

feat(auth): add new auth interceptor

Adds a new http interceptor to authenticate requests with our new auth system via the auth service added in #340. This is disabled behind a flag until we are ready to migrate from the legacy auth system to the new one.

Implements #342

Type Scope
feat(auth): add new auth interceptor ← Description

Body
Adds a new http interceptor to authenticate requests with our new auth system via the auth service added in #340. This is disabled behind a flag until we are ready to migrate from the legacy auth system to the new one.

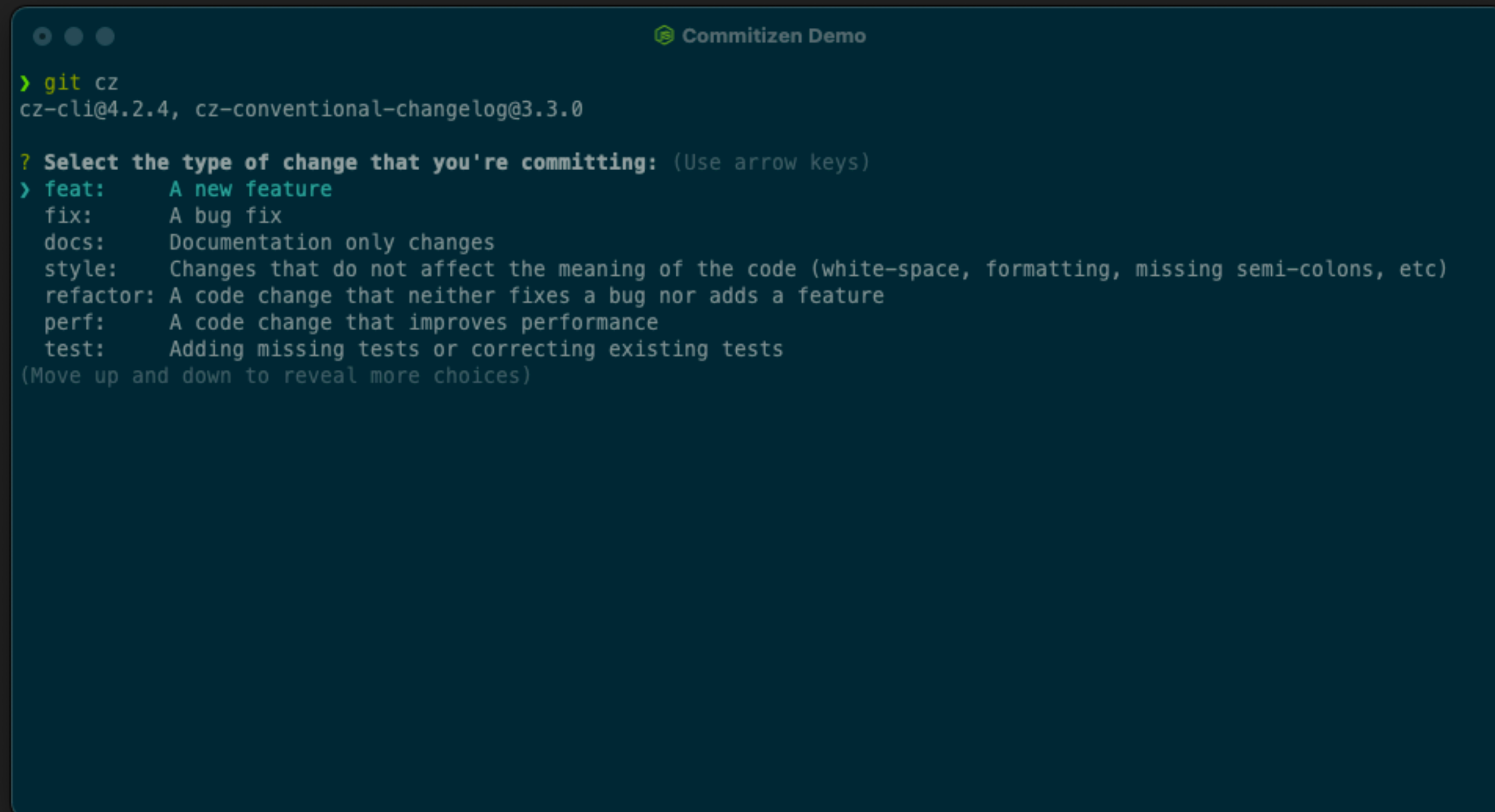
Implements #342 ← Footer

TOOLS TO HELP EASE ADOPTION OF CONVENTIONAL COMMITS

Commitizen - CLI to help format commits

Commitlint - Lint your commits

Husky - Git Hooks

A terminal window titled "Commitizen Demo" with a dark blue background. It shows the command `git cz` being executed, which opens the Commitizen CLI. The CLI displays the version `cz-cli@4.2.4, cz-conventional-changelog@3.3.0` and prompts the user to select a commit type. The available options are: `feat:` (A new feature), `fix:` (A bug fix), `docs:` (Documentation only changes), `style:` (Changes that do not affect the meaning of the code), `refactor:` (A code change that neither fixes a bug nor adds a feature), `perf:` (A code change that improves performance), and `test:` (Adding missing tests or correcting existing tests). A note at the bottom says "(Move up and down to reveal more choices)".

```
> git cz
cz-cli@4.2.4, cz-conventional-changelog@3.3.0

? Select the type of change that you're committing: (Use arrow keys)
> feat:      A new feature
  fix:       A bug fix
  docs:      Documentation only changes
  style:     Changes that do not affect the meaning of the code (white-space, formatting, missing semi-colons, etc)
  refactor:  A code change that neither fixes a bug nor adds a feature
  perf:      A code change that improves performance
  test:      Adding missing tests or correcting existing tests
(Move up and down to reveal more choices)
```