Table of content

[Install and Uninstall Test 2](#_heading=h.gjdgxs)

[UI JS test 2](#_heading=h.30j0zll)

[CSS backstop 2](#_heading=h.1fob9te)

[Django test 3](#_heading=h.3znysh7)

[Function test 3](#_heading=h.2et92p0)

[Button 3](#_heading=h.tyjcwt)

[QA Code 5](#_heading=h.3dy6vkm)

[Navigation 5](#_heading=h.1t3h5sf)

[QA Code 5](#_heading=h.4d34og8)

[Input 6](#_heading=h.2s8eyo1)

[QA Code 6](#_heading=h.17dp8vu)

[Compatibility test 7](#_heading=h.3rdcrjn)

[User experience test 9](#_heading=h.26in1rg)

[User range 9](#_heading=h.lnxbz9)

[Method 9](#_heading=h.35nkun2)

[Task 9](#_heading=h.1ksv4uv)

[Simple User 12](#_heading=h.44sinio)

[Acceptance test 13](#_heading=h.2jxsxqh)

[Method 13](#_heading=h.z337ya)

[Verifiability Item 13](#_heading=h.3j2qqm3)

[Result 15](#_heading=h.1y810tw)

[Integration &Unit test 16](#_heading=h.4i7ojhp)

[Backend function test 16](#_heading=h.2xcytpi)

[task 16](#_heading=h.1ci93xb)

[Summary Report 20](#_heading=h.3whwml4)

[Frontend function test report 21](#_heading=h.2bn6wsx)

[UI test 22](#_heading=h.qsh70q)

[login 22](#_heading=h.3as4poj)

[Register 22](#_heading=h.1pxezwc)

[New project 22](#_heading=h.49x2ik5)

[User Profile 23](#_heading=h.2p2csry)

[Project list 23](#_heading=h.147n2zr)

[View project 23](#_heading=h.3o7alnk)

# Install and Uninstall Test

## UI JS test

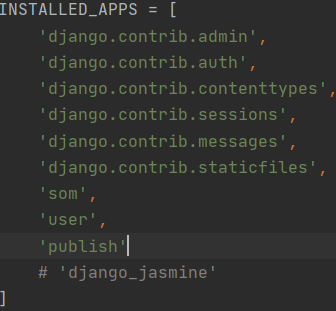
Step1: install the jasmine package into the server

*pip install Django-jasmine*

*Pip uninstall Django-jasmine <uninstall>*

Step2: enable package under the project

* Add APP to Setting



* Add JASMINE\_TEST\_DIRECTORY



* Enable URL from project. URLs



Step3: access URL in bowser

*Ex. Localhost:8080/jasmine-test-suite/*

## CSS backstop

Step1: install the backstop application into the desktop

*npm install -g backstopjs*

Step2: initialize the project

*backstop init*

Step3: modify the url under generated json

*To UI url*

Step 4: Approve the file as reference

*backstop approve*

Step 5: modify the url under generated json

*To web url*

Step 6: test the page

*backstop test*

## Django test

Following command under server terminal manually running the test

cd <project dir –SOM\_PROJECT>

*python manage.py test*

# Function test

Test whether the functions of all buttons meet the expectations. The specific method is to reach the conditions of clicking the buttons, and then click the buttons to test whether the expected effects can be produced.

## Button

|  |  |  |  |
| --- | --- | --- | --- |
| Button | Location | Method | Result |
| Login | Login page | Input correct user email and password, Click button | Login successfully |
| Register now | Login page | Click button | Return to register page |
| Register | Register page | Input unappropriated username, email and adequate password，Click button | Register successfully |
| Login now | Register page | Click button | Return to login page |
| Choose file | Main page | Click button and choose a file from PC | Open File successfully |
| Upload | Main page | After choosing the file, Click the upload button, based on different situations, there are four results. | 1.Upload file successfully.  2.Please select file before upload.  3.Not supporting a given data file.  4. Not valid file. |
| Train | Main page | After upload file，and set parameters，click button | Generate Self-Organizing Map |
| Save | Main page | After train，click button | Open save page |
| Save and publish | Save page | Input name and description, then click save and publish button | Save data to profile and publish page |
| Save | Save page | Input name and description, then click save button | Save data to profile page |
| Edit | Profile page | Click button | Activates boxes for modifying phone numbers, passwords, and user names |
| Update | Profile page | After input adequate phone numbers and user name，Click button | Save successfully |
| Save | Profile page | Input new password twice then click button | Save successfully |
| Your project | Profile page | Click button | Return to all saved projects of current user |
| Next page | Profile page | Click button | Return to next project page |
| Last page | Profile page | Click button | Return to last project page |
| Next page | Publish page | Click button | Return to next publish page |
| Last page | Publish page | Click button | Return to last publish page |
| View | Publish page | Click button | Return to som map page of current record |
| Back | Som map page | Click button | Return to publish page |
| Project name | Profile page | Click button | Return to som map page of current record |

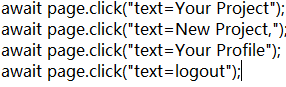
### QA Code



## Navigation

|  |  |  |
| --- | --- | --- |
| Navigation bar | Method | Result |
| User | Click button | Return to profile page |
| New project | Click button | Return to main page |
| Logout | Click button | Return to login page |
| Publish | Click button | Return to publish page |

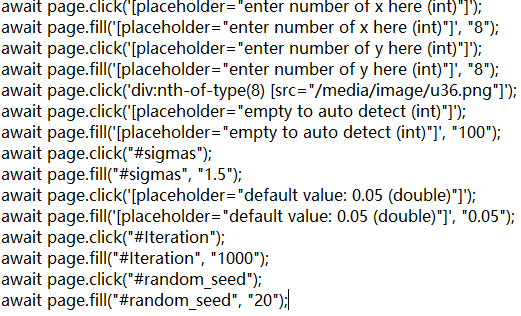
### QA Code



## Input

|  |  |  |
| --- | --- | --- |
| Input Box | Method | Result |
| Email | Attempt to input | Input Successfully |
| Username | Attempt to input | Input Successfully |
| Password | Attempt to input | Input Successfully |
| Personal info | Attempt to input | Input Successfully |
| Change password | Attempt to input | Input Successfully |
| Set parameters | Attempt to input | Input Successfully |

### QA Code



Based on the test, there is no error happens during the process.

# Compatibility test

Verify that users can use our web pages in different browsers.

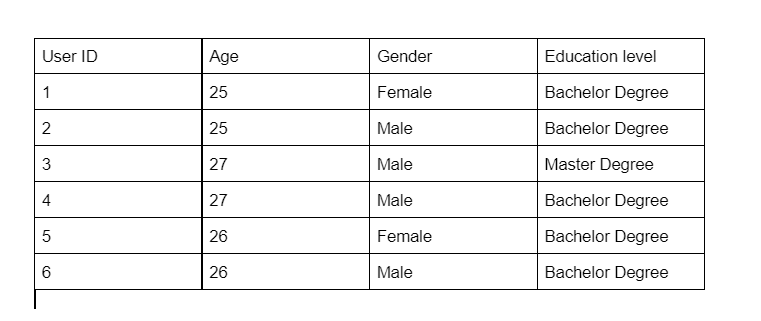
|  |  |  |
| --- | --- | --- |
|  | **Chrome** | **Microsoft Edge** |
| Register |  |  |
| Login |  |  |
| Project list |  |  |
| View Project |  |  |
| User Profile |  |  |
|  |  |  |
| New Project |  |  |
|  |  |  |

After compare, the page can correct render the page under Chrome and Microsoft Edge.

# User experience test

## User range

The simple user test includes 6 users under the age between 25 to 27 with both male and female, their education level above or equal bachelor degree.



## Method

The method here providing user the target task to complete, then record the time, issues and process of the completation.

## Task

Simulate the process of users using our web pages and the problems they may encounter.

|  |  |  |  |
| --- | --- | --- | --- |
| Task ID | Abstract Task | Website | Concrete Task |
| 1 | The user can register a personal account on the website. | Register | Suppose you have opened the “Register” website.  1.Suppose you want to enrol a new account, please Input personal “Username”, “Email”, “Password” to create a personal account.  2.Suppose you have an account and want to login, please switch the Register website to the Login website. |
| 2 | The user can login their personal account on the website. | Login | Suppose you have opened the “Login” website.  1.Suppose you want to login your account, please input “Email” and “Password” to login to your personal account.  2.Suppose you do not have an account or want to create a new one, please switch the Register website to the Login website. |
| 3 | The user can upload data and train Som visualization map. | Main | Suppose you have opened the “Main” website.  1.Suppose you want to train a Som map,  Firstly, please choose a data file and upload it. Secondly, Please choose the parameters in the table of this website.Finally, click the “train” button to get the Som map.  2.Suppose you want to save your training result, please enter visualization name, user name and user description. Suppose you want to publish the result, please click “Save and Publish”. Suppose you want to save your Som map and data privately, please just click the “Save”.  3.Suppose you want to log out of your account, please click the “logout” label next to your account name. |
| 4 | The user can interaction with map | Main | 1.Suppose you want to change the training parameters to see different training results, please change the training parameters in the table and click the “train” button.  2.Suppose you want to choose two node colors that correspond to the maximum and minimum value in the Som map, please choose one as maximum, and another as minimum in the related table.  3.Suppose you want to change the font size of the label in the Som map, please choose in the related table. |
| 5 | The user can view different authors and publish their general information Som-map. | Publish | Suppose you have opened the “Publish” website.  1.Suppose you want to browse the next page of the general information published by users, please click “Next Page”.  2.Suppose you want to view the details of the Som map, please click “view” in the table.  3.Suppose you want to log out of your account, please click the “logout” label next to your account name. |
| 6 | The user can view the detailed information about the Som map. | View | Suppose you have opened the “view” page.  1. Suppose you want to go back to the “publish” website, please click the “Back” button or “house” icon.  2. Suppose you want to create a new project, please click the “New project” label.  3. Suppose you want to log out of your account, please click the “logout” label next to your account name. |

## Simple User

|  |  |  |
| --- | --- | --- |
| User ID | 1 | Gary |
| Task ID | 1,2,3,4,5,6 | |
| Overall  Successful | Mostly successful | |
| Clock Time  11:08  11:09  11:09  11:10  11:11  11:12  11:13  11:14  11:14  11:15  11:15  11:15  11:17  11:18  11:19  11:20  11:20  Total time:  12 mins | At first, the user doesn’t have an account, so he clicks register now button  After register, the web page is automatically converted to the login page  After login, the user comes to the main page  User clicks choose file button  User clicks upload button  User set parameters  User clicks train button  User tries to drag and zoom in the map  User try to save the map  User clicks save button  User inputs name &description  User clicks save and publish button  User try to find publish button, but doesn’t success  User clicks user icon  User clicks edit button  User edit personal information  User click logout | |
| Errors | User can’t find the publish data | |
| Other Notes |  | |

# Acceptance test

## Method

The method acceptance test focus on validate the requirements steps by steps under the client’s operation.

## Verifiability Item

Verify that our project meets the requirements we set for the project in the initial stage.

|  |  |  |  |
| --- | --- | --- | --- |
| Abstract | Location | Method | Acceptance criteria |
| The system should allow user to login. | Login page | Enter the correct email and password and try the wrong input | Successful registered users can log in by entering the correct email and password on the login page. If the mail and password do not match, or if the password or mailbox is empty, they cannot log in |
| The system should allow user to register. | Register page | Enter a new user name, email, and appropriate password to verify that the registration is successful. Enter an existing user name, email, and inappropriate password to verify that the registration fails | Only when the user enters an unregistered email and user name can the registration be successful, and the password needs to meet a certain length and complexity |
| The system should allow user to reset password. | Profile page | Enter the new password twice, verify that the update is successful, enter the old password, or enter the password inconsistently twice, will leads to the update fails | Enter a new password （that is different from the previous one）twice， then the password can be updated successfully |
| The system should allow user to logout. | Main page & Profile page | Click logout button，Verify that the user is logged out | After click Logout, return to the login page |
| The system should allow user to modify profile include name. | Profile page | Click edit button in this page and change name, email, phone number and DOB | The user can modify the name, email, phone number and DOB on this page and satisfy the corresponding format |
| The system should allow user to upload SOM data. | Main page | After choosing file successfully, click upload button | The uploaded data is stored in the Mengo database and can be used for training |
| The system should allow user to train SOM data with application in weights/parameter. | Main page | After uploading file, change parameters in the page, and then, click the train button | All parameters can be input. Different parameter settings will lead to different clustering results, and the clustering results can be displayed |
| The system should allow user to visualize SOM data. | Main page | After training data, map can be displayed | Map with label/legend/color be showing |
| The system should allow user drag the som-map with edge connected. |  | Try dragging and moving the map | Drag and move maps are allowed  Edge connected so there is relation still existed after drag the map |
| The system should allow user to post SOM data with username/publish date/comment. | Publish page & Main page | After training data，click save button in this page, input name and description, click save and publish button in this page, Check whether the map and record appears on the Publish page | The published data can be viewed by all users on publish page |
| The system should allow user to review previous data/visualization. | Profile page | After training data，click save button in this page, input name and description, click save button in this page, Check whether the map and record appears on the Profile page | Whether public or non-public, as long as the training data stored by this user can be viewed here |
| The system should allow user interaction with web-based GUI. | All pages | Check whether all buttons and input boxes meet the requirements | On the web page, the button has feedback, the input box can enter, enter one, and display one |

## Result

# Integration &Unit test

**Javascript test**

**1.Each new commit you made to the project could potentially break existing features, having already-written unit tests run successfully/passed will build your confidence in knowing that your program will just work.**

**2.As Javascript tests don't conflict with any other module and run with fewer dependencies on each other, doing Javascript tests can help reduce bugs in later development stages.**

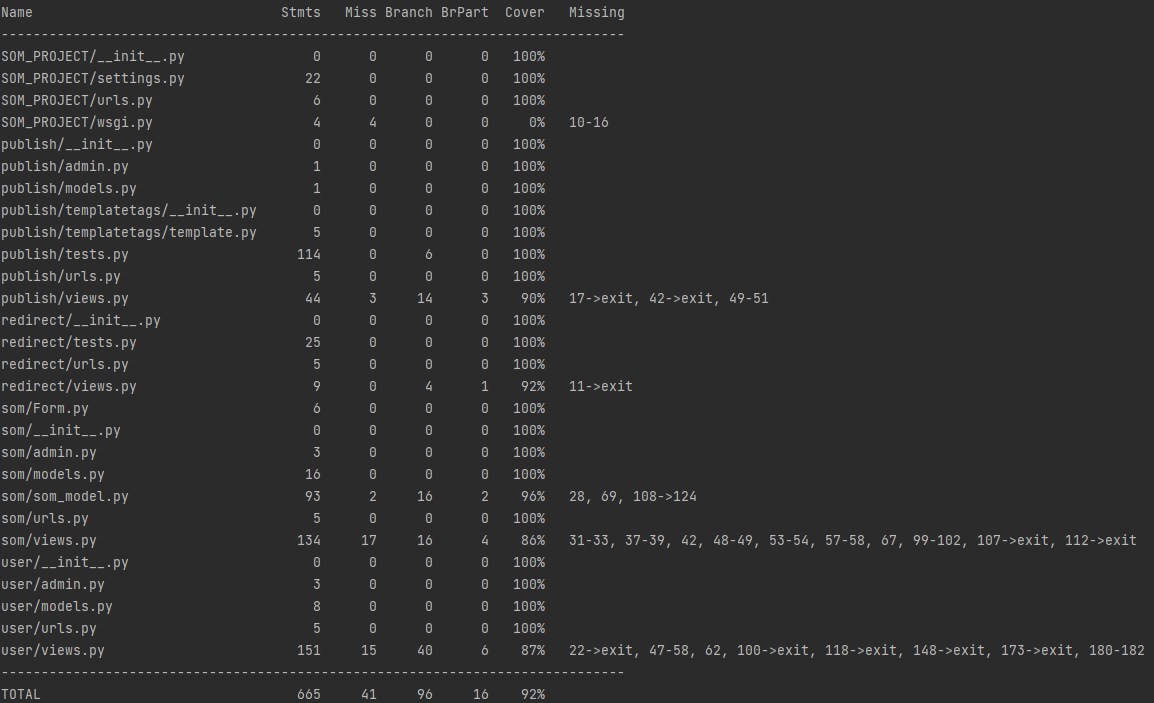
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | test content | input | predicted output | true output | description | cover |
| 1 | Main page | click function call | save menu visible | save menu visible | should be able to visit menu | save Pop-ups in main page |
| 2 | Main page | click function call | save menu invisible | save menu invisible | should be able to hidden menu | save Pop-ups in main page |
| 3 | Main page | ajax call | receive ajax call response | receive ajax call response | should be able to just save the map | save map button in the main page |
| 4. | Main page | ajax call | receive ajax call response | receive ajax call response | should be able to save and publish map | publish button in the main page |
| 5 | Main page | function call | modified data | receive modified data | should be able to set/get data | modify the data in the main page |
| 6 | Main page | function call | modified data | receive modified data | should be able to clean data | modify the data in the main page |
| 7 | Main page | ajax call | receive ajax call response | receive ajax call response | should be able to change rows | set the rows of the map in the main page |
| 8 | Main page | function call | get generated map data | get generated map data | should be able to get point | generate the map data in the main page |
| 9 | Main page | function call | receive function call response | receive function call response | should be able to move map right by data | drag the map in the main page |
| 10 | Main page | function call | receive function call response | receive function call response | should be able to move map left by data | drag the map in the main page |
| 11 | Main page | function call | receive function call response | receive function call response | should be able to move map up by data | drag the map in the main page |
| 12 | Main page | function call | receive function call response | receive function call response | should be able to move map down by data | drag the map in the main page |
| 13 | Main page | function call | receive function call response | receive function call response | should be able to display color legend of the map | display the color legend in the main page |
| 14 | Main page | ajax call | receive ajax call response | receive ajax call response | plotting the generated map | generate the visualization in the main page |
| 15 | Profile page | function call | receive function call response | receive function call response | should be able to update user’s general information | update user’s general information in the profile page |
| 16 | Profile page | function call | receive function call response | receive function call response | should be able to update user’s password | update password in the profile page |
| 18 | publish page | function call | receive function call response | receive function call response | should be able to see previous page button | previous page button in the publish page |
| 19 | publish page | function call | receive function call response | receive function call response | should be able to hidden previous page button | previous page button in the publish page |

## Backend function test

### task

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | test content | input | predicted output | true  output | description | cover |
| 1 | login page | get login page,  email:test1@test1.com  password:test11234 | template used: login.html,  return to som page，  status code=302 | template used: login.html,  return to som page，  status code=302 | post userinfo to login function | login button  in login page |
| 2 | register page | get register page,  username:test  email:test@test.com  password:test1234 | template used: register..html,  return to som page，  status code=302 | template used: register..html,  return to som page，  status code=302 | post register info to register function | register button  in register page |
| 3 | login  errors | create userinfo,  null email & password, wrong email & password | error code:164  msg: “username or password incorrect” | error code:164  msg: “username or password incorrect” | test all of login errors | login button  in login page |
| 4 | register  errors | create userinfo,  null username,email & password, existed username & email | error code:111,222,333,444,555  msg:  “user error or user existed”  “email error or email existed” | error code:111,222,333,444,555  msg:  “user error or user existed”  “email error or email existed” | test all of register errors | register page  register button  in register page |
| 5 | update  password | old password  new password | success code: 200  msg:”password changed successful” | success code: 200  msg:”password changed successful” | post new password to UpdatePassw  function and check update | profile page  save button |
| 6 | update  user | old username  new username | success code:  200  msg:”success” | success code:  200  msg:”success” | post new username to UpdateUser  function and check update | profile page  edit and update button |
| 7 | profile view | confirm login,  get user profile | return to profile.html,  status code:200 | return to profile.html,  status code:200 | get profile view | Upper right corner user image in main page |
| 8 | post new profile | confirm login,  create new profile information(file\_name=”som\_1”) | return to view.html,  status code:200,  som\_1 in the view page | return to view.html,  status code:200,  som\_1 in the view page | post new profile info to profile view function |  |
| 9 | next profile page | user info, create 21 profile data, dataframe filter uid, page\_n = 0 | return to next profile page, input profile  data equal to output profile data, page+1 equal to  context[page] | return to next profile page, input profile data equal to output profile data, page+1 equal to context[page] | post inserted data to user profile function, Simulate the command to click on the next page | next page button in profile page |
| 10 | last profile  page | user info ,create  21 profile data, dataframe filter uid, page\_l = 1 | return to last publish page, input publish data equal to output publish data, page-1 equal to  context[page] | return to last publish page, input publish data equal to output publish data, page-1 equal to  context[page] | post inserted data to user profile function, Simulate the command to click on the last page | last page button in profile page |
| 11 | zero data in next profile page | user info, create  21 profile data, dataframe filter uid, page\_n = 4 | return to current  page, page equal to  context[page] | return to current page, page equal to  context[page] | Since there is no information on the next page, the function returns to the current page | next page button in profile page |
| 12 | logout | get logout function | status code: 200  return to login.html | status code: 200  return to login.html | Simulate the reaction of clicking this button | Upper right corner  logout button |
| 13 | dataframe  model | create filename, autor, uid, description | som\_1,spike,1, this is a test | som\_1,spike,1, this is a test | use dataframe model to create the data and verify the storage of the data |  |
| 14 | upload  file | use InMemoryUploadedFile read opened file | data\_id existed | data\_id existed | simulate the process of file upload | upload button in main page |
| 15 | som map | np.array([[[1., 0.], [0., 1.]], [[1., 0.], [0., 1.]]]). | np.array([[0. , 1. , 0. , 1.],[0. , 1. , 0. , 1.],[0. , 1. , 0. , 1.],[0. , 1. , 0. , 1.]]) | np.array([[0. , 1. , 0. , 1.],[0. , 1. , 0. , 1.],[0. , 1. , 0. , 1.],[0. , 1. , 0. , 1.]]) | Interaction with map by drawing. | Visualization in map maker opage. |
| 16 | som  page | get som | return to visualization0.1.html,  status code:  200 | return to visualization0.1.html,  status code:  200 | check som template used |  |
| 17 | test save map | create user, create dataframe, author, vis\_name, description | user\_id, data\_id, spike, som\_project1,  this is a test | user\_id, data\_id, spike, som\_project1,  this is a test | check that the data is stored in the correct location | save button in main page: save button |
| 18 | test save and publish map | create user, create dataframe, author, vis\_name, description,  post to publish | user\_id, data\_id, spike, som\_project1,  this is a test in publish list | user\_id, data\_id, spike, som\_project1,  this is a test in publish list | check that the data is stored in the correct location | save button in main page: save and publish button |
| 19 | publish list | filename, uid, description | som\_1, 1 , this is a test,  return to publish.html | som\_1, 1 , this is a test,  return to publish.html | view the contents of the Publish page | Upper left corner publish button |
| 20 | publish  view | ObjectID, data\_id, filename | som\_1,  return to view.html | som\_1,  return to view.html | create a publish data first, and then view it | view button in publish page |
| 21 | next publish page | user info, create 21 publish data, dataframe filter publish = True, page\_n = 0 | return to next publish page, input publish data equal to output publish data, page+1 equal to  context[page] | return to next publish page, input publish data equal to output publish data, page+1 equal to context[page] | post inserted data to publish list function, Simulate the command to click on the next page | next page button in publish page |
| 22 | last publish  page | user info, create 21 publish data, dataframe filter publish = True, page\_l = 1 | return to last publish page, input publish data equal to output publish data, page-1 equal to  context[page] | return to last publish page, input publish data equal to output publish data, page-1 equal to  context[page] | post inserted data to publish list function, Simulate the command to click on the last page | last page button in publish page |
| 23 | zero data in next publish page | user info, create 21 publish data, dataframe filter publish = True,  page\_n = 4 | return to current  page, page equal to  context[page] | return to current  page, page equal to  context[page] | Since there is no information on the next page, the function returns to the current page | next page button in publish page |
| 24 | redirect app | create user info,  login succeeded, get redirect view function | status code = 302,  response.url =  publish list | status code = 302,  response.url = publish list | user can only go to the publish function when the user logs in successfully |  |
| 25 | redirect app | create user info,  login failed, get redirect view function | status code = 302,  response.url =  user login | status code = 302,  response.url =  user login | if user login failed,  let him login first |  |

### Summary Report



The tested method and coverage showing here with more that 80% method tested. After using test coverage tool, we can find that most of our back-end codes have been covered by the test.

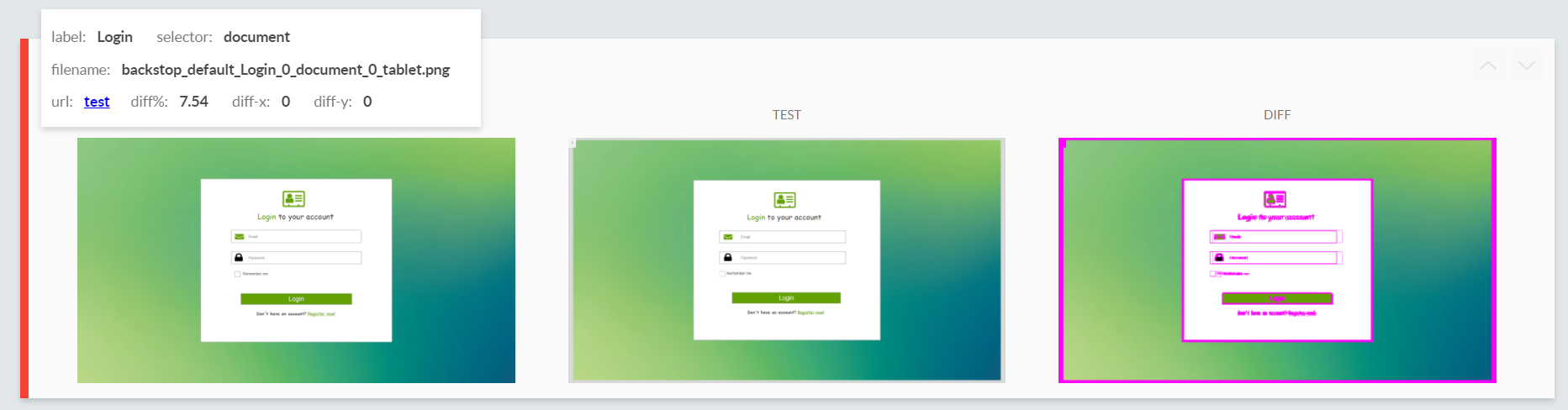
## Frontend function test report

The Java Script unit passed the test by using Jasmine tool.

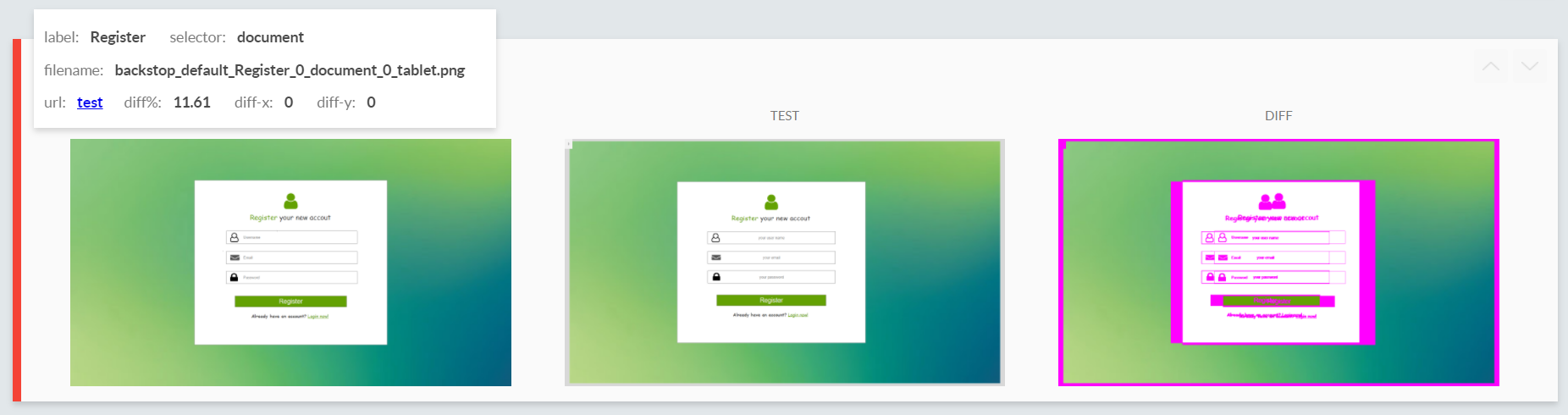
# UI test

Reference（the left one） is the actual effect displayed in the project，The test is the original of the Axure design，differences shows the difference between the Reference and test.

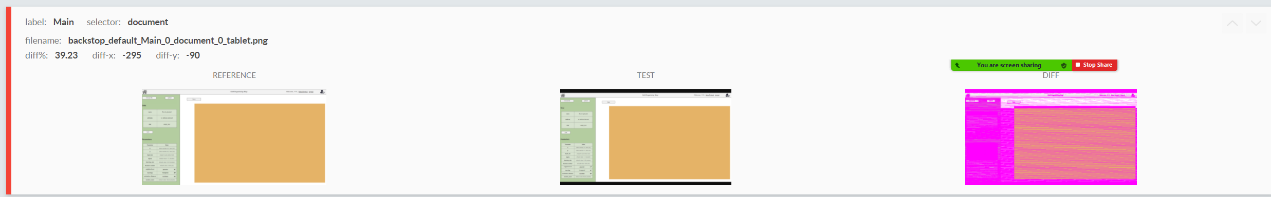
## login



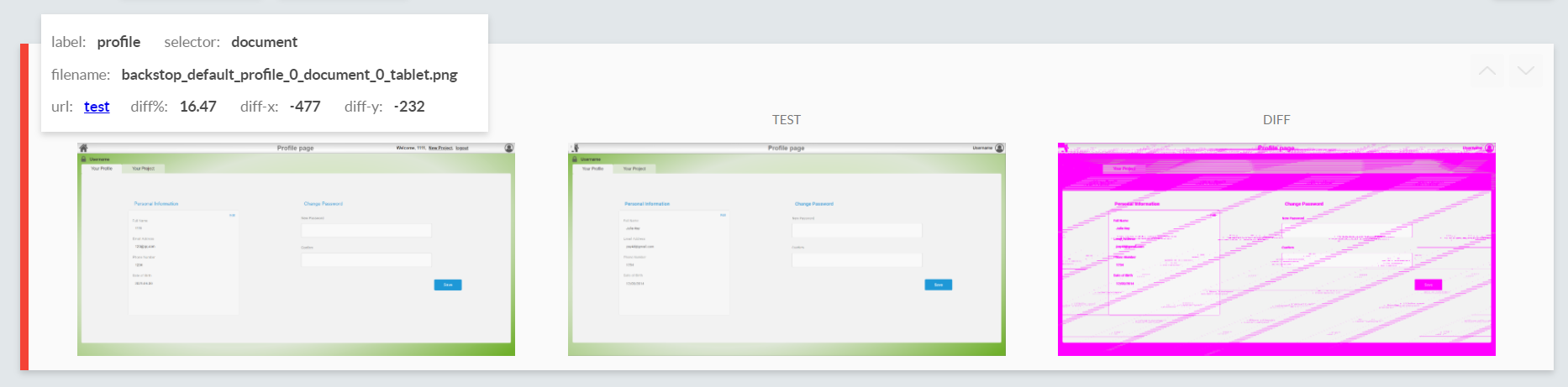
## Register



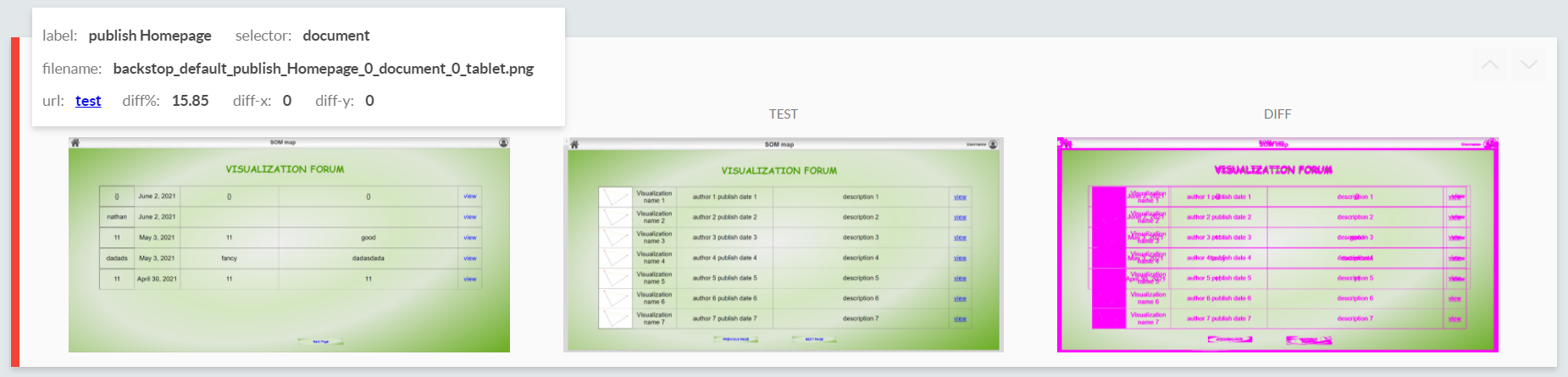
## New project



## User Profile



## Project list



## View project

