Code Review. Cypress

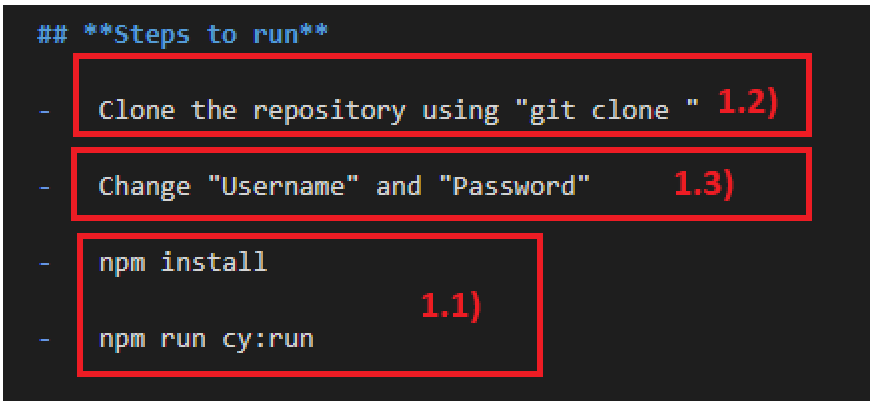
The testing repository is https://github.com/AZANIR/cypressAllure.

1) README.md file

1.1) We can see the *npm* commands in the "Steps to Run" section. But there is no information about the Node.js required to run these commands.

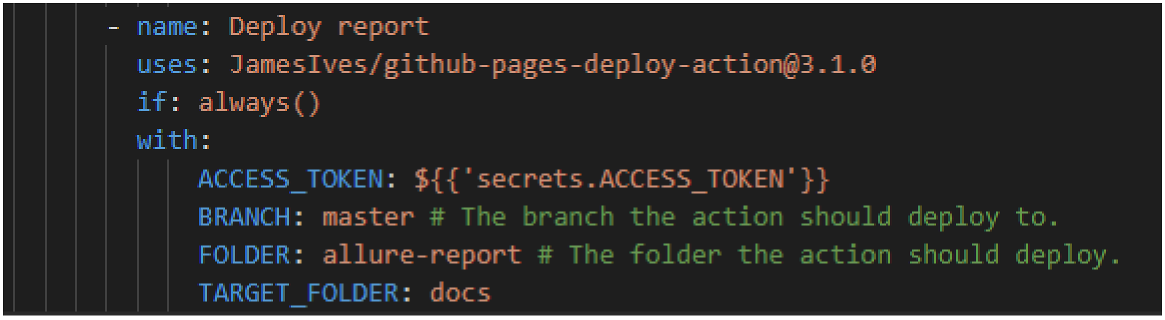
1.2) We can see the *clone* command in the same "Steps to get started" section, but the link to the GitHub repository is missing.

1.3) Also in this section "Steps to start", there is a step: *Change "Username" and "Password"*. But it is not clear what it is for, where it is and how it should be used.



```
## **Steps to run**

- Clone the repository using "git clone "

- Change "Username" and "Password"

- npm install

- npm run cy:run
```

1.4) In the ‘yml’ file, we see that the report should be deployed on the GitHub pages, but nothing is said about this in ‘README.md’



```
      - name: Deploy report
        uses: JamesIves/github-pages-deploy-action@3.1.0
        if: always()
        with:
          ACCESS_TOKEN: ${{'secrets.ACCESS_TOKEN'}}
          BRANCH: master # The branch the action should deploy to.
          FOLDER: allure-report # The folder the action should deploy.
          TARGET_FOLDER: docs
```

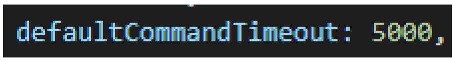
1.5) There is no information about test cases.

2) Project structure

2.1) File ‘commands.js’ in “support” folder is useless.

2.2) File ‘README.md’ in “docs” folder is useless. It contains only one sentence - ### report folder

3) **config.config.ts** file

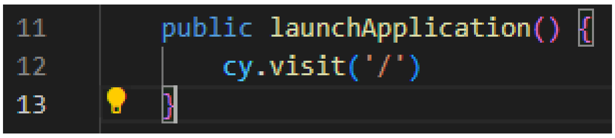


```
defaultCommandTimeout: 5000,
```

By default, the test execution time is 4 seconds. In this case - 5 seconds. This is not considered a big mistake, but it increases the time of passing all tests. If you need to wait for an element to appear in the test, you can use the ‘timeout’ command for a specific element.

4) Pages and tests(specs)

4.1) **loginPage.ts** . This command should open a specific page. Perhaps it is better to name the command - openBasicPage

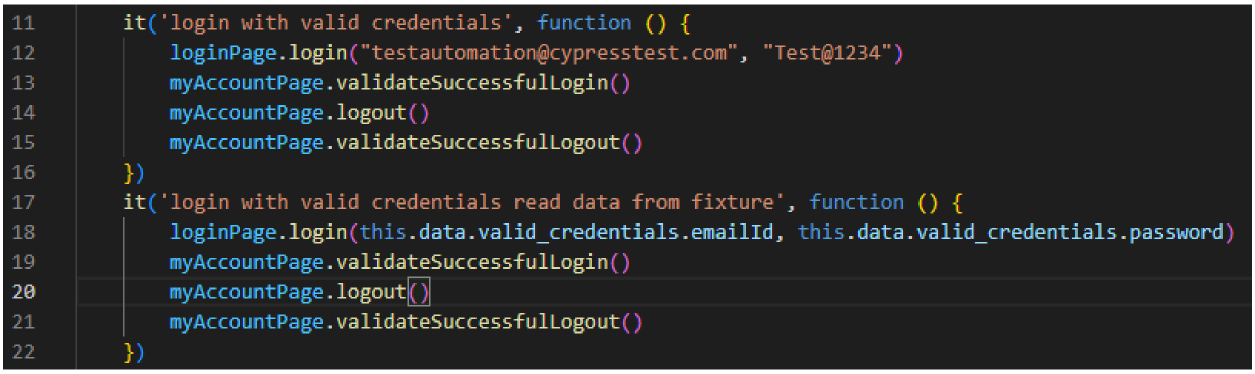


```
    public launchApplication() {
        cy.visit('/')
    }
```

4.2) **login.test.ts** .

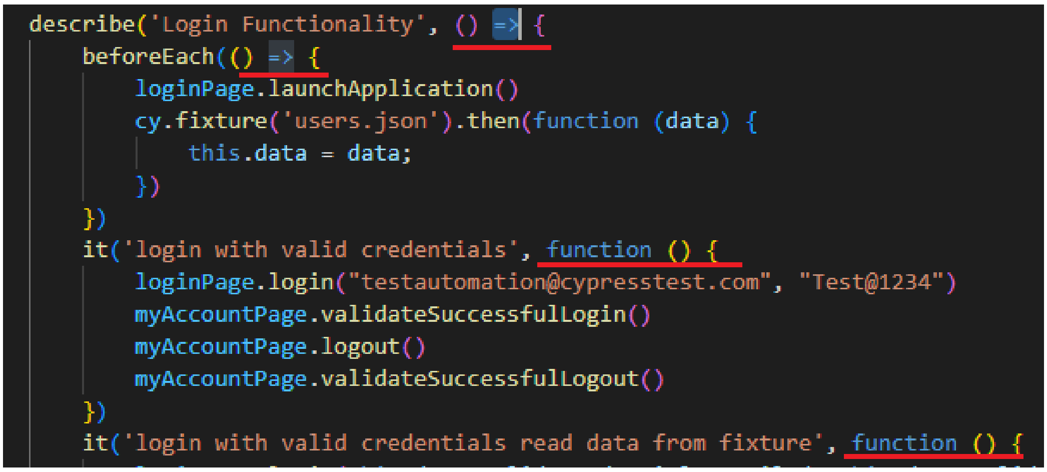
4.2.1) In my opinion, it would be good to add ID to the tests.

4.2.2) Testi 'login with valid credentials' and 'login with valid credentials read data from fixture' are identical. The difference is that in the test that starts on line 17, the login and password are taken from file ‘users.json’ of the “fixtures” folder. It is really more practical (for example, when you have a lot of tests where you need to enter a login and password) and safer. Therefore, I would leave this test and delete the previous one.



```
11      it('login with valid credentials', function () {
12          loginPage.login("testautomation@cypresstest.com", "Test@1234")
13          myAccountPage.validateSuccessfulLogin()
14          myAccountPage.logout()
15          myAccountPage.validateSuccessfulLogout()
16      })
17      it('login with valid credentials read data from fixture', function () {
18          loginPage.login(this.data.valid_credentials.emailId, this.data.valid_credentials.password)
19          myAccountPage.validateSuccessfulLogin()
20          myAccountPage.logout()
21          myAccountPage.validateSuccessfulLogout()
22      })
```

4.2.3) In tests, the function is most often indicated by the word 'function', and sometimes by the sign ' => '. You need to choose one thing to use.



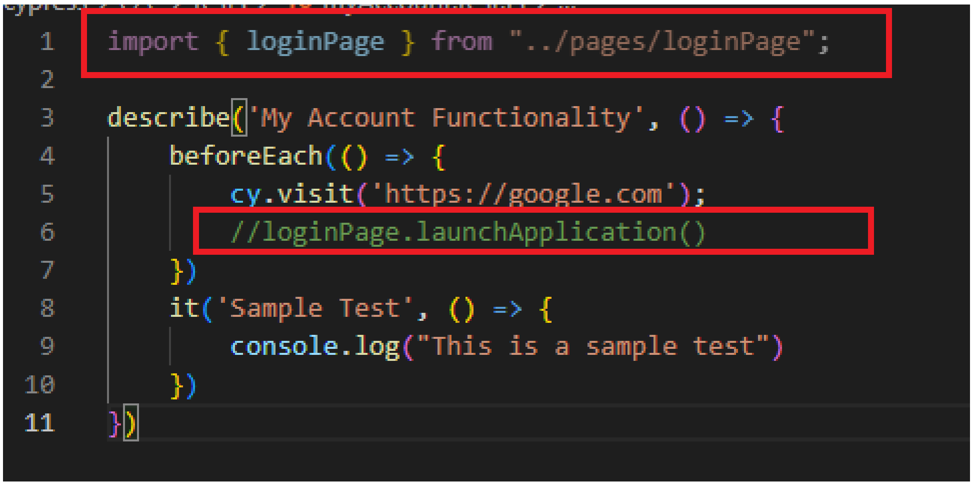
```
describe('Login Functionality', () => {
    beforeEach(() => {
        loginPage.launchApplication()
        cy.fixture('users.json').then(function (data) {
            this.data = data;
        })
    })
    it('login with valid credentials', function () {
        loginPage.login("testautomation@cypresstest.com", "Test@1234")
        myAccountPage.validateSuccessfulLogin()
        myAccountPage.logout()
        myAccountPage.validateSuccessfulLogout()
    })
    it('login with valid credentials read data from fixture', function () {
```

4.3) **myAccount.test.ts**

4.3.1) In general, the essence of the test is unclear. The description says that this is 'My Account Functionality' and the link is used to the general page of Google https://google.com. At the same time, the test itself is called "Sample test"

4.3.2) The 'before Each' command is used here, but there is only one test.

4.3.3) Inactive code elements are highlighted in the picture below. If they are not needed, it is better to remove them.



```
import { loginPage } from "../pages/loginPage";

describe('My Account Functionality', () => {
    beforeEach(() => {
        cy.visit('https://google.com');
        //loginPage.launchApplication()
    })
    it('Sample Test', () => {
        console.log("This is a sample test")
    })
})
```

5) .gitignore

5.1) Duplication ' reports ' and ' reports/** '

5.2) Maybe I would add ' docs ' folder.

6) package.json file

6.1) We can see the "cy:e2e:run" comand in package.json file, this command combines the sequential execution of three commands but it is not used in the ‘yml’ files or described in the ‘README’ file