WSO2 Identity Server

USER GUIDE OF PRODUCT MIGRATION TESTING AUTOMATION TOOL

Table of Contents

1.Overview

This guide will walk you through how to use the wso2 Product Migration Testing Automation Tool.

Notion documentation of the project:
https://www.notion.so/User-Guide-Of-Product-Migration-Testing-Automation-Feature-17676cd5205a49a898d6527a03347199?pvs=4

External reading links of the project:
https://medium.com/@Jayana_Gunaweera/unleashing-the-power-of-automation-how-wso2-identity-server-migration-tool-can-revolutionize-67abe21d6d40

ProjectLink:https://github.com/wso2/product-is/actions/workflows/migration-automation.yml

Project Description:
wso2 is committed in providing a seamless transition for our users as they migrate wso2 Identity Server across different product versions. To facilitate this, we offer a separate migration client, specifically designed to manage the migration process. If users are planning to upgrade from an older version to the most recent release of the wso2 Identity Server, they can do so by executing the migration client and adhering to our detailed migration documentation.

With the introduction of every new product release, it is imperative for us to test these migrations meticulously. The testing process must cover various infrastructure combinations, such as different databases and operating systems, to ensure the compatibility and stability of the product across all supported environments. However, manual migration testing with all potential infrastructure combinations is time-consuming and resource intensive.

Solution: Automation

To reduce the time and resources required for exhaustive testing, and to enhance our efficiency, we have automated the migration client test execution. By automating the migration client test execution, we aim to streamline our processes and improve our overall productivity, while maintaining our commitment to delivering an identity server that meets the highest standards of quality, performance, and reliability.

Supported wso2 Identity server versions - wso2IS 5.9, wso2IS 5.10, wso2IS 5.11, wso2IS 6.0, wso2IS 6.1

Supported Databases – Mysql version 8, Mssql version 12, Postgresql version 42

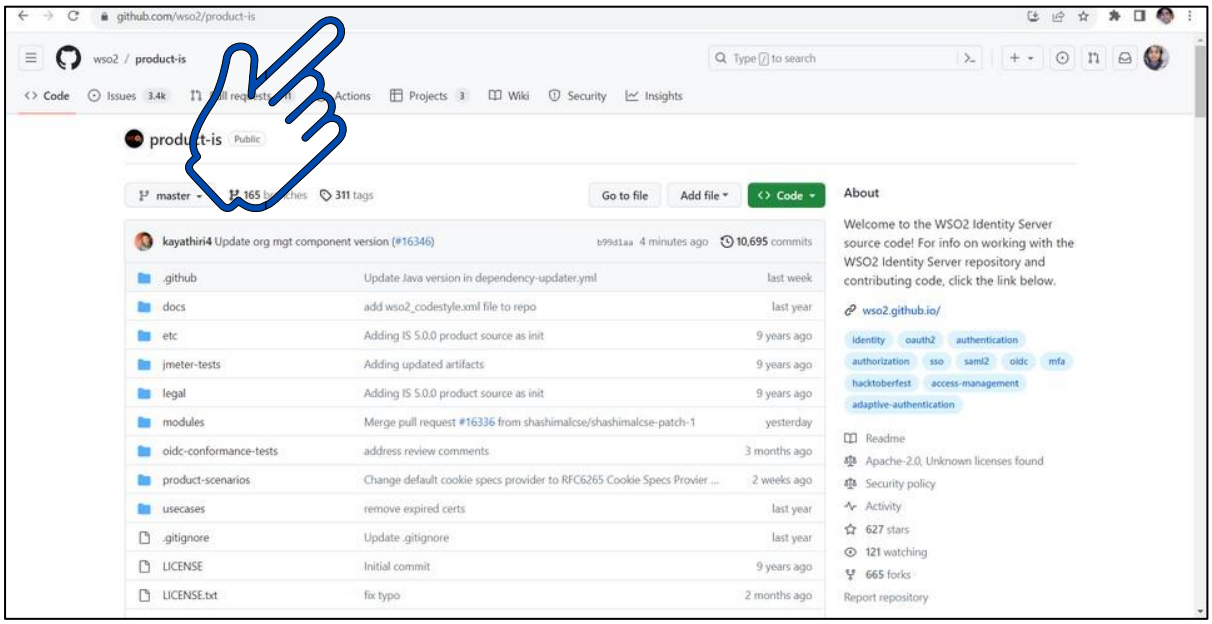
Supported Operating Systems – Ubuntu 22.04, MacOs 12 (inside GitHub-hosted runners)

2.Usage

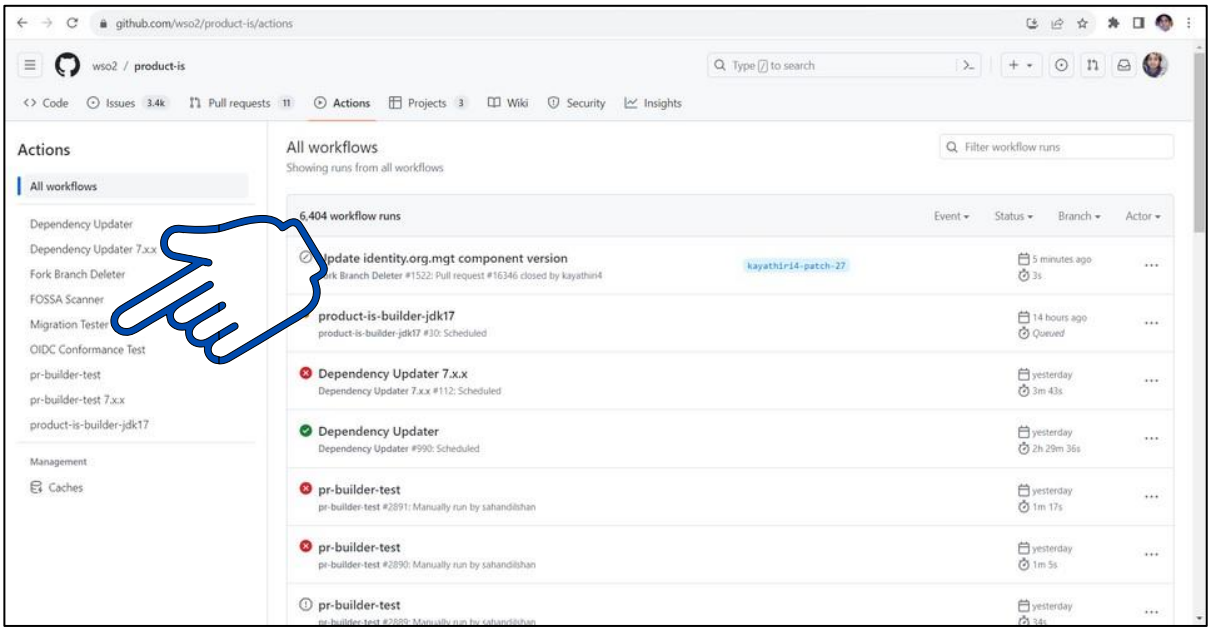
Running the Migration Testing Automation Tool-User can manually trigger workflow from the GitHub user interface.

1.Open wso2/product-is repository on GitHub.
https://github.com/wso2/product-is

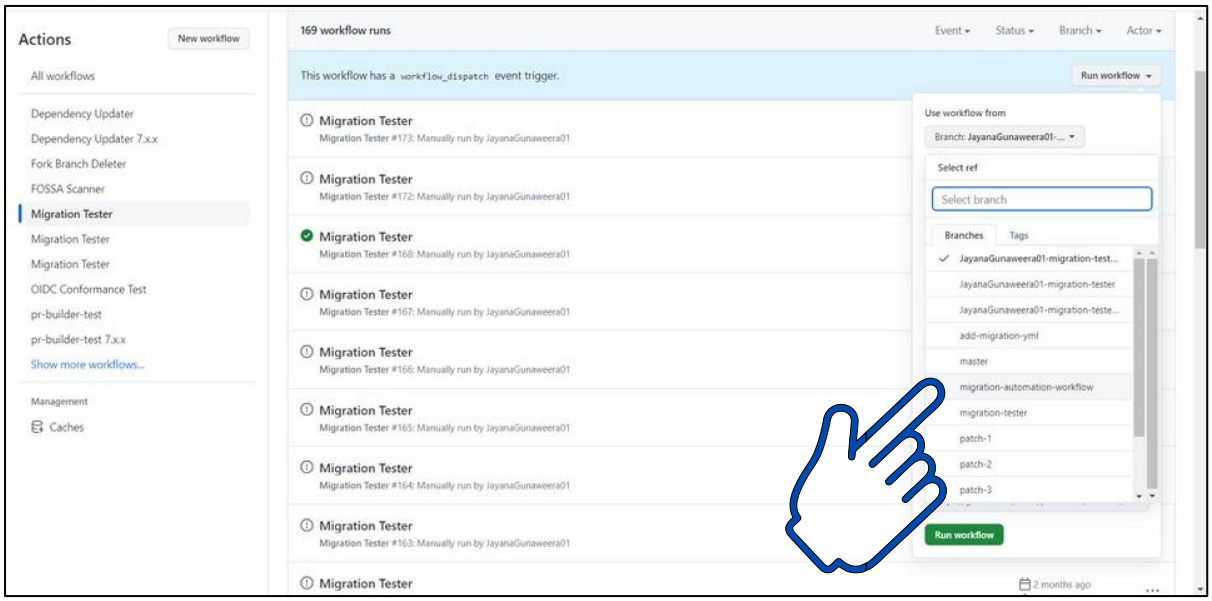
2.At the top of the repository, click on the Actions tab.



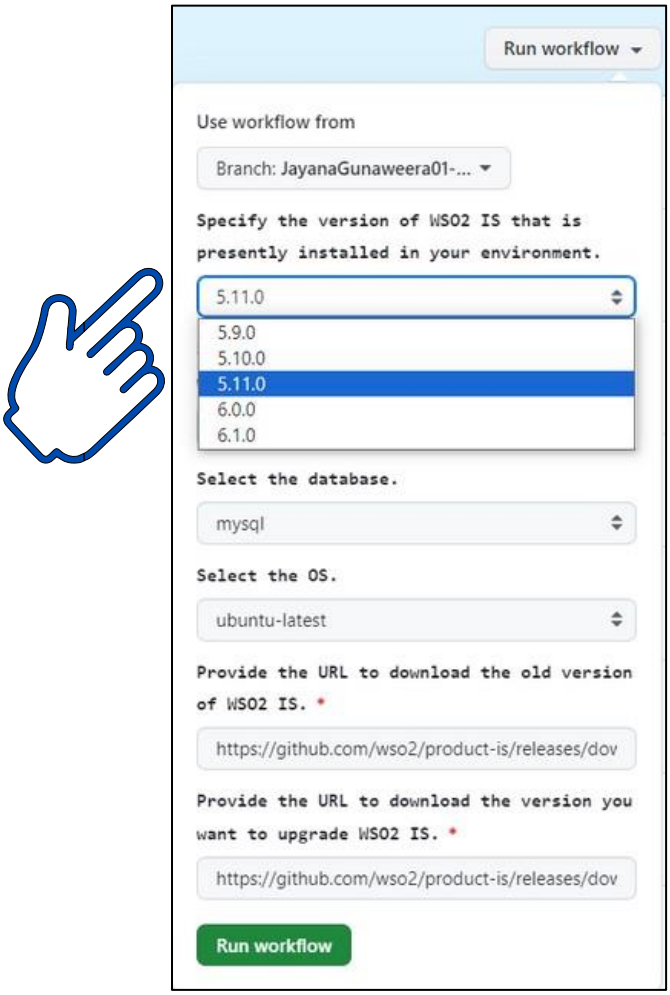
3.In the left sidebar, click the Migration Tester



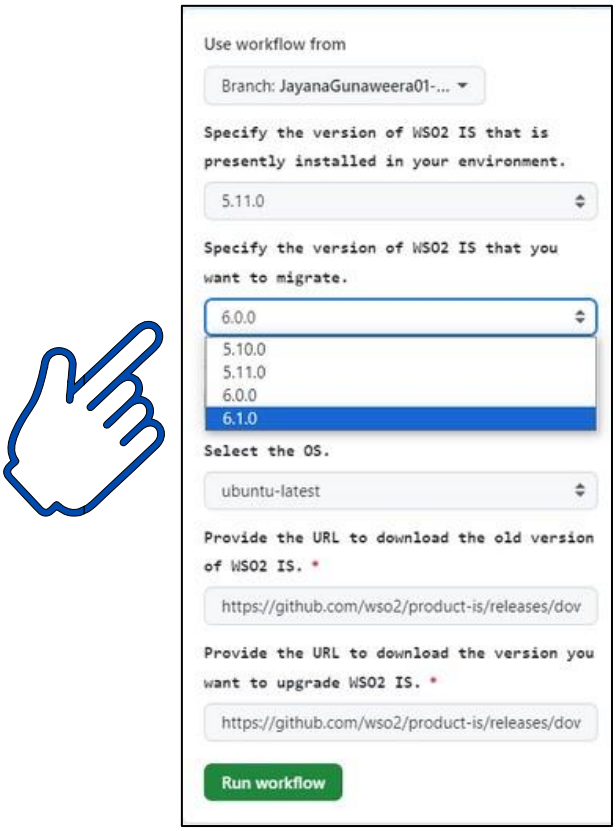
4.Select the branch which you want to trigger the workflow.



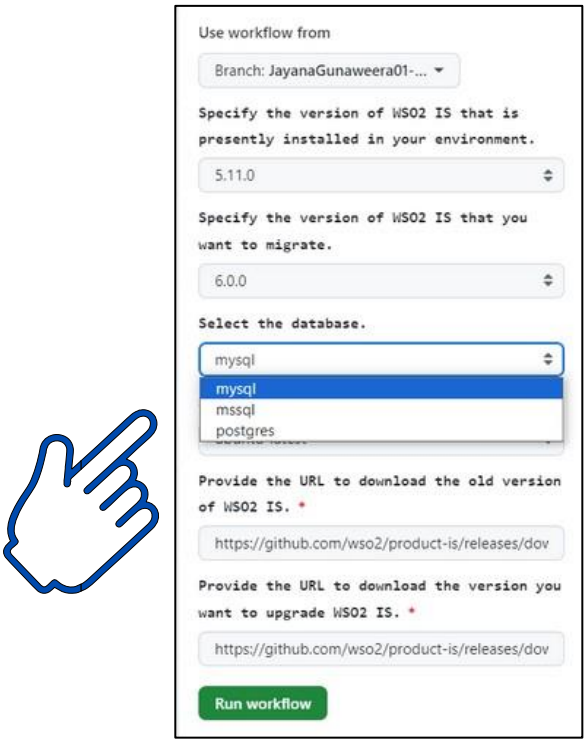
5.Select the version that you want to migrate from



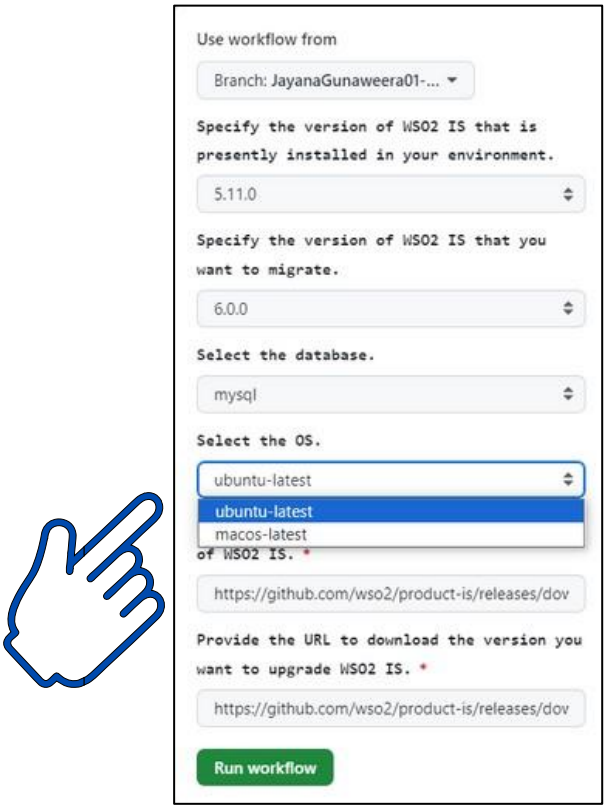
6.Select the version that you want to migrate.



7.Select the Database.



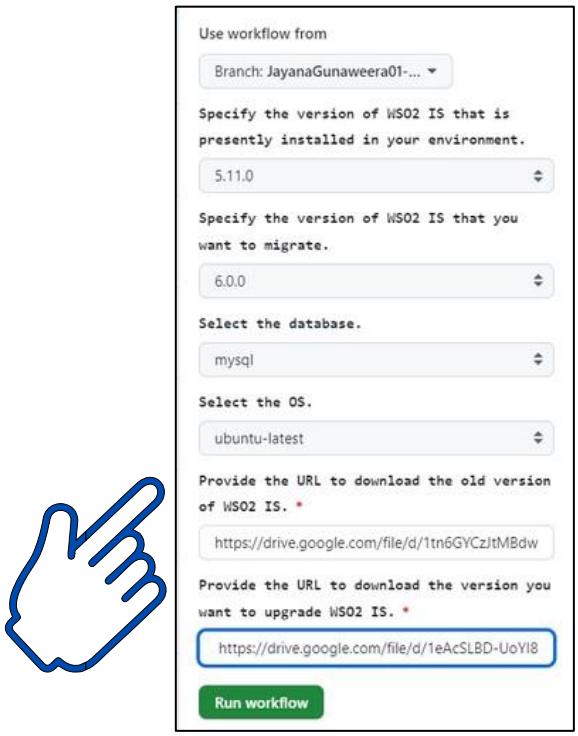
8.Select the Operating System.



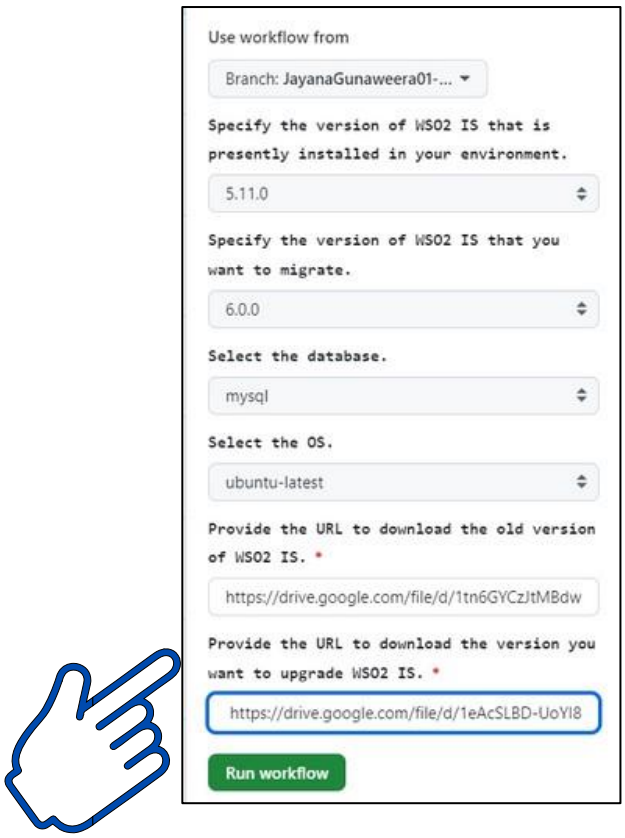
9.Provide the URL of the base pack stored inside google drive to download the old/existing version of wso2 IS.*

You can get URLs of base packs stored inside google drive from below permalink (Hardcoded base pack file URLs available here)

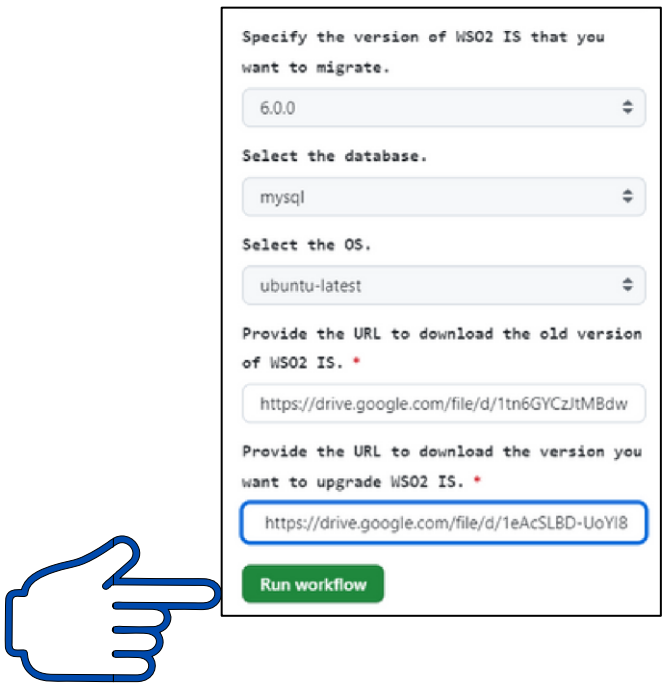
https://github.com/wso2/product-is/blob/c7109e57d92516d224548beae0bdef7e1d24ba33/.github/migration-tester/migration-automation/env.sh#L538C1-L543C97



10.Provide the URL of base pack stored inside google drive to download the migrating/updating version of wso2 IS.*

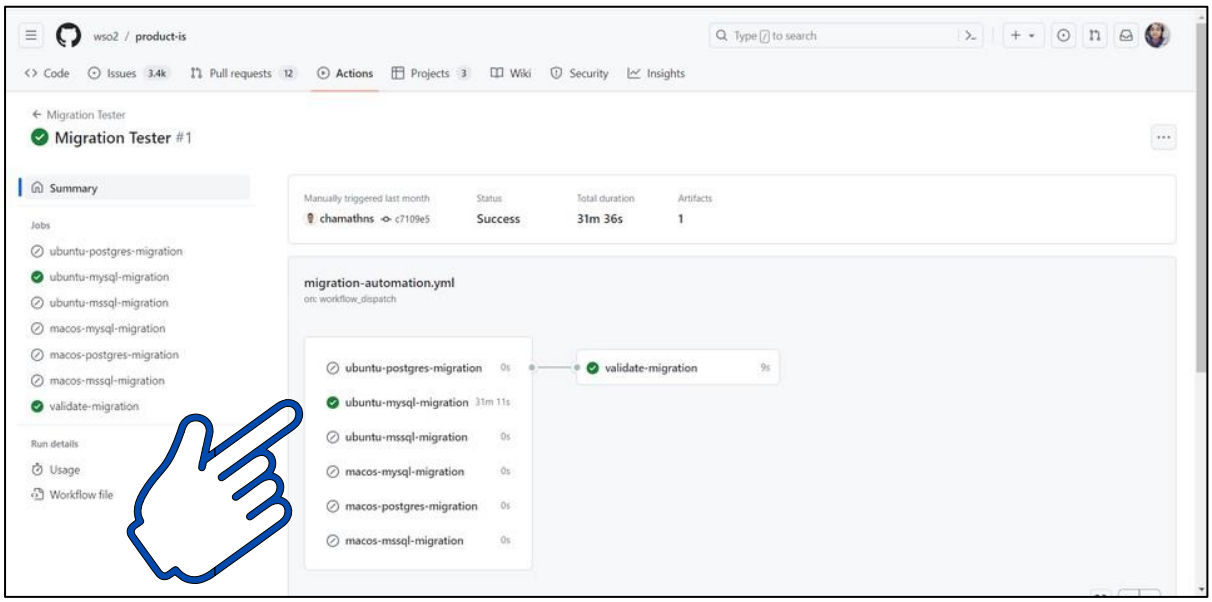


11.Click the Run workflow button to start the workflow.

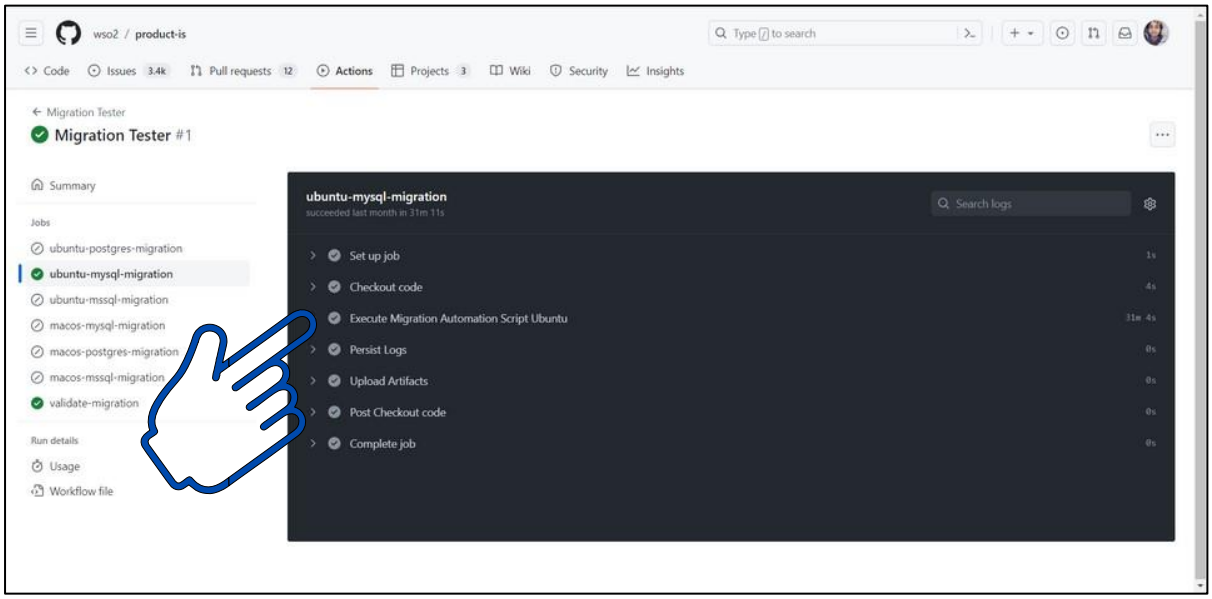


To observe the Real -Time Shell Script Execution of Migration Testing:

12.Please click the job marked with a green checkmark to observe the real-time execution of the migration process.



13.Click on Execute Migration Automation Script



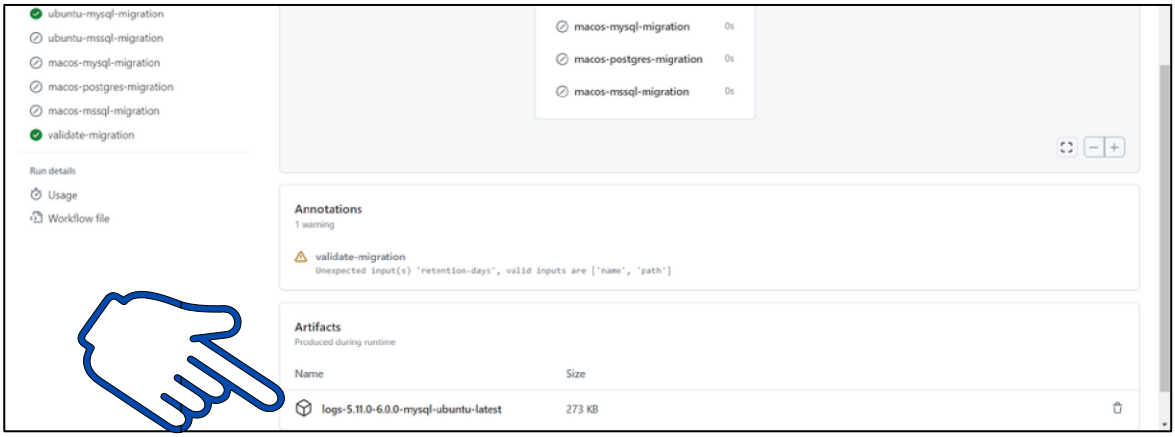
Analyzing Test Results:

You can analyze test results in 2 ways.

1. Checking Artifacts built while shell script execution.
2. Checking validate migration job.

1. Checking Artifacts built while shell script execution.

After workflow execution, github action generates a log report that provides all the logs prints in the cmd generate while the workflow execution takes place. They can be found in artifacts. It is named uniquely based on the current version, migrating version, database, and operating system.



2.Checking validate migration job.

Here the logs from the downloaded artifacts folder are accessed and processed line by line.

➔Each line is checked for the presence of error messages by using a pattern matching condition.

➔If an error is detected (indicated by the presence of "ERROR," "error," or "Error"), the line is printed in red color to highlight the error.

➔Otherwise, the line is printed as it is.

The "validate-migration" job ensures that the migration process is validated after the completion of the migration jobs for different databases and operating systems. It retrieves the logs from the previous migration jobs and performs postmigration testing by analyzing the logs for any error messages.

Ex:

```
spawn ./wso2update_linux
wso2update: Error while initializing updates config file: Error while initializing product config: Unable to identify the product in
/home/runner/work/product-is/product-is/.github/migration-tester/utils/updates/product.txt
```

3.Extensions

1.Adapting Migration Scripts for Extended Testing Capabilities: GA, RC Packs, and Beyond

At present, this tool is designed to facilitate testing of base packs stored in a wso2 Google Drive.

Should there be a requirement to test RC/GA packs, or any other types, modifications can be made to the existing code. This can be accomplished by altering the 'migration-script-ubuntu.sh' or 'migration-script-mac.sh' files, as appropriate for your operating system.

```
# Download needed wso2IS zip # Generate access token
response=$(curl --location --request POST 'https://oauth2.googleapis.com/token' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode "client_id=$gcpClientId" \
--data-urlencode "client_secret=$gcpClientSecret" \
--data-urlencode "refresh_token=$gcpRefreshToken" \ --data-urlencode 'grant_type=refresh_token')
# Extract the access token from the response using jq access_token=$(echo "$response" | jq -r '.access_token')
# Initialize file_id variable file_id=""
# Check the value of currentVersion and assign the corresponding environment variable to file_id
case $currentVersion in
  5.9.0)     file_id="$FILE_ID_5_9"
    ;;
  5.10.0)     file_id="$FILE_ID_5_10"
    ;;
  5.11.0)     file_id="$FILE_ID_5_11"
    ;;
  6.0.0)     file_id="$FILE_ID_6_0"
    ;;
  6.1.0)     file_id="$FILE_ID_6_1"

    ;;   *)     echo "No action taken.Please assign a value in env.sh if you haven't assigned a value for file ID."
    ;;
esac
```

```
# Use the file_id variable in downloading the IS zip echo "file_id: $file_id"
# Specify the Google Drive file URL
file_url="https://www.googleapis.com/drive/v3/files/"$file_id"?alt=media"
# Download the file using the access token response=$(curl "$file_url" \
  --header "Authorization: Bearer $access_token" \
  --header "Accept: application/json" \
  --compressed -o wso2IS.zip) wait $!

# Check if the response contains any error message
 If
echo "$response" | grep -q '"error":'; then
  # If there is an error, print the failure message with the error description   error_description=$(echo "$response" | jq -r '.error_description')
echo -e "${RED}${BOLD}Failure in downloading Identity Server $error_description${NC}"
else
  # If there is no error, print the success message
  echo -e "${PURPLE}${BOLD}Success: IS Pack downloaded successfully. ${NC}"
fi
```

To facilitate testing of RC/GA packs, the existing code must be replaced with the following provided code.

```
# Download needed wso2IS zip
wget -qq --waitretry=5 --retry-
connrefused "$urlOld" wait $!
```

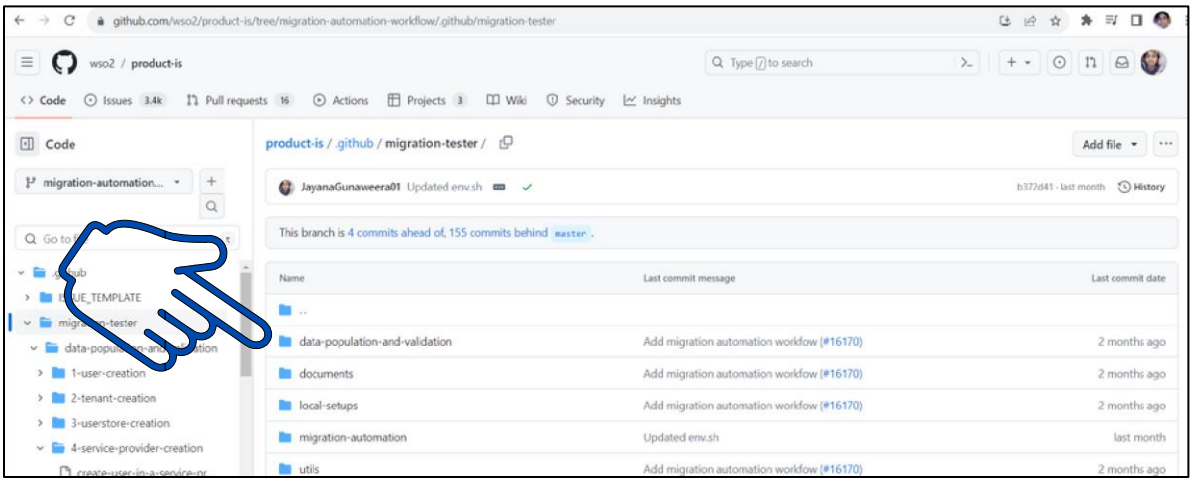
Subsequently, use the corresponding URL of the pack during testing via workflow dispatch. Ex:

https://github.com/wso2/product-is/releases/download/v5.10.0/wso2IS-5.10.0.zip

https://github.com/wso2/product-is/releases/download/v6.0.0-rc2/wso2IS-6.0.0rc2.zip

2.Incorporating Additional Features into the Designated Directory for Automatic Execution

If additional features, such as data population or verification, need to be included, kindly integrate them within the designated directory labeled 'data-population-and validation. During the execution of the main shell script, all scripts residing within this directory are executed automatically. Therefore, please ensure to incorporate your additional features exclusively within this specified directory.



4.Local Setups

Executing Automated Product Migration Testing in a Local Environment.

Prerequisites

1.Before proceeding, ensure that necessary utilities, specifically **wget** and **jq**, are installed.

2.Ensure **Java Open JDK 11** is set up according to the required specifications.

3.Install and configure **docker** and **MySQL 8.3**. If MySQL is already installed and running locally, please make sure to stop it.

4.Place the appropriate mysql JDBC driver, compatible with your specific version, in the **utils** folder.

5.Create a MySQL database to the wso2 Identity Server and change the database configurations in the **deployment.toml** file which is located in the local-setups/ubuntu-os/directory.

6.Update env.sh file according to your requirements. (Updating paths, metadata.)

Demystifying the Execution:

1. Clone the repository.
2.Open the terminal
3.Navigate to local-setups/ubuntu-os/
4.Make the migration-automation-script-linux.sh script executable (chmod +x migration-automation-script-linux.sh)
5.Run the script (sh migration-automation-script-linux.sh / sudo migrationautomation-script-linux.sh)

A log report of migration testing can be obtained for further analysis and review after a successful execution of the script.

log_file=$Home/Downloads/Automating-Product-Migration-Testing/localsetups/migration.log

5.Future Development

1.Expanding Database Compatibility

Currently, this tool is designed to support only MySQL, MSSQL, and PostgreSQL databases. However, future development plans include expanding its capabilities to accommodate all databases supported by wso2 Identity Server.

2.Extending OS Compatibility

As of now, the tool is compatible with the Ubuntu and MacOS operating systems. Our roadmap also involves extending support to additional operating systems, such as Windows OS, to enhance versatility and user convenience.

3.Enhancing Flexibility in Local Setup Scenarios

The local setup scripts contained in this repository are presently designed to facilitate migration solely from wso2 IS 5.11 to 6.00, and are limited to MySQL on the Ubuntu operating system. In the future, it is feasible to expand the scope of these scripts to support other environments for enhanced flexibility in local setup scenarios.

4.Enhanced Validation Checks

1.https://github.com/wso2/product-is/blob/migration-automationworkflow/.github/migration-tester/data-population-and-validation/4-serviceprovider-creation/validate-database-mac.sh

2.https://github.com/wso2/product-is/blob/migration-automationworkflow/.github/migration-tester/data-population-and-validation/4-serviceprovider-creation/validate-database-ubuntu.sh

Above scripts presently offers fundamental database validation, which involves generating an access token and a refresh token from a service provider registered within the updated identity server. However, in light of enhancing the robustness of the script, we anticipate incorporating expanded validation checks into our future development plan. These enhancements may involve augmenting error detection mechanisms, introducing performance checks, and implementing data integrity assessments, dependent upon the existing capabilities of the script. These steps will serve to significantly bolster the reliability and efficiency of our validation process, paving the way for a more comprehensive database validation approach.

6.Additional Readings

Directory Structure

```
├── product-is
│   ├── .github
│   │   └── workflows
│   │       └── migration-automation.yml
│   ├── data-population-and-validation
│   │   ├── 1-user-creation
│   │   │   ├── create-bulk-users.sh
│   │   │   └── create-user.sh
│   │   ├── 2-tenant-creation
│   │   │   ├── create-tenant-soapAPI.sh
│   │   │   └── create-tenant.sh
│   │   ├── 3-userstore-creation
│   │   │   ├── create-user-in-userstore.sh
│   │   │   ├── create-userstore-soapAPI.sh
│   │   │   └── create-userstore.sh
│   │   ├── 4-service-provider-creation
│   │   │   ├── create-user-in-a-service-provider.sh
│   │   │   ├── register-a-service-provider-get-access-token-mac.sh
│   │   │   └── register-a-service-provider-get-access-token-ubuntu.sh
│   │   │   ├── register-a-service-provider.sh
│   │   │   ├── validate-database-mac.sh
│   │   │   └── validate-database-ubuntu.sh
│   │   ├── 5-group-creation
│   │   │   ├── create-group.sh
│   │   │   └── create-groups-with-users.sh
│   │   └── automated-data-poputation-and-validation-script-mac.sh
│   │   └── automated-data-poputation-and-validation-script-ubuntu.sh
│   ├── documents
│   │   └── Automating Product Migration Testing.word
│   ├── local-setups
│   │   ├── mac-os
│   │   │   ├── migration-automation-script-macos.sh
│   │   ├── ubuntu-os
│   │   │   ├──automated-data-population-and-validation-script-ubuntu-local-setup.sh
│   │   │   ├──automating-product-migration-testing.sh
│   │   │   └── backup_db.sql
│   │   │   ├──change-deployment-toml.sh
│   │   │   ├── change-migration-configyaml.sh
│   │   │   └── copy-jar-file-mysql.sh
│   │   │   ├──create-new-database.sh
│   │   │   ├── deployment.toml
│   │   │   └── enter-login-credentials.sh
│   │   │   ├── env.sh
│   │   │   ├── humanoid.jpg
│   │   │   └── migration-automation-script-linux.sh
│   │   │   ├── migration-terminal.sh
│   │   │   ├── migration.log
│   │   │   └── server-start-newIS.sh
│   │   │   ├── server-start.sh
│   │   │   ├── validate-database-ubuntu-local-setup.sh
│   ├── migration-automation
│   │   ├── deployment-tomls
│   │   │   ├── IS-5.9
│   │   │   │   ├── deployment-mssql.toml
│   │   │   │   ├── deployment-mysql.toml
│   │   │   │   └── deployment-postgre.toml
│   │   │   │   ├── deployment-mssql-migration.toml
│   │   │   │   ├── deployment-mysql-migration.toml
│   │   │   │   └── deployment-postgre-migration.toml
│   │   │   ├── IS-5.10
│   │   │   │   ├── deployment-mssql.toml
│   │   │   │   ├── deployment-mysql.toml
│   │   │   │   └── deployment-postgre.toml
│   │   │   │   ├── deployment-mssql-migration.toml
│   │   │   │   ├── deployment-mysql-migration.toml
│   │   │   │   └── deployment-postgre-migration.toml
│   │   │   ├── IS-5.11
│   │   │   │   ├── deployment-mssql.toml
│   │   │   │   ├── deployment-mysql.toml
│   │   │   │   └── deployment-postgre.toml
│   │   │   │   ├── deployment-mssql-migration.toml
│   │   │   │   ├── deployment-mysql-migration.toml
│   │   │   │   └── deployment-postgre-migration.toml
│   │   │   ├── IS-6.0
│   │   │   │   ├── deployment-mssql.toml
│   │   │   │   ├── deployment-mysql.toml
│   │   │   │   └── deployment-postgre.toml
│   │   │   │   ├── deployment-mssql-migration.toml
│   │   │   │   ├── deployment-mysql-migration.toml
│   │   │   │   └── deployment-postgre-migration.toml
│   │   │   ├── IS-6.1
│   │   │   │   ├── deployment-mssql.toml
│   │   │   │   ├── deployment-mysql.toml
│   │   │   │   └── deployment-postgre.toml
│   │   │   │   ├── deployment-mssql-migration.toml
│   │   │   │   ├── deployment-mysql-migration.toml
│   │   │   │   └── deployment-postgre-migration.toml
│   │   │   └── IS-6.2
│   │   │       ├── deployment-mssql.toml
│   │   │       ├── deployment-mysql.toml
│   │   │       └── deployment-postgre.toml
│   │   │       ├── deployment-mssql-migration.toml
│   │   │       ├── deployment-mysql-migration.toml
│   │   │       └── deployment-postgre-migration.toml
```

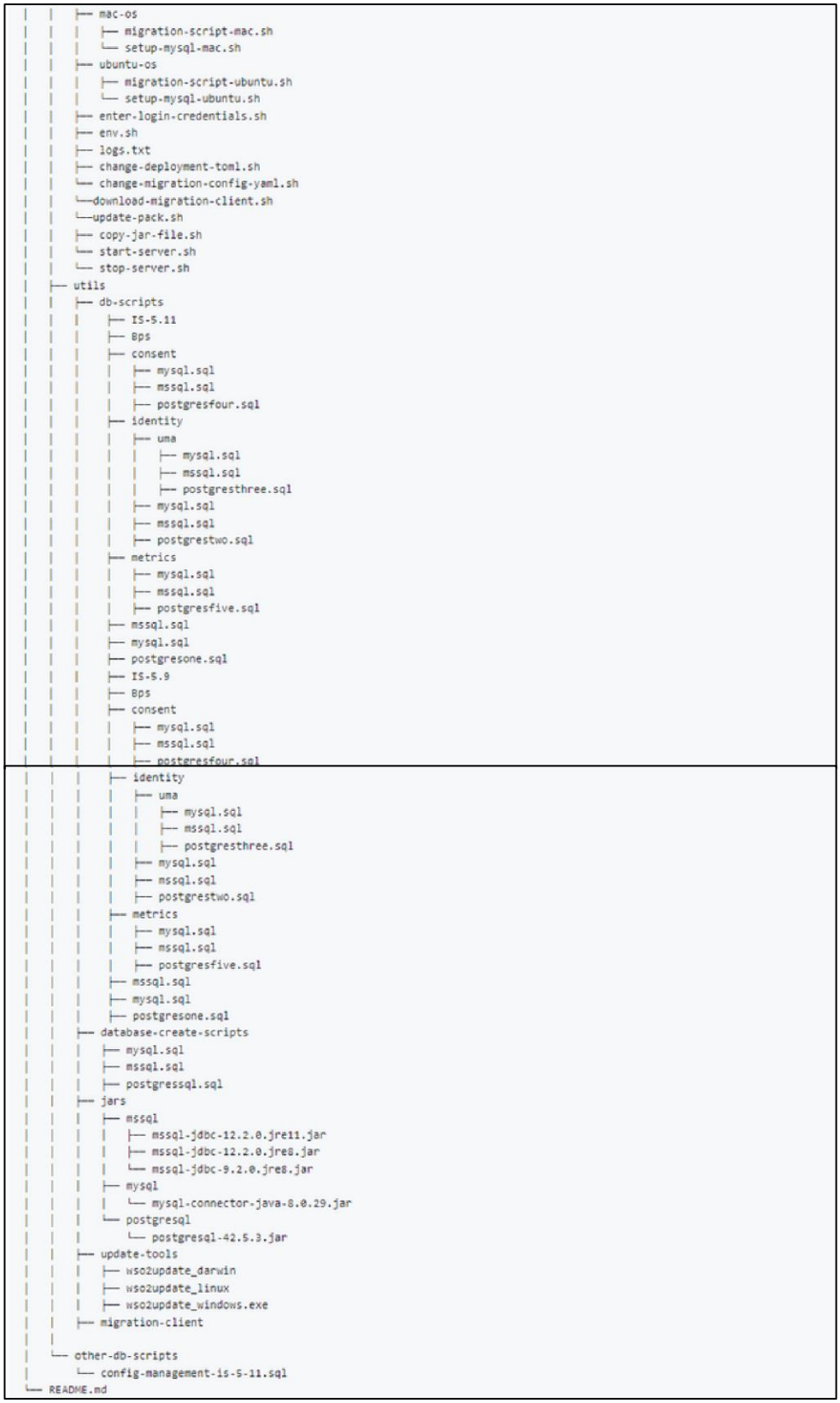


Image Link : https://github.com/wso2/product-is/blob/migration-automationworkflow/.github/migration-tester/README.md

Directory Structure – Explanation

1.github/workflows:

Contains the workflow file migration-automation.yml, which defines the main migration workflow for the Migration Automation Testing.

2.data-population-and-validation:

Contains subdirectories for different operating systems: mac-os, ubuntu-os
Each OS directory includes scripts for data population and validation, such as user creation, tenant creation, user store creation, service provider creation, and group creation.

Additionally, the directory encompasses common scripts named **automateddata-population-and-validation-script-mac.sh** and **automated-datapopulation-and-validation-script-ubuntu.sh**, specifically designed to automate the execution of data population shell scripts.

3.documents:

Contains the document file- User Guide of Product Migration Testing Automation tool.

4.local-setups:

Contains subdirectories for different operating system: ubuntu-os
It includes setup scripts specific to that operating system, such as changing deployment toml files, migration automation files, copying jar files, migration scripts, and MySQL setup scripts inside a local environment.

5.migration-automation:

Contains subdirectories for different operating systems: mac-os, ubuntu-os
Each OS directory includes scripts specific to that operating system for migration automation, such as changing deployment toml files, changing migration configuration YAML files, copying jar files, migration scripts, and MySQL setup scripts.

The deployment-tomls sub-directory includes subdirectories for different versions of the migration target (e.g., IS-5.10, IS-5.11) and respective deployment toml files for MSSQL, MySQL, and Postgres databases.

The deployment-tomls sub-directory is a key component within our directory structure. It houses multiple subdirectories, each specifically tailored to a different version of the migration target, namely IS-5.9, IS5.10, IS-5.11, IS-6.0, IS-6.1, and IS-6.2.
Contained within these subdirectories are dummy toml files, each specifically configured to be compatible with a certain type of database. For example,

within the IS 5.10 subdirectory, the deployment-mssql.toml file includes configurations for an MSSQL database specific to the IS 5.10 version. This file is designed to replace the default deployment toml in the current (or old) Identity Server during the data population process.

In addition, there is also a deployment-mssql-migration.toml file that carries configurations for MSSQL databases, which are compatible with the IS 5.10 version. This file is destined to replace the default deployment toml in the new (or updating) Identity Server before the migration process.

If you wish to make any deployment.toml config changes, these directories and files are where such modifications should be made.

Under the mac-os and ubuntu-os subdirectories, you'll find main shell scripts.migration-script-mac.sh,migration-script-ubuntu.shrespectively. These are scripts for executing migrations specific to each OS. Additionally, there are setup-mysql-mac.sh and setup-mysql-ubuntu.sh scripts for setting up MySQL databases in macOS and Ubuntu environments, respectively.

In the parent directory, you'll find several utility shell scripts:

- enter-login-credentials.sh: A script for inputting login
- credentials. env.sh: Contains environment-specific
- configurations. logs.txt: A log file generated during the
- execution of scripts.

change-deployment-toml.sh and change-migration-config-yaml.sh: The scripts change-deployment-toml.sh and change-migration-config-yaml.sh serve crucial roles within the project. They facilitate the modification of the deployment toml and migration configuration YAML files, respectively. **In the event that adjustments to the migration config.yaml are necessary, please note that such changes should be implemented within this changemigration-config-yaml.sh** download-migration-client.sh: A script to download the migration client from migration resources folder.

6.update-pack.sh:

A script for updating the product pack. copy-jar-file.sh: This script is for copying the necessary JAR files to lib folder in the IS.

7.start-server.sh and stop-server.sh:

Scripts to start and stop the server, respectively.start-server.sh will start the downloaded identity server in a background terminal inside github actions.

8.utils:

Contains utility scripts and helper tools.

9.db-scripts:

Contains subdirectories for different versions of the migration target (e.g., IS-5.11, IS-5.9) and respective subdirectories for database create scripts. Additionally, the directory includes deployment toml files for MSSQL databases.

Contains subdirectories for different database types: MsSQL, MySQL, and PostgreSQL.

- The MSSQL directory includes mssql-jdbc-12.2.0.jre11.jar file for MSSQL database connectivity.
- The MySQL directory includes the mysql-connector-java-8.0.29.jar file for MySQL database connectivity.
- The PostgreSQL directory includes the postgresql-42.5.3.jar file for PostgreSQL database connectivity.

10.migration-client:

This is an empty folder to download wso2IS-migration-1.0.225.zip file, which represents a migration client for performing specific migration tasks.

11.update-tools:

Comprises of U2 update tools for updating u2 base packs compatible to different OS types.

12.README.md:

Represents the readme file (README.md) for the repository, which provides information about the project, its purpose, and instructions for usage or contribution.

Feel free to explore each directory to find more details about the specific components and scripts.

Explanation of workflow (migration-automation.yml) and migration shell scripts (migration-script-ubuntu.sh and migration-scriptmac.sh)

- The workflow automation is implemented using GitHub Actions, which provides a flexible and customizable framework for continuous integration and deployment.

WORKFLOW OVERVIEW:

The "migration tester" workflow allows users to specify the necessary parameters for testing the migration of wso2 IS from one version to another. The workflow can be manually triggered using the GitHub Actions workflow_dispatch event. It collects inputs such as the current version of wso2 IS, the target migrating version, the database type, the operating system, and the download URLs for the old and new versions.

Key Steps in the Workflow:

a. **Trigger**:
The workflow can be triggered manually using the GitHub Actions workflow_dispatch event. This gives users control over when to initiate the migration testing process.

b. **Inputs**:
The workflow defines several input parameters that users can provide when triggering the workflow. These parameters include:

currentVersion: Users specify the version of wso2 IS presently installed in their environment. The options include versions 5.9.0, 5.10.0, 5.11.0, 6.0.0 and 6.1.0

migratingVersion: Users specify the version of wso2 IS they want to migrate to. The available options include versions 5.10.0, 5.11.0, 6.0.0 and 6.1.0
database: Users select the database type to be used during migration, with options including "mysql", "mssql", and "postgres".

os: Users select the operating system relevant to their environment. Options include "ubuntu-latest" and "macos-latest"

urlOld: Users provide the URL to download the old version of wso2 IS that corresponds to their currentVersion selection.

urlNew: Users provide the URL to download the version they want to upgrade wso2 IS to, based on the migratingVersion specified.

This workflow file comprises 6 Jobs. These jobs automate the testing process for product migration in the context of wso2 IS. These jobs are triggered based on specific conditions and are designed to handle different database migrations on different Operating Systems.

1.**ubuntu-postgres-migration**: Migrates PostgreSQL databases on Ubuntu.
2.**ubuntu-mssql-migration**: Migrates Microsoft SQL Server (MSSQL) databases on Ubuntu.
3.**macos-mssql-migration**: MSSQL databases on macOS.
4.**macos-postgres-migration**: Migrates PostgreSQL databases on macOS.
5.**ubuntu-mysql-migration**: Migrates MySQL databases on Ubuntu.
6.**macos-mysql-migration**: Migrates MySQL databases on macOS.
7.**validate-migration**: Validates migration processes and performs post-migration testing.

These jobs cover a range of migration tasks, including setup, SQL script execution, and running migration automation scripts. Logs are persisted and uploaded as artifacts for analysis. The "validate-migration" job checks the logs for errors and provides a log report of the migration process.

Migration Testing Automation Shell Script Overview:

Two shell scripts called migration-script-ubuntu.sh and migration-scriptmac.sh have been written as main shell scripts specific to OS in order to automate product migration testing from one version of wso2IS to another in a Linux (Ubuntu) or a MacOS environment. These scripts focus on migrating databases such as MySQL, MSSQL, or PostgreSQL using GitHub Actions. Both scripts follow the same procedure but all the steps in doing a manual migration and migration testing are automated here.

- The script begins by defining color variables for enhanced output formatting.
- The system is updated using the apt-get command to ensure the latest packages are available.
- The script navigates to the relevant directory for the migration automation process.

Input Variables

- The script retrieves input values from the workflow dispatch, such as URLs, versions, databases, operating systems, and credentials.
- These inputs are used throughout the script to customize the migration process.

Pre-migration Steps

- The script performs various operations to set up the migration environment before starting the migration process.
- The env.sh file is modified to replace placeholders with actual version numbers.
- Java is installed using the apt-get command.

The IS_HOME_OLD directory is created to store the existing wso2IS installation.

- The required wso2 zip file is downloaded based on the provided version using Google Drive API and an access token.
- The downloaded zip file is extracted into the IS_HOME_OLD directory.
- Update IS Packs-The script updates the IS packs using the update-pack.sh script, which takes email and password parameters.
- Deployment Configuration-The script modifies the deployment.toml file based on the provided database, operating system, and versions.
- If the selected database is MySQL, additional setup is performed using the setup-mysql-ubuntu.sh script.
- JAR Files and Data Population-The script copies the necessary JAR files based on the database and operating system.
- The data population script (automated-data-population-and-validation-scriptubuntu.sh) is executed to populate data in the database, create users, tenants, user stores, and generate OAuth tokens.
- Stopping the Existing wso2IS-The script stops the running wso2IS instance
- using the stop-server.sh script.

New wso2IS Installation

- The IS_HOME_NEW directory is created to store the new wso2IS
- installation. The latest wso2IS zip file is downloaded using Google Drive API and an access token.
- The downloaded zip file is extracted into the IS_HOME_NEW directory.

- Update IS Packs-The script updates the IS packs using the update-pack.sh script, which takes email and password parameters.

Migration Client Setup

- The migration client is downloaded and extracted from the provided zip
- file. The migration client and resources are copied to the appropriate directories in the new wso2IS installation.

Configuration Changes

- The migration-config.yaml file is modified to include the current and migrating versions and the operating system.

Copying Files

- Various files, such as JAR files, userstores, and tenants, are copied from the old wso2IS installation to the new one.
- Check if all files are copied successfully.
- Change the deployment.toml file in the newIS version based on the provided parameters.
- Execute consent management DB scripts for IS 5.11.0 - MySQL if the migrating version and database match the specified conditions.
- Execute consent management DB scripts for IS 5.11.0 - MSSQL if the migrating version and database match the specified conditions.
- Execute consent management DB scripts for IS 5.11.0 - PostgreSQL if the migrating version and database match the specified conditions.
  Run the migration client.
- Display migration details such as currentVersion, migratingVersion, database, operating system, and time and date.
- Start the migration server.
- Go to the automation home.

- Stop the WSO2 IS

Post-migration Steps

Apply a special configuration change if migrating from IS 5.9 by modifying the deployment.toml file.

If the current version is "5.9.0," this step is included to make a special configuration change in the deployment.toml file.

The script navigates to the deployment directory and modifies the deployment.toml file by commenting out the line type = "database" and uncommenting the line type = "database_unique_id".

It then prints the content of the modified deployment.toml file.

- Start the migrated wso2IS with the migrating version.
- Enter login credentials (admin) and access the management console home page.

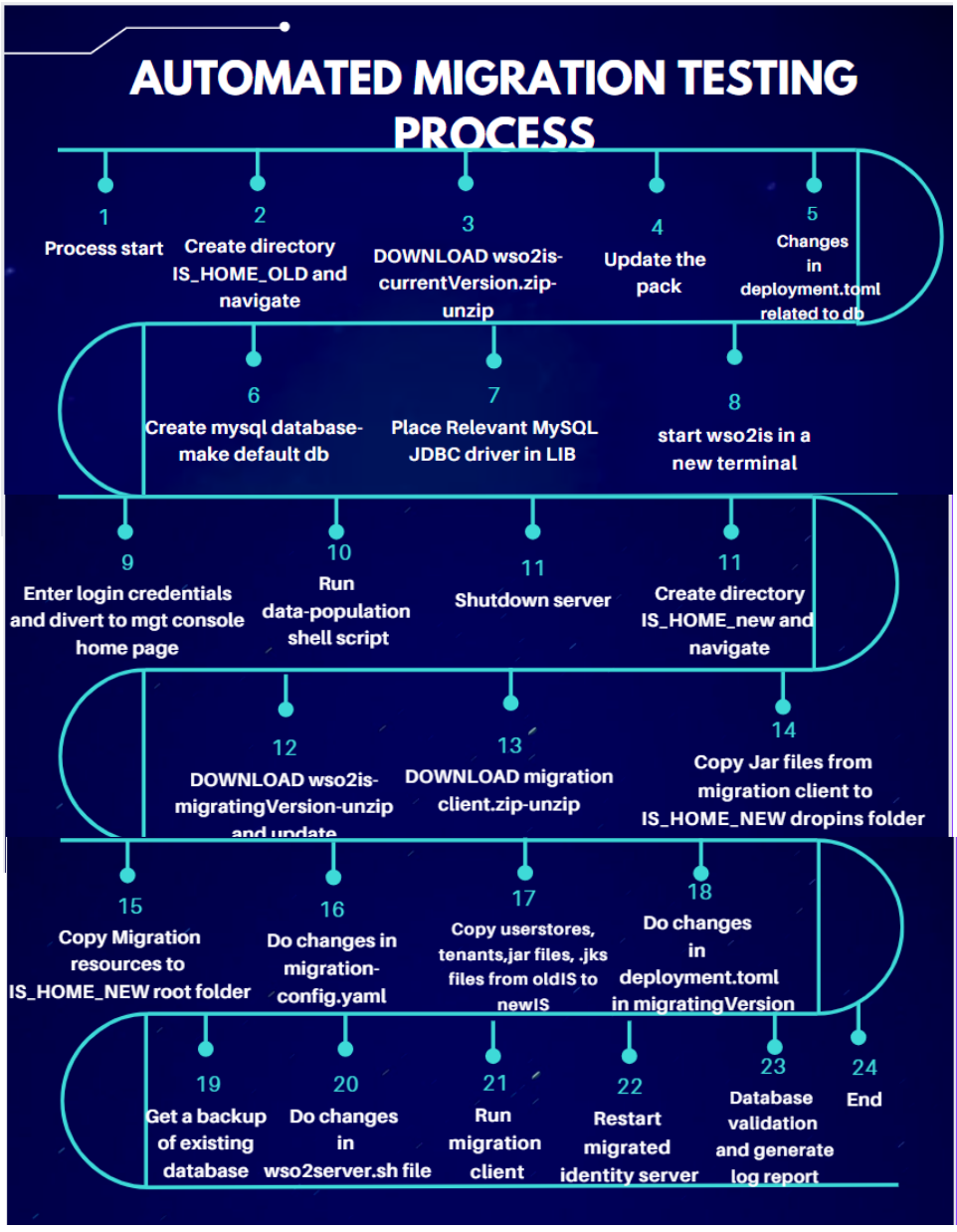
- Go to the data population directory for service provider creation.

Database-validation

Run the scripts to validate database.

- Go back to the automation home.
- Stop the migrated wso2IS.
- Display the end message for automating product migration testing.

7.Rich Picture Diagram



8.End

Intern Project Number: 255

Mentored by: Ashen Weerathunga

Co - Mentored by: Chamath Samarawickrama

Intern: Jayana Gunaweera