**Practical Exercise 21 - Working with Environments**

**Exercise Description**

**In this practical exercise, our goal is to explore how to work with Environments in GitHub Actions, as well as their limitations for non-public repositories.**

Here are the instructions for the exercise:

1. Create a file named 16-environments.yaml under the .github/workflows folder at the root of your repository.
2. Name the workflow 16 - Working with Environments.
3. Add the following triggers with event filters and activity types to your workflow:
   1. workflow\_dispatch: the workflow\_dispatch should also receive an input named target-env, of type environment and with a default value of staging.
4. Set the run-name option of the workflow to 16 - Working with Environments | env - <retrieve target-env input here>
5. Add a single job named echo to the workflow.
   1. It should run on ubuntu-latest.
   2. It should have the environment parameter set to the received input value.
   3. It should define an env variable named my-env-value to the value of MY\_ENV\_VALUE, which should be retrieved from the Environment variables via the vars context. If no value is available, it should default to 'default value'.
   4. The job should contain a single step, named Echo vars, which should print "Env variable: <retrieve the value of my-env-value here>" to the screen.
6. If you have not yet create a prod and a staging Environment, do so (if you have already created it, you can just skip steps a-d below):
   1. Create and configure the prod Environment:
      1. Under the repository page, click on Settings.
      2. On the left-side menu, click on Environments.
      3. Click on New Environment.
      4. Name it prod, and then click on Configure Environment.
      5. Mark the option Required reviewers.
      6. Add yourself as a required reviewer.
      7. Mark the option Wait time.
      8. Set its value to 1 minute.
      9. Make sure that the Environment has a variable named MY\_ENV\_VALUE with the value prod value (it can be added by scrolling to the bottom of the page and clicking on Add variable).
   2. Create and configure the staging Environment:
      1. Under the repository page, click on Settings.
      2. On the left-side menu, click on Environments.
      3. Click on New Environment.
      4. Name it staging, and then click on Configure Environment.
      5. There is no further configuration needed, you can return to the repository page.
7. Commit the changes and push the code. Trigger the workflow manually from the UI and take a few moments to inspect the result of the workflow run.
8. Now modify the workflow by:
   1. Removing the run-name option.
   2. Removing the target-env input.
   3. Renaming the echo job to deploy-staging.
   4. Hard-coding the environment of deploy-staging to staging.
   5. Changing the statement from the Echo vars step to print "Deploying to staging".
   6. Adding a new job named e2e-tests:
      1. It should run on ubuntu-latest.
      2. It should be executed if and only if the deploy-staging job successfully completes.
      3. It should contain a single step, named E2E tests, which prints "Running E2E" to the screen.
   7. Adding another job named deploy-prod:
      1. It should run on ubuntu-latest.
      2. It should be executed if and only if the e2e-tests job successfully completed.
      3. It should have the environment parameter set to prod.
      4. It should define an env variable named my-env-value to the value of MY\_ENV\_VALUE, which should be retrieved from the Environment variables via the vars context. If no value is available, it should default to 'default value'.
      5. The job should contain a single step, named Echo vars, which should print "Deploying to prod" to the screen.
9. Commit the changes and push the code. Trigger the workflow manually from the UI and take a few moments to inspect the result of the workflow run. What happened to the prod environment job? How did you approve it?