

## 1. The LoginPage.js file has an extra line indent between lines:

```
cypressAllure / cypress / e2e / pages / LoginPage.ts
AZANIR update test and add allure

Code Blame 28 lines (23 loc) · 879 Bytes

1  /// <reference types="cypress" />
2
3  class LoginPage {
4      get signInLink() { return cy.get('.login') }
5      get emailAddressTxt() { return cy.get('#email') }
6      get passwordTxt() { return cy.get('#passwd') }
7      get signInBtn() { return cy.get('#SubmitLogin') }
8      get alertBox() { return cy.get('p:contains("error")') }
9      get alertMessage() { return cy.get('.alert-danger > ol > li') }
10
11     public launchApplication() {
12         cy.visit('/')
13     }
14
15     public login(emailId: string, password: string) {
16         this.signInLink.click()
17         this.emailAddressTxt.type(emailId)
18         this.passwordTxt.type(password)
19         this.signInBtn.click()
20     }
21
22     public validateLoginError(errorMessage: string) {
23         this.alertBox.should('be.visible')
24         this.alertMessage.should('have.text', errorMessage)
25     }
26 }
27
28 export const loginPage = new LoginPage()
```

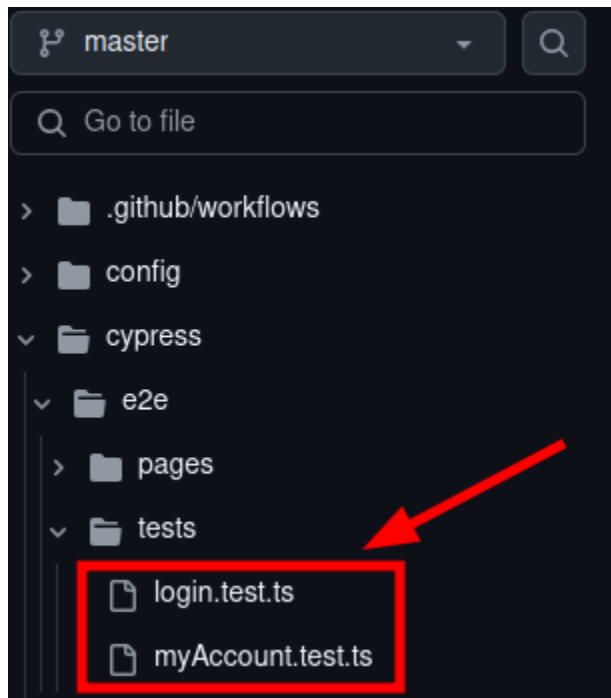
## 2. There are no semicolons in test and pages files:

```
cypressAllure / cypress / e2e / tests / login.test.ts
AZANIR final update yml

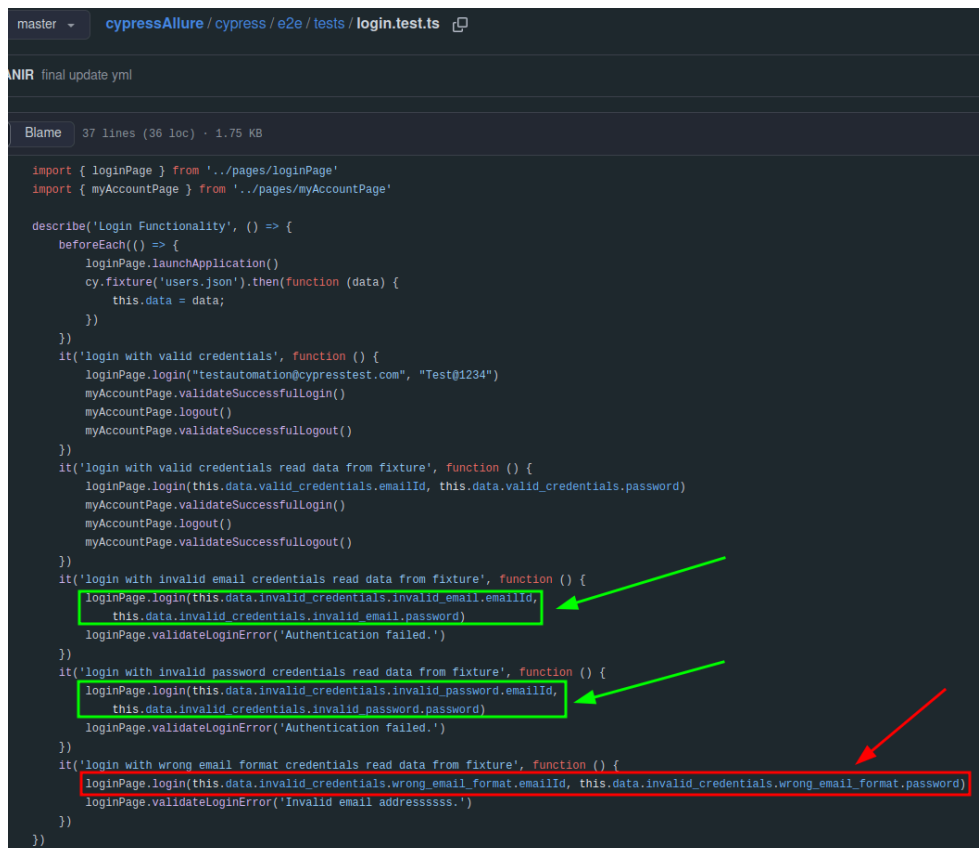
Code Blame 37 lines (36 loc) · 1.75 KB

1  import { loginPage } from '../pages/loginPage'
2  import { myAccountPage } from '../pages/myAccountPage'
3
4  describe('Login Functionality', () => {
5      beforeEach(() => {
6          loginPage.launchApplication()
7          cy.fixture('users.json').then(function (data) {
8              this.data = data;
9          })
10     })
11     it('login with valid credentials', function () {
12         loginPage.login("testautomation@cyresstest.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     })
23     it('login with invalid email credentials read data from fixture', function () {
24         loginPage.login(this.data.invalid_credentials.invalid_email.emailId,
25             this.data.invalid_credentials.invalid_email.password)
26         loginPage.validateLoginError('Authentication failed.')
27     })
28     it('login with invalid password credentials read data from fixture', function () {
29         loginPage.login(this.data.invalid_credentials.invalid_password.emailId,
30             this.data.invalid_credentials.invalid_password.password)
31         loginPage.validateLoginError('Authentication failed.')
32     })
33     it('login with wrong email format credentials read data from fixture', function () {
34         loginPage.login(this.data.invalid_credentials.wrong_email_format.emailId, this.data.invalid_credentials.wrong_email_format.password)
35         loginPage.validateLoginError('Invalid email addressssss.')
36     })
37 })
```

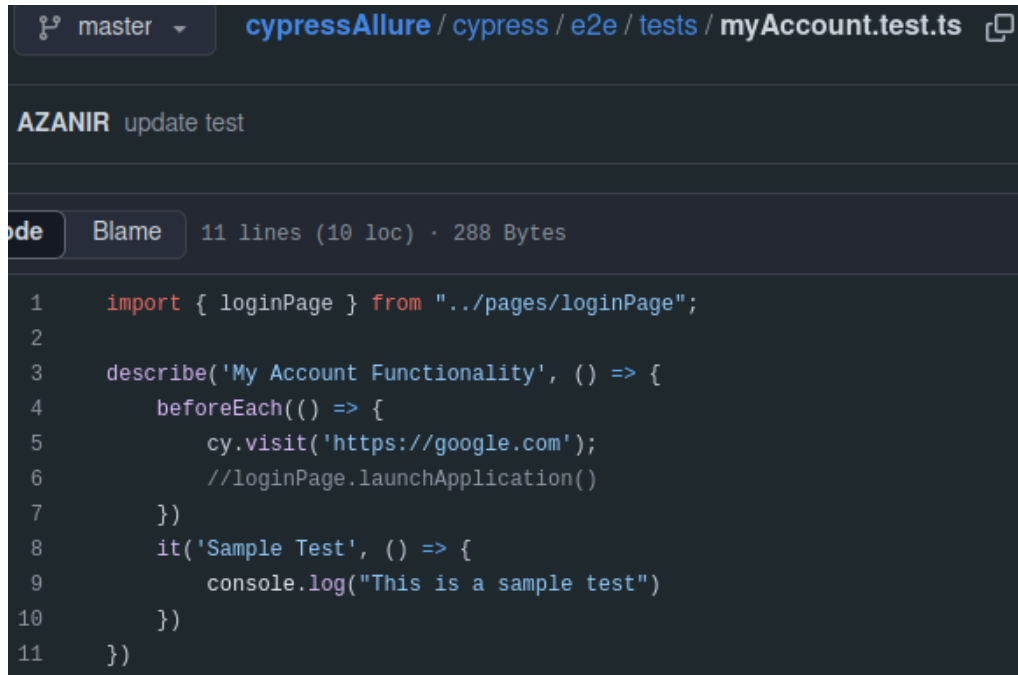
3. The project uses specific test names: test.ts instead of: cy.ts:



4. Different formatting in the test of similar components:

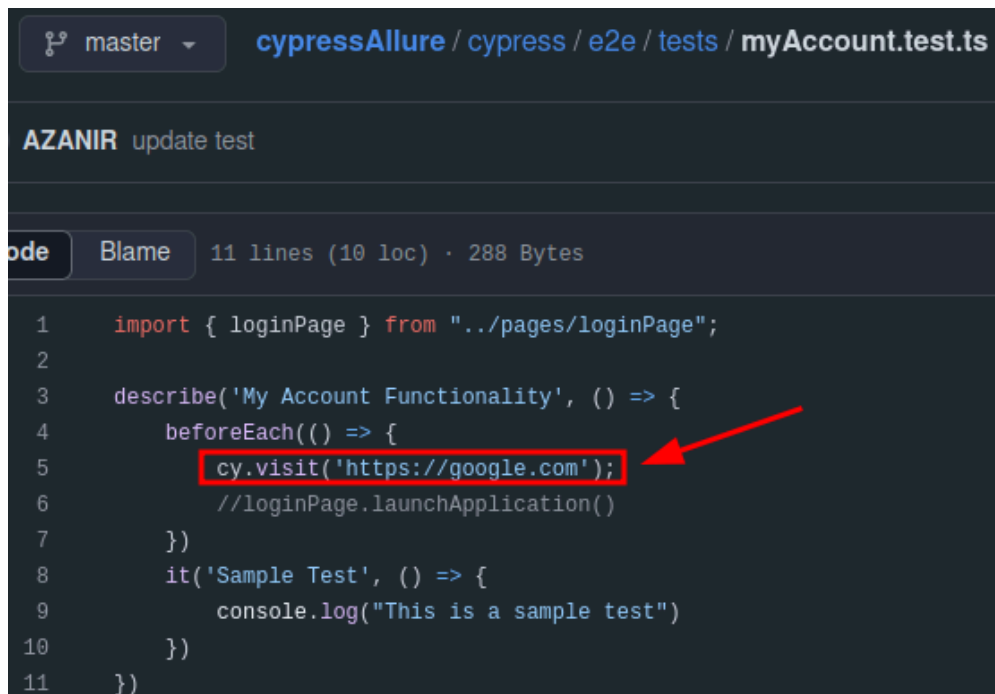


5. The test looks unnecessary and should be deleted. It may have been used when the project was created:



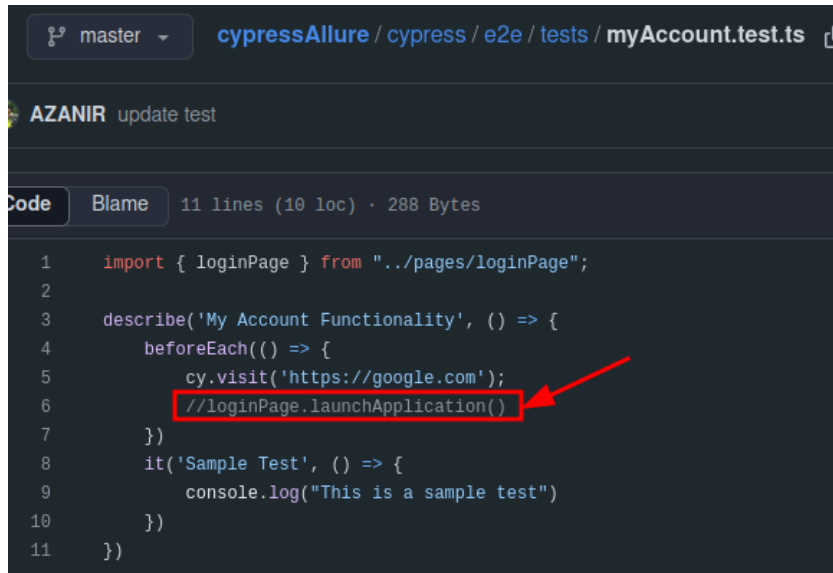
```
1 import { loginPage } from "../pages/loginPage";
2
3 describe('My Account Functionality', () => {
4   beforeEach(() => {
5     cy.visit('https://google.com');
6     //loginPage.launchApplication()
7   })
8   it('Sample Test', () => {
9     console.log("This is a sample test")
10  })
11 })
```

6. The test does not use base url:



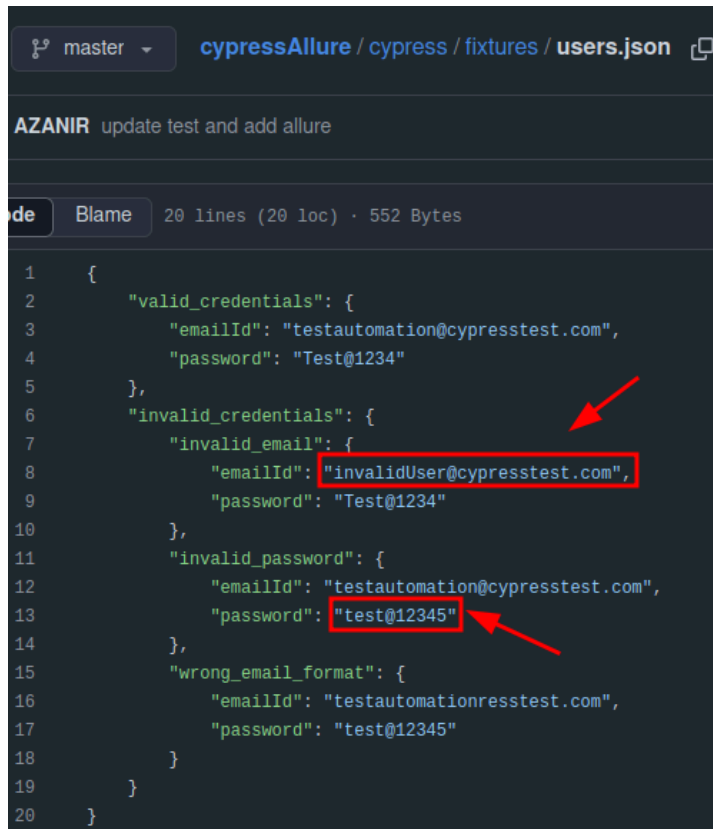
```
1 import { loginPage } from "../pages/loginPage";
2
3 describe('My Account Functionality', () => {
4   beforeEach(() => {
5     cy.visit('https://google.com');
6     //loginPage.launchApplication()
7   })
8   it('Sample Test', () => {
9     console.log("This is a sample test")
10  })
11 })
```

7. The test has a commented code:




```
1  import { loginPage } from "../pages/loginPage";
2
3  describe('My Account Functionality', () => {
4    beforeEach(() => {
5      cy.visit('https://google.com');
6      //loginPage.launchApplication()
7    })
8    it('Sample Test', () => {
9      console.log("This is a sample test")
10    })
11  })
```


8. The project uses static data for invalid data in the users.json file. The best practice is to use for that the faker.js library:



```
1  {
2    "valid_credentials": {
3      "emailId": "testautomation@cypresstest.com",
4      "password": "Test@1234"
5    },
6    "invalid_credentials": {
7      "invalid_email": {
8        "emailId": "invalidUser@cypresstest.com",
9        "password": "Test@1234"
10      },
11      "invalid_password": {
12        "emailId": "testautomation@cypresstest.com",
13        "password": "test@12345"
14      },
15      "wrong_email_format": {
16        "emailId": "testautomationresstest.com",
17        "password": "test@12345"
18      }
19    }
20  }
```

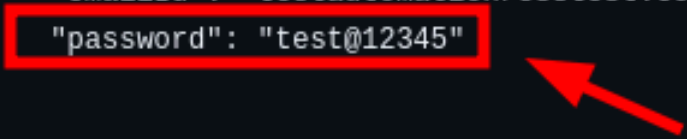
9. Incorrect password data in the users.json file. In this place, the password must be valid:

**cypressAllure** / **cypress** / **fixtures** / **users.json** 

 **AZANIR** update test and add allure

**Code** **Blame** 20 lines (20 loc) · 552 Bytes

```
1  {
2    "valid_credentials": {
3      "emailId": "testautomation@cypresstest.com",
4      "password": "Test@1234"
5    },
6    "invalid_credentials": {
7      "invalid_email": {
8        "emailId": "invalidUser@cypresstest.com",
9        "password": "Test@1234"
10     },
11     "invalid_password": {
12       "emailId": "testautomation@cypresstest.com",
13       "password": "test@12345"
14     },
15     "wrong_email_format": {
16       "emailId": "testautomationresstest.com",
17       "password": "test@12345"
18     }
19   }
20 }
```



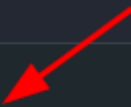
10. Data can be optimized in the users.json file:

master cypressAllure / cypress / fixtures / users.json


AZANIR update test and add allure

Code Blame 20 lines (20 loc) · 552 Bytes

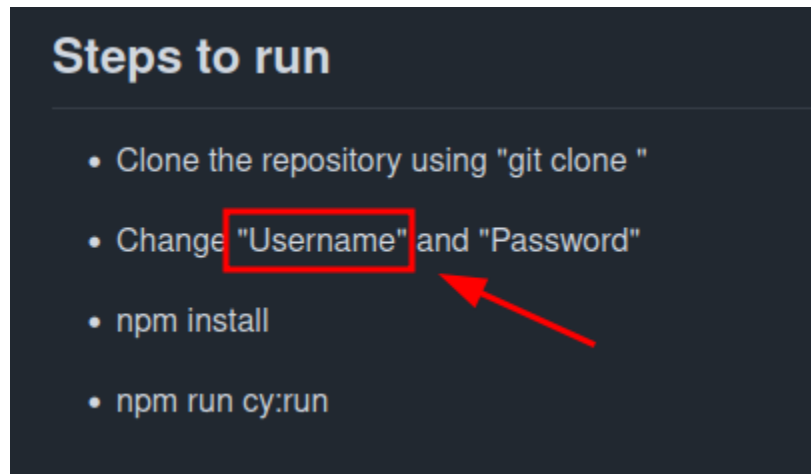
```
1  {
2    "valid_credentials": {
3      "emailId": "testautomation@cypresstest.com",
4      "password": "Test@1234"
5    },
6    "invalid_credentials": {
7      "invalid_email": {
8        "emailId": "invalidUser@cypresstest.com",
9        "password": "Test@1234"
10     },
11     "invalid_password": {
12       "emailId": "testautomation@cypresstest.com",
13       "password": "test@12345"
14     },
15     "wrong_email_format": {
16       "emailId": "testautomationresstest.com",
17       "password": "test@12345"
18     }
19   }
20 }
```



```
{
  "valid_credentials": {
    "emailId": "testautomation@cypresstest.com",
    "password": "Test@1234"
  },
  "invalid_credentials": {
    "invalid_email": "invalidUser@cypresstest.com",
    "wrong_email_format": "testautomationresstest.com",
    "invalid_password": "test@12345"
  }
}
```



11. The README.md file describes the Username field, which is not used in the tests:



12. The README.md file is too short, you can add more details. A good practice is to add the node, cypress, and allure versions.