**CI/CD Pipeline Documentation for fwf-infrastructure**

This document outlines the GitHub Actions workflow for creating and configuring the Azure infrastructure required for the Flight Booking Aggregator (fwf-infrastructure). The workflow automates the provisioning of Azure resources using ARM templates and SQL configuration scripts.

**Purpose**

This pipeline automates the creation of Azure infrastructure components, ensuring consistency and reliability. The main components created include:

* **Resource Groups**
* **SQL Databases**
* **Azure Key Vault**
* **App Service Plans and App Services**
* **Azure Data Factory**

**Pipeline Trigger**

* **Manual Trigger (workflow\_dispatch)**  
  This workflow must be initiated manually from the GitHub Actions tab.

**Environment Variables**

The pipeline uses the following environment variables:

| **Variable Name** | **Description** |
| --- | --- |
| AZURE\_SUBSCRIPTION\_ID | Azure Subscription ID (stored in Secrets). |
| AZURE\_RESOURCEGROUP\_NAME | Resource group name (fwf-rg). |
| SQLSERVER\_NAME | SQL Server name (fwf-db-srv). |
| DATABASE\_NAME | SQL Database name (fwfdb). |
| SQLADMIN\_LOGIN | SQL Admin login (stored in Secrets). |
| SQLADMIN\_PASS | SQL Admin password (stored in Secrets). |
| FWF\_APP\_SERVICE\_PLAN\_NAME | Name of the App Service Plan. |
| FWF\_FLIGHT\_SERVICE\_NAME | Name of the Flight Search Service App. |

**Pipeline Workflow**

**1. Code Checkout**

* **Step:** actions/checkout@v2
* **Purpose:** Clones the repository to allow access to ARM templates, parameter files, and SQL scripts.

**2. Setup .NET SDK**

* **Step:** actions/setup-dotnet@v1
* **Purpose:** Installs the required .NET SDK to execute any .NET-based tasks in the workflow.

**3. Azure Login**

* **Step:** azure/login@v1
* **Purpose:** Authenticates the workflow with Azure using a Service Principal stored in GitHub Secrets (FWF\_ENV\_SPN).

**4. Deploy Azure Resource Group**

* **Template:** fwf-resource-group.json
* **Parameters:** fwf-resource-group.parameters.json
* **Scope:** Subscription-level deployment.
* **Purpose:** Creates a resource group to host related Azure resources.

**5. Deploy SQL Server and Database**

* **Template:** fwf-database.json
* **Parameters:** fwf-database.parameters.json
* **Purpose:** Provisions an Azure SQL Server and Database using the credentials and names provided.

**6. Create SQL Connection String**

* **Command:**

CONN\_STR=$(az sql db show-connection-string --client ado.net --server ${{ env.SQLSERVER\_NAME }} --name ${{ env.DATABASE\_NAME }} -o tsv)

* **Purpose:** Dynamically generates and sets the SQL connection string for use in subsequent steps.

**7. Execute SQL Scripts**

* **Command:**

for script in $(ls repo/FWF/deployment/DBScripts/\*.sql | sort); do

sqlcmd -S ${{ env.SQLSERVER\_NAME }}.database.windows.net -U ${{ secrets.SQLADMIN\_LOGIN }} -P ${{ secrets.SQLADMIN\_PASS }} -d ${{ env.DATABASE\_NAME }} -i $script

done

* **Purpose:** Executes all SQL scripts located in the DBScripts folder to initialize and configure the database.

**8. Deploy Azure Key Vault**

* **Template:** fwf-key-vault.json
* **Parameters:** Includes the SQL database connection string as a secret.
* **Purpose:** Creates an Azure Key Vault to securely store application secrets.

**9. Deploy App Service Resources**

* **Template:** fwf-service-resources.json
* **Parameters:** Configures the App Service Plan and App Service.
* **Purpose:** Provisions the App Service Plan and the Flight Search Service App.

**10. Deploy Azure Data Factory**

* **Template:** fwf-data-factory.json
* **Purpose:** Provisions an Azure Data Factory to handle ETL processes.

**11. Deploy Data Factory Resources**

* **Template:** fwf-data-factory-resources.json
* **Parameters:** Configures linked services and pipelines for the Data Factory.
* **Purpose:** Sets up Azure Data Factory resources for integration with the database and other components.

**Conclusion**

This pipeline automates the provisioning of Azure resources, ensuring:

* Consistency in resource deployment.
* Secure handling of secrets using Azure Key Vault.
* Proper configuration of the database and other services.

For customization, update the ARM templates, parameter files, or SQL scripts to fit the specific requirements of your project.