Exercise 4.08 - Serverless application deployment

Step 1 - Create the structure

- Create the recommended set of files in the workspace/root folder
  - providers.tf, terraform.tfvars, variables.tf can be copied from ex07
  - versions.tf can be copied from ex06
  - main.tf and outputs.tf should be create empty

Step 2 - Group 1 - Resource group

- Declare an Azure resource group using the existing variables

Step 3 - Group 2 - Storage container for built artefacts and table storage

- Declare a "random_string" resource from "random" provider. This resource will be used to name the storage account resource. Some resources in Azure must be unique globally.
- Declare a storage container resource from "azurerm" provider. The storage container will be used to store the application source code and the documents in a NoSQL database. The provisioning of a storage container is a two-step process:
  - Create a storage account
  - Create the container itself

```
resource "azurerm_storage_account" "________" {
  name                     = _________
  resource_group_name      = _________
  location                 = _________
  account_tier             = "Standard"
  account_replication_type = "LRS"
}
```

```
resource "azurerm_storage_container" "________" {
  name                  = _________
  storage_account_name  = _________
  container_access_type = "private"
}
```

Step 4 - Group 3 - Put built artefacts as a blob in storage container

- For this exercise, we'll run from a zip package referenced by a public URL. This can be done in one single Terraform command. For simplicity, the source code has been packaged into a Terraform module. This module is currently located on the Terraform Registry and will be downloaded from there.

```
module "tweetish" {
  source  = "MaxDehaut/azure/tweetish"
  version = "1.0.0"
}
```

- The output value of the module will be placed into the Blob storage (source). It might be worth to check the source of the module.

```
resource "azurerm_storage_blob" "________" {
  name                   = "server.zip"
  storage_account_name   = ________
  storage_container_name = ________
  type                   = "Block"
  source                 = ________
}
```

Step 5 - Group 4 - Configure and launch Azure Functions App

- Azure Function needs to be able to download the application source code from the private storage container, which requires a URL presigned by a Shared Access Signature Token (SAS Token). To do so, we will use a data source.

```
data "azurerm_storage_account_sas" "________" {
  connection_string =
azurerm_storage_account.________.primary_connection_string

  resource_types {
    service   = false
    container = false
    object    = true
  }

  services {
    blob  = true
    queue = false
    table = false
    file  = false
  }

  start  = "2021-09-01T00:00:00Z"
  expiry = "2021-12-31T00:00:00Z"

  permissions {
    read    = true
    write   = false
    delete  = false
    list    = false
```

```
    add     = false
    create  = false
    update  = false
    process = false
  }
}
```

- The next step is to generate the presigned URL. For simplicity, let's declare a local value. The structure of the url should be as follow:
  - https://
  - «storage account name».blob.core.windows.net/
  - «storage container name»/
  - «storage blob name»«storage account sas.sas»

```
locals {
  package_url = ""
}
```

- The following steps are to create the application_insights and the function_app from azurerm provider. Application Insights is used here for instrumentation and logging.

```
resource "azurerm_app_service_plan" "________" {
  name                = ________
  location            = ________
  resource_group_name = ________
  kind                = "functionapp"

  sku {
    tier = "Dynamic"
    size = "Y1"
  }
}
```

```
resource "azurerm_application_insights" "________" {
  name                = ________
  location            = ________
  resource_group_name = ________
  application_type    = "web"
}
```

```
resource "azurerm_function_app" "________" {
  name                = ________
  location            = ________
  resource_group_name = ________
```

```
  app_service_plan_id = _________
  https_only          = true

  storage_account_name       = _________
  storage_account_access_key = _________
  version                    = "~2"

  app_settings = {
    FUNCTIONS_WORKER_RUNTIME       = "node"
    WEBSITE_RUN_FROM_PACKAGE       = _________
    WEBSITE_NODE_DEFAULT_VERSION   = "10.14.1"
    APPINSIGHTS_INSTRUMENTATIONKEY = _________
    TABLES_CONNECTION_STRING       = _________
    AzureWebJobsDisableHomepage    = true
  }
}
```

Step 6 - Version lock

- As we are accessing a zip file, the "archive" provider will be required.

Step 7 - Output the website URL

- The output value should be as follow:
  - https://
  - «function app name».azurewebsites.net/

Step 8 - Deployment

- Execute the typical deployment process