**Practical Exercise 09 - Organization, Repository, and Environment Variables**

**Exercise Description**

**In this practical exercise, our goal is to deepen our knowledge around how to use organization, repository, and Environment (GitHub Environments) variables.**

Here are the instructions for the exercise:

1. If you are using organizations, you will be able to create organization-level variables. If not, then simply skip the steps and script lines of this exercise that require organization-level variables. All the steps that require organization features are marked as such.
2. **[Requires Organizations]** - Create two organization-level variables:
   1. ORG\_VAR, should contain the value 'organization value'
   2. OVERWRITTEN\_VAR, should contain the value 'organization value 2'
3. Create a single repository-level variable:
   1. REPOSITORY\_VAR, should contain the value 'repository value'
4. Create two new Environments (if you are not sure how to do it, check the **Tips** section below):
   1. The first Environment should be named prod. Additionally, create two Environment variables:
      1. TARGET\_VAR, should contain the value 'prod'
      2. OVERWRITTEN\_VAR, should contain the value 'prod value'
   2. The second Environment should be named staging. Additionally, create one Environment variable:
      1. TARGET\_VAR, should contain the value 'staging'
5. Extend the workflow named 08-variables.yaml:
   1. Add a third env variable at the workflow-level definition (next to the OVERWRITTEN env variable):
      1. Name it UNDEFINED\_VAR\_WITH\_DEFAULT
      2. Retrieve its value from the UNDEFINED\_VAR variable from within the vars context
      3. If UNDEFINED\_VAR is not defined, provide 'default value' as the default value.
   2. Add a second job named echo2:
      1. The job should run on ubuntu-latest
      2. It should contain a single step, named Print Variables, which prints the following information on the screen:
         1. **[Requires Organizations]** - "Org var: <retrieve the value of ORG\_VAR here>"
         2. **[Requires Organizations]** - "Org overwritten var: <retrieve the value of OVERWRITTEN\_VAR here>"
         3. "Repo var: <retrieve the value of REPOSITORY\_VAR here>"
   3. Add a third job named echo-prod:
      1. The job should run on ubuntu-latest
      2. Set the environment of this job to prod. If you are not sure how to set the job's environment, check the **Tips** section below.
      3. It should contain a single step, named Print Prod Variables, which prints the following information on the screen:
         1. **[Requires Organizations]** - "Org var: <retrieve the value of ORG\_VAR here>"
         2. "Org overwritten var: <retrieve the value of OVERWRITTEN\_VAR here>"
         3. "Repo var: <retrieve the value of REPOSITORY\_VAR here>"
         4. "Environment var: <retrieve the value of TARGET\_VAR here>"
   4. Add a fourth job named echo-undefined:
      1. The job should run on ubuntu-latest
      2. It should contain a single step, named Print Undefined Variables, which prints the following information on the screen:
         1. "Org var: <retrieve the value of UNDEFINED\_VAR\_WITH\_DEFAULT here>"
6. Commit the changes and push the code. Take a few moments to inspect the result of the workflow run triggered by the push event.
7. Try to alternate the Environments between prod and staging. How does this impact the output of the workflow runs?
8. Change the workflow triggers to contain only workflow\_dispatch to prevent this workflow from running with every push and pollute the list of workflow runs.

**Tips**

**Creating Environments in GitHub**

To create Environments in GitHub:

1. Navigate to Your Repository: Go to your repository on GitHub where you want to create the environments.
2. Access Repository Settings: Click on the **“Settings”** tab at the top of the repository page.
3. Go to Environments: In the left sidebar, scroll down to the **“Code and automation”** section and click on **“Environments”**.
4. Create a New Environment: Click the **“New environment”** button on the right, enter the environment name, and click on "Configure Environment".
5. Add Environment Variables:
   1. Scroll down to the **“Environment variables”** section.
   2. Click on **“Add variable”**.
   3. Enter the **Name** (e.g., TARGET\_VAR) and **Value** (e.g., prod or staging).
   4. Click **“Add variable”** to save.
   5. Repeat this step to add additional variables as needed.
6. Once you’ve added all necessary secrets and variables, your environment is ready to use.

**Setting the Environment for a specific job**

To set the environment for a specific job in your GitHub Actions workflow:

1. Edit Your Workflow File: Open the respective workflow file in your repository.
2. Specify the Environment in the Job: Under the specific job you want to set the environment for (e.g., echo-prod), add the environment key. Here is one example of configuring the environment:
3. jobs:
4. echo-prod:
5. runs-on: ubuntu-latest
6. environment: prod # Set the environment to 'prod'
7. steps:
8. - name: Print Prod Variables
9. run: |
10. # Your commands here