Task 1

Steps:

1. Create a cluster with a load balancer using the command ($ k3d cluster create mongo -p "8080:8081@loadbalancer") which has the port mapping of 8080:8081 using K3D. The host running the application is 8081.
2. Then create a yaml file called mongo.yaml and add the following:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongodb
  template:
    metadata:
      labels:
        app: mongodb
    spec:
      containers:
        - name: mongodb
          image: mongo:4.4
          ports:
            - containerPort: 27017
          env:
          - name: MONGO_INITDB_ROOT_USERNAME
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-username
          - name: MONGO_INITDB_ROOT_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-password
---
apiVersion: v1
kind: Service
```

metadata:
  name: mongo-service
spec:
  selector:
    app: mongodb
  ports:
    - protocol: TCP
      port: 27017
      targetPort: 27017

```
  GNU nano 4.8                                                    mongo.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongodb
  template:
    metadata:
      labels:
        app: mongodb
    spec:
      containers:
        - name: mongodb
          image: mongo
          ports:
            - containerPort: 27017
          env:
          - name: MONGO_INITDB_ROOT_USERNAME
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-username
          - name: MONGO_INITDB_ROOT_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-password
---
apiVersion: v1
kind: Service
metadata:
  name: mongo-service
spec:
  selector:
    app: mongodb
  ports:
    - protocol: TCP
      port: 27017
      targetPort: 27017
```

Task 2

3. Now we need to create a secret.yaml file for the username and password. We will need to encode the string to base64. Open WSL (Windows) and use the following commands ($ echo -n mongo-root-username | base64) and ($ echo -n mongo-root-password | base64).

   $echo -n mongo-root-username | base64
   bW9uZ28tcm9vdC11c2VybmFtZQ==

   $echo -n mongo-root-password | base64
   bW9uZ28tcm9vdC1wYXNzd29yZA==

4. Now create a secret.yaml file and make the necessary changes to the the mongo-root-username and mongo-root-password with the correct values.

   apiVersion: v1
   kind: Secret
   metadata:
     name: mongodb-secret
   type: Opaque
   data:
     mongo-root-username: bW9uZ28tcm9vdC11c2VybmFtZQ==
     mongo-root-password: bW9uZ28tcm9vdC1wYXNzd29yZA==

```
  GNU nano 4.8                                          secret.yaml
apiVersion: v1
kind: Secret
metadata:
    name: mongodb-secret
type: Opaque
data:
    mongo-root-username: bW9uZ28tcm9vdC11c2VybmFtZQ==
    mongo-root-password: bW9uZ28tcm9vdC1wYXNzd29yZA==
```

Task 3

5. Now create the secret.yaml using the following commands:

kubectl create -f secret.yaml

kubectl create -f mongo.yaml

Note: If receiving the following error ensure that you check your spacing in the yaml file.

```
error: error validating "secret.yaml": error validating data: [ValidationError(Secret): unknown field "mongo-root-passwo
rd" in io.k8s.api.core.v1.Secret, ValidationError(Secret): unknown field "mongo-root-username" in io.k8s.api.core.v1.Sec
ret, ValidationError(Secret): unknown field "name" in io.k8s.api.core.v1.Secret]; if you choose to ignore these errors,
turn validation off with --validate=false
```

Task 4

6. To allow our application to connect to our database, create a config_map.yaml and paste the following:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: mongodb-configmap
data:
  database_url: mongo-service
```

7. Then create a mongo-express.yaml file and paste the following:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongoexp-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongo-express
  template:
    metadata:
      labels:
        app: mongo-express
    spec:
```

```
      containers:
        - name: mongo-express
          image: mongo-express
          ports:
            - containerPort: 8081
          env:
          - name: ME_CONFIG_MONGODB_ADMINUSERNAME
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-username
          - name: ME_CONFIG_MONGODB_ADMINPASSWORD
            valueFrom:
              secretKeyRef:
                name: mongodb-secret
                key: mongo-root-password
          - name: ME_CONFIG_MONGODB_SERVER
            valueFrom:
              configMapKeyRef:
                name: mongodb-configmap
                key: database_url
---
apiVersion: v1
kind: Service
metadata:
  name: mongo-exp-service
spec:
```

```
selector:
  app: mongo-express
type: LoadBalancer
ports:
  - protocol: TCP
    port: 8081
    targetPort: 8081
```

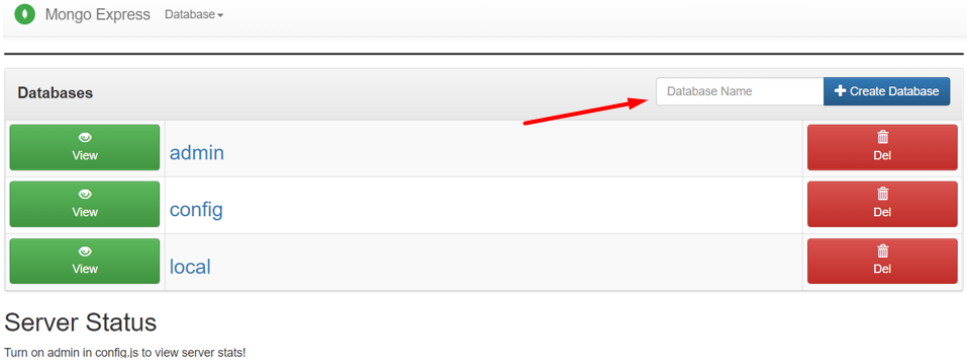
8. Then create both the config_map and the mongo-express.yaml using the commands ($ kubectl create -f config_map.yaml) and ($ kubectl create -f mongo-express.yaml).

Task 5

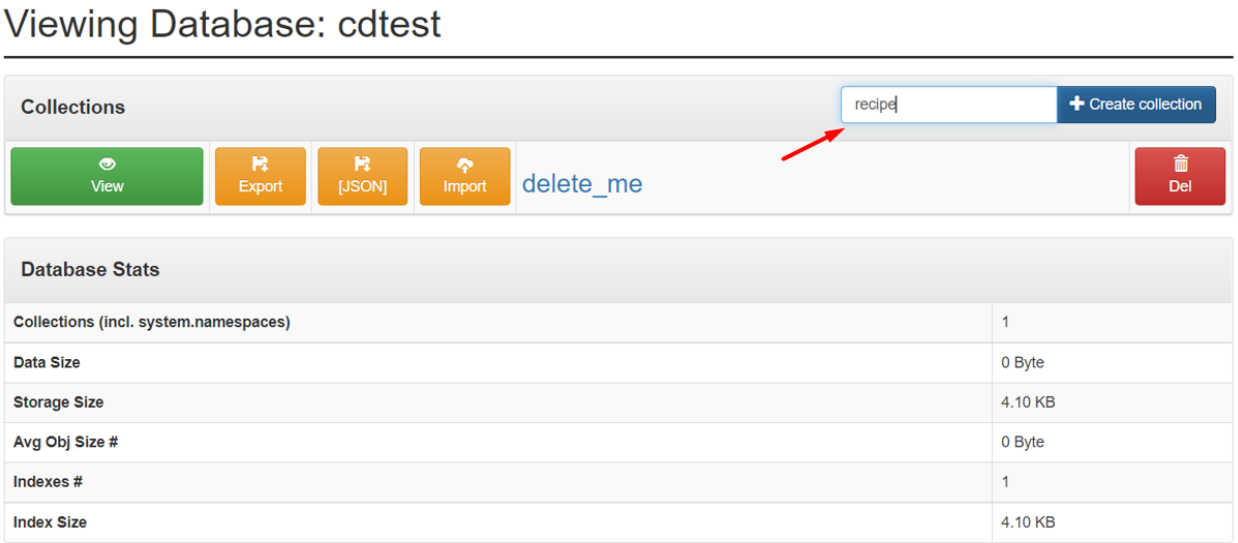
9. Now type localhost:8080 to access the application

Task 6

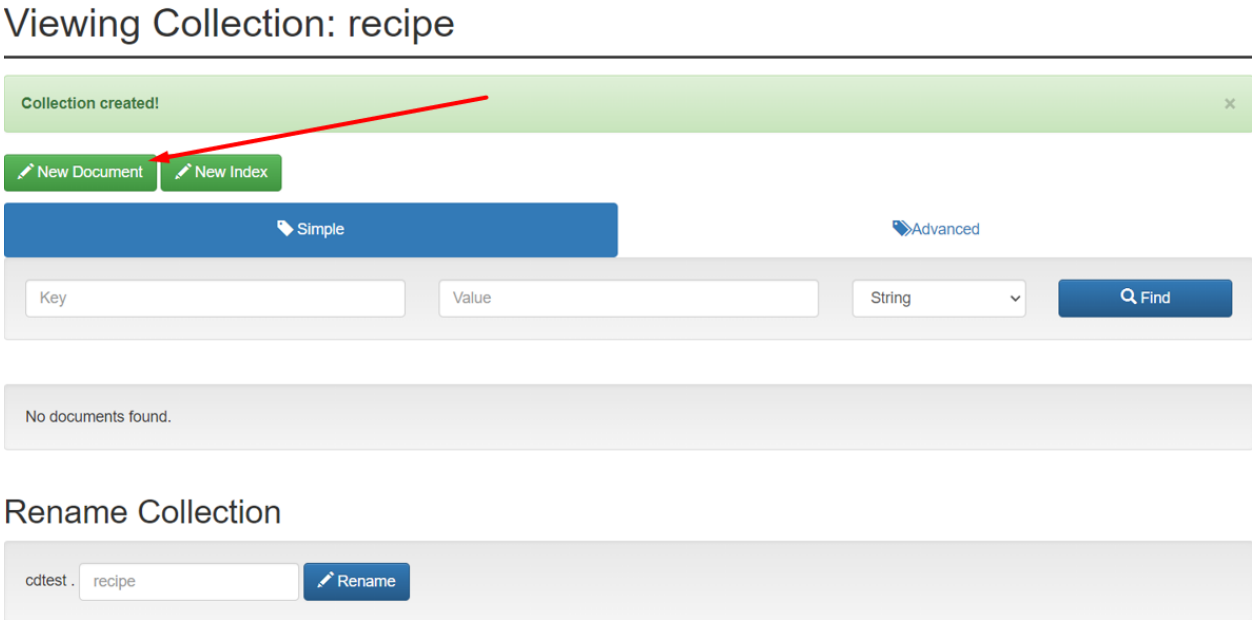
10. Now create a database by giving it a name in the top right corner.

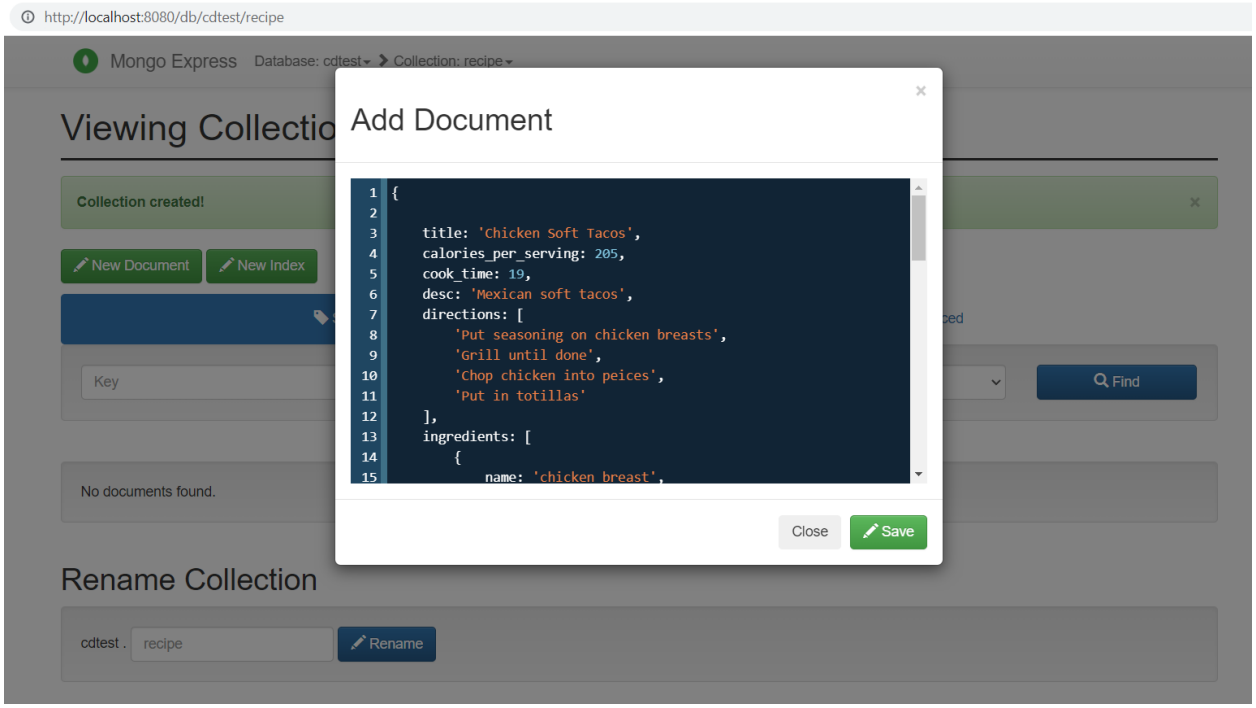


11. Now create a new collection.



12. Then in New Document paste the following:

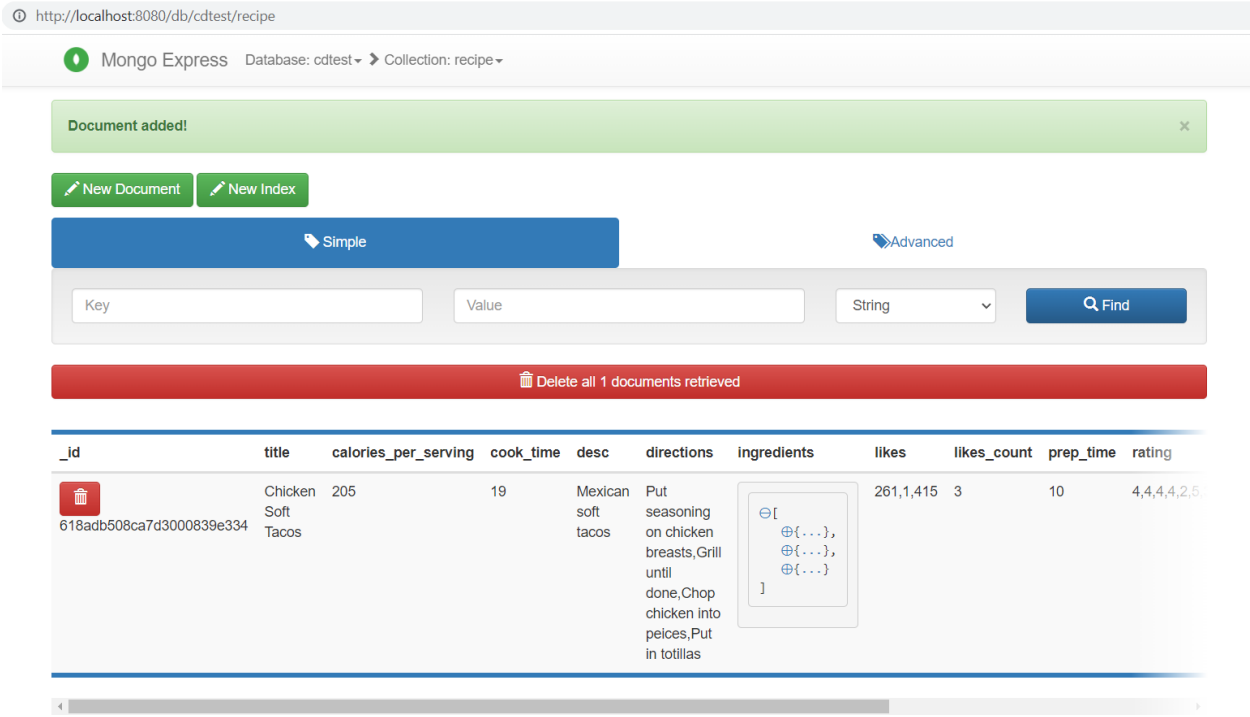




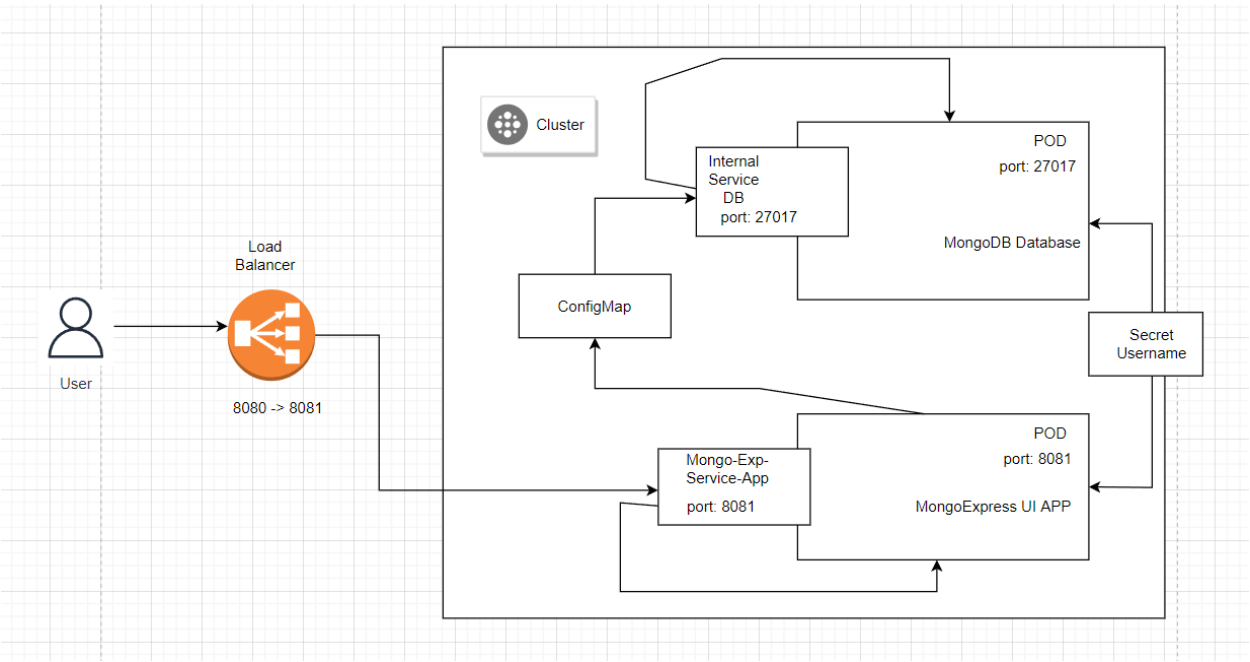

```
{

    title: 'Chicken Soft Tacos',
    calories_per_serving: 205,
    cook_time: 19,
    desc: 'Mexican soft tacos',
    directions: [
        'Put seasoning on chicken breasts',
        'Grill until done',
        'Chop chicken into peices',
        'Put in totillas'
    ],
    ingredients: [
        {
            name: 'chicken breast',
            quantity: {
                amount: 1,
                unit: 'lbs'
            }
        },
        {
            name: 'taco seasoning',
```

```
            quantity: {
                amount: 2,
                unit: 'oz'
            }
        },
        {
            name: 'small flour totillas',
            quantity: {
                amount: 12,
                unit: 'oz'
            }
        }
    ],
    likes: [
        261,
        1,
        415
    ],
    likes_count: 3,
    prep_time: 10,
    rating: [
        4,
        4,
        4,
        4,
        2,
        5,
        3
    ],
    rating_avg: 3.71,
    servings: 5,
    tags: [
        'mexican',
        'quick',
        'easy',
        'chicken'
    ],
    type: 'Dinner'
}
```



13. Now delete the cluster using the following command ($ k3d cluster delete mongo).



When a user accesses the database, the load balancer which is using port 8080 directs them to the Mongo-Exp-Service App. The applications runs on an external port of 27017. The node port is 27017(entry point), therefore if anything needs to communicate with the targetport which has the container it needs to open that pod as well.