**Q3 - SCENARIO**

A Toy Retail company ToyTrex has it retail application deployed as 3-tier application - Web App(UI), Web API(middle layer) and Database as Azure SQL.

The user load started increasing multiple fold every month and complex programs getting implemented, the application started performing poorly.

As a result, company decided to re-architect the middle layer as microservices using Azure Kubernetes Services.

The new architecture has below design decisions.

*1) The middle layer should be implemented as Microservices using Azure AKS*

*2) The middle layer API should be deployed as containerized application images*

*3) The container images will use Azure Container Repository (ACR) as the private image repository*

*4) The CI/CD pipelines for microservices should be implemented using Azure DevOps services.*

*5) The Azure DevOps should be able to access ACR and download the container images for microservices deployment*

*6) The image should be deployed as templates such as <image\_name>:<build\_id>*

*Explain the DevOps configuration and steps in detail for above requirements*

**Solution:**

Yes we can implement this scenario we need few resources like ACR we need to create an Azure Container Registry and Azure Kubernetes Cluster under Azure subscription.

We have to create a docker file based on the code that we are using for that API and have to pass the instructions correctly in that docker file to build a docker image.

Also we need to create a deployment.yaml file ( descriptor file ) we have to mention all the things like kind, containers, secrets, name of the image and image path ( acr path ) etc that are required to deploy on AKS cluster.

**CI Pipeline:**

We have to add the tasks docker build and docker push have to configure all the required details from azure subscription, Container registry type, Azure container registry name and the command which needs to be executed like build or push and the docker file path and <APIservice>:$(Build.BuildId).

Add a task to **copy the files to the artifact staging directory** that are included in the deployment folder.( have to fill the fields that are required )

Add another task that needs to be published ( have to fill the fields that are required which includes with a artifact name )

**CD Pipeline:**

We need to add the artifact that we already published in the CI pipeline.  
We need to add **Kubectl** tasks and add the required fields like Azure subscription, AKS Cluster name, service connection type, resource group, command ( apply ) because we are executing and file path of your package.

we have to fill the fields under secrets in that task, type of secret, registry type, container registry etc.,

After that we need to expose the service (APIservice) that we mentioned in the deployment descriptor file for that application, then we have to check the pods are running or not and then execute the command to get the service Kubectl get service -- watch. Once if you see the IP then we can check and browse with that ip whether our application got deployed or not.

We can conclude that here we are dockerizing that image and deploying into the AKS cluster as a Microservice.