# Project: Eduardo Perez

# Team: 6Strings

Completion date: 21st December 2018

# This document is to explain and show how 6Strings implemented Jenkins into the project of developing a Web application online order system.

# Jenkins – automated Pipeline for software integration for Continuous Delivery.

# Jenkins

What is Jenkins? It is a self-contained open source automation server which can be used to automate all sorts of tasks related to building, testing and deployment of software. <https://jenkins.io/doc/>

Jenkins is installed through native system packages, Docker (container)or run standalone by any machine with a JRE installed.(Java Runtime Environment)

## What is Jenkins Pipeline?

It is a suite of plugins which supports implementing and integrating continuous delivery pipelines into Jenkins.

For 6Strings the task is to create an automated pipeline by using Jenkins, GITHUB, GIT and Eclipse. Then show run a test class and show its output in the Jenkins Dashboard.

A CD pipeline is an automated expression of your process for getting software from version control right through to your users and customers. Every change in your software goes through multiple states of testing and deployment.

# Overview

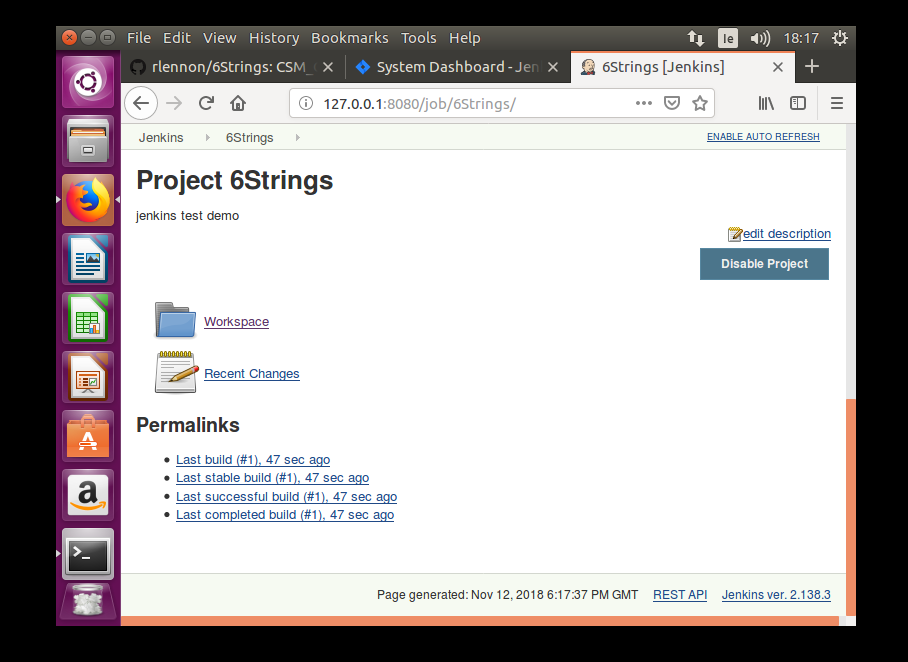
Jenkins (and in fact any CI systems) works best when builds are triggered after each and every commit. That provides you, the developer, with the most instant feedback possible. That, in turn, means you get fix(es) to the problems you introduced before you leave the context of what you were working on.

* Good analogy: Polling(“dad are we there yet are we there yet technique”) where you ask source control system at regular intervals
* Push –where the source control system tells you there has been a change.

The first technique requires no support from the source control system but puts a lot more load on both the CI system and the source control system. As well as that, it can only pick up changes as often as the polling frequency. The 6Strings build is set to every 15mins.  
The second technique requires support, in both the source control system and the CI (command line) system, for the sending and receiving of notifications. SCM –source control management.  
  
The Jenkins GitHub plugin provides support for receiving push notifications from Github post-commit hooks and using those notifications to trigger jobs within Jenkins. It can set-up the post-commit hooks in GitHub for you.  
  
In addition, GitHub decorates the Jenkins “Changes” pages with links to the pretty GitHub commit, issues pages and adds a sidebar link to the corresponding GitHub project page

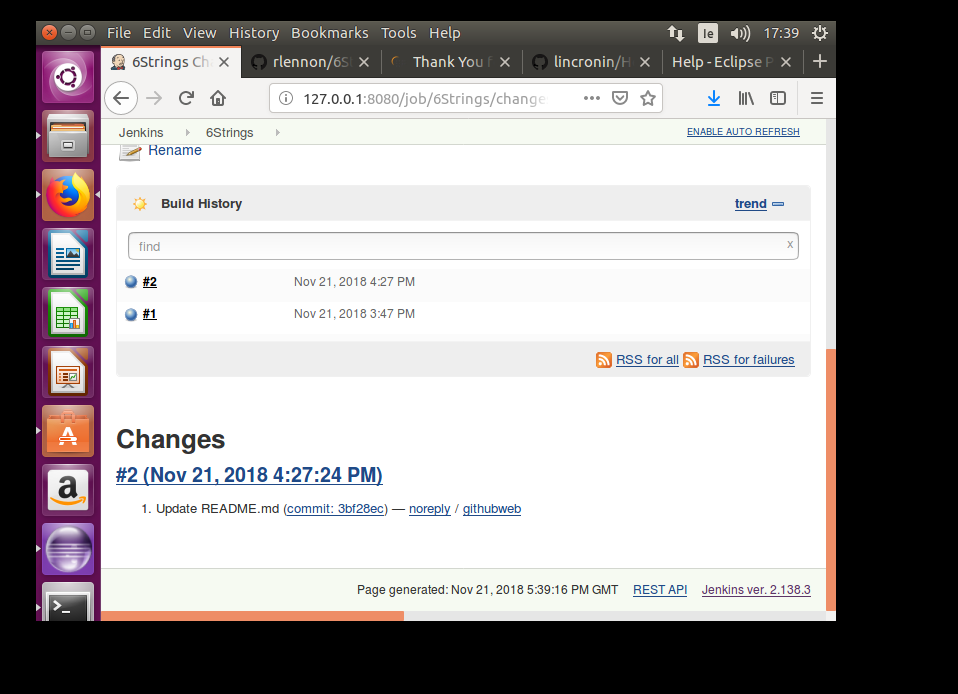
Github Plugin:

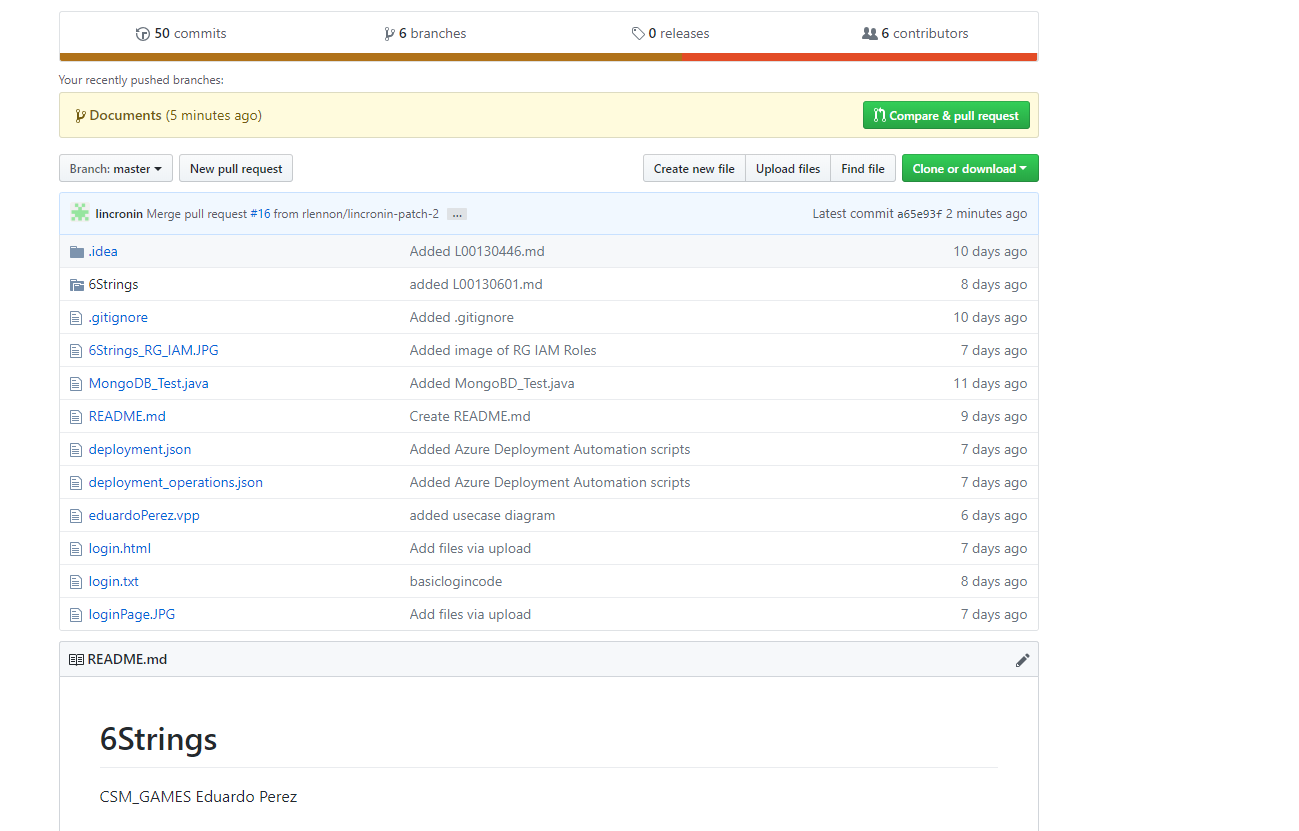
Integrates Jenkins with Github projects.



#### 

I tested this by editing the Readme file and ran the build history and it showed the changes in polling log also.





Functionalities:

* Create hyperlinks between your Jenkins projects and GitHub
* Trigger a job when you push to the repository by invoking HTTP POSTs from post-receive hook and optionally auto-managing the hook setup.
* Report build status result back to Github as Commit status.
* Base features for other plugins

Benefits of Jenkins:

* Automatically creates a Pipeline build for all branches and pull requests
* Code review and iteration on pipeline alongside of remain source code
* Audit trail for pipeline
* Single source of truth for pipeline, can be viewed and edited by other members of team.

Next stage is to install visual plugin to show dashboard of a testing of code and its status.

