**NTVN\_SD\_001\_Guideline**

Automation testing with javascript - Cypress

Version:1.0

**Jul 20 2020**

# 

**Author: Minh Tuan Pham**

**Security Classification: Confidential**

110 Bishopsgate, London EC2N 4AY

Tel: +44 (0)20 7333 0033

Email: [info@nashtechglobal.com](mailto:info@nashtechglobal.com)



Table of contents

[1](#_Toc46502011)

[Document History 1](#_Toc46502012)

[1. Cypress overview 1](#_Toc46502013)

[2. Installation 2](#_Toc46502014)

[1.1 List of needed for implement script: 2](#_Toc46502015)

[1.2 Import source code: 2](#_Toc46502016)

[3. Source code architecture 3](#_Toc46502017)

[3.1 Configuration files 3](#_Toc46502018)

[3.2 Cypress folder 4](#_Toc46502019)

[4. Run automation test script by command line 4](#_Toc46502020)

[5. Run automation test script with cypress GUI mode 5](#_Toc46502021)

[6. Continuous Integration with Jenkins 6](#_Toc46502022)

# Document History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Document Version History | | | | | |
| **Version** | **Effective Date** | **Author** | **Details** | **Reviewer** | **Approvers** |
| 1.0 | Jul 24 2020 | Minh Tuan Pham | Initial guideline |  |  |

Confidentiality

This document is distributed on a restricted basis, is commercial in confidence to the recipient, and may not be used for any purpose other than that associated with a NashTech project. The contents of this document may not be disclosed to any third parties without the expressed advance written authorisation of NashTech.

# Cypress overview

Cypress is a next generation front end testing tool built for the modern web. We address the key pain points developers and QA engineers face when testing modern applications.

Cypress is both fundamentally and architecturally different. Cypress is not constrained by the same restrictions as Selenium.

This enables you to write faster, easier and more reliable tests.

Browser support:

* [Canary](https://www.google.com/chrome/browser/canary.html)
* [Chrome](https://www.google.com/chrome/browser/desktop/index.html)
* [Chromium](https://www.chromium.org/Home)
* [Edge](https://www.microsoft.com/edge)
* [Edge Beta](https://www.microsoftedgeinsider.com/download)
* [Edge Canary](https://www.microsoftedgeinsider.com/download)
* [Edge Dev](https://www.microsoftedgeinsider.com/download)
* [Electron](https://electron.atom.io/)
* [Firefox](https://www.mozilla.org/firefox/) (Beta support)
* [Firefox Developer Edition](https://www.mozilla.org/firefox/developer/) (Beta support)
* [Firefox Nightly](https://www.mozilla.org/firefox/nightly/) (Beta support)

There are some pros and cons about that approach:

* **Pros**:
  + Easy to use because the cypress contains almost everything to implement a test script such as assertion, wait element method and support run scripts.
  + Fast in building and running the test script
  + Support run parallel test cases (Enterprise option)
* **Cons**
  + Cypress is relatively new, and it does not have the vast community that selenium does
  + No cross-browser testing, this is huge and will cause less adoption until it can be cured
  + As stated earlier, it’s JavaScript or bust. You won’t write cypress tests in the tired old static languages of C# and java
  + Because it runs inside the browser, you won’t be able to support multiple tabs
  + There is no cross-browser support other than Chrome and Electron
  + At this time of writing, there is no shadow DOM support

# Installation

## 1.1 List of needed for implement script:

* **Nodejs** and **Npm** (download: <https://www.npmjs.com/get-npm>)
* **Visual Studio Code** tool (download: <https://code.visualstudio.com/>)
* **Jenkins** (download: <https://www.jenkins.io/>)
* **Npx** by npm: use command **npm i npx (see** <https://www.npmjs.com/package/npx>**)**

Note:

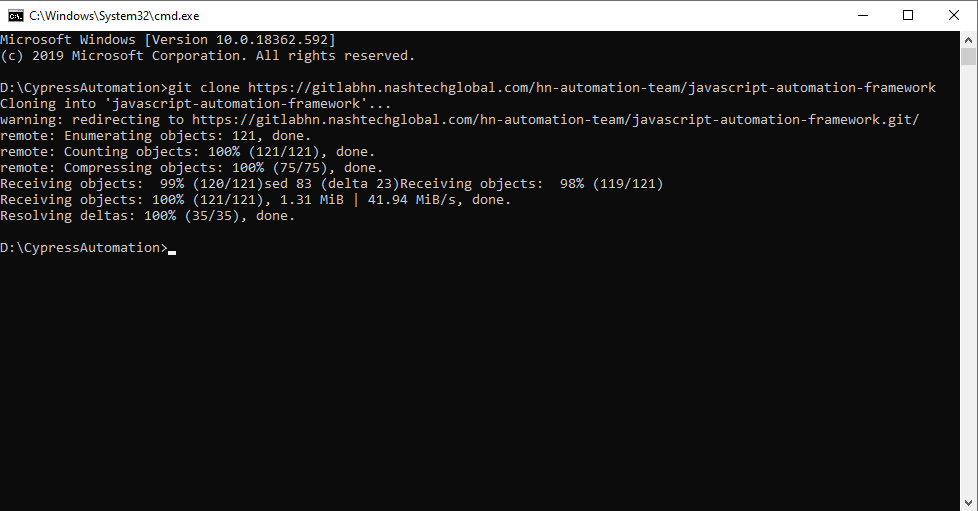
* **Nodejs** and **Npm** is use for running javascript application.
* **Visual studio code** is IDE for development so you can you the other tools such as visual studio, WebStorm.
* **Jenkins** is the tools to setup CICD
* **Npx** is helper for run some script easily like generate HTML report

## 1.2 Import source code:

The source code of automation script will be managed by gitlab at [**Link**](https://gitlabhn.nashtechglobal.com/hn-automation-team/javascript-automation-framework)

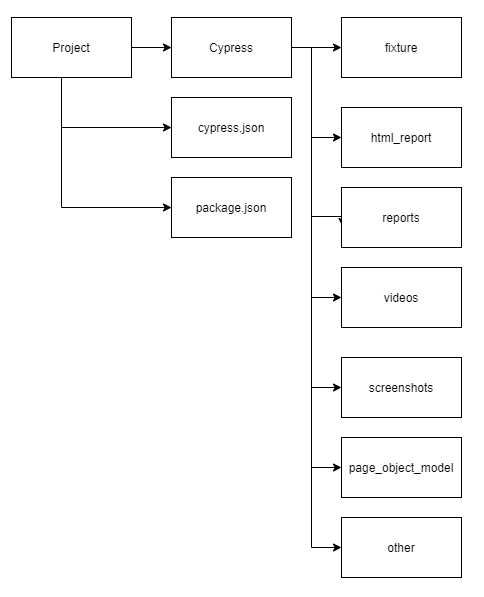
To import automation script by git:

* Open the empty folder to save source code (Example: D:\CypressAutomation)
* Open command line at that folder
* Clone code by git command (**git clone** [**https://gitlabhn.nashtechglobal.com/hn-automation-team/javascript-automation-framework**](https://gitlabhn.nashtechglobal.com/hn-automation-team/javascript-automation-framework))
* Open the source code by visual studio code



# 3. Source code architecture

The automation script source contains 2 main parts contain **configuration file** and **cypress folder**. For more details, cypress is the folder which was generated by installing cypress (to install cypress by npm: **npm install cypress --save-dev**)



## Configuration files

* **Cypress.json**: Contain configuration information for cypress. There are some common attribute such as (see: [Configurations](https://docs.cypress.io/guides/references/configuration.html))
  + reporter: contain name of reporting library
  + baseUrl: to setup the base Url of the website that is automated
  + reporterOptions: options of reporter library
  + defaultCommandTimeout: to set default timeout for the cypress command
  + video: flag say that to record video or not
  + viewPortWidth: to set the width of the testing window on browser
  + viewPortHeight: to set the height of the testing window on browser
* Package.json: Contain configuration to build and run the project.
  + Dependencies: consist all dependencies of the project.
  + Script: consist the group of script to run the project.
  + Some project information like name, version, etc.

## Cypress folder

Cypress folder is the folder is generate when install cypress library by npm that contains:

* Fixture: are used as external pieces of static data that can be used by your tests
* Integration: locate test files in that folder but can be configured to another directory.
* Screenshots: Contain screenshots when running script got failure.
* Videos: Contains the video recording
* Others

There are others folder inside the cypress folder:

* Html\_report: Contain html reports of the test scripts.
* Reports: Contain reports in json type in each time running scripts.

# 4. Run automation test script by command line

First of all, to run the script you have to install Npm by the command:

**npm clean-install** to create the node\_modules folder and set up the dependencies.

The source code already set up some script command to run the script such as:

* **npm run test** to run the script with default browser
* **npm run test:chrome** to run the script with chrome browser
* **npm run test:electron** to run the script with electron browser
* **npm run test:chrome:headless** to run the script with chrome headless browser
* **npm run test:chrome** to run the script with chrome browser

For more details, the script always consists 3 parts are **pretest**, **running** and **posttest** script

* **Pretest** script: To clean the report folder to prepare for new running time.
* **Running** script: To run the script with options like type of browser and headless mode.
* **Posttest** script: To generate report

Note:

* You can customize the script as the requirement of the project
* In the documented time, you can note run with **Firefox browser** because it is in the beta status and have the closing browser conflict error when run by command line.
* The report with be generate into **html\_report** folder with format is “report\_${datetime}” which can setup in the package.json file.

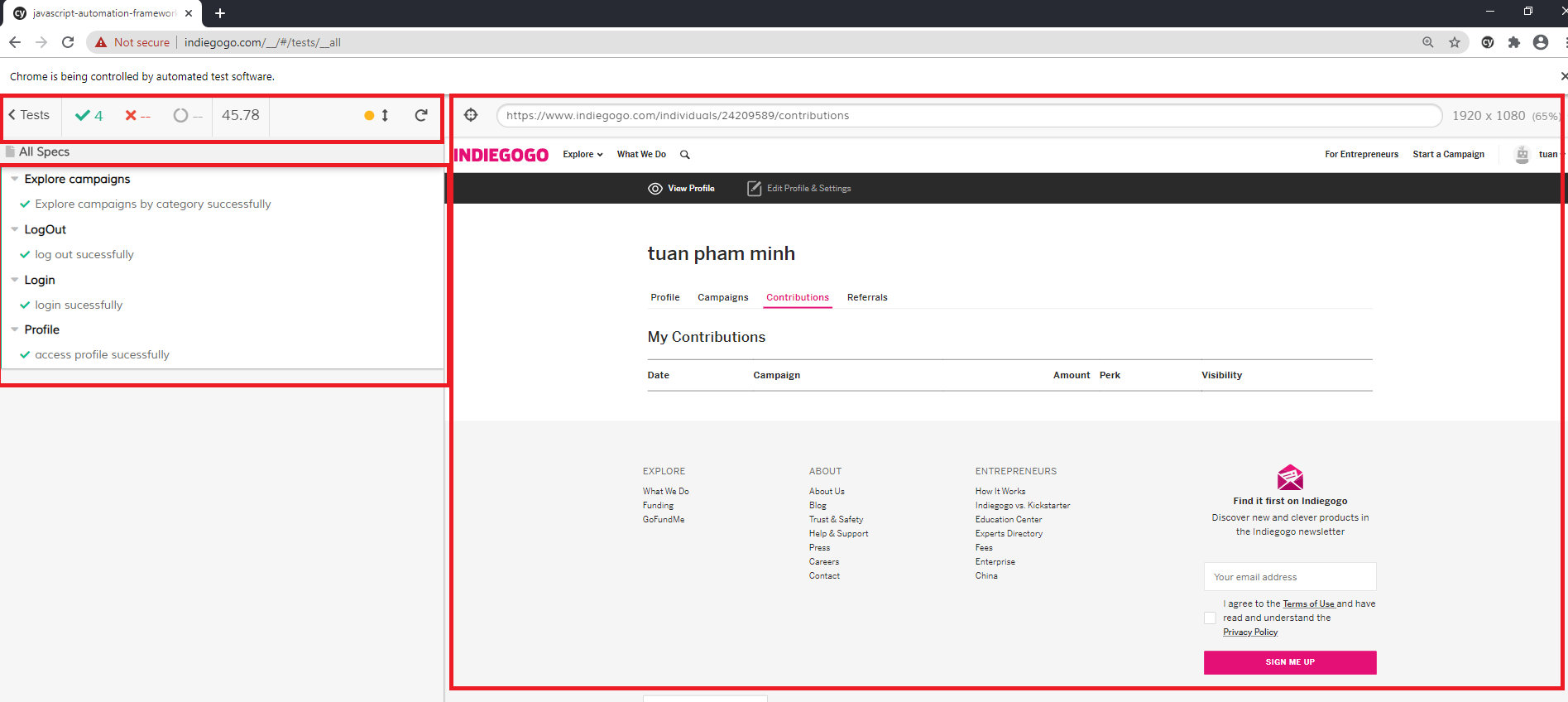
# 5. Run automation test script with cypress GUI mode

There are another way to run the test script, you can run the test script with GUI mode by run the command line **npx cypress open**. It is the friendly way to run and find the error when developing.

In the cypress GUI you can see all of the test case and you just click on that test case so it can run or run all test cases by clicking on “Run all specs”. For more information, you can chose the browser you want to run easily.



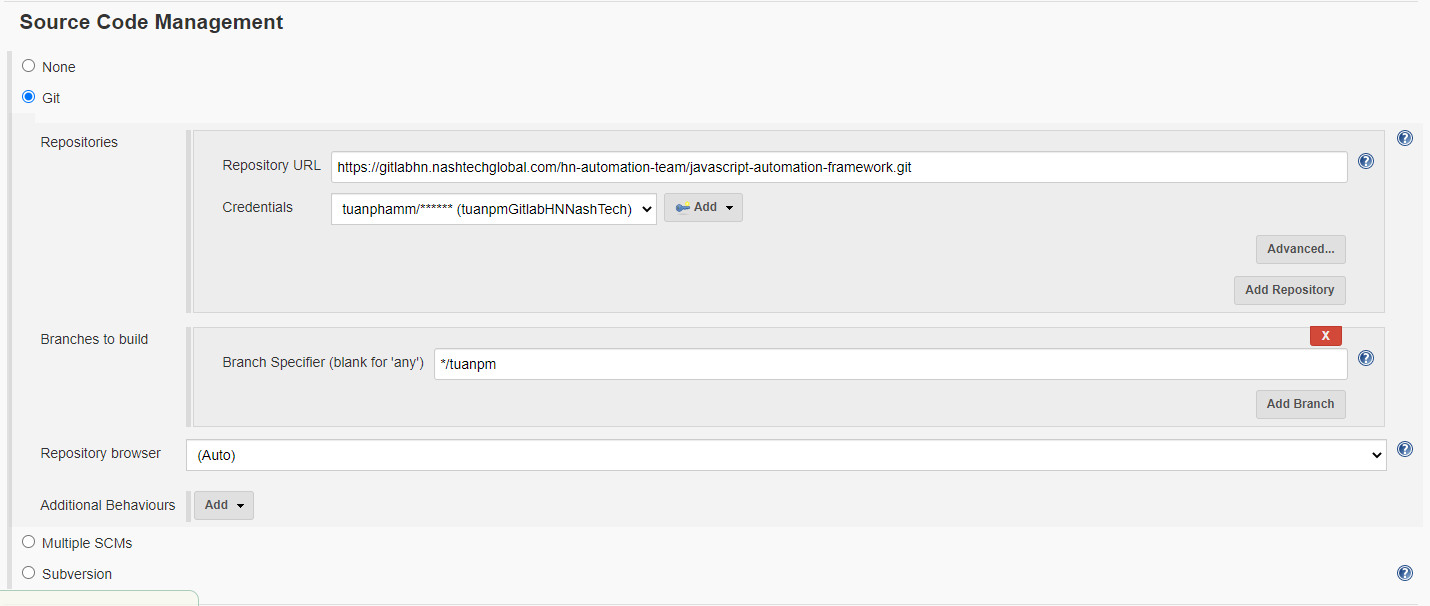
When run the test case, you can see the GUI of cypress contains 3 parts are **general result**, **detail result** **and the window to see how the script is running**. With this mode you can easily investigate the bug and fix and run again in the short time because, it will re-run every time you change the source code automatically.



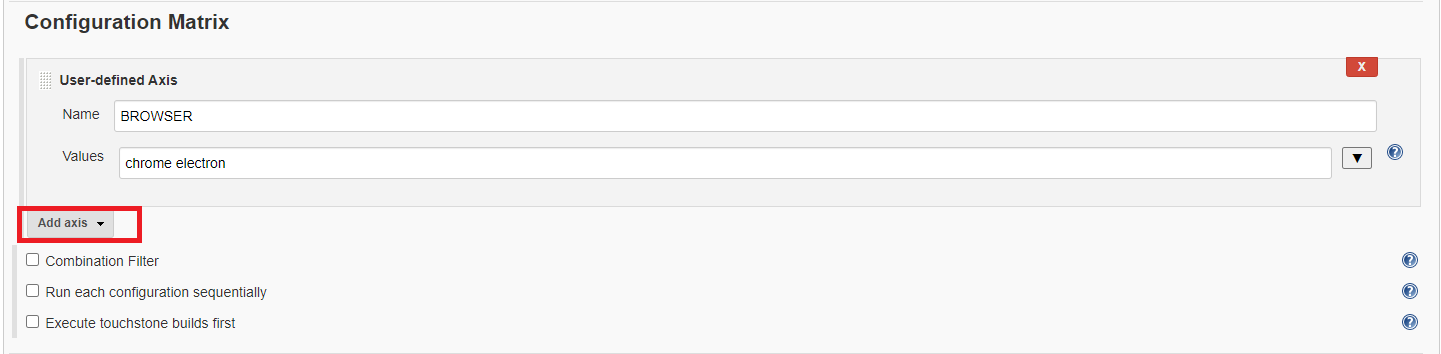
# 6. Continuous Integration with Jenkins

Assume that you already install Jenkins, so we can create a job to run the script in parallel for multi-browser (Chrome and electron) following these steps:

* Click on New Item
* Enter an item name (example: Cypress demo), select Multi-configuration project and click OK
* Configuration project:
  + Setup source code management



* + Configuration Matrix: to set variable **BROWSER** with 2 value is **chrome** and **electron** (It can be more than 2 value)



* + Build: create command to run the script



* + Click on “Save”
  + Click on “Build Now” to run the script in cross browser is Chrome and Electron with parallel process.

A close up of a logo

Description automatically generated

For more details, please send your enquiry to [info@nashtechglobal.com](mailto:info@nashtechglobal.com)

or visit our website [www.nashtechglobal.com](https://nashtechglobal.com/)

[@NashTech](https://twitter.com/NashTechHN)HN

[NashTech Limited](https://www.linkedin.com/company/nashtech-global/)

We are experts in technology, delivering smart solutions that solve business challenges and create value. Our award-winning teams apply deep expertise and passion to deliver complex IT projects globally.

**Author**

**Author**