**SCENARIO 1:**

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

and Web API Service.

The on-premises system had 3 environments Dev, QA, Sand 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.*

Solution: This can be done by the two approaches:

* By using the UI option provided by Azure DevOps in the triggers section.
* Steps for Approach1:
  + Navigate to the pipeline
  + Edit the pipeline
  + Click on the triggers option present on the top header (if the pipeline is built in Classic UI)
  + Click on the top right ellipsis and navigate to triggers (if the pipeline is built using YAML)
  + Enable the continuous integration trigger
  + And, set the branch as 'master' in the include section.
* Appraoch2 (only for YAML pipeline):
  + Add the following snippet in the YAML
    - trigger:
    - branches:
    - include:
      * "master"

*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.*

Flowchart for the solution:

Stage: Build 
Jobl: Build Projecti 
Job2: Build Project2 
Job3: Build Test Project 
Stage: Test 
Jobl: Execute Smoke Test 
If any Of the buid Job fails, the 
execution Will Stop there, else the 
test Stage Will initiate and on 
successful completion only, the 
pipeline Will be grees 

Job3 will produce the test artifact and those will be used to run the smoke test job in test stage. Also, find the dummy YAML in the following folder structure.

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

Solution: In the continuation of about scenario i.e., once the smoke test gets passed and required artifacts are generated in the build pipeline, the auto deployment can be achieved by using any of the below two methods.

Approach1: Using the YAML pipeline

* With the new updates from MS, we have the option to use the YAML in the release pipelines.
* Steps:
  + Create a YAML with the stage->Job->task to deploy the changes to dev environment.
  + Create a pipeline using that YAML
  + Add the build completion trigger to this pipeline for the build pipeline and that will make the auto deployment to the dev environment.

Approach2: Using release pipeline (Classic UI)

* Steps:
  + Create a new release pipeline.
  + In the artifact section, provide the CI pipeline name and details.
  + In the stages, create a stage->Job->tasks to deploy to the dev environment
  + On the stage level, navigate and select, all the required artifacts (by default it will download all the artifacts).
  + Enable the continuous deployment trigger (at the level of artifacts).

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

Solution: In continuation with the approach1 i.e., using YAML,

* This can be done either by adding multiple jobs/stages. We can add the checks using environments and that will stop the execution from automatic deployment to QA and prod. However, the cons will be that one of the agents will be on hold waiting for the response of the approvers.
* Using the YAML, we can also re-use a same YAML and create multiple pipelines using that and control that by passing pipeline variables.

Recommended by MS, we should use release pipeline for such scenarios,

So, flowchart will be like:

Add 
Artifacts 
Schedule 
not set 
I + Add v 
Stages 
Dev Deployment 
I job, 9 ta5k5 
QA Deploymnet 
I job, 9 tasks 
Prod Deployment 
I job, 9 tasks 

Wherein, the Dev will be automatic deployed, and the QA and prod can be auto triggered but with approvals i.e., will be on hold until the approvers approves, or the other option can be using the manual trigger and with/without approvals (as per the requirement).

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

Approaches:

Approach1: In case of YAML, this can be done using the environment.

Steps:

* Create an environment
* Add approvals and checks
* And add that environment to the QA and Prod deployment job

Approach2:

Steps:

* As mentioned in the point4, we can enable the pre-deployment approvals for QA and prod deployment stages.