## Template Repositories

This document provides an overview of the Module Template and Division Template repositories, including their setup and associated workflows.

## Module Template Repository

**Setup Instructions:**

1. A repository has been created on GitHub with all required files.

2. The "Template repository" option has been enabled in the repository settings.

3. A developer token has been created and stored under repository secrets to allow creation of new repositories from this template using GitHub Actions.

**Workflows:**

The following workflows have been implemented:

1. \*\*Pre-commit Check Validation\*\*: Ensures code quality and consistency before commits.

2. \*\*PR Title Validation\*\*: Checks Pull Request titles for conformity to conventional commit standards.

3. \*\*Automatic Release\*\*: Manages version releases based on commit messages.

Division Template Repository

**Setup Instructions:**

1. A repository has been created on GitHub with all required files.

2. The "Template repository" option has been enabled in the repository settings.

3. A developer token has been created and stored under repository secrets to allow creation of new repositories from this template using GitHub Actions.

**Workflows :**

The following workflow has been implemented:

1. \*\*Pre-commit Check Validation\*\*: Ensures code quality and consistency before commits.

Repository Creation Workflow

**Module:**

This workflow allows you to create new repositories from a template using GitHub Actions.

1. \*\*Access the Actions Tab\*\* - Navigate to the "Actions" tab in your GitHub repository.

2. \*\*Select the Workflow\*\* - Find and select the "module-repository-creation" workflow.

3. \*\*Initiate the Workflow\*\* - Click the "Run workflow" button.

4. \*\*select the below Details\*\* - In the workflow form: -

* Organization name
* Infrastructure as code tool
* Cloud provider
* Resource name
* Description(optional)

5. \*\*Execute the Workflow\*\* - Click "Run workflow" to start the repository creation process.

What This Workflow Does

- Creates a new public repository based on the specified template.

- You can modify the `private: false` setting to `private: true` in the workflow file if you prefer to create private repositories.

**Note:** The personal access token used in this workflow must have both the `repo` and `workflow` scopes enabled. These permissions are necessary to create repositories and run workflows.

**Troubleshooting If you encounter any issues:**

- Ensure your personal access token has the correct permissions.

- Check that the template repository exists and is accessible.

- Verify that you have the necessary permissions in the organization (if creating repositories within an org).

**Division:**

This workflow allows you to create new repositories from a template using GitHub Actions.

1. \*\*Access the Actions Tab\*\* - Navigate to the "Actions" tab in your GitHub repository.

2. \*\*Select the Workflow\*\* - Find and select the "module-repository-creation" workflow.

3. \*\*Initiate the Workflow\*\* - Click the "Run workflow" button.

4. \*\*select the below Details\*\* - In the workflow form: -

* Template name
* Organization name
* Division name
* Platform(optional)- (e.g.: terragrunt/terraform)
* Description(optional)

5. \*\*Execute the Workflow\*\* - Click "Run workflow" to start the repository creation process.

What This Workflow Does:

- Creates a new repository based on the specified template.

- You can modify the `private: false` setting to `private: true` in the workflow file if you prefer to create private repositories.

**Note:** The personal access token used in this workflow must have both the `repo` and `workflow` scopes enabled. These permissions are necessary to create repositories and run workflows.

**Troubleshooting If you encounter any issues:**

- Ensure your personal access token has the correct permissions.

- Check that the template repository exists and is accessible.

- Verify that you have the necessary permissions in the organization (if creating repositories within an org).

Automatic versioning for module repositories:

This project uses automatic versioning and release creation:

1. When changes are merged to `main`, a GitHub Action is triggered

2. This action automatically:

- Determines the next version number based on commit messages

- Creates a new git tag

- Generates a GitHub release with release notes

**Guidelines for pull request and commit messages:**

**Pull Requests:**

- All changes must be made through pull requests (PRs)

- PRs should target the `develop` branch unless hot fixing

- PR titles must follow the Conventional Commits specification

**Commit Messages**

We use Conventional Commits for clear and machine-readable commit messages. Your commit messages should be structured as follows:

``` <type>[optional scope]: <description> [optional body] [optional footer(s)] ```

Types include:

- feat: A new feature

- fix: A bug fix

- docs: Documentation changes

- chore: Routine tasks, maintenance, etc

A GitHub Action validates PR titles against this convention.