IFoundMovie
Code Test Plan and Result

Xufeng Liu
Yixu Zhou

Table of Contents

1.0 Summary

The IFoundMovie website can show our users information about current and upcoming movies.And our users can go to the detail page of the movie which they just choose. What's more, the user can comment on the detail page and even share the movie to their friends. In this report I will test if all of the information can be displayed normally, and all of the functions can meet our expectations. I mainly use unit testing and integration testing in this report. There will be a total of 9 unit tests and 14 integration tests.

2.0 Test Method

The first method we used is unit testing. A unit test is the way to test a unit such as a function, a subroutine, a method or property. It is easy to use and we can easily see the result of the test. If we find any function that didn't succeed we just know and change it immediately.

The other method we used is integration testing. Integration testing involves checking individual components or units of a software project to expose defects and problems to verify that they work together as designed. It is also easy for us to see the result.

3.0 Test Plan

For the unit testing, first of all, we will test the function of login and registration. Then we will test if the main page can display the TMDB movie information. In the next step, we will test if we can open the popular recommended movies list page and the detail page.Finally, we will test the search and the comment function.

For the integration testing, we will also test some functions same with unit testing, but we need to see if it can display normally on our website. Moreover, we will test some more detailed things. For example, if some buttons can jump to the exact page.

4.0 Test cases and Results

Unit Testing:

The below image shows the test for registration function.

```
it("Test registration function:", async() => {
    const login=await axios({
        url:Py_URL+"/register",
        method:"POST",
        data:{username:userInfo.username,password:userInfo.password},
        headers: {
            'content-type': 'application/x-www-form-urlencoded'
        },
    });
    expect(login.data.message).to.be.equal("Registered user successfully");
    expect(login.data.status_code).to.be.equal(200);
```

The below image shows the test for login function.

```
it("Test login function:", async() => {
    const login=await axios({
        url:Py_URL+"/login",
        method:"POST",
        data:{username:userInfo.username,password:userInfo.password},
        headers: {
            'content-type': 'application/x-www-form-urlencoded'
        },
    });
    expect(login.data.message).to.be.equal("successfully login");
    expect(login.data.status_code).to.be.equal(200);
```

The below image shows the test for if the home page can display normally.

```
it("Open the home page", async() => {
    const home=await axios({
        url:API_URL+"/",
        method:"get",
        headers: {
            'content-type': 'application/x-www-form-urlencoded'
        },
    });
    expect(home.status).to.be.equal(200);
```

The below image shows if the popular movies list can be opened.

```
it("Open the Popular Recommended Movies List page", async() => {
    const popularList=await axios({
        url:API_URL+"/popularList",
        method:"get",
        headers: {
            'content-type': 'application/x-www-form-urlencoded'
        },
    });
    expect(popularList.status).to.be.equal(200);
});
```

The below image shows if the detail page of the popular movies list can be opened.

```
it("Open the Popular Recommended Movies Detail page", async() => {
    const proplueDetail=await axios({
        url:API_URL+"/proplue?proplue_id="+movie.proplue_id,
        method:"get",
        headers: {
            'content-type': 'application/x-www-form-urlencoded'
        },
    });
    expect(proplueDetail.status).to.be.equal(200);
});
```

The below image shows if the search function works well.

```
it("Open the movie search information page and search for information", async() => {
    const searchMovie=await axios({
        url:API_URL+"/search/"+movie.search_name,
        method:"get",
        headers: {
            'content-type': 'application/x-www-form-urlencoded'
        },
    });
    expect(searchMovie.status).to.be.equal(200);
});
```

The below image shows if the detail page after search can be opened.

```
it("Open the Movies Detail page", async() => {
    const movieDetail=await axios({
        url:API_URL+"/movie?movie_id="+movie.movie_id,
        method:"get",
        headers: {
            'content-type': 'application/x-www-form-urlencoded'
        },
    });
    expect(movieDetail.status).to.be.equal(200);
});
```

The below image shows if the comment function works well.

```
describe("Test comment function:", async() => {
    it("Query all reviews of the current movie:", async() => {
        const movieDetail=await axios({
            url:Py_URL+"/comments/"+movie.movie_id,
            method:"get",
            headers: {
                'content-type': 'application/x-www-form-urlencoded'
            },
        });
        expect(movieDetail.data.message).to.be.equal("successfully");
        expect(movieDetail.data.status_code).to.be.equal(200);
    });

    it("Test the new comment feature:", async() => {
        var date=new Date();
        var nowdate=date.getFullYear()+"-"+(date.getMonth()+1)+"-"+date.getDate()+" "+date.getHours()+":"+date.getMinutes()+":"+date.getSeconds();
        const login=await axios({
            url:Py_URL+"/login",
            method:"POST",
            data:{username:userInfo.username,password:userInfo.password},
            headers: {
                'content-type': 'application/x-www-form-urlencoded'
            },
        });
        const commentAdd=await axios({
            url:Py_URL+"/comment",
            method:"POST",
            headers: {
                'content-type': 'application/x-www-form-urlencoded',
                "Authorization":"Bearer "+login.data.data.token,
            },
            data:{mid:movie.movie_id,content:"test comment",start_time:nowdate}
        });
        expect(commentAdd.data.message).to.be.equal("successfully comment");
        expect(commentAdd.data.status_code).to.be.equal(200);
    });
});
```

After those tests, the result displayed below.

```
test
  Displays the TMDB movie information page:
    ✓ Open the home page (1762ms)
    ✓ Open the Popular Recommended Movies List page (7662ms)
    ✓ Open the Popular Recommended Movies Detail page (616ms)
    ✓ Open the movie search information page and search for information (464ms)
    ✓ Open the Movies Detail page (7418ms)
  Test the login and registration functions:
    ✓ Test registration function:
    ✓ Test login function:
  Test comment function:
    ✓ Query all reviews of the current movie:
    ✓ Test the new comment feature:


 9 passing (18s)
```

Integration Testing:

Test Case	Expected ouput	Test Result
Click the right button in the scroll bar	The movie poster jump to the next one	Passed
Click the left button in the scroll bar	The movie poster jump to the previous one	Passed
Click the number button in the scroll bar	The movie poster jump to that page	Passed
Click the movie poster in the	Display the detail page of	Passed

scroll bar	the movie	
Click the movie poster below the theatrical movie section	Display the detail page of the movie	Passed
Click the movie poster below the upcoming movie section	Display the detail page of the movie	Passed
Click the movie poster below the popular movie section	Display the detail page of the movie	Passed
Click the popular movies lists button	The page jump to that page	Passed
Click the learn more button from any lists	The page jump to the detail page of that list	Passed
Click the Login button	Jumping to the login page	Passed
Click the register button	Jumping to the register button	Passed
Type some words and click search button	Jumping to the search result page of that key words	Passed
Click one of the movie poster from the search result page	Jumping to the detail page of that movie	Passed
Click the share button	copy the current url	Passed

5.0 Conclusion

We used 9 unit tests and 14 integration tests in our testing period, all of the tests were passed.

In the future, if we add any new features in our website we will also test them by using unit testing and integration testing and add to this file.