**Practical Exercise 16 - Using Multiple Jobs for Better Caching Functionality**

**Exercise Description**

**In this practical exercise, our goal is to explore how we can use a dedicated job for caching so that it prevents the parallel installation of dependencies once the package-lock.json file changes.**

In order to achieve that, we will extend the workflow 13-caching.yaml. Here are the instructions for the exercise:

1. Add a new job named install-deps:
   1. The job should run on ubuntu-latest
   2. The job should set 13-caching/react-app as the default working-directory option for run commands.
   3. The job should contain five steps:
      1. The first one, named Checkout code, should checkout the repository code into the current working directory.
      2. The second one, named Setup Node, should setup Node with the version provided as an input.
      3. The third one, named Calculate cache key, should have an id of cache-key and generate an output named CACHE\_KEY and with the value deps-node-modules-${{ hashFiles('13-caching/react-app/package-lock.json') }}
      4. The fourth one should be similar to the already implemented Download cached dependencies step. However, it should use the output of the cache-key step instead of hard-coding the cache key.
      5. The fifth one should be similar to the already implemented Install dependencies step.
   4. The job should contain a single output named deps-cache-key, and its value should come from the CACHE\_KEY output of the cache-key step.
2. Add a new job named linting:
   1. The job should run on ubuntu-latest
   2. The job should run if and only if the install-deps job successfully completes.
   3. The job should set 13-caching/react-app as the default working-directory option for run commands.
   4. The job should contain five steps:
      1. The first one, named Checkout code, should checkout the repository code into the current working directory.
      2. The second one, named Setup Node, should setup Node with the version provided as an input.
      3. The third one should be similar to the already implemented Download cached dependencies step. However, it should use the output of the install-deps job instead of hard-coding the cache key, and it should not depend on the use-cache input.
      4. The fourth one, named Testing, should run the npm run test script.
      5. The fifth one, named Linting, should print "Linting..." on the screen.
3. Modify the build job:
   1. Add a dependency to the install-deps job.
   2. Remove the Install dependencies step, since the dependencies will always come from the cache.
   3. Update the Download cached dependencies step to use the output of the install-deps job instead of hard-coding the cache key. Additionally, it should not be influenced by the use-cache input anymore, since the cache becomes necessary for the correct execution of the job.
4. Commit the changes and push the code. Trigger the workflow a few times and inspect the running times of each step. How much time does it take to install the dependencies? How much time would it take if the workflow ran 1000 times?