

Nama : Nissa `Uzzulfa
Kelas : QE B

Penjelasan fitur mobile testing pada aplikasi Login dan Register.

Berikut adalah script Login dan Register feature.

Feature: As an user, I want to success login, So that i can see my home page

#TESTCASE-1

@TestCase-LoginWrongValue

Scenario: As an user, i cannot login because wrong email or password

Given user on the login page

When user input "yulastari74@gmail.com" on email field

And user input "password" on password field

And user click login button

Then user see error message "Wrong Email or Password"

Pada test case ini menjelaskan bahwa user tidak dapat login jika terdapat pengisian email atau password yang salah.

Berikut adalah hasil report pengujian pada test case tersebut :

As An User, I Want To Success Login And Success Register, So That I Can See My Home Page			Testcase-Loginwrongvalue (tag)
✓ As an user, i cannot login because wrong email or password			
Steps	Outcome		
Given user on the login page	SUCCESS	24s	497ms
✓ Is on login page	SUCCESS	23s	764ms
When user input "yulastari74@gmail.com" on email field	SUCCESS	359ms	
✓ Input email: yulastari74@gmail.com	SUCCESS	345ms	
And user input "password" on password field	SUCCESS	715ms	
✓ Input password: password	SUCCESS	709ms	
And user click login button	SUCCESS	855ms	
✓ Click login button	SUCCESS	849ms	
Then user see error message "Wrong Email or Password"	SUCCESS	903ms	
✓ Get error message	SUCCESS	896ms	
	SUCCESS	27.58s	

Berikut adalah LoginSteps yang diimplementasikan pada feature diatas :

```
public class LoginSteps {  
  
    @Steps  
    LoginScreen loginScreen;
```

```

@Given("user on the login page")
public void userOnTheLoginPage() {
    Assert.assertTrue(loginScreen.isOnLoginPage());
}

@When("user input {string} on email field")
public void inputEmail(String email) {
    loginScreen.inputEmail(email);
}

@And("user input {string} on password field")
public void inputPassword(String password) {
    loginScreen.inputPassword(password);
}

@And("user click login button")
public void clickLoginButton() {
    loginScreen.clickLoginButton();
}

@Then("user see error message {string}")
public void userSeeErrorMessage(String message) {
    Assert.assertEquals(message, loginScreen.getErrorMessage());
}

```

Kelas ini terdiri dari beberapa metode yang masing-masing merepresentasikan satu langkah dalam skenario uji, seperti berikut:

- `userOnTheLoginPage()`: Metode ini mengecek apakah pengguna berada di halaman login atau tidak.
- `inputEmail(String email)`: Metode ini mengisi kolom email dengan nilai email yang diberikan.
- `inputPassword(String password)`: Metode ini mengisi kolom password dengan nilai password yang diberikan.
- `clickLoginButton()`: Metode ini mengklik tombol login.
- `userSeeErrorMessage(String message)`: Metode ini memastikan bahwa pesan error yang muncul di layar sama dengan pesan error yang diharapkan.

#TESTCASE-2

@TestCase-LoginValidValue

Scenario: As an user, i can login with a valid email and valid password

Given user on the login page

When user input "nissa@gmail.com" on email field

And user input "Ridwan123" on password field

And user click login button

Then user see home page

Berikut adalah LoginSteps yang diimplementasikan pada feature diatas :

```
public class LoginSteps {  
  
    @Steps  
    LoginScreen loginScreen;  
  
    @Given("user on the login page")  
    public void userOnTheLoginPage() {  
        Assert.assertTrue(loginScreen.isOnLoginPage());  
    }  
    @When("user input {string} on email field")  
    public void inputEmail(String email) {  
        loginScreen.inputEmail(email);  
    }  
  
    @And("user input {string} on password field")
```

```

public void inputPassword(String password) {
    loginScreen.inputPassword(password);
}

@And("user click login button")
public void clickLoginButton() {
    loginScreen.clickLoginButton();
}

@Then("user see home page")
public void userSeeHomePage() {
    Assert.assertTrue(loginScreen.homePage());
}

```

Kelas ini terdiri dari beberapa metode yang masing-masing merepresentasikan satu langkah dalam skenario uji, seperti berikut:

- `userOnTheLoginPage()`: Metode ini mengecek apakah pengguna berada di halaman login atau tidak.
- `inputEmail(String email)`: Metode ini mengisi kolom email dengan nilai email yang diberikan.
- `inputPassword(String password)`: Metode ini mengisi kolom password dengan nilai password yang diberikan.
- `clickLoginButton()`: Metode ini mengklik tombol login.
- `userSeeHomePage(String message)`: Metode ini memastikan dan memvalidasi jika pengisian email dan password valid maka akan menampilkan ke halaman home

#TESTCASE-3

@TestCase-RegisterValidValue

Scenario: As an user, i can register with a valid value

Given user on the login page

When user click Create One button

And user input name "Nisa" on name field

And user input email "nisa@gmail.com" on email field

And user input password "123456" on password field

And user input confirm password "123456" on confirm password field

And user click register button

Then user see validate message "Registration Successful"

Pada test case ini menjelaskan bahwa user dapat register akun dengan melengkapi pengisian semua field

Berikut adalah hasil report pengujian pada test case tersebut :

As An User, I Want To Success Login And Success Register, So That I Can See My Home Page			Testcase-RegisterValidValue (tag)
As an user, i can register with a valid value			
Steps	Outcome		
+ Given user on the login page	SUCCESS	27s	424ms
✓ Is on login page	SUCCESS	27s	403ms
+ When user click Create One button	SUCCESS	25s	908ms
✓ Click create button	SUCCESS	24s	531ms
+ And user input name "Aulia" on name field	SUCCESS	1s	689ms
✓ Input name: Aulia	SUCCESS	1s	680ms
+ And user input email "aulia@dummy.com" on email field	SUCCESS	599ms	
✓ Input email: aulia@dummy.com	SUCCESS	591ms	
+ And user input password "123456" on password field	SUCCESS	643ms	
✓ Input password: 123456	SUCCESS	632ms	
+ And user input confirm password "123456" on confirm password field	SUCCESS	644ms	
✓ Input confirm password: 123456	SUCCESS	629ms	
+ And user click register button	SUCCESS	888ms	
✓ Click register button	SUCCESS	876ms	
+ Then user see validate message "Registration Successful"	SUCCESS	1s	106ms
✓ Get validate message	SUCCESS	1s	099ms
	SUCCESS	58.94s	

Berikut adalah RegisterSteps yang diimplementasikan pada feature diatas :

```
public class RegisterSteps {  
  
    @Steps  
    RegisterScreen registerScreen;
```

```

@When("user click Create One button")
public void ClickCreateButton() {
    registerScreen.clickCreateButton();
}

@And("user input name {string} on name field")
public void InputName(String Name) {
    registerScreen.InputName(Name);
}

@And("user input email {string} on email field")
public void InputEmail(String Email) {
    registerScreen.InputEmail(Email);
}

@And("user input password {string} on password field")
public void InputPassword(String Password) {
    registerScreen.InputPassword(Password);
}

@And("user input confirm password {string} on confirm password field")
public void InputConfirmPassword(String ConfirmPassword) {
    registerScreen.InputConfirmPassword(ConfirmPassword);
}

@And("user click register button")
public void ClickRegisterButton() {
    registerScreen.ClickRegisterButton();
}

@Then("user see validate message {string}")
public void userSeeValidateMessage(String validate) {
    Assert.assertEquals(validate, registerScreen.getValidateMessage());
}

```

Kelas ini terdiri dari beberapa metode yang masing-masing merepresentasikan satu langkah dalam skenario uji, seperti berikut:

- `ClickCreateButton()`: Metode ini mengklik tombol "Create One" untuk membuka halaman registrasi.
- `InputName(String Name)`: Metode ini mengisi kolom "name" dengan nilai nama yang diberikan.
- `InputEmail(String Email)`: Metode ini mengisi kolom "email" dengan nilai email yang diberikan.
- `InputPassword(String Password)`: Metode ini mengisi kolom "password" dengan nilai password yang diberikan.
- `InputConfirmPassword(String ConfirmPassword)`: Metode ini mengisi kolom "confirm password" dengan nilai password konfirmasi yang diberikan.
- `ClickRegisterButton()`: Metode ini mengklik tombol "register" untuk menyelesaikan proses registrasi.

- `userSeeValidateMessage(String validate)`: Metode ini memastikan bahwa pesan validasi yang muncul di layar sama dengan pesan validasi yang diharapkan.

#TESTCASE-4

@TestCase-RegisterEmptyName

Scenario: As an user, i cannot register because empty name field
Given user on the login page
When user click Create One button
And user input name " " on name field
And user input email "nisa@dummys.com" on email field
And user input password "123456" on password field
And user input confirm password "123456" on confirm password field
And user click register button
Then user see error alert message "Enter Full Name"

Pada test case ini menjelaskan bahwa user dapat register akun dengan mengosongkan pengisian field Name

Berikut adalah hasil report pengujian pada test case tersebut :

As An User, I Want To Success Login And Success Register, So That I Can See My Home Page			Testcase-Registeremptyname (tag)
As an user, i cannot register because empty name field			
Steps	Outcome		
+ Given user on the login page	SUCCESS	28s 100ms	
✓ Is on login page	SUCCESS	28s 072ms	
+ When user click Create One button	SUCCESS	22s 923ms	
✓ Click create button	SUCCESS	22s 890ms	
+ And user input name " " on name field	SUCCESS	1s 292ms	
✓ Input name:	SUCCESS	1s 278ms	
+ And user input email "aulia@dummys.com" on email field	SUCCESS	657ms	
✓ Input email: aulia@dummys.com	SUCCESS	641ms	
+ And user input password "123456" on password field	SUCCESS	624ms	
✓ Input password: 123456	SUCCESS	611ms	
+ And user input confirm password "123456" on confirm password field	SUCCESS	626ms	
✓ Input confirm password: 123456	SUCCESS	615ms	
+ And user click register button	SUCCESS	1s 456ms	
✓ Click register button	SUCCESS	1s 430ms	
✓ Then user see error alert message "Enter Full Name"	SUCCESS	2s 203ms	
	SUCCESS	57.94s	

Berikut adalah RegisterSteps yang diimplementasikan pada feature alert message "Enter Valid Full Name" diatas :

```
@Then("user see error alert message {string}")
public void userSeeErrorAlertMessage(String alert) {
    Assert.assertEquals(alert, registerScreen.getAlertMessage());
}
```


#TESTCASE-5

@TestCase-RegisterEmptyEmail

Scenario: As an user, i cannot register because empty email field

Given user on the login page

When user click Create One button

And user input name "nisa" on name field

And user input email " " on email field

And user input password "123456" on password field

And user input confirm password "123456" on confirm password field

And user click register button

Then user see alert message "Enter Valid Email"

Pada test case ini menjelaskan bahwa user dapat register akun dengan mengosongkan pengisian field Email

Berikut adalah hasil report pengujian pada test case tersebut :

As An User, I Want To Success Login And Success Register, So That I Can See My Home Page			Testcase-Registeremptyemail (tag)
As an user, i cannot register because empty email field			
Steps	Outcome		
+ Given user on the login page	SUCCESS	22s	159ms
✓ Is on login page	SUCCESS	22s	131ms
+ When user click Create One button	SUCCESS	22s	883ms
✓ Click create button	SUCCESS	22s	852ms
+ And user input name "aulia" on name field	SUCCESS	1s	740ms
✓ Input name: aulia	SUCCESS	1s	731ms
+ And user input email " " on email field	SUCCESS	658ms	
✓ Input email:	SUCCESS	653ms	
+ And user input password "123456" on password field	SUCCESS	704ms	
✓ Input password: 123456	SUCCESS	698ms	
+ And user input confirm password "123456" on confirm password field	SUCCESS	697ms	
✓ Input confirm password: 123456	SUCCESS	685ms	
+ And user click register button	SUCCESS	831ms	
✓ Click register button	SUCCESS	827ms	
✓ Then user see alert message "Enter Valid Email"	SUCCESS	2s	484ms
	SUCCESS	52.2s	

Berikut adalah RegisterSteps yang diimplementasikan pada feature alert message "Enter Valid Email" diatas :

```
@Then("user see alert message {string}")
public void userSeeAlertMessage(String alertMessage) {
    Assert.assertEquals(alertMessage, registerScreen.getAlert());
}
```