# SENDCLOUD’s CRISPY SUCCOSTASH APPLICATION

## TEST APPROACH

## &

## AUTOMATION SOLUTION

INTRODUCTION ..... 3
TEST STRATEGY ..... 4
SOLUTION ..... 4
SCOPE ..... 4
TEST COVERAGE ..... 5
ROLES & RESPONSABILITIES ..... 6
AUTOMATION SOLUTION ..... 7
DESIGN ..... 8
EXECUTION ..... 10
REQUIREMENTS ..... 10
1) INTERACTIVELY EXECUTION ..... 10
2) NON-INTERACTIVELY EXECUTION ..... 12
3) REPORTING ..... 13
4) VISUAL VALIDATION TESTS ..... 15

## INTRODUCTION

This document contains the description of the test approach taken for the ***Crispy Succotash*** application as well as the details about the automation solution along with instructions on how to run it, its design and reporting.

# TEST STRATEGY

Given main goal of solution is to focus in automated test cases implementation and time is limited, this is a brief test strategy with the approach, scope and other definitions to consider in the test process.

## SOLUTION

Crispy-Succotash provided solution is a web application which has been developed to cover the following requirements:

- ➔ User account creation and login
- ➔ Support to follow multiple feeds with at least support for the following ones:
    - http://www.nu.nl/rss/Algemeen
    - https://feeds.feedburner.com/tweakers/mixed
- ➔ Ability to add a bookmark/favorite to a feed item.
- ➔ Ability to add Markdown based comments to the feed items.

## SCOPE

Given time available for implementation is limited, test suite will include only the main flows/cases of the mentioned requirements related use cases. Some additional test cases might be added with data driven approach in some additional test cases.

All test cases will be automated.

Tests will be 95% functional tests and 1 test case sample of visual validation will be added.

## TEST COVERAGE

| Functionality | Tests Included |
|---|---|
| USER CREATION | • *Validate default sign up page*<br>• *Validate Positive User sign up*<br>• *Validate sign up with empty fields*<br>• *Validate sign up with different passwords*<br>• *Validate sign up with short passwords ->* data driven test cases (8 in total) |
| LOG IN | • *Validate default log in page*<br>• *Validate positive user log in* |
| FEEDS | • *Validate default new feed page*<br>• *Validate submit empty feed*<br>• *Validate My Feeds when empty*<br>• *Validate All Feeds when added*<br>• *Validate specific feed added*<br>• *Validate empty bookmark feeds page*<br>• *Validate bookmarked feed*<br>• *Validate New Feed creation*<br>• *Validate Feed Entry when Logged Out*<br>• *Validate Feed Entry when Logged In*<br>• *Validate add comment in Feed Entry* |
| VISUAL VALIDATION | • *Validate application aspect for logged out user.* |

## ROLES & RESPONSABILITIES

| Role | Assigned to | Responsibilities |
|---|---|---|
| Software QA Automation Engineer | Matías Cárdenas | ➔ Write test strategy<br>➔ Design and implement test automation solution<br>➔ Document test automation solution<br>➔ Upload solution and provide instructions on how to execute it |

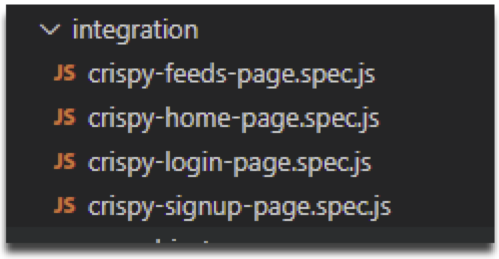
## AUTOMATION SOLUTION

Automation solution has been developed in order to have more tests executed, make them faster and more efficient.

Solution has been built using ***Cypress***, which allow us to create faster, easier and more reliable tests.

## DESIGN

- Tests are divided in 4 main parts: **Home, Sign Up, Log In and Feeds** pages.



```
integration
  crispy-feeds-page.spec.js
  crispy-home-page.spec.js
  crispy-login-page.spec.js
  crispy-signup-page.spec.js
```

**HOME TESTS**
*(crispy-home-page.spec.js)*

- *Validate default home page*
- *Validate Log In opens*
- *Validate All Feeds opens*
- *Validate Sign Up opens*

**SIGN UP TESTS**
*(crispy-signup-page.spec.js)*

- *Validate default sign up page*
- *Validate Positive User sign up*
- *Validate sign up with empty fields*
- *Validate sign up with different passwords*
- *Validate sign up with short passwords ->* data driven test cases (8 in total)

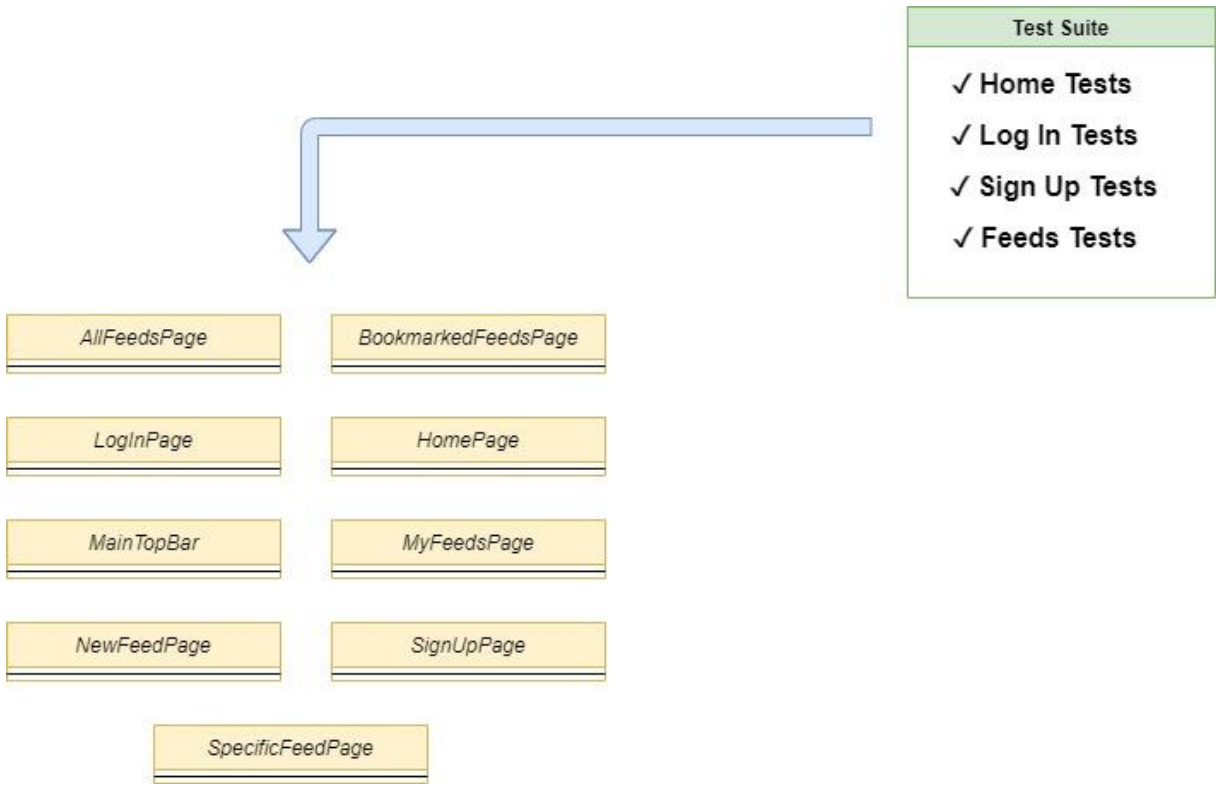
**LOG IN TESTS**
*(crispy-login-page.spec.js)*

- *Validate default log in page*
- *Validate positive user log in*

**FEEDS TESTS**
*(crispy-feeds-page.spec.js)*

- *Validate default new feed page*
- *Validate submit empty feed*
- *Validate My Feeds when empty*
- *Validate All Feeds when added*
- *Validate specific feed added*
- *Validate empty bookmark feeds page*
- *Validate bookmarked feed*
- *Validate New Feed creation*
- *Validate Feed Entry when Logged Out*
- *Validate Feed Entry when Logged In*
- *Validate add comment in Feed Entry*

- Also ***PAGE OBJECT*** pattern has been implemented as a neat, reusable and efficient way of instantiating web pages objects and performing manipulations on them.



## EXECUTION

### REQUIREMENTS

- NodeJS and NPM
- Cypress
- A Browser installed (Chrome, Firefox)
- Get the solution:
    - ➢ *git clone https://github.com/maticardenas/sendcloud-test-automation.git*

### 1) INTERACTIVELY EXECUTION

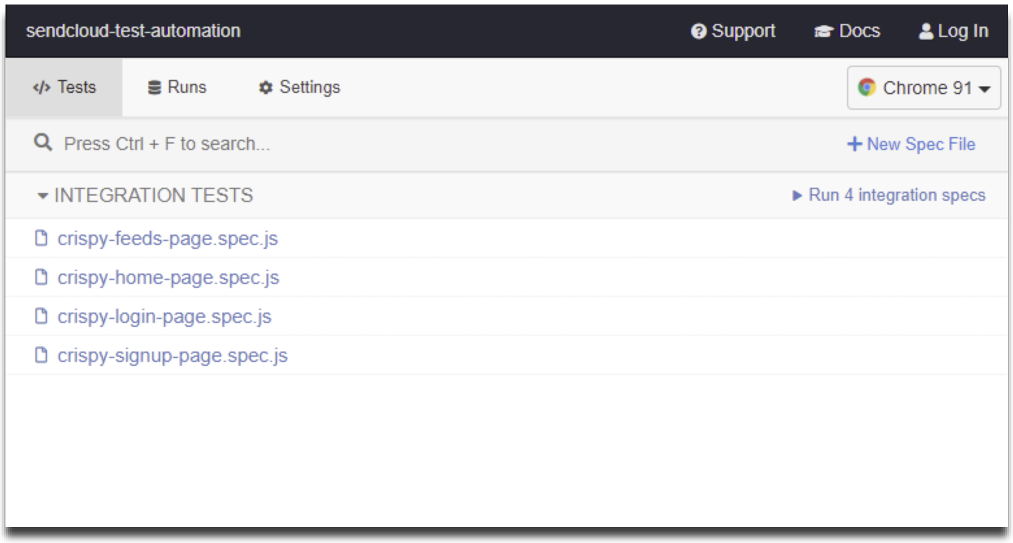
Scripts shortcuts have been added at ***package.json*** file, for executing all tests from npm.

All tests can be run **INTERACTIVELY** running this command:

```
➢ npm run cypress
```

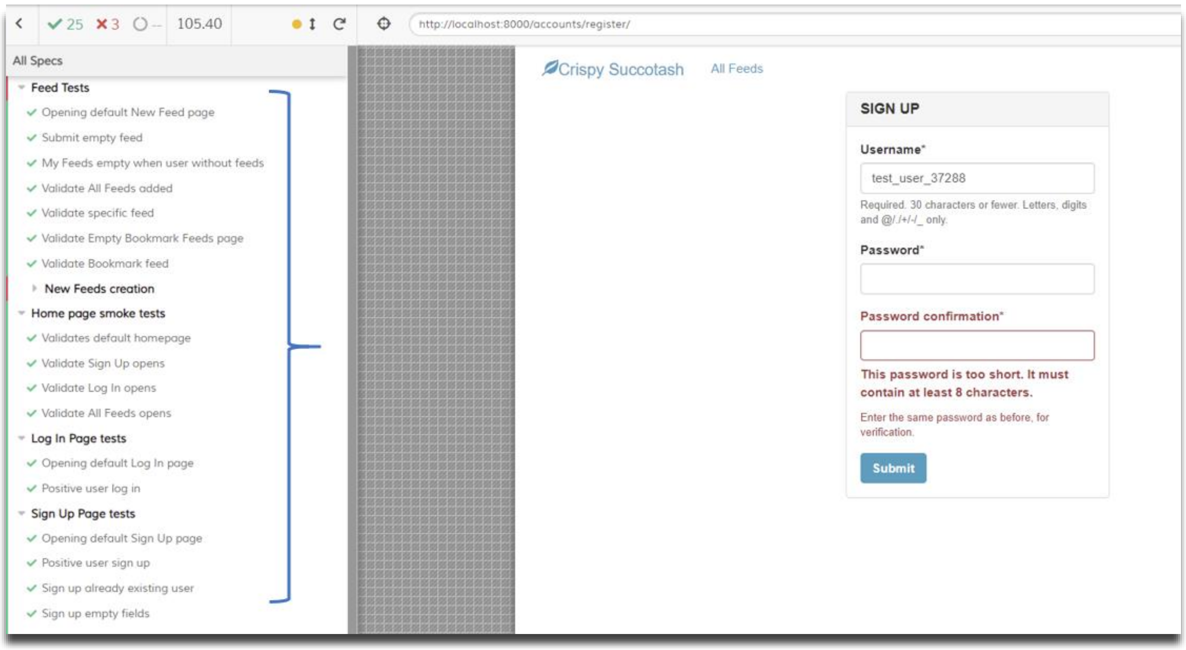
This command executes the Test Runner from cypress, from where is possible to execute all of the tests or the desired ones, in interactive mode.

O



crispy-feeds-page.spec.js
crispy-home-page.spec.js
crispy-login-page.spec.js
crispy-signup-page.spec.js

Once executed, tests results as well as its steps can be checked next to the web page being displayed while performing the execution.



✓ 25 ✗ 3 ○ -- 105.40

http://localhost:8000/accounts/register/

All Specs

- Feed Tests
  - ✓ Opening default New Feed page
  - ✓ Submit empty feed
  - ✓ My Feeds empty when user without feeds
  - ✓ Validate All Feeds added
  - ✓ Validate specific feed
  - ✓ Validate Empty Bookmark Feeds page
  - ✓ Validate Bookmark feed
  - New Feeds creation
- Home page smoke tests
  - ✓ Validates default homepage
  - ✓ Validate Sign Up opens
  - ✓ Validate Log In opens
  - ✓ Validate All Feeds opens
- Log In Page tests
  - ✓ Opening default Log In page
  - ✓ Positive user log in
- Sign Up Page tests
  - ✓ Opening default Sign Up page
  - ✓ Positive user sign up
  - ✓ Sign up already existing user
  - ✓ Sign up empty fields

Crispy Succotash  All Feeds

SIGN UP

Username*

test_user_37288

Required. 30 characters or fewer. Letters, digits and @/./+/-/_ only.

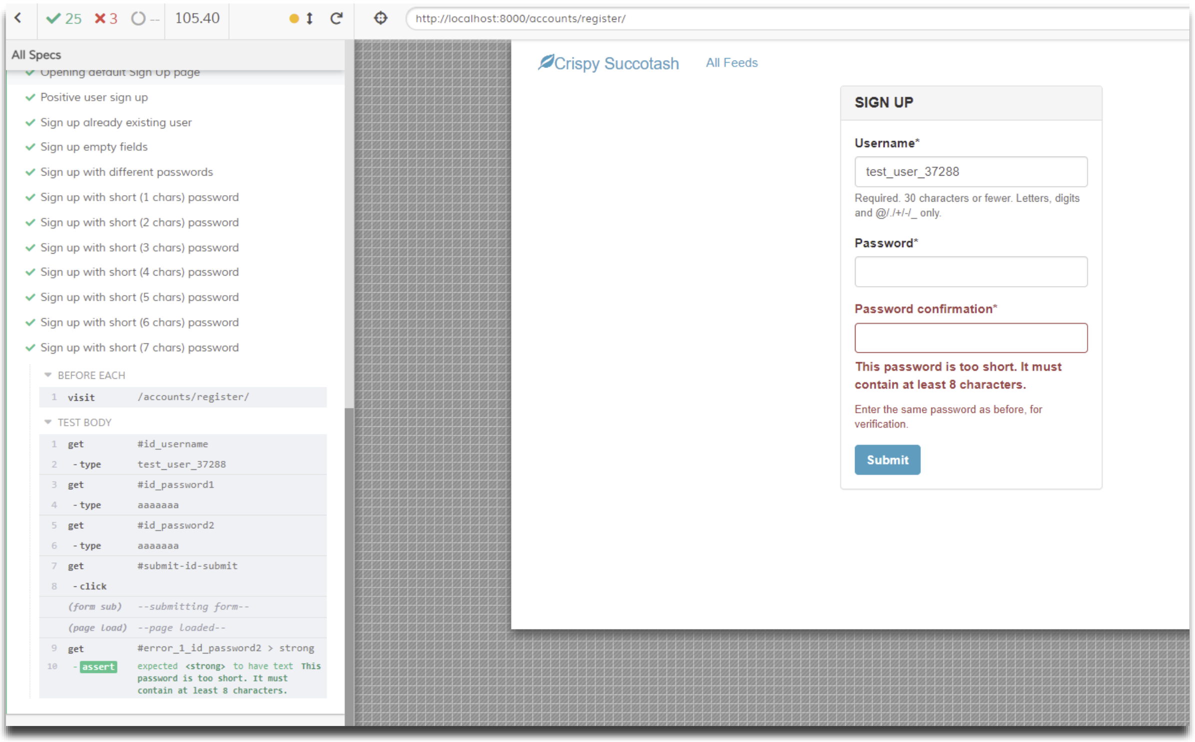
Password*

Password confirmation*

This password is too short. It must contain at least 8 characters.

Enter the same password as before, for verification.

Submit



✓ 25 ✗ 3 ○ -- 105.40

http://localhost:8000/accounts/register/

All Specs

- ✓ Opening default Sign Up page
- ✓ Positive user sign up
- ✓ Sign up already existing user
- ✓ Sign up empty fields
- ✓ Sign up with different passwords
- ✓ Sign up with short (1 chars) password
- ✓ Sign up with short (2 chars) password
- ✓ Sign up with short (3 chars) password
- ✓ Sign up with short (4 chars) password
- ✓ Sign up with short (5 chars) password
- ✓ Sign up with short (6 chars) password
- ✓ Sign up with short (7 chars) password

```
BEFORE EACH
visit          /accounts/register/
TEST BODY
get            #id_username
 - type        test_user_37288
get            #id_password1
 - type        aaaaaaa
get            #id_password2
 - type        aaaaaaa
get            #submit-id-submit
 - click
(form sub)     --submitting form--
(page load)    --page loaded--
get            #error_1_id_password2 > strong
 - assert      expected <strong> to have text This
               password is too short. It must
               contain at least 8 characters.
```

Crispy Succotash  All Feeds

SIGN UP

Username*

test_user_37288

Required. 30 characters or fewer. Letters, digits and @/./+/-/_ only.

Password*

Password confirmation*

This password is too short. It must contain at least 8 characters.

Enter the same password as before, for verification.

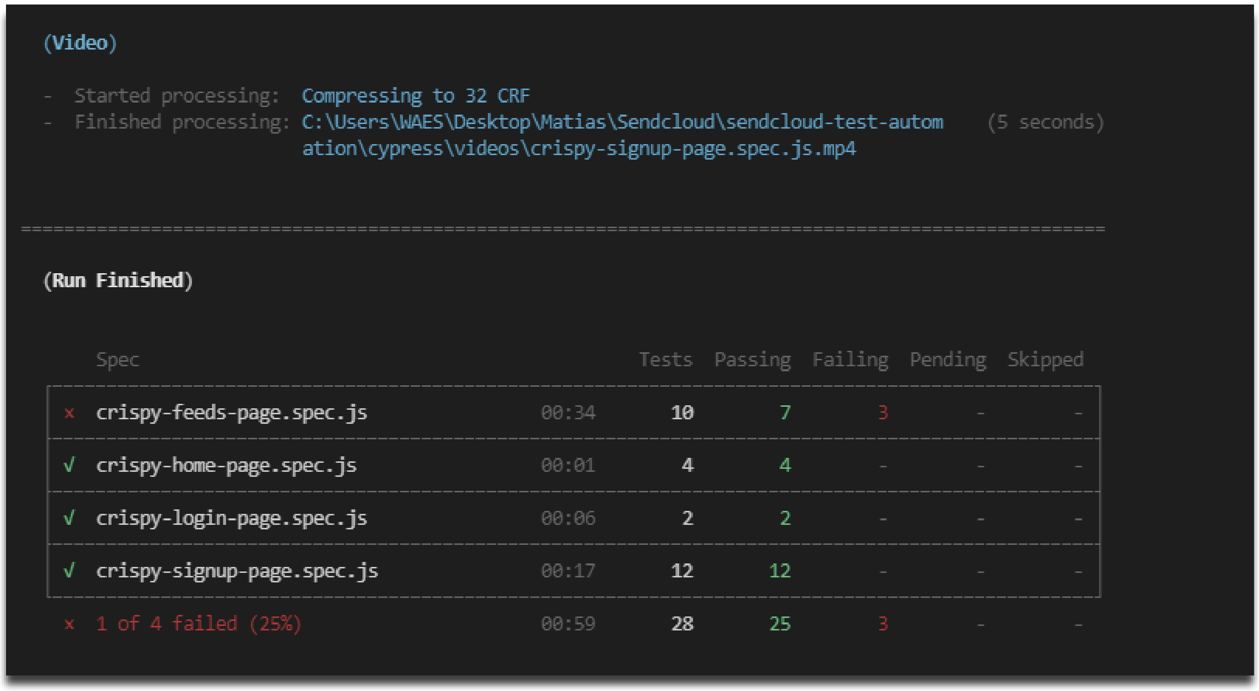
Submit

## 2) NON-INTERACTIVELY EXECUTION

All tests can be run **NON-INTERACTIVELY** running this command:

- ***npm run test***

This command will execute all tests in non-interactive mode, displaying results in the console without opening the interactive runner. It will generate though the videos of the execution at */videos/* folder of the repository.



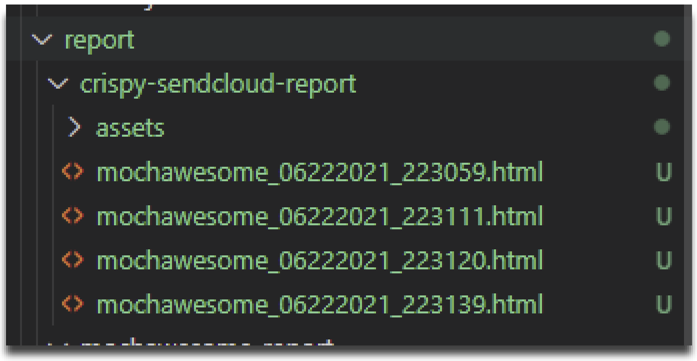
```
  (Video)

  -  Started processing:  Compressing to 32 CRF
  -  Finished processing: C:\Users\WAES\Desktop\Matias\Sendcloud\sendcloud-test-autom     (5 seconds)
                          ation\cypress\videos\crispy-signup-page.spec.js.mp4


====================================================================================================

  (Run Finished)


       Spec                                              Tests  Passing  Failing  Pending  Skipped
  ┌──────────────────────────────────────────────────────────────────────────────────────────────┐
  │ ×  crispy-feeds-page.spec.js                00:34       10        7        3        -        - │
  ├──────────────────────────────────────────────────────────────────────────────────────────────┤
  │ √  crispy-home-page.spec.js                 00:01        4        4        -        -        - │
  ├──────────────────────────────────────────────────────────────────────────────────────────────┤
  │ √  crispy-login-page.spec.js                00:06        2        2        -        -        - │
  ├──────────────────────────────────────────────────────────────────────────────────────────────┤
  │ √  crispy-signup-page.spec.js               00:17       12       12        -        -        - │
  └──────────────────────────────────────────────────────────────────────────────────────────────┘
    ×  1 of 4 failed (25%)                      00:59       28       25        3        -        -
```

For executing each test suite individually the following commands can be used:

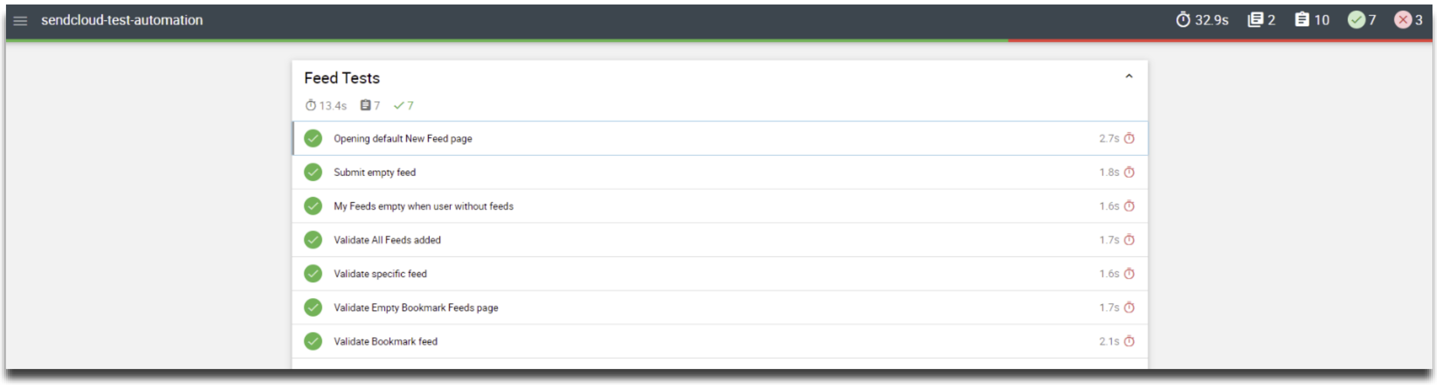
- ***npm run home-tests***
- ***npm run login-tests***
- ***npm run signup-tests***
- ***npm run feeds-tests***

## 3) REPORTING

Making use of ***mochawesome*** framework, at the end of the execution a report for the tests is generated at */report/crispy-sendcloud-report* directory:



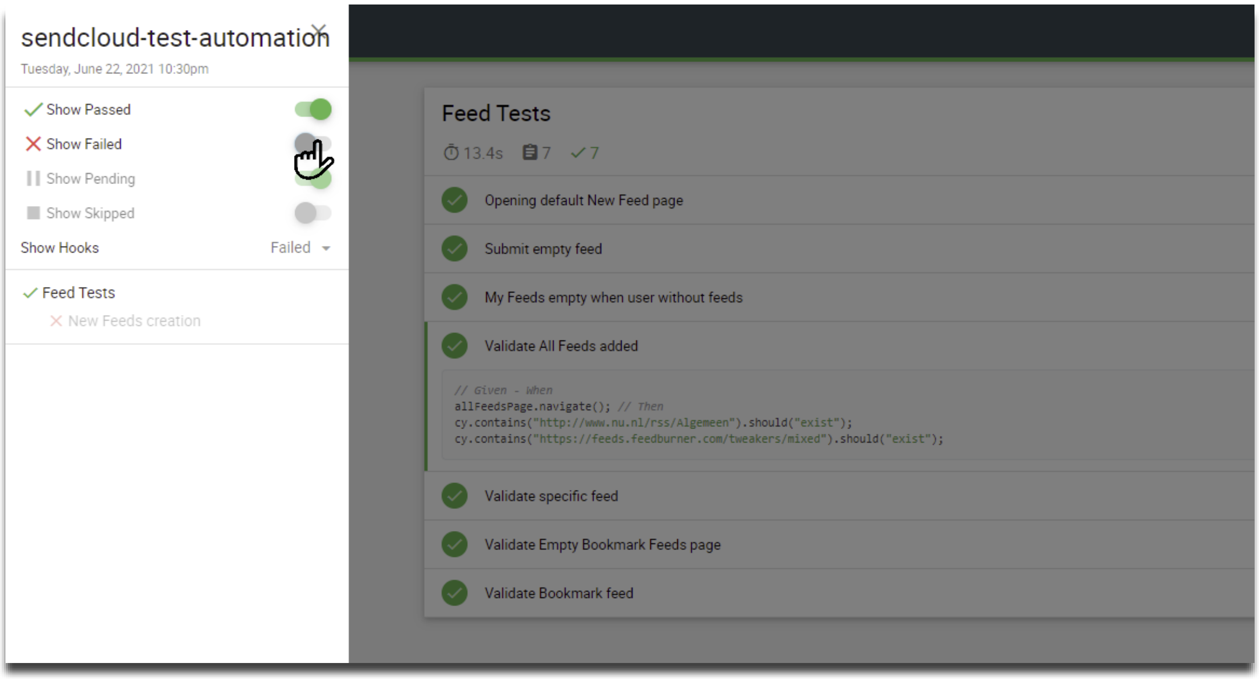
```
report
  crispy-sendcloud-report
    > assets
    mochawesome_06222021_223059.html    U
    mochawesome_06222021_223111.html    U
    mochawesome_06222021_223120.html    U
    mochawesome_06222021_223139.html    U
```



sendcloud-test-automation — 32.9s | 2 | 10 | 7 | 3

Feed Tests — 13.4s | 7 | ✓ 7

| Test | Duration |
|---|---|
| ✓ Opening default New Feed page | 2.7s |
| ✓ Submit empty feed | 1.8s |
| ✓ My Feeds empty when user without feeds | 1.6s |
| ✓ Validate All Feeds added | 1.7s |
| ✓ Validate specific feed | 1.6s |
| ✓ Validate Empty Bookmark Feeds page | 1.7s |
| ✓ Validate Bookmark feed | 2.1s |



Opening default New Feed page

Submit empty feed

My Feeds empty when user without feeds

Validate All Feeds added

```
// Given - When
allFeedsPage.navigate(); // Then
cy.contains("http://www.nu.nl/rss/Algemeen").should("exist");
cy.contains("https://feeds.feedburner.com/tweakers/mixed").should("exist");
```

Validate specific feed

Validate Empty Bookmark Feeds page

Validate Bookmark feed

**<u>NOTE:</u> MOCHAWESOME REPORTS ARE ONLY GENERATED WHEN RUNNING THE TESTS IN NON-INTERACTIVELY MODE.**

## 4) VISUAL VALIDATION TESTS

All previous mentioned and implemented tests are functional tests, there is also added a visual validation test, to validate the aspect of the application.

Test added as sample is for **logged out navigation** (crispy-visual-valid.spec.js) and with the same way much more several tests can be added to tests different aspects in different scenarios of the application. (no more were added given limited time to work on it).

For running the visual validation tests **applitools / eyes-cypress** dependency is required. Also to see it's required to signup an account at applitools.com (it is possible to sign up with github account) and retrieve the API key from there.

### *REQUIREMENTS*

- Install dependency:

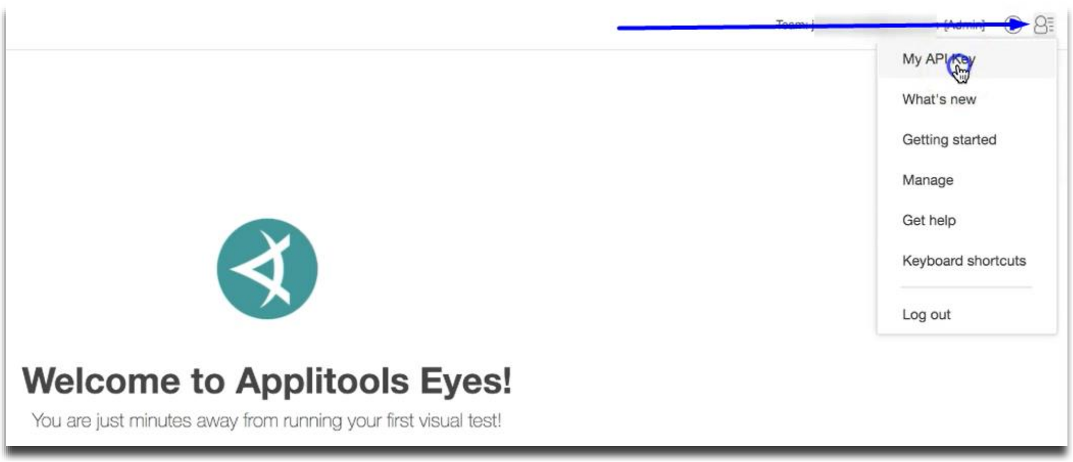
    ➢ **`npm install @applitools/eyes-cypress`**

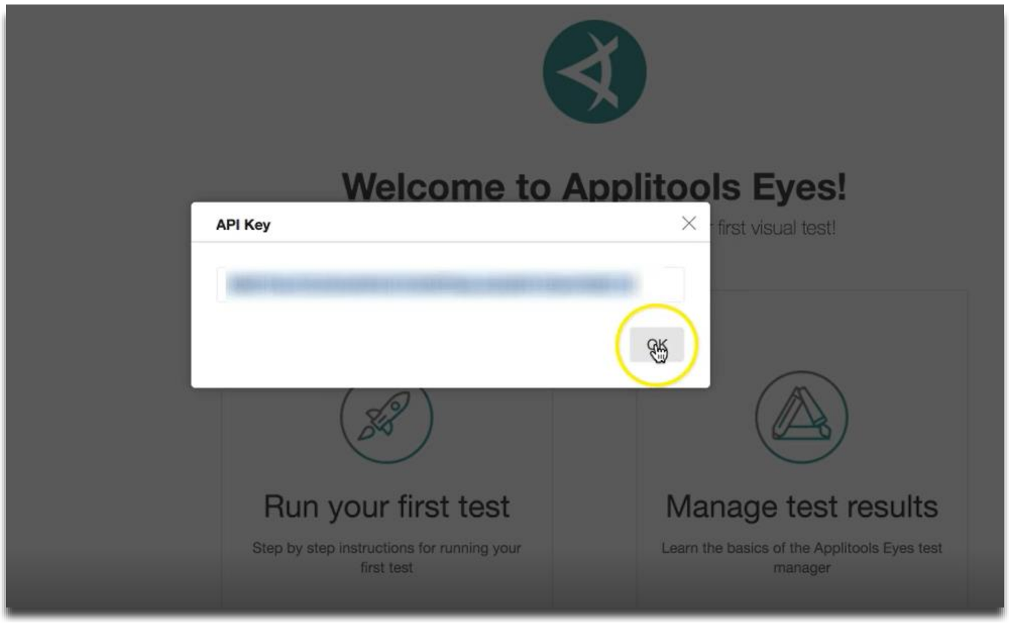
- Link to cypress:

    ➢ **`npx eyes-setup`**

- Setup API Key:

    1) Retrieve API key linked to your account from applitools.com


[Admin]
My API Key
What's new
Getting started
Manage
Get help
Keyboard shortcuts
Log out
Welcome to Applitools Eyes!
You are just minutes away from running your first visual test!


Welcome to Applitools Eyes!
first visual test!
API Key
OK
Run your first test
Step by step instructions for running your first test
Manage test results
Learn the basics of the Applitools Eyes test manager

2) Set API Key as environment variable Linux terminal / Git bash / CMD (where you are executing the tests) :

Linux/Git bash:

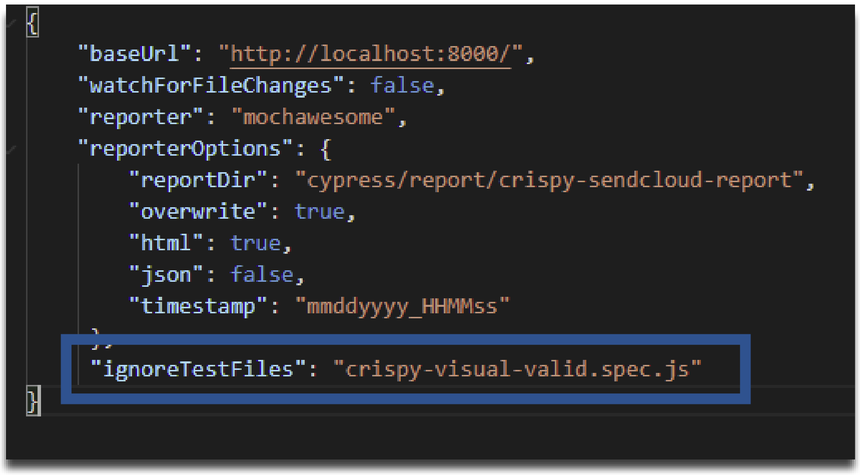
➢ `export APPLITOOLS_API_KEY=<API_KEY_VALUE>`

Windows:

➢ `set APPLITOOLS_API_KEY=<API_KEY_VALUE>`

## *EXECUTION*

Given the requirements step, this test is disabled by default in the suite. In order to enable it, it is just needed to remove the ignore line at *cypress.json* file:

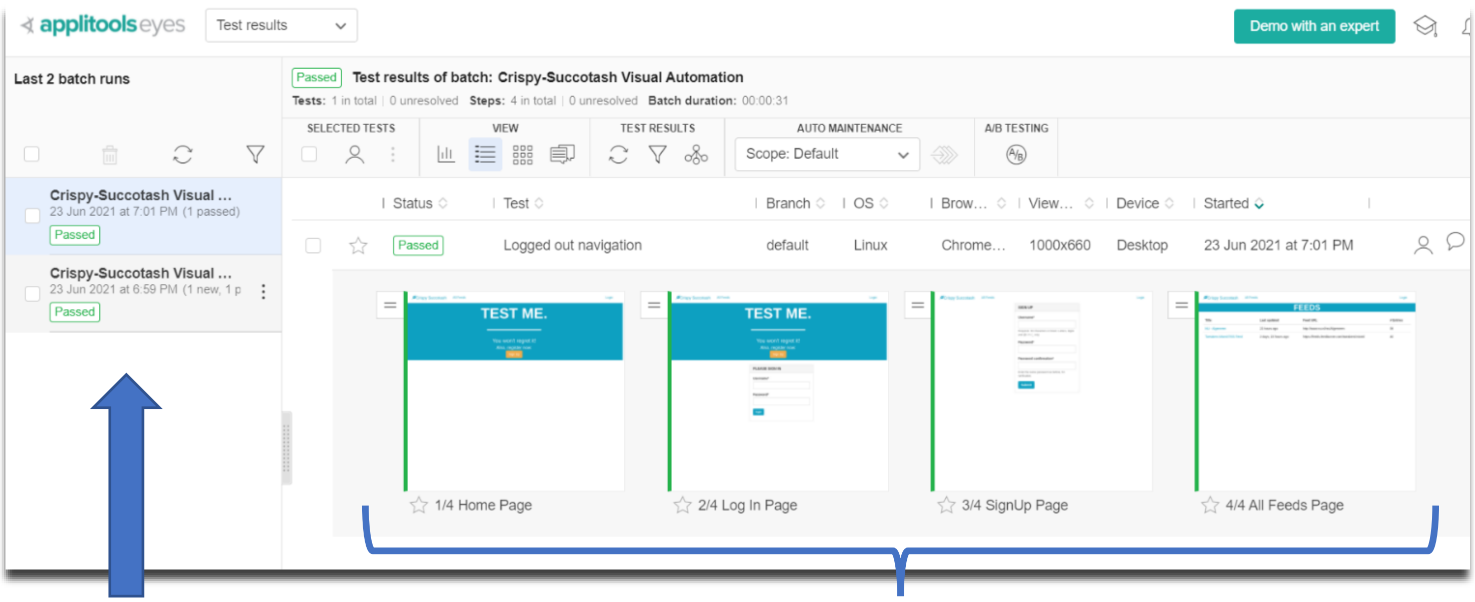


```
{
    "baseUrl": "http://localhost:8000/",
    "watchForFileChanges": false,
    "reporter": "mochawesome",
    "reporterOptions": {
        "reportDir": "cypress/report/crispy-sendcloud-report",
        "overwrite": true,
        "html": true,
        "json": false,
        "timestamp": "mmddyyyy_HHMMss"
    },
    "ignoreTestFiles": "crispy-visual-valid.spec.js"
}
```

Test can be normally executed interactively or non-interactively running:

```
npm run visual-validation
```

After finishing, at APPLITOOLS account there will be a record of the test executed with the snapshots taken for the visual validation:



Last 2 batch runs

Crispy-Succotash Visual … 23 Jun 2021 at 7:01 PM (1 passed) Passed

Crispy-Succotash Visual … 23 Jun 2021 at 6:59 PM (1 new, 1 p Passed

Passed **Test results of batch: Crispy-Succotash Visual Automation**

**Tests:** 1 in total | 0 unresolved **Steps:** 4 in total | 0 unresolved **Batch duration:** 00:00:31

AUTO MAINTENANCE: Scope: Default

| Status | Test | Branch | OS | Brow… | View… | Device | Started |
|---|---|---|---|---|---|---|---|
| Passed | Logged out navigation | default | Linux | Chrome… | 1000x660 | Desktop | 23 Jun 2021 at 7:01 PM |

1/4 Home Page

2/4 Log In Page

3/4 SignUp Page

4/4 All Feeds Page