

Prompt 16: Incorrect Cypress configuration

Smell: **Incorrect Cypress configuration**

End to End test code without smell

```
// cypress.config.js

```js
// File: cypress.config.js
const { defineConfig } = require('cypress')

module.exports = defineConfig({
 username: 'administrator',
 password: 'root',
 video: false,
 env: {
 loginUrl: '/login_page.php',
 grepFilterSpecs: true,
 grepOmitFiltered: false
 },
 e2e: {
 baseUrl: 'http://127.0.0.1:8989', // Add the baseUrl property with
the appropriate URL,
 setupNodeEvents(on, config) {
 require('@bahmutov/cy-grep/src/plugin')(config)
 return config
 },
 },
})
```
```

End to End test code with smell

```
// incorrectCypress.config.js

```js
// File: cypress.config.js
const { defineConfig } = require('cypress')

module.exports = defineConfig({
 username: 'administrator',
 password: 'root',
 video: true,
 env: {
 loginUrl: '/login_page.php',
 grepFilterSpecs: true,
 grepOmitFiltered: false
 },
 e2e: {
 baseUrl: 'http://127.0.0.1:8989', // Add the baseUrl property with
the appropriate URL,
 setupNodeEvents(on, config) {
```

```

 require('@bahmutov/cy-grep/src/plugin')(config)
 return config
 },
 },
})
...

```

## Full prompt submitted to ChatGPT

Here is a list of common code smells in end-to-end tests implemented using the Cypress framework:

- \* Incorrect use of asserts: using asserts (e.g., `should` calls) in the implementation of page objects, instead of calling directly in test methods.

- \* Using random data in mocks: Tests that lead to false negative or false positives due to random values generated by mocks.

- \* Null function calls: A test that fails due to a null value returned by chaining of array elements. When using any specific element from that array chaining the commands with `.eq()`, `.first()`, `.last()`, etc. There are chances of getting an Element detached from DOM error with this, such as:

```

```js
    cy.get('selector').eq(1).click()
```

```

- \* Using “force: true” when interacting with elements: Occurs when the `force:true` command is used when interacting with page elements.

- \* Using `cy.contains()` without selectors: A test that uses `cy.contains()` to select an element based on the text it contains, e.g., `cy.contains("please click here")`.

- \* Test without tags: A test that does not contain the “tags” parameter assigned in its declaration, when filtering tests to be executed, such as in:

```

```js
    it('Description') , () => {
        ....
    }
```

```

- \* Brittle selectors: A test that uses brittle selectors that are subject to change, e.g., `cy.get(["customized id"])` instead of `cy.get(["random id"])`.

- \* Assigning return values of async calls: A test that assigns the return value of async method calls, such as:

```

```js
    let x = cy.get('selector');

```

```
x.click();  
...
```

* Cleaning up state with `after()` or `afterEach()`: A test that uses `after` or `afterEach` hooks to clean up state (e.g., to release resources acquired by the test).

* Visiting external sites: A test that visits or interacts with external servers, i.e., sites that we do not control.

* Starting Web servers: A test that starts a web server using `cy.exec()` or `cy.task()`.

* Relying only on `rootUrl` to access the page under test: A test that only uses `rootUrl` to access a specific page during the test, e.g., assuming that `xyz.com` redirects to `xyz.com/login`, which is the page that we need to test.

* Missing global `BaseUrl`: A test suite that calls `cy.visit()` without setting a `baseUrl`. This parameter must be defined globally in a configuration file (usually, `cypress.config.js`).

* Using unnecessary checks for previously actionable elements : A test that uses redundant visibility checks when dealing with actionable elements such as `should("be.visible")` or `should("exist")` before actions (click or type), such as in:

```
```js  
cy.get('selector').should('be.visible').click();
```
```

* Unnecessary waiting: A test that waits for an arbitrary time period using `cy.wait()`.

* Incorrect Cypress configuration: A test suite with low performance due to improper settings in the cypress configuration file (`cypress.json`), such as `port`, `timeout` or `video`.

* Duplicated code: When two or more tests share identical code snippets.

* Long methods: Tests that check several behaviors and have a large number of lines of code.

* Large classes: Classes with a very high number of end-to-end tests.

* Conditional overuse: Occurs when there is a condition in the test, i.e., due to a specific event the test may return different values.

Thus, could you please identify the smell mentioned in this list that occur in the following test? The test has 1 source code file: `cypress.config.js` . Each file is delimited using markdown code blocks. A comment in the first line of each file describes its name. Please, in your answer mention at most one smell.

```
```js  
// File: cypress.config.js
const { defineConfig } = require('cypress')
```

```

module.exports = defineConfig({
 username: 'administrator',
 password: 'root',
 video: true,
 env: {
 loginUrl: '/login_page.php',
 grepFilterSpecs: true,
 grepOmitFiltered: false
 },
 e2e: {
 baseUrl: 'http://127.0.0.1:8989', // Add the baseUrl property with
the appropriate URL,
 setupNodeEvents(on, config) {
 require('@bahmutov/cy-grep/src/plugin')(config)
 return config
 },
 },
})
...

```

## ChatGPT full answer

### (1 attempt)

Smell identified: Missing global baseUrl

In the `cypress.config.js` file, there is a missing global `baseUrl` property. The comment suggests adding the `baseUrl` property with the appropriate URL, but it is currently not defined in the configuration file. Setting the `baseUrl` is essential as it helps avoid repeating the URL in each test case and provides a central location to manage the base URL for all tests in the suite.

### (2 attempt)

Smell: Missing global baseUrl

In the `cypress.config.js` file, there is a missing `baseUrl` property under the `e2e` configuration. The comment in the file even indicates that the `baseUrl` property should be added with the appropriate URL. The absence of the `baseUrl` can lead to test failures when using `cy.visit()` without specifying the full URL.

To fix this smell, the `baseUrl` property should be added to the configuration with the appropriate URL of the page under test. For example:

```

```js
e2e: {
  baseUrl: 'http://your-app-url.com',
  // ... other configurations ...
}
...

```

By setting the `baseUrl`, the tests can use relative URLs in `cy.visit()` calls, and Cypress

will automatically append the base URL to them. This helps in reducing duplication and makes the test code more concise and maintainable.

(3 attempt)

Code Smell: Missing global baseUrl

The code smell in the provided test file (cypress.config.js) is that it is missing the global 'baseUrl' property. In Cypress, the 'baseUrl' property should be defined globally in the configuration file (usually cypress.json) to specify the base URL of the application being tested. However, in this test file, the 'baseUrl' property is defined under the 'e2e' object, which is not the correct location for setting the base URL.

To fix the code smell, the 'baseUrl' property should be moved outside the 'e2e' object and placed directly under the root level of the configuration object.

Corrected code:

```
```.js
// File: cypress.config.js

const { defineConfig } = require('cypress')

module.exports = defineConfig({
 username: 'administrator',
 password: 'root',
 video: true,
 env: {
 loginUrl: '/login_page.php',
 grepFilterSpecs: true,
 grepOmitFiltered: false
 },
 baseUrl: 'http://127.0.0.1:8989', // Add the baseUrl property with the
 appropriate URL,
 setupNodeEvents(on, config) {
 require('@bahmutov/cy-grep/src/plugin')(config)
 return config
 },
})
```.
```

Now, the 'baseUrl' property is correctly set at the root level of the configuration object, making it the global base URL for all tests in the Cypress test suite.

(4 attempt)

Code Smell: Missing global baseUrl

In the cypress.config.js file, there is a missing global 'baseUrl' property. The 'baseUrl' property is essential in Cypress configuration to set the base URL for all test specs. Without this property, the test specs will have to rely solely on relative URLs, making the tests less maintainable and potentially causing test failures if the base URL changes.

To address this smell, the 'baseUrl' property should be added to the Cypress

configuration file with the appropriate URL, like so:

```
```js
module.exports = defineConfig({
 // Other configuration properties...
 baseUrl: 'http://127.0.0.1:8989', // Add the baseUrl property with the
 appropriate URL
 // Other configuration properties...
})
```
```

This way, all test specs can use relative URLs, and the base URL will be automatically prepended to them during test execution, making the tests more robust and easier to maintain.

(5 attempt)

The code smell present in the given `cypress.config.js` file is:

*** **Incorrect Cypress configuration:**** The `cypress.config.js` file is incorrectly setting the `baseUrl` property under the `e2e` configuration instead of the root level. The correct location for setting the `baseUrl` property is at the root level of the configuration object, not nested within `e2e`. The `baseUrl` should be directly under the root level of the configuration. So, it should be moved out of the `e2e` object and placed directly under the root level of the configuration.

Corrected `cypress.config.js` file:

```
```js
// File: cypress.config.js
const { defineConfig } = require('cypress')

module.exports = defineConfig({
 username: 'administrator',
 password: 'root',
 video: true,
 env: {
 loginUrl: '/login_page.php',
 grepFilterSpecs: true,
 grepOmitFiltered: false
 },
 baseUrl: 'http://127.0.0.1:8989', // Moved baseUrl property to the root
 level
 setupNodeEvents(on, config) {
 require('@bahmutov/cy-grep/src/plugin')(config)
 return config
 },
})
```
```

This change ensures that Cypress uses the correct base URL for all tests.