**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 Pipeline service for below requirements:

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

Create a separate repository for the pipelines which will trigger the builds in any of the api, web, test git projects when a commit is made to them. Complete pipeline is part of the git hub repo

**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.**

resources:

  repositories:

  - repository: web

    type: git

    name: FastCarz/web

    trigger:

    - master

  - repository: api

    type: git

    name: FastCarz/api

    trigger:

    - master

  - repository: test

    type: git

    name: FastCarz/test

    trigger:

    - master

The build should not be successful if any test fails.

CI Stage build all the 3 projects in order and API, WEB and TEST and is only successful when all the three passes.

stages:

- stage: CI

  jobs:

  - job: API

    steps:

    - checkout: api

    - task: Maven@3

      displayName: Maven Install

      inputs:

        mavenPomFile: 'pom.xml'

        mavenOptions: '-Xmx3072m'

        javaHomeOption: 'JDKVersion'

        jdkVersionOption: '1.8'

        jdkArchitectureOption: 'x64'

        publishJUnitResults: true

        testResultsFiles: '\*\*/surefire-reports/TEST-\*.xml'

        goals: 'install'

  - job: WEB

    dependsOn: API

    steps:

    - checkout: web

    - task: Maven@3

      inputs:

        mavenPomFile: 'pom.xml'

        mavenOptions: '-Xmx3072m'

        javaHomeOption: 'JDKVersion'

        jdkVersionOption: '1.8'

        jdkArchitectureOption: 'x64'

        publishJUnitResults: true

        testResultsFiles: '\*\*/surefire-reports/TEST-\*.xml'

        goals: 'install'

  - job: TEST

    dependsOn: WEB

    steps:

    - checkout: test

    - task: Maven@3

      inputs:

        mavenPomFile: 'pom.xml'

        mavenOptions: '-Xmx3072m'

        javaHomeOption: 'JDKVersion'

        jdkVersionOption: '1.8'

        jdkArchitectureOption: 'x64'

        publishJUnitResults: true

        testResultsFiles: '\*\*/surefire-reports/TEST-\*.xml'

        goals: 'install'

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

Stage DEPLOY\_DEV is called when the CI stage is successfully completed. I have put dummy tasks in the deployment steps.

- stage: Deploy\_DEV

  dependsOn: CI

  jobs:

  - deployment: deploy

    displayName: Deploy to Dev

    environment: dev

    strategy:

      runOnce:

        deploy:

          steps:

          - task: Bash@3

            inputs:

              targetType: 'inline'

              script: |

                echo 'Hello world'

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

This can be done either through calling the DEPOY\_QA and DEPLOY\_PROD stage after the DEPLOY\_DEV is successful and putting the Manual Validation step that will notify users and wait for the approval.

- stage: Deploy\_QA

  dependsOn: Deploy\_DEV

  jobs:

  - job: waitForQA\_approval

    displayName: Wait forApproval for QA

    pool: server

    timeoutInMinutes: 1440 # job times out in 1 days

    steps:

    - task: ManualValidation@0

      timeoutInMinutes: 1440 # task times out in 1 hour

      inputs:

        notifyUsers: |

          abhishek.yadav@igtsolutions.com

        instructions: 'Please Provide approval to deploy to QA'

        onTimeout: 'reject'

  - deployment: deploy

    dependsOn: waitForQA\_approval

    displayName: Deploy to QA

    environment: qa

    strategy:

      runOnce:

        deploy:

          steps:

          - task: Bash@3

            inputs:

              targetType: 'inline'

              script: |

                echo 'Hello world'

- stage: Deploy\_PROD

  dependsOn: Deploy\_QA

  jobs:

  - job: waitForPROD\_approval

    displayName: Wait forApproval for PROD

    pool: server

    timeoutInMinutes: 1440 # job times out in 1 days

    steps:

    - task: ManualValidation@0

      timeoutInMinutes: 1440 # task times out in 1 hour

      inputs:

        notifyUsers: |

          abhishek.yadav@sita.aero

        instructions: 'Please Provide approval to deploy to PROD'

        onTimeout: 'reject'

  - deployment: deploy

    dependsOn: waitForPROD\_approval

    displayName: Deploy to PROD

    environment: prod

    strategy:

      runOnce:

        deploy:

          steps:

          - task: Bash@3

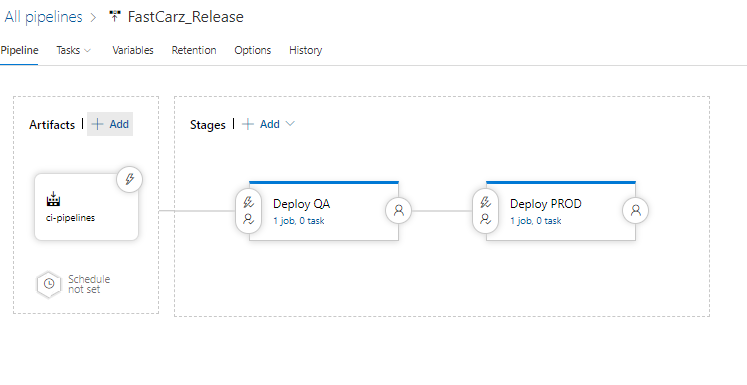
            inputs:

              targetType: 'inline'

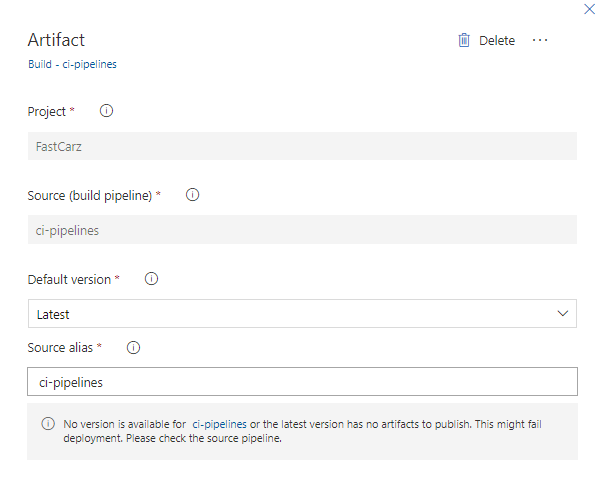
              script: |

                echo 'Hello world'

Other approach will be to call the Azure Release pipeline that will be triggered when the DEPLOY\_DEV stage is completed, so the dev deployment will be part of the yaml pipeline and QA and prod deployment will be part of the release pipeline.

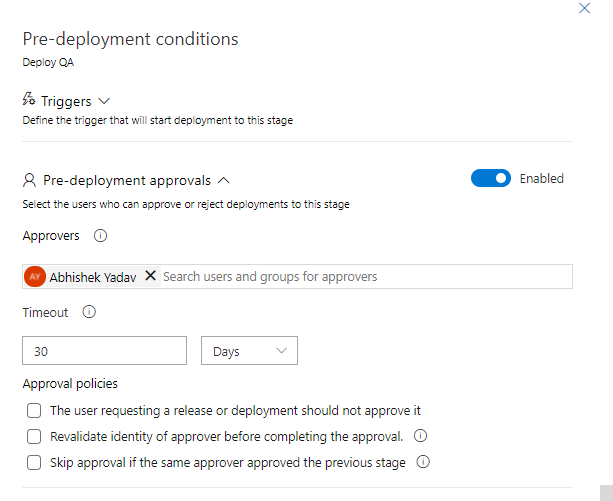


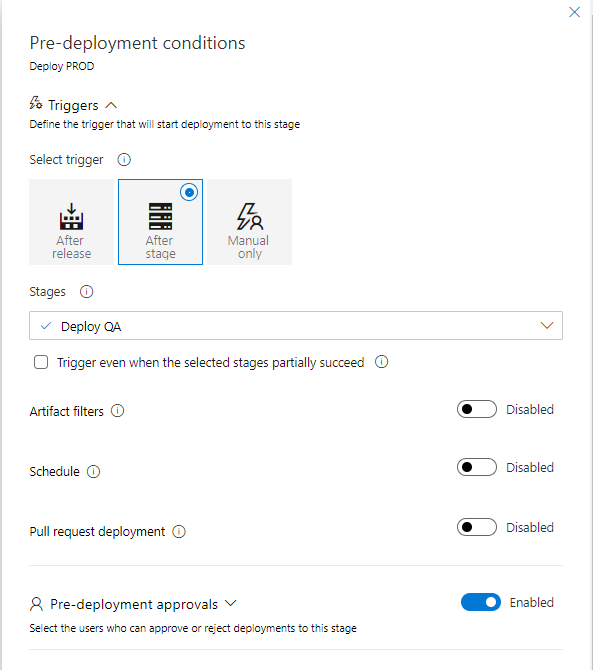
Trigger for the Release pipeline will be the CI Pipeline



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

Approvals can be set in the release pipelines as below or with ManualValidation steps mentioned above in the pipeline





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

Explain the steps with configuration details.