Continuous Integration (CI) Build Caching

To improve build performance, Next.js saves a cache to `.next/cache` that is shared between builds.

To take advantage of this cache in Continuous Integration (CI) environments, your CI workflow will need to be configured to correctly persist the cache between builds.

If your CI is not configured to persist `.next/cache` between builds, you may see a No Cache Detected error.

Here are some example cache configurations for common CI providers:

Vercel

Next.js caching is automatically configured for you. There's no action required on your part.

CircleCI

Edit your `save_cache` step in `.circleci/config.yml` to include `.next/cache`:

```
steps:
  - save_cache:
      key: dependency-cache-{{ checksum "yarn.lock" }}
      paths:
        - ./node_modules
        - ./.next/cache
```

If you do not have a `save_cache` key, please follow CircleCI's documentation on setting up build caching ↗.

Travis CI

Add or merge the following into your `.travis.yml`:

```
cache:
  directories:
    - $HOME/.cache/yarn
    - node_modules
    - .next/cache
```

GitLab CI

Add or merge the following into your `.gitlab-ci.yml`:

```
cache:
  key: ${CI_COMMIT_REF_SLUG}
  paths:
    - node_modules/
    - .next/cache/
```

Netlify CI

Use Netlify Plugins ↗ with `@netlify/plugin-nextjs` ↗.

AWS CodeBuild

Add (or merge in) the following to your `buildspec.yml`:

```
cache:
  paths:
    - 'node_modules/**/*' # Cache `node_modules` for faster `yarn` or `npm i`
    - '.next/cache/**/*' # Cache Next.js for faster application rebuilds
```

GitHub Actions

Using GitHub's actions/cache↗, add the following step in your workflow file:

```
uses: actions/cache@v3
with:
  # See here for caching with `yarn` https://github.com/actions/cache/blob/main/examples#
  path: |
    ~/.npm
    ${{ github.workspace }}/.next/cache
  # Generate a new cache whenever packages or source files change.
  key: ${{ runner.os }}-nextjs-${{ hashFiles('**/package-lock.json') }}-${{ hashFiles('**
  # If source files changed but packages didn't, rebuild from a prior cache.
  restore-keys: |
    ${{ runner.os }}-nextjs-${{ hashFiles('**/package-lock.json') }}-
```

Bitbucket Pipelines

Add or merge the following into your `bitbucket-pipelines.yml` at the top level (same level as `pipelines`):

```
definitions:
  caches:
    nextcache: .next/cache
```

Then reference it in the `caches` section of your pipeline's `step`:

```
- step:
    name: your_step_name
    caches:
      - node
      - nextcache
```

Heroku

Using Heroku's custom cache↗, add a `cacheDirectories` array in your top-level package.json:

```
"cacheDirectories": [".next/cache"]
```

Azure Pipelines

Using Azure Pipelines' Cache task↗, add the following task to your pipeline yaml file somewhere prior to the task that executes `next build`:

```
- task: Cache@2
  displayName: 'Cache .next/cache'
  inputs:
    key: next | $(Agent.OS) | yarn.lock
    path: '$(System.DefaultWorkingDirectory)/.next/cache'
```