# Kubernetes (K8S) YAML Exercise: Secure and Scalable Web Application Deployment on Minikube

This exercise will guide you through deploying a secure and scalable web application using Kubernetes (K8S) on Minikube. The application will include a MongoDB database, a backend server, and a frontend server. You will also learn to work with RBAC permissions in namespaces, perform health checks, and manage replicas.

## Prerequisites

- Minikube installed and running on your local machine  
- Basic knowledge of Kubernetes and YAML syntax

## Exercise Overview

1. Create a Namespace  
2. Set a Resource Quota  
3. Set Up RBAC Permissions  
4. Deploy a MongoDB Database  
5. Deploy a Backend Server  
6. Deploy a Frontend Server  
7. Configure Services (ClusterIP, NodePort, ExternalName)  
8. Use ConfigMaps for Configuration  
9. Use Secrets for Sensitive Data  
10. Perform Health Checks on Replicas

## Instructions

### 1. Create a Namespace

Create a namespace called `secure-app` to isolate the resources.

### 2. Set a Resource Quota

Set a resource quota for the namespace to limit resource usage.

### 3. Set Up RBAC Permissions

Create Role and RoleBinding to manage permissions within the namespace.

- Create a Role that grants permissions to manage pods and services.  
- Create a RoleBinding that assigns the Role to a specific user.

### 4. Deploy a MongoDB Database

Create a Secret for the MongoDB credentials and a ConfigMap for database configuration.

- Create a Secret for MongoDB credentials.  
- Create a ConfigMap for MongoDB configuration.  
- Deploy MongoDB with the created Secret and ConfigMap.

### 5. Deploy a Backend Server

- Deploy the backend server with environment variables configured to connect to the MongoDB database.  
- Ensure the backend server has three replicas.  
- Include readiness and liveness probes for health checks.

### 6. Deploy a Frontend Server

- Deploy the frontend server with environment variables configured to connect to the backend server.  
- Ensure the frontend server has three replicas.  
- Include readiness and liveness probes for health checks.

### 7. Configure Services

- Create a ClusterIP service for the MongoDB database.  
- Create a ClusterIP service for the backend server.  
- Create a NodePort service for the frontend server.  
- Create an ExternalName service to map an external DNS name.

### 8. Use ConfigMaps for Configuration

Create a ConfigMap for the backend server configuration.

### 9. Use Secrets for Sensitive Data

Create a Secret for storing API keys used by the backend server.

### 10. Perform Health Checks on Replicas

Add liveness and readiness probes to the backend and frontend deployments to perform health checks.

## Testing the Setup

Once everything is deployed, you can test the setup using `curl` commands.

1. \*\*Check the Frontend Service:\*\*  
```sh  
curl http://<minikube-ip>:30003  
```

2. \*\*Access the Backend Service from the Frontend Pod:\*\*  
```sh  
kubectl exec -it <frontend-pod> -n secure-app -- curl http://backend:8080  
```

3. \*\*Verify Database Connection from the Backend Pod:\*\*  
```sh  
kubectl exec -it <backend-pod> -n secure-app -- curl http://mongodb:27017  
```

## Submission

Please submit the following YAML files:  
1. Namespace definition  
2. Resource Quota definition  
3. Role and RoleBinding definitions  
4. Secret for MongoDB credentials  
5. ConfigMap for MongoDB configuration  
6. MongoDB Deployment and Service  
7. Backend Deployment and Service with health checks  
8. Frontend Deployment and Service with health checks  
9. ConfigMap for Backend configuration  
10. Secret for API keys  
11. ExternalName Service definition