Created by: Krotchenko Nataliia

Introduction. The purpose of this document is to validate the Cypress end-to-end test project code. Checking typical errors found in the code.

Common Mistakes:

1. All Locators are Initialized in the Constructor of the File

Problem: Locators are initialized in the class constructor, which can cause issues if the DOM tree is reloaded during tests.

File: loginPage.ts

Instead of:

```
class LoginPage {
    get signinLink() { return cy.get('.login') }
    get emailAddressTxt() { return cy.get('#email') }
    get passwordTxt() { return cy.get('#passwd') }
    get signinBtn() { return cy.get('#SubmitLogin') }
    get alertBox() { return cy.get('p:contains("error")')}
    get alertMessage() { return cy.get('.alert-danger > ol > li') }
}
```

Use next:

```
const signinLink = '.login';
const emailAddressTxt = '#email';
const passwordTxt = '#passwd';
const signinBtn = '#SubmitLogin';
const alertBox = 'p:contains("error")';
const alertMessage = '.alert-danger > ol > li';
```

File: myAccountPage.ts

Instead of:

```
class MyAccountPage {
    get signoutLink() { return cy.get('.logout') }
    get pageHeading() { return cy.get('.page-heading') }
}
```

Use next:

```
const signoutLink = '.logout';
const pageHeading = '.page-heading';
```

2. Using the Same Testing Data for Fields

Problem: Using always the same data for tests, which can lead to the Pesticide Paradox

File: login.test.ts

Instead of:

```
    it('login with invalid email credentials read data from fixture', function ()
{
```

```
        loginPage.login(this.data.invalid_credentials.invalid_email.emailId,
            this.data.invalid_credentials.invalid_email.password)
        loginPage.validateLoginError('Authentication failed.')
    })
    it('login with invalid password credentials read data from fixture', function
() {
        loginPage.login(this.data.invalid_credentials.invalid_password.emailId,
            this.data.invalid_credentials.invalid_password.password)
        loginPage.validateLoginError('Authentication failed.')
    })
```

Use next:

```
it('login with invalid email credentials read data from fixture', function () {
    cy.generateRandomText(7).then((randomText) => {
        loginPage.login(randomText, this.data.valid_credentials.password);
      });
    loginPage.validateLoginError('Authentication failed.')
})
it('login with invalid password credentials read data from fixture', function() {
    cy.generateRandomText(7).then((randomText) => {
        loginPage.login(this.data.valid_credentials.emailId, randomText);

     });
        loginPage.validateLoginError('Authentication failed.')
})
```

3. Verification is Absent or Using console.log Instead

Problem: Using console.log for verification is not reliable.

File: myAccount.test.ts

```
it('Sample Test', () => {
        console.log("This is a sample test")
    })
```

4. You also need to add a file random_data.js to generate random data.

For example:

```
Cypress.Commands.add('generateRandomText', (length) => {
    const charset = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
    let randomText = "";

    for (let i = 0; i < length; i++) {
      const randomIndex = Math.floor(Math.random() * charset.length);
      randomText += charset[randomIndex];
    }

    return randomText;
  });
```