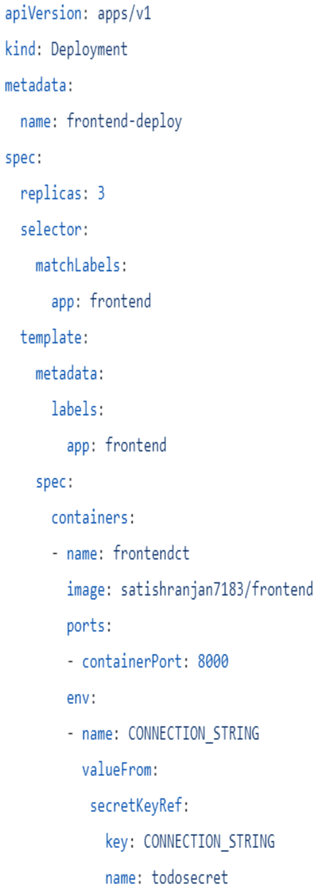
- Now Update the frontend manifest yaml file (deployment.yaml, ingress.yaml and service.yaml)
- In service.yaml need to put port and target port 80 (both port same 80)



```yaml
apiVersion: v1
kind: Service
metadata:
  name: frontend-svc
spec:
  selector:
    app: frontend
  ports:
  - port: 80
    targetPort: 80
```



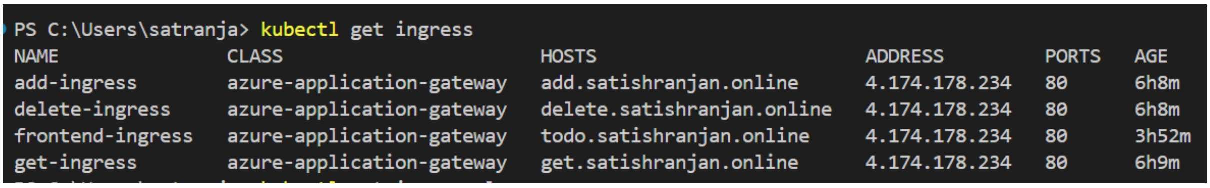
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: frontend-ingress
  labels:
    app: frontend
spec:
  ingressClassName: azure-application-gateway
  rules:
  - host: todo.satishranjan.online
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: frontend-svc
            port:
              number: 80
```



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend-deploy
spec:
  replicas: 3
  selector:
    matchLabels:
      app: frontend
  template:
    metadata:
      labels:
        app: frontend
    spec:
      containers:
      - name: frontendct
        image: satishranjan7183/frontend
        ports:
        - containerPort: 8000
        env:
        - name: CONNECTION_STRING
          valueFrom:
           secretKeyRef:
             key: CONNECTION_STRING
             name: todosecret
```

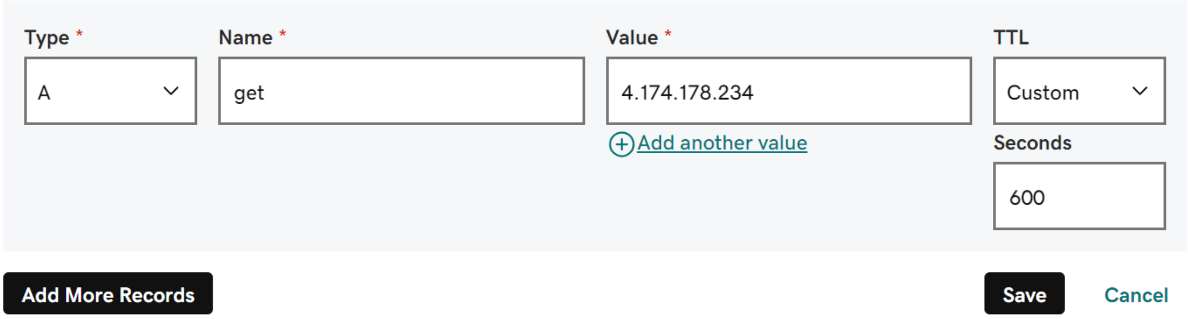
- Now go inside the frontend manifest folder and deploy:
  Kubectl apply -f .
  Kubectl get pod
  Kubectl get all

Step9: Now check ingress IP address and then add host in godaddy



```
PS C:\Users\satranja> kubectl get ingress
NAME               CLASS                       HOSTS                        ADDRESS         PORTS   AGE
add-ingress        azure-application-gateway   add.satishranjan.online      4.174.178.234   80      6h8m
delete-ingress     azure-application-gateway   delete.satishranjan.online   4.174.178.234   80      6h8m
frontend-ingress   azure-application-gateway   todo.satishranjan.online     4.174.178.234   80      3h52m
get-ingress        azure-application-gateway   get.satishranjan.online      4.174.178.234   80      6h9m
```

- Now need to login in go daddy and add host



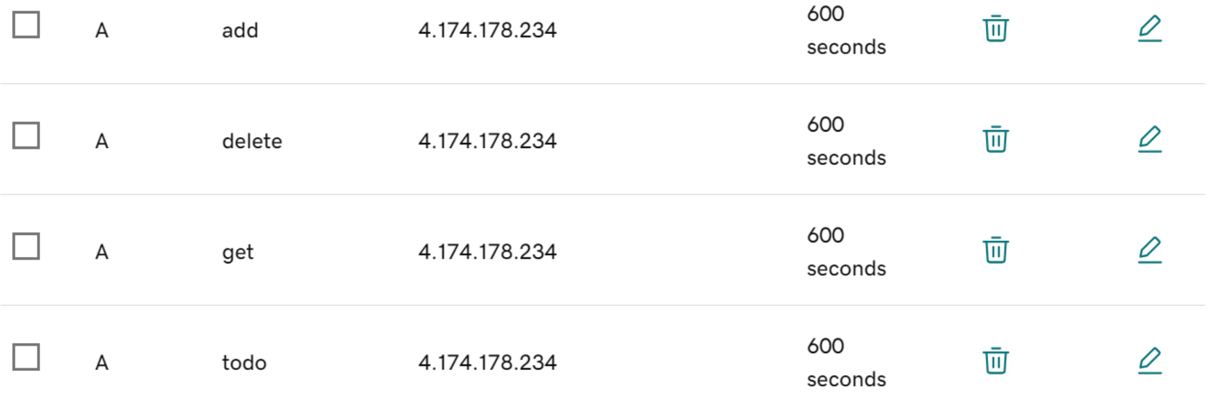
Type * A

Name * get

Value * 4.174.178.234

TTL Custom

Seconds 600



□	A	add	4.174.178.234	600 seconds	🗑	✎
□	A	delete	4.174.178.234	600 seconds	🗑	✎
□	A	get	4.174.178.234	600 seconds	🗑	✎
□	A	todo	4.174.178.234	600 seconds	🗑	✎

- Wait some time and then try below link in browser:
  add.satishranjan.online
  delete.satishranjan.online
  todo.satishranjan.online
  get.satishranjan.online

Step10: Now successfully deployed our Microservices in kubernetes