**Q1 - SCENARIO**

A car rental company called FastCarz has a .net Web Application and Web API which are recently migrated from on-premise system to Azure cloud using Azure Web App Service

and Web API Service.

The on-premises system had 3 environments Dev, QA and Prod.

The code repository was maintained in TFS and moved to Azure GIT now. The TFS has daily builds which triggers every night which build the solution and copy the build package to drop folder.

deployments were done to the respective environment manually. The customer is planning to setup Azure DevOps service for below requirements:

*1) The build should trigger as soon as anyone in the dev team checks in code to master branch.*

*2) There will be test projects which will create and maintained in the solution along the Web and API. The trigger should build all the 3 projects - Web, API and test.*

*The build should not be successful if any test fails.*

*3) The deployment of code and artifacts should be automated to Dev environment.*

*4) Upon successful deployment to the Dev environment, deployment should be easily promoted to QA and Prod through automated process.*

*5) The deployments to QA and Prod should be enabled with Approvals from approvers only.*

Explain how each of the above the requirements will be met using Azure DevOps configuration.

Explain the steps with configuration details.

**Ans: Scenario\_01 (My implementation with AWS Cloud)**

The car rental company called FastCarz has Web Application and Web API which are recently migrated from on-premise system to Azure cloud using Azure Web App Service

and Web API Service. The on-premises system had 3 environments Dev, QA and Prod

**AWS codecommit:**

* Consider the entire codebase is in AWS Codecommit Git VCS (version control system)
* Implement PR (Pull Request) approval functionality in AWS Codecommit service which can restrict user’s direct code merge with branch. And allow only merge code with branch with PR approval.

**Environment (Dev | QA | Prod): AWS ECS service with three cluster for dev qa and prod respectively:**

* Create three cluster for 3 environment desired resource.
* Maintain Autoscaling respectively for cluster as well as service level.

**Orchestration Tool (Jenkins) for CI/CD:**

* List of Jenkins jobs will get created for all 3 projects, i.e. Web, API & Test
* In-order to achieve immediate build and deliver Artefacts.

Please find answers in green color:

* *1) The build should trigger as soon as anyone in the dev team checks in code to master branch.*

In Jenkins job, poll SCM mechanism can be implemented to trigger the build as soon as team checks in the changes into master branch.

* *2) There will be test projects which will create and maintained in the solution along the Web and API. The trigger should build all the 3 projects - Web, API and test.*

Jenkins pipeline job can be created to achieve such requirement, i.e. as long as the job get trigger it can cover the pipeline for Web and API. And the same for all 3 projects.

* *The build should not be successful if any test fails.*

The Jenkins job configuration can be done is such a way that the job should not proceed for build if in case found any such hiccups i.e. if the test case gets failed. This can be achieved by implementing check point in the Jenkins job configuration like conditional plugins in the pipeline. If there any test case failure or static SONARQUBE failure the job gets break the build and notify to the respective team over the email or Slack/teams channel.

* *3) The deployment of code and artifacts should be automated to Dev environment.*

The Jenkins job should be prepared in a way that the pipeline job should build the artefacts and deploy the same over Dev Environment by providing dev env credential/argument/parameter details*.*

* *4) Upon successful deployment to the Dev environment, deployment should be easily promoted to QA and Prod through automated process.*

Once the Dev env release get complete and upon successful functional/sanity testing, the respective artefacts can be freezed with TAG and the same can be stored in any artefacts storage tool like Nexus or JFrog or AWS ECR (Docker Image artefacts), and the respective TAG artefacts can be released to the next env (QA & Prod) through Jenkins automation process job.

* *5) The deployments to QA and Prod should be enabled with Approvals from approvers only.*

While creating CD job, approval mechanism can be implemented by installing/configuring approval process with the help of **Promotion plugins**. Upon Job A approval the next pipeline job B will get trigger.

*End….*