AZ-204$^{Q\&As}$

Developing Solutions for Microsoft Azure

Pass Microsoft AZ-204 Exam with 100% Guarantee

QUESTION 1

You are developing a solution that will use Azure messaging services.

You need to ensure that the solution uses a publish-subscribe model and eliminates the need for constant polling.

What are two possible ways to achieve the goal? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

A. Service Bus

B. Event Hub

C. Event Grid

D. Queue

Correct Answer: AC

It is strongly recommended to use available messaging products and services that support a publish-subscribe model, rather than building your own. In Azure, consider using Service Bus or Event Grid. Other technologies that can be used for pub/sub messaging include Redis, RabbitMQ, and Apache Kafka.

Reference: https://docs.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber

QUESTION 2

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while

others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You develop a software as a service (SaaS) offering to manage photographs. Users upload photos to a web service which then stores the photos in Azure Storage Blob storage. The storage account type is General-purpose V2.

When photos are uploaded, they must be processed to produce and save a mobile-friendly version of the image. The process to produce a mobile-friendly version of the image must start in less than one minute.

You need to design the process that starts the photo processing.

Solution: Convert the Azure Storage account to a BlockBlobStorage storage account.

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: B

Not necessary to convert the account, instead move photo processing to an Azure Function triggered from the blob upload..

Azure Storage events allow applications to react to events. Common Blob storage event scenarios include image or video processing, search indexing, or any file-oriented workflow.

Note: Only storage accounts of kind StorageV2 (general purpose v2) and BlobStorage support event integration. Storage (general purpose v1) does not support integration with Event Grid.

Reference:

https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blob-event-overview

QUESTION 3

You have a new Azure subscription. You are developing an internal website for employees to view sensitive data. The website uses Azure Active Directory (Azure AD) for authentication.

You need to implement multifactor authentication for the website.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Configure the website to use Azure AD B2C.

B. In Azure AD, create a new conditional access policy.

C. Upgrade to Azure AD Premium.

D. In Azure AD, enable application proxy.

E. In Azure AD conditional access, enable the baseline policy.

Correct Answer: BC

B: MFA Enabled by conditional access policy. It is the most flexible means to enable two-step verification for your users. Enabling using conditional access policy only works for Azure MFA in the cloud and is a premium feature of Azure AD.

C: Multi-Factor Authentication comes as part of the following offerings:

Azure Active Directory Premium licenses - Full featured use of Azure Multi-Factor Authentication Service (Cloud) or Azure Multi-Factor Authentication Server (On-premises).

Multi-Factor Authentication for Office 365

Azure Active Directory Global Administrators

Reference:

https://docs.microsoft.com/en-us/azure/active-directory/authentication/howto-mfa-getstarted

QUESTION 4

HOTSPOT

You have a single page application (SPA) web application that manages information based on data returned by Microsoft Graph from another company\\'s Azure Active Directory (Azure AD) instance.

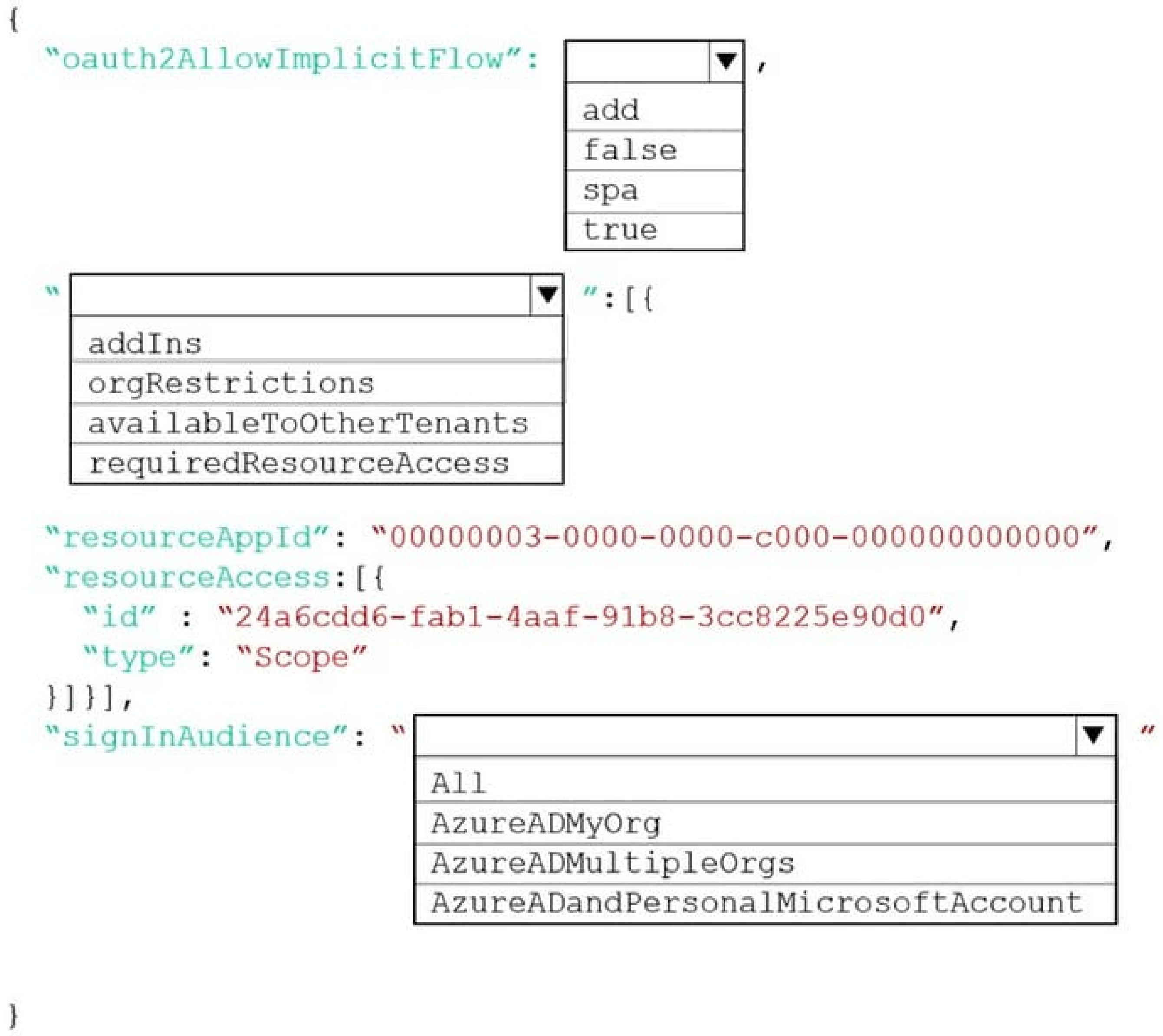
Users must be able to authenticate and access Microsoft Graph by using their own company\\'s Azure AD instance.

You need to configure the application manifest for the app registration.

How should you complete the manifest? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:



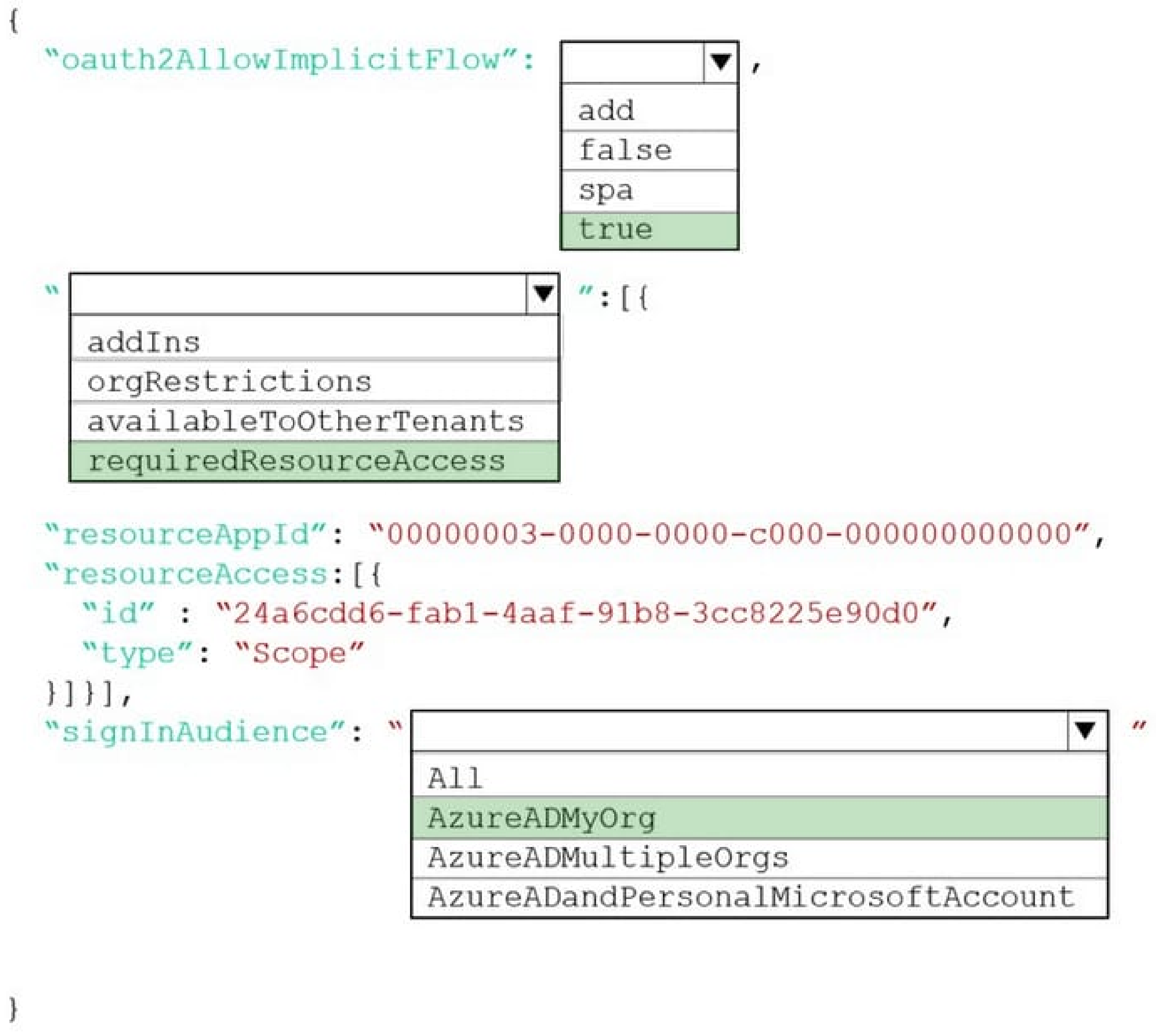
```
{
  "oauth2AllowImplicitFlow":  [          ▼] ,
                              add
                              false
                              spa
                              true

  " [                          ▼] ":[{
     addIns
     orgRestrictions
     availableToOtherTenants
     requiredResourceAccess

  "resourceAppId": "00000003-0000-0000-c000-000000000000",
  "resourceAccess":[{
    "id" : "24a6cdd6-fab1-4aaf-91b8-3cc8225e90d0",
    "type": "Scope"
  }]}],
  "signInAudience": " [                          ▼] "
                       All
                       AzureADMyOrg
                       AzureADMultipleOrgs
                       AzureADandPersonalMicrosoftAccount

}
```

Correct Answer:



```
{
  "oauth2AllowImplicitFlow": [          ▼] ,

  "[                              ▼]":[{

  "resourceAppId": "00000003-0000-0000-c000-000000000000",
  "resourceAccess":[{
    "id" : "24a6cdd6-fab1-4aaf-91b8-3cc8225e90d0",
    "type": "Scope"
  }]}],
  "signInAudience": "[                              ▼]"

}
```

Box 1: true

The oauth2AllowImplicitFlow attribute Specifies whether this web app can request OAuth2.0 implicit flow access tokens. The default is false. This flag is used for browser-based apps, like JavaScript single-page apps.

In implicit flow, the app receives tokens directly from the Azure Active Directory (Azure AD) authorize endpoint, without any server-to-server exchange. All authentication logic and session handling is done entirely in the JavaScript client with

either a page redirect or a pop-up box.

Box 2: requiredResourceAccess

With dynamic consent, requiredResourceAccess drives the admin consent experience and the user consent experience for users who are using static consent. However, this parameter doesn\\'t drive the user consent experience for the

general case.

resourceAppId is the unique identifier for the resource that the app requires access to. This value should be equal to the appId declared on the target resource app.

resourceAccess is an array that lists the OAuth2.0 permission scopes and app roles that the app requires from the specified resource. Contains the id and type values of the specified resources.

Example: "requiredResourceAccess": [

```
 {
 "resourceAppId": "00000002-0000-0000-c000-000000000000",
 "resourceAccess": [
 {
 "id": "311a71cc-e848-46a1-bdf8-97ff7156d8e6",
 "type": "Scope"
 }
 ]
 }
 ],
```

Incorrect Answers:

1.

The legacy attribute availableToOtherTenants is no longer supported.

2.

The addIns attribute defines custom behavior that a consuming service can use to call an app in specific contexts. For example, applications that can render file streams may set the addIns property for its "FileHandler" functionality. This parameter will let services like Microsoft 365 call the application in the context of a document the user is working on.

Example: "addIns": [

```
 {
 "id": "968A844F-7A47-430C-9163-07AE7C31D407",
 "type":" FileHandler",
 "properties": [
 {
 "key": "version",
 "value": "2"
 }
```

```
]

}

],
```

Box 3: AzureADMyOrg

The signInAudience attribute specifies what Microsoft accounts are supported for the current application. Supported values are:

1.

AzureADMyOrg - Users with a Microsoft work or school account in my organization\\'s Azure AD tenant (for example, single tenant)

2.

AzureADMultipleOrgs - Users with a Microsoft work or school account in any organization\\'s Azure AD tenant (for example, multi-tenant)

3.

AzureADandPersonalMicrosoftAccount - Users with a personal Microsoft account, or a work or school account in any organization\\'s Azure AD tenant

Reference: https://docs.microsoft.com/en-us/azure/active-directory/develop/reference-app-manifest
https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-implicit-grant-flow

QUESTION 5

HOTSPOT

A company is developing a Java web app. The web app code is hosted in a GitHub repository located at https://github.com/Contoso/webapp.

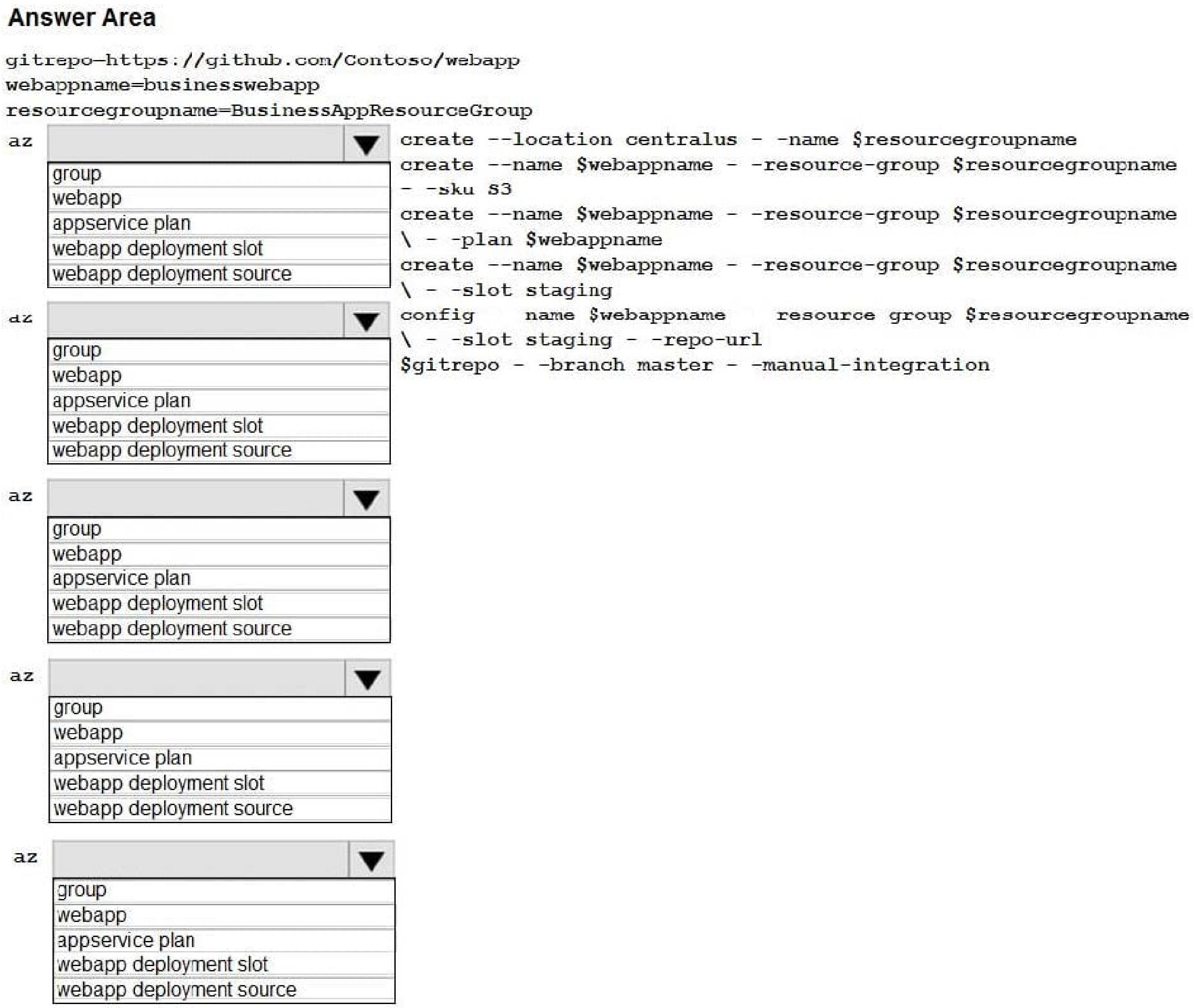
The web app must be evaluated before it is moved to production. You must deploy the initial code release to a deployment slot named staging.

You need to create the web app and deploy the code.

How should you complete the commands? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:

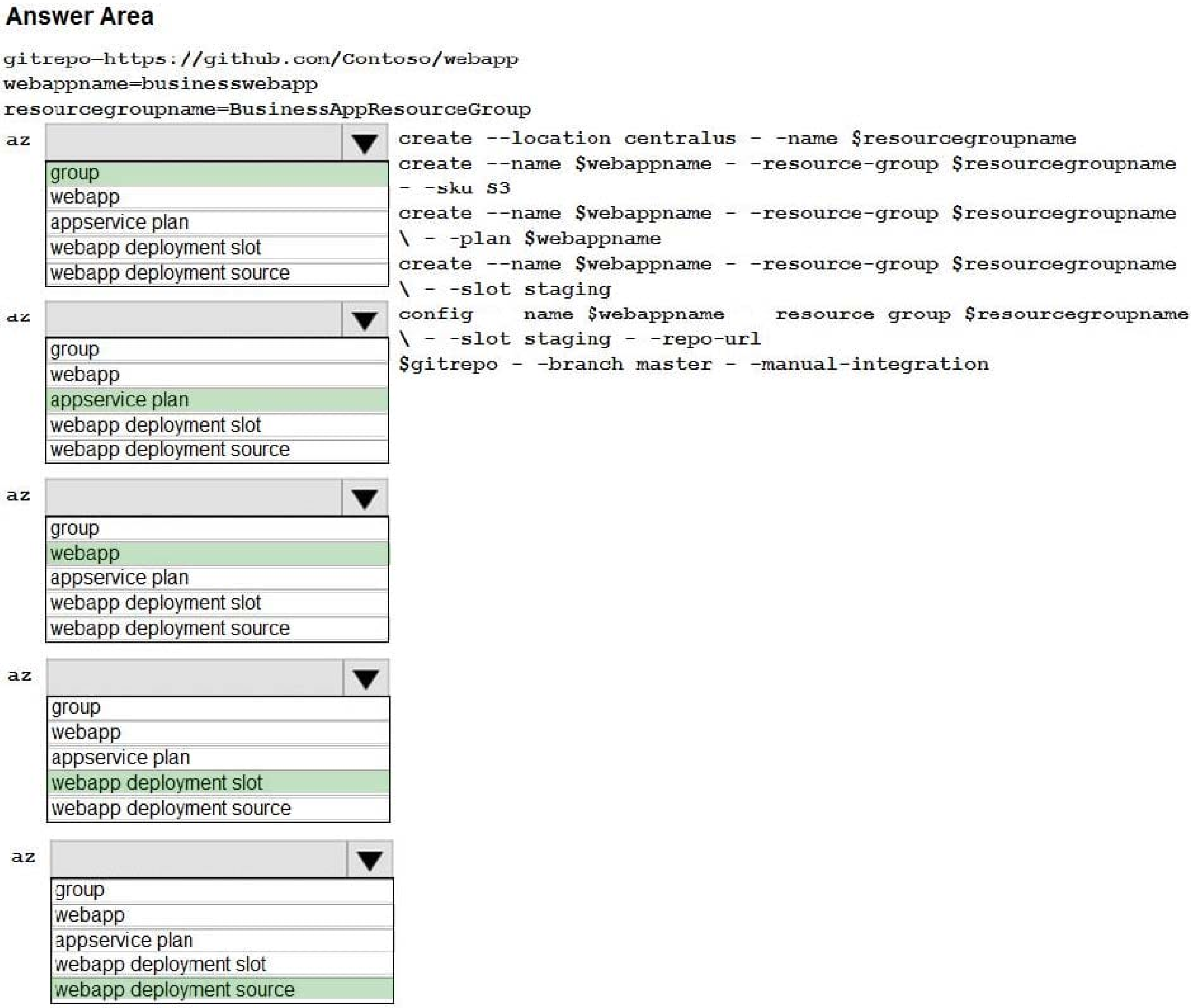


Answer Area

```
gitrepo—https://github.com/Contoso/webapp
webappname=businesswebapp
resourcegroupname=BusinessAppResourceGroup
```

```
create --location centralus - -name $resourcegroupname
create --name $webappname - -resource-group $resourcegroupname
- -sku S3
create --name $webappname - -resource-group $resourcegroupname
\ - -plan $webappname
create --name $webappname - -resource-group $resourcegroupname
\ - -slot staging
config    name $webappname    resource group $resourcegroupname
\ - -slot staging - -repo-url
$gitrepo - -branch master - -manual-integration
```

Correct Answer:



Answer Area

```
gitrepo-https://github.com/Contoso/webapp
webappname=businesswebapp
resourcegroupname=BusinessAppResourceGroup
az [dropdown 1]     create --location centralus - -name $resourcegroupname
az [dropdown 2]     create --name $webappname - -resource-group $resourcegroupname
                    - -sku S3
az [dropdown 3]     create --name $webappname - -resource-group $resourcegroupname
                    \ - -plan $webappname
az [dropdown 4]     create --name $webappname - -resource-group $resourcegroupname
                    \ - -slot staging
az [dropdown 5]     config    name $webappname    resource group $resourcegroupname
                    \ - -slot staging - -repo-url
                    $gitrepo - -branch master - -manual-integration
```

Dropdown 1: [✓ group] / webapp / appservice plan / webapp deployment slot / webapp deployment source

Dropdown 2: group / webapp / [✓ appservice plan] / webapp deployment slot / webapp deployment source

Dropdown 3: group / [✓ webapp] / appservice plan / webapp deployment slot / webapp deployment source

Dropdown 4: group / webapp / appservice plan / [✓ webapp deployment slot] / webapp deployment source

Dropdown 5: group / webapp / appservice plan / webapp deployment slot / [✓ webapp deployment source]

Box 1: group

# Create a resource group.

az group create --location westeurope --name myResourceGroup

Box 2: appservice plan

# Create an App Service plan in STANDARD tier (minimum required by deployment slots).

az appservice plan create --name $webappname --resource-group myResourceGroup --sku S1

Box 3: webapp

# Create a web app.

az webapp create --name $webappname --resource-group myResourceGroup \

--plan $webappname

Box 4: webapp deployment slot

#Create a deployment slot with the name "staging".

```
az webapp deployment slot create --name $webappname --resource-group myResourceGroup \
--slot staging
```

Box 5: webapp deployment source

# Deploy sample code to "staging" slot from GitHub.

```
az webapp deployment source config --name $webappname --resource-group myResourceGroup \
--slot staging --repo-url $gitrepo --branch master --manual-integration
```

References:

https://docs.microsoft.com/en-us/azure/app-service/scripts/cli-deploy-staging-environment