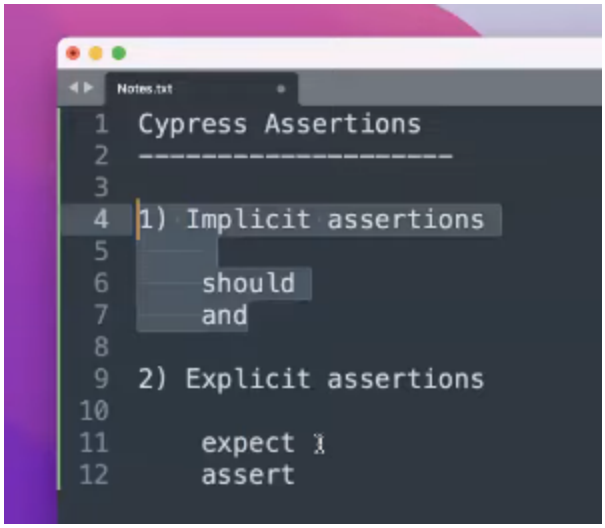
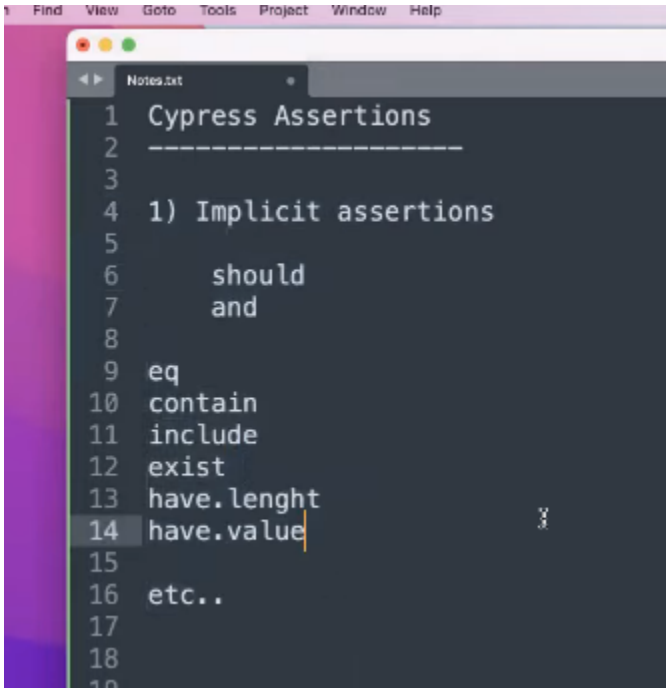


Implicit already present in cypress internally. Default assertions.
Implicit and explicit both comes from chai library.



Expect is bdd style assertions.
Assert is tdd style assertions.
In bdd and tdd - capture the actual value, use js functions to validate the captured value with expected value.

```
describe("Assertions demo", ()⇒{

    it("Implicit assertions", ()⇒{

        cy.visit("https://opensource-demo.orangehrmlive.com/web/index.php/auth/login")

        // should   and
        //cy.url().should('include','orangehrmlive.com')
       // cy.url().should('eq','https://opensource-demo.orangehrmlive.com/web/index.php/auth/login')
       // cy.url().should('contain','orangehrm')

       //multiple assertions.
       //multiple should will also work.
       /*cy.url().should('include','orangehrmlive.com')
       .should('eq','https://opensource-demo.orangehrmlive.com/web/index.php/auth/login')
       .should('contain','orangehrm')*/

       //multiple assertions.
       //instead of should repeating use and.
       cy.url().should('include','orangehrmlive.com')
       .and('eq','https://opensource-demo.orangehrmlive.com/web/index.php/auth/login')
       .and('contain','orangehrm')
       .and('not.contain','greenhrm')

        cy.title().should('include','Orange')
        .and('eq', "OrangeHRM")
        .and('contain',"HRM")

        cy.get('.orangehrm-login-branding > img').should('be.visible')  //Logo  pvisible
        .and('exist')   // logo exist

        cy.xpath("//a").should('have.length','5')  // No of links

        cy.get("input[placeholder='Username']").type("Admin") // provide a value into inputbox

        //check if the value we enter in text box is the same as we enter.
        cy.get("input[placeholder='Username']").should('have.value','Admin') //value
    })
```

```
it("explicit assertions", ()⇒{

        cy.visit("https://opensource-demo.orangehrmlive.com/web/index.php/auth/login")

        cy.get("input[placeholder='Username']").type("Admin")
        cy.get("input[placeholder='Password']").type("admin123")
        cy.get("button[type='submit']").click()

        //for explicit assertions we need to use functions and capture the value and then
        //assert.

        //this is expected result.
        let expName="Paul Collings";

        //capture the user name from the right hand side corner
        cy.get(".oxd-userdropdown-name").then( (x)⇒{

            //get the text of the variable.
            //this is actual result.
                let actName=x.text()

                //BDD Style

                //use actual name with expect.
                //then we compare with expected result.
                expect(actName).to.equal(expName)
                expect(actName).to.not.equal(expName)

                //TDD Style

                //first actual result, then expected result.
                //assert keyword is used.
                assert.equal(actName,expName)
                assert.notEqual(actName,expName)
           })

    })

})
```