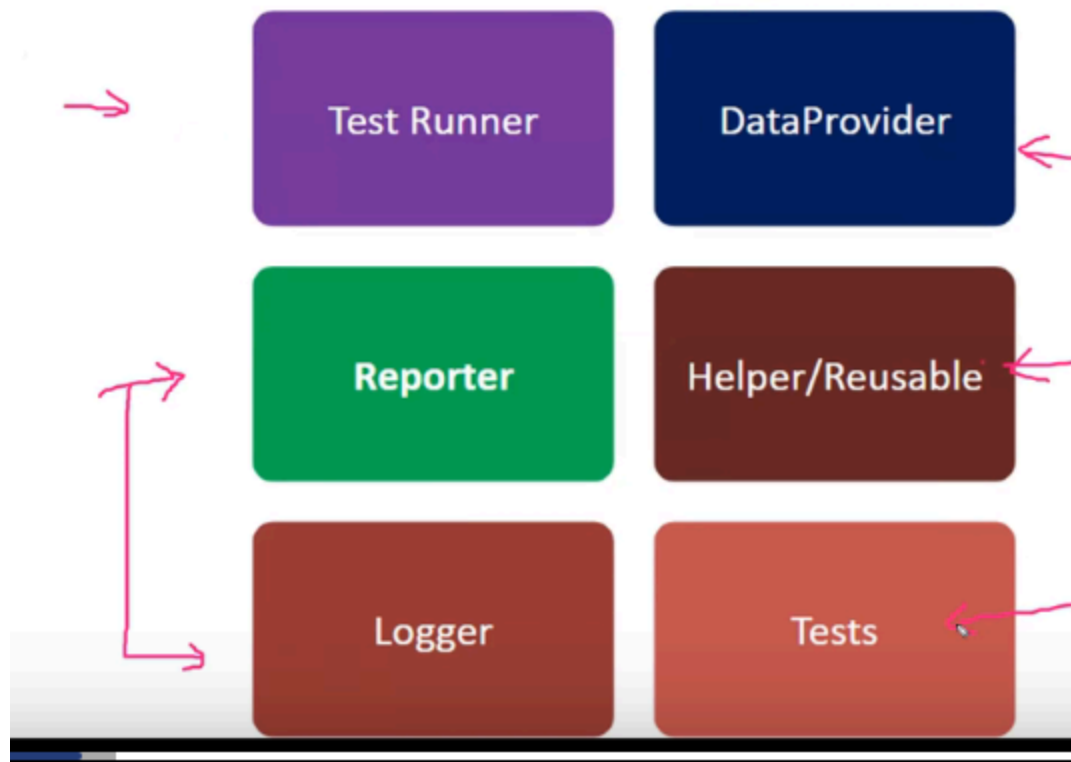


## Basic framework design



Import {login} from “../pages/[login.cy.js](#)”

.. means go to the root directory.

Pom working—  
Login page class

```
1 class login{
2
3     enterusername(){
4         return cy.get('#email1')
5     }
6
7     enterpassword(){
8         return cy.get('#password1')
9     }
10
11    clickonloginbutton(){
12        return cy.get("button[type='submit']")
13    }
14
15    //more optimised way to club methods
16    loginIntoApplication(username, password){
17        cy.get('#email1').type(username)
18        cy.get('#password1').type(password)
19        cy.get("button[type='submit']").click()
20    }
21 }
22
23 //see how to export
24 export default login;
```

codesnap.dev

Another login page class created but with get and set methods-

```

1 class login{
2
3     //use get and set directly
4
5     get enterUserName(){
6 return cy.get('#email1')
7     }
8
9     //how to write above method with set.
10    set enterUserNameWithSet(email){
11 return cy.get('#email1').type(email)
12    }
13
14
15    get enterPassword(){
16        return cy.get('#password1')
17    }
18
19    //how to write above method with set.
20    set enterPasswordWithSet(password){
21 return cy.get('#password1').type(password)
22    }
23
24    get clickOnLoginButton(){
25        return cy.get("button[type='submit']")
26    }
27
28    //how to write above method with set.
29    //this is not allowed in set.
30    //compile error - A 'set' accessor must have exactly one parameter.ts(1049)
31
32    //    set clickOnLoginButtonWithSet(){
33    // return cy.get("button[type='submit']").click()
34    //    }
35
36    set clickOnLoginButtonWithSet(locator){
37 return cy.get(locator).click()
38    }
39 }
40
41 //see how to export
42 export default login;

```

codesnap.dev

Login test class-

Dividing the code as not completely pastable.

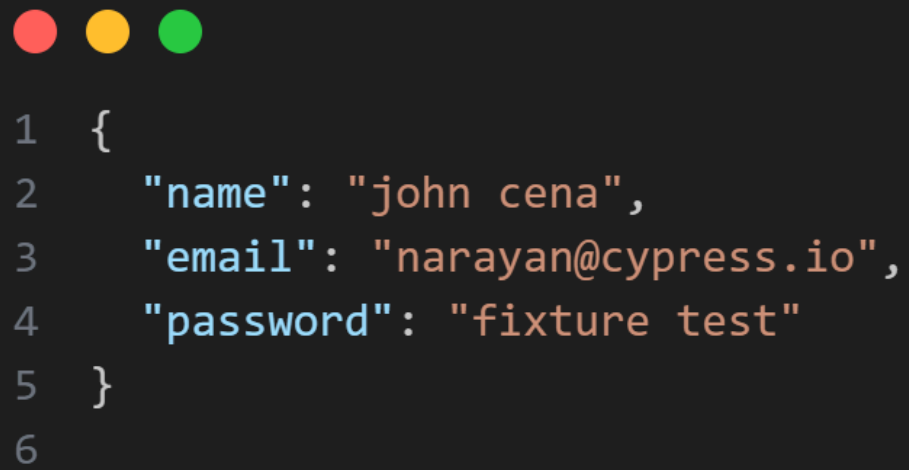
```
1 import loginpage from "../pages/login.cy.js"
2 import loginpage1 from "../pages/loginwithgettersetter.cy.js"
3 import newuserpage from "../pages/registration.cy.js"
4
5 describe("login with pom", function(){
6     it("valid login", function(){
7
8         //create object of login page class.
9         //here if we use same name login for the page class, for the object, for the import.
10        //we get error as Cannot access 'login' before initialization
11        //js is confused what this login refers to.
12        //so when importing change some name and then use that name for object creation.
13
14        const login=new loginpage();
15
16        //visit url
17        cy.visit("https://ineuron-courses.vercel.app/login")
18
19        //enter user name password and click login
20        login.enterusername().type("ineuron@ineuron.ai")
21        login.enterpassword().type("ineuron")
22        login.clickonloginbutton().click()
23
24
25    })
26 })
```

```
1  it("valid login functionality all steps clubbed into one method", function(){
2
3      //call the clubbed method at once.
4
5      const login=new loginpage();
6
7      //visit url
8      cy.visit("https://ineuron-courses.vercel.app/login")
9
10     //enter user name password and click login
11     login.loginIntoApplication("dummy@gmail.com", "dummy")
12
13
14 })
15
16
17 it("login with getter and setter", function(){
18
19     const login=new loginpage1();
20
21     //visit url
22     cy.visit("https://ineuron-courses.vercel.app/login")
23
24     //enter user name password and click login
25     //you dont need parenthesis next to method names
26     login.enterUserName.type("mukesh@gmail.com")
27     login.enterPassword.type("tester")
28     login.clickOnLoginButton.click()
29
30
31 })
```



```
1  it("login with setter only", function(){
2
3      const login=new loginpage1();
4
5      //visit url
6      cy.visit("https://ineuron-courses.vercel.app/login")
7
8      //how to use setter now.
9
10     login.enterUserNameWithSet="tiger@gmail.com"
11     login.enterPasswordWithSet="programmer"
12     login.clickOnLoginButtonWithSet="button[type='submit']"
13
14
15 })
16
17
18 it("register new user on the page", function(){
19
20     const newuser=new newuserpage();
21
22     //visit url
23     cy.visit("https://ineuron-courses.vercel.app/login")
24
25     //click on new user link
26     newuser.clickonnewuserlink().click()
27
28     //enter first name
29     newuser.enterfirstName().type("karan tiger")
30
31
32
33
34 })
35 })
```

Fixture data -



```
1  {  
2    "name": "john cena",  
3    "email": "narayan@cypress.io",  
4    "password": "fixture test"  
5  }  
6
```


Register page class-



```
1  class newuser{
2
3      clickonnewuserlink(){
4          return cy.contains("New user? Signup")
5      }
6
7      enterfirstName(name){
8          return cy.get('#name').type(name);
9      }
10
11     enterEmail(email){
12         return cy.get('#email').type(email);
13     }
14
15     enterPassword(password){
16         return cy.get('#password').type(password);
17     }
18 }
19
20 export default newuser;
```

Test class-





```
1 import registrationpage from "../pages/registration.cy.js"
2
3
4 //working with fixture data.
5 describe("login with pom with fixture", function(){
6
7     //we need before each to load the fixture data.
8     //pass json file name, extension not mandatory.
9     beforeEach("load fixture data", function(){
10         cy.fixture("pomframework.json").then(function(data){
11             this.data=data;
12
13         })
14     })
15     it("valid login", function(){
16
17         const newuser=new registrationpage();
18
19         //visit url
20         cy.visit("https://ineuron-courses.vercel.app/login")
21
22         //click on new user link
23         newuser.clickonnewuserlink().click()
24
25         //enter first name
26         newuser.enterfirstName(this.data.name);
27
28         //enter email
29         newuser.enterEmail(this.data.email)
30
31         //enter password
32         newuser.enterPassword(this.data.password)
33
34     })
35
36 })
37
```

```
//we can have such validations also in test class
newuser.clickonnewuserlink().should("be.visible").click()
```

For api assertions use expect not should.

Should be used with webelements like cy.get().should(have something).

For put post, delete, patch just send body like this –

```
4
5     cy.request({
6       method: 'DELETE',
7       url: "https://gorest.co.in/public/v2/users/4434",
8       headers: {
9         "Authorization": "Bearer 7a99fa78ea0075bf8f8105a75e72e9539fd169ddab9327297f1809a8c5c24240"
10      },
11       data: {}
12     })
13
14     }).then(function (response) {
```


Api code block by block –




```
1
2 describe("get all users" , function(){
3
4     it("verify status code", function(){
5
6         //request for api testing.
7         //if we dont pass which method, it will take as get.
8         //we can pass explicitly also using method.
9
10        cy.request({
11            method: 'GET',
12            url: "https://restful-booker.herokuapp.com/booking",
13            // headers: {
14            //     "Authorisation": "Bearer 34234324weewrewrewrw"
15            // }
16        })
17        .then(function(data){
18            cy.log(data) //this will print object{8} means there is object with 8 entries.
19
20            //get the response in string format.
21            cy.log(JSON.stringify(data))
22        })
23    })
24 }
```




```
1  it("verify status code with stringify", function(){
2
3      //request for api testing.
4      //if we dont pass which method, it will take as get.
5      //we can pass explicitly also using method.
6
7      cy.request({
8          method: 'GET',
9          url: "https://restful-booker.herokuapp.com/booking",
10         // headers: {
11             //     "Authorisation": "Bearer 34234324weewrewrewr"
12         // }
13     })
14     .then(function(data){
15
16         //get the response in string format.
17         cy.log(JSON.stringify(data))
18     })
19 })
```



```
1  it("assert should be done with expect not should as no webelement", function(){
2
3      //request for api testing.
4      //if we dont pass which method, it will take as get.
5      //we can pass explicitly also using method.
6
7      cy.request({
8          method: 'GET',
9          url: "https://restful-booker.herokuapp.com/booking",
10         // headers: {
11         //     "Authorisation": "Bearer 34234324weewrewrewr"
12         // }
13     })
14     .then(function(data){
15
16         //check status code
17         expect(data.status).to.eq(200)
18     })
19 })
```



```
1  it("verify response for one particular user", function(){
2
3      //request for api testing.
4      //if we dont pass which method, it will take as get.
5      //we can pass explicitly also using method.
6
7      cy.request({
8          method: 'GET',
9          url: "https://restful-booker.herokuapp.com/booking/441",
10         // headers: {
11             //     "Authorisation": "Bearer 34234324weewrewnewrw"
12         // }
13     })
14     .then(function(data){
15
16         //check status code
17         expect(data.status).to.eql(200)
18     })
19 })
```



```
1  it("verify the last name for an user", function(){
2
3      //request for api testing.
4      //if we dont pass which method, it will take as get.
5      //we can pass explicitly also using method.
6
7      cy.request({
8          method: 'GET',
9          url: "https://restful-booker.herokuapp.com/booking/441",
10         // headers: {
11             //     "Authorisation": "Bearer 34234324weewrewnewrw"
12         // }
13     })
14     .then(function(data){
15
16         //check if response has property called status
17         expect(data.body).has.property("lastname")
18     })
19 })
```



```
1  it("verify the property and value for an user", function(){
2
3      //request for api testing.
4      //if we dont pass which method, it will take as get.
5      //we can pass explicitly also using method.
6
7      cy.request({
8          method: 'GET',
9          url: "https://restful-booker.herokuapp.com/booking/202",
10         // headers: {
11             //     "Authorisation": "Bearer 34234324weewrewrewrw"
12         // }
13     })
14     .then(function(data){
15
16         cy.log(JSON.stringify(data))
17
18         //verify the property and value
19         expect(data.body).has.property("firstname", "John")
20     })
21 })
22 })
```