Name: Mahesh Valikar

Q1 – SCENARIO

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

Answer: We should select check for “**Enable continuous integration**” to build for every check-in. Also specify branch you want to associate for this pipeline. Please refer screenshot below

Graphical user interface

Description automatically generated

1. 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.

Answer: We should include tasks to restore, build, test and publish. Please refer screenshot below

Graphical user interface, text, application

Description automatically generated

To keep build to be failed when any test fails, we should enable control option on test execution task. As shown belowGraphical user interface, text, application

Description automatically generated

Additionally, we can configure on immediate next task as show below

Graphical user interface, text, application

Description automatically generated

1. The deployment of code and artifacts should be automated to Dev environment.

Answer: First we need link artifacts from build with this release as shown below(\_fastcarz.ci).

Then we need to select trigger type as “After release” for this pipeline. This will automatically gets deployed to Dev environment whenevern new build artifact is ready.

A picture containing text

Description automatically generated

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

Answer: we should create stage for QA and PRD as shown below.

Diagram

Description automatically generated with medium confidence

And we should select trigger type as “After Stage” for subsequent stages as shown below

Diagram

Description automatically generated

1. The deployments to QA and Prod should be enabled with Approvals from approvers only.

Answer: We should enable “**Pre-deployment approval**” setting on stage as shown below

Graphical user interface

Description automatically generated

**Q2 – SCENARIO**

1. What are different artifacts you need to create - name of the artifacts and its purpose

Answer: Following artifacts are required

1. Azure service principal -- Authenticate to Azure
2. Azure DevOps git -- to store all terraform scripts
3. List the tools you will create and store the Terraform templates.

Answer:

1. Azure-storage-account to store state file
2. Azure git repo for code base storage
3. Azure keyvault to store credentials or sensitive data
4. Explain the process and steps to create automated deployment pipeline.

Answer: Deployment should contain task to “**init, validate, plan & apply**” as shown below

Graphical user interface, text, application, email

Description automatically generated

4) Create a sample Terraform template you will use to deploy Below services:

Vnet

2 Subnet

NSG to open port 80 and 443

1 Window VM in each subnet

1 Storage account

Answer:

We need to use following resource in terraform to create resources in azure

azurerm\_resource\_group --- Resource group

azurerm\_network\_security\_group – For NSG

azurerm\_network\_security\_rule – to open port 80 and 443 on NSG

azurerm\_virtual\_network -- to create vnet

azurerm\_subnet -- to create subnet

azurerm\_network\_interface -- to create nic

azurerm\_windows\_virtual\_machine – to create windows vm

azurerm\_storage\_account -- to create storage account



1. Explain how you will access the password stored in Key Vault and use it as Admin Password in the VM Terraform template.

Answer: We need to use “azurerm\_key\_vault\_secret” data to fetch keyvault secret as shown below

data "azurerm\_key\_vault\_secret" "maersktestusername" {

name = "username"

key\_vault\_id = data.azurerm\_key\_vault.maersktestkv.id

}

output "username" {

value = data.azurerm\_key\_vault\_secret.maersktestkv.value

}