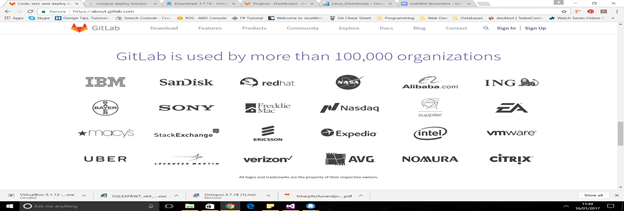
GitLab

History

* Launched in 2011 - 6 years ago
* Developed using Ruby - recently using Go - Google language
* Public, Private and Internal repositories
  + Internal - can be cloned by a logged in user

Overview

* Removes worry about installation, hosting or upgrades
* > 1400 open source contributors
* Friendly UI
* Detailed Documentation
* Internet application
* Native application available for: open source OS
  + Ubuntu
  + Debian
  + CentOS
  + OpenSUSE
  + Raspberry PI 2
* Similar to github
* Large host of organisations use GitLab
* Integrated CI & CD



Editions Available

* Editions available
  + Community Edition - free
    - Git Repository Hosting
    - Code Reviews
    - Issue Tracking
    - GitLab CI
    - Cycle Analytics
  + Enterprise Edition Starter - $39 per user per year
    - Community plus…
    - LDAP group sync
    - Repository mirroring
    - Merge request approvals
    - Push rules
    - Next business day support
  + Enterprise Edition Premium
    - GitLab GEO
    - Premium Support
    - Time Tracking
    - File locking
    - pivotal title
* GitLab.com Hosting - free
  + Hosting personal projects
  + Run Enterprise Edition starter
  + unlimited no of public & private repos
  + 10GB disk space per project
  + Unlimited Contributors
  + GitLab CI & Unlimited Runners
* Gitlab Hosted - $35 per month for up to 30 users
  + for teams who want their own private & secure GitLab instance without having to worry about installation, hosting or upgrades
  + private server hosted by GitLab
  + Run GitLab CE or EE
  + Automatic updates & backups

Features

Code, test & deploy together - move faster to production

Shorten feedback cycles

LDAP Group sync, git hooks, automated testing & audit trains - GitLab Enterprise

Gives administrators management capabilities without imposing on developers workflows

\*LDAP Lightweight Directory Access Protocol - Internet Protocol that email and other programs use to look up information from a server

Git hooks are scripts that Git executes before or after events such as: commit, push & receive

Gitlab pages - hosting static websites

Built in wiki / documentation

Import repositories from Github, Bitbucket, Google code, fogbugz gittea or any git repos

Chat

User roles

* Guest
* Reporter
* Developer
* Master
* Owner/Admin

Write permissions

ctrl read

built in wiki / documentation

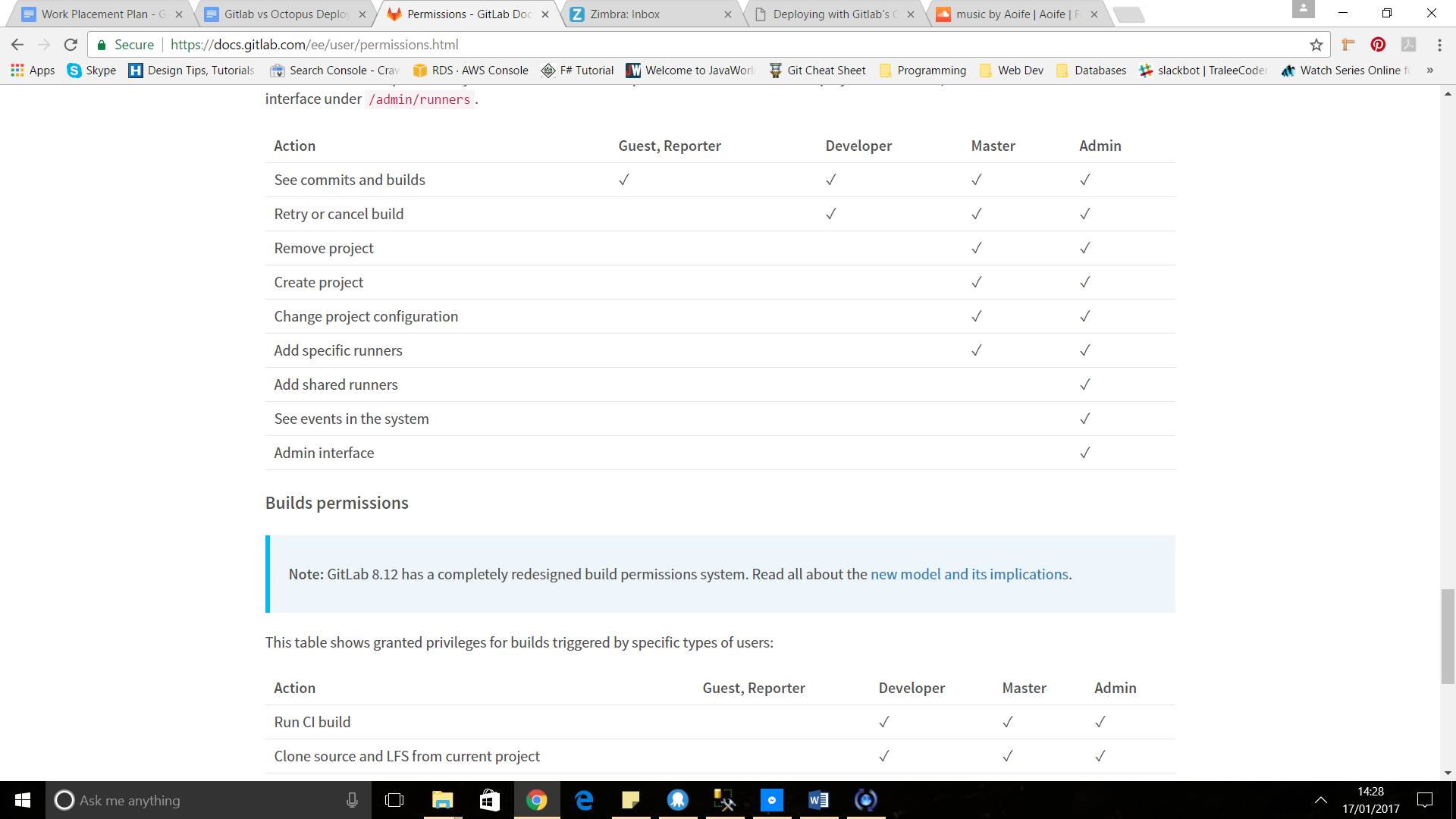
import from Github, Bitbucket, Google code, fogbugz gittea or any git repo

chat

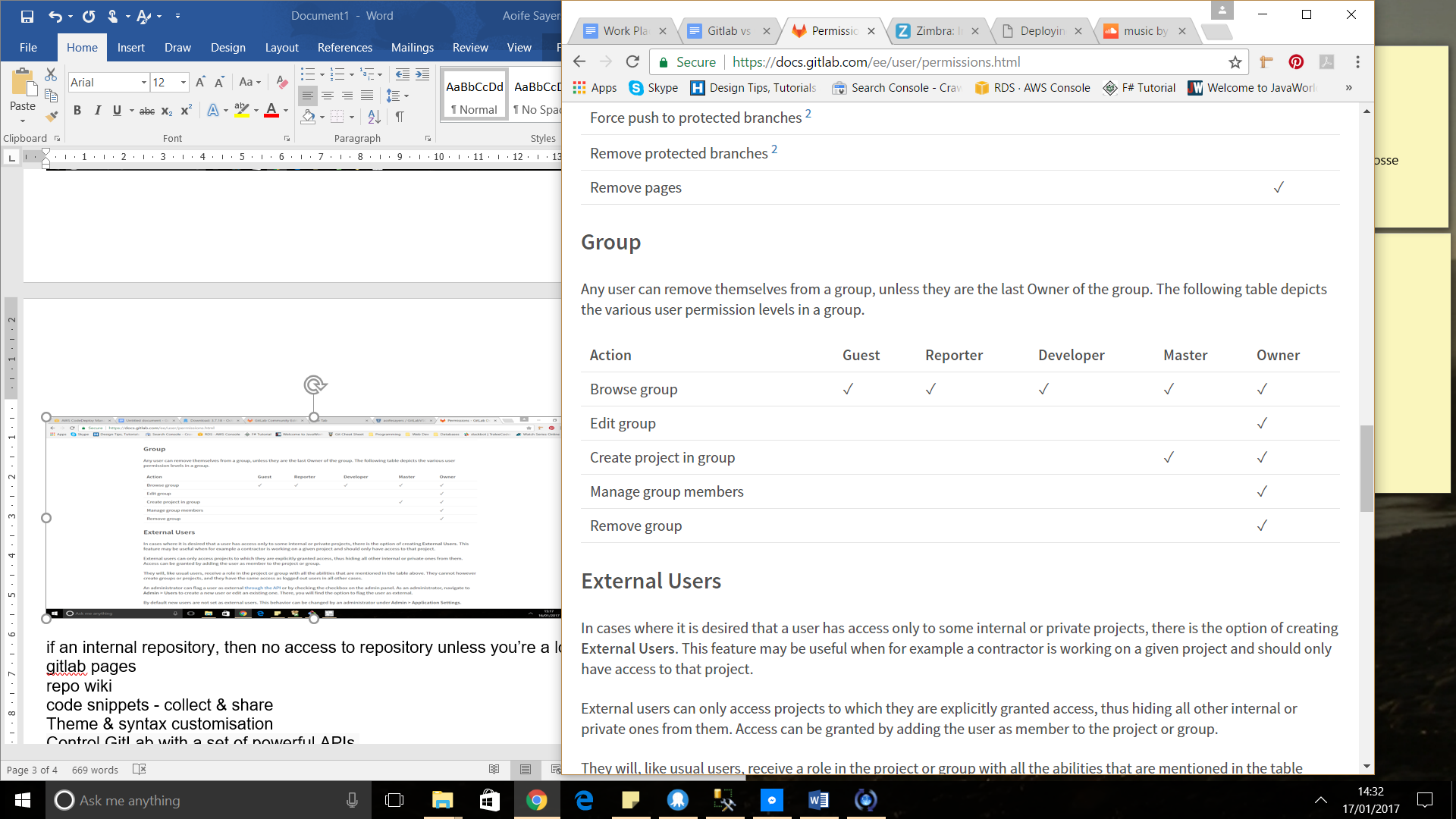
user roles

* Guest
* Reporter
* Developer
* Master
* Owner/Admin

Read & Write permissions



Groups



Code snippets - collect & share

Control GitLab with a set of powerful APIs.

Track & manage releases

Issues, issues board, todos, Labels, milestones, start & due dates, powerful search

Merge requests, resolve merge conflicts in UI, run multiple jobs in parallel, merge request version

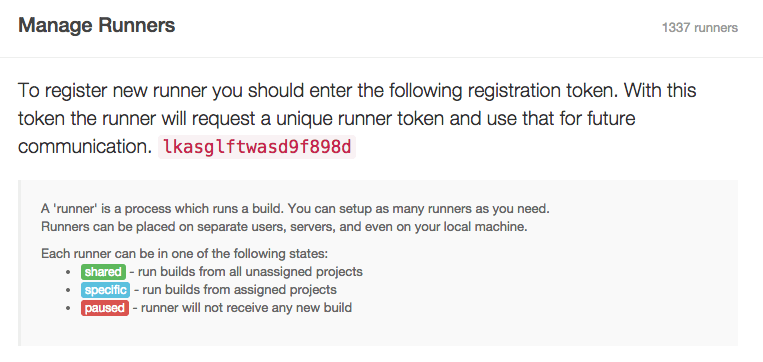
cherry pick changes

review apps - preview branch in the live environment

pipelines - build in stage

Control GitLab with a set of powerful APIs

Runners



**Shared runners** are useful for jobs that have similar requirements, between multiple projects. Rather than having multiple runners idling for many projects, you can have a single or a small number of runners that handle multiple projects. This makes it easier to maintain and update runners.

**Specific runners** are useful for jobs that have special requirements or for projects with a specific demand. If a job has certain requirements, you can set up the specific runner with this in mind, while not having to do this for all runners. For example, if you want to deploy a certain project, you can setup a specific runner to have the right credentials for this.

Pipelines

Merge Requests

Container Registry

an integrated Docker container registry means every project can have its own space to store its docker images

Versioned Tests

Autoscaling You can automatically spin up and down VMs to make sure your builds get processed immediately and minimize costs.

* Native Apps
  + GitLab Control - iOS
  + LabCoat - Android
  + Chrome GitLab notifier
* Command Line Interface CLI
* API Client
* Utilities
* Continuous Integration
* CMS Plugins
* GUI Git Clients
* Performance Measurement
* Login with your GitLab.com account
* Scrum Boards
* Built with GitLab
* GitLab Runners

Rivals

**Bitbucket**

**Atlassian**

**Amazon’s CodeCommit & CodePipeline**

**Microsoft Team Foundation Server**

**Github Enterprise**

GitLab does virtually everything Github does

GitLab open source

File & Repository Size limiations

github Recommended under 1gb

gitlab 10gb - Enterprise Edition - unlimited

User Limits

github file size limit 100mb

gitlab unlimited disk space - experience timeouts

lfs - large file storage

Pricing

|  |  |
| --- | --- |
| For GitLab | For Github |
|  |  |

LDAP

Easy deployment

Continuousintegration

Continuous Integration & Continuous Delivery & Continuous Deployment

Continuous Integration

Requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build

Solves problems quickly

* Developers check out code into their private workspaces.
* When done, commit the changes to the repository.
* The CI server monitors the repository and checks out changes when they occur.
* The CI server builds the system and runs unit and integration tests.
* The CI server releases deployable artefacts for testing.
* The CI server assigns a build label to the version of the code it just built.
* The CI server informs the team of the successful build.
* If the build or tests fail, the CI server alerts the team.
* The team fix the issue at the earliest opportunity.
* Continue to continually integrate and test throughout the project.

Continuous Delivery

Automatically tests the application

Manual deployment to production

Depends on organisations’ needs

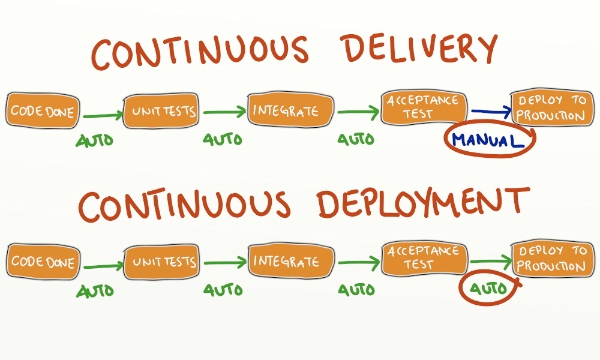
With Continuous Delivery your software is **always release-ready**, yet the timing of when to push it into production is a business decision, and so the final deployment is a **manual step**.

Continuous Delivery is the ability to get changes of all types—including new features, configuration changes, bug fixes and experiments—into production, or into the hands of users, safely and quickly in a sustainable way.

Continuous Deployment

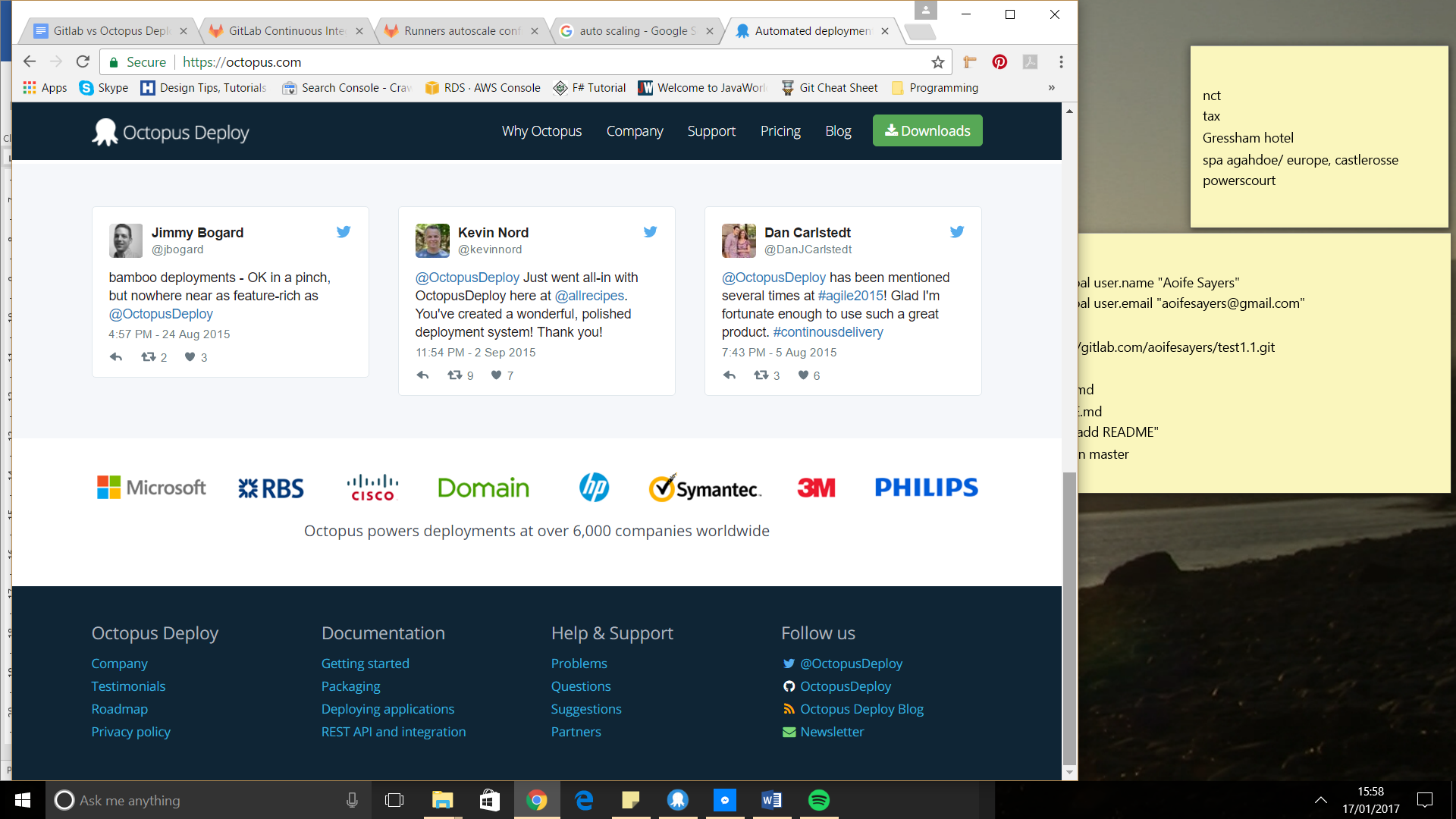
Every code change goes through the entire pipeline and is put into production, automatically, resulting in many production deployments every day.Continuous Deployment is closely related to Continuous Integration and refers to the release into production of software that passes the automated tests. Essentially, “it is the practice of releasing every good build to users,” explains Jez Humble, author of Continuous Delivery.

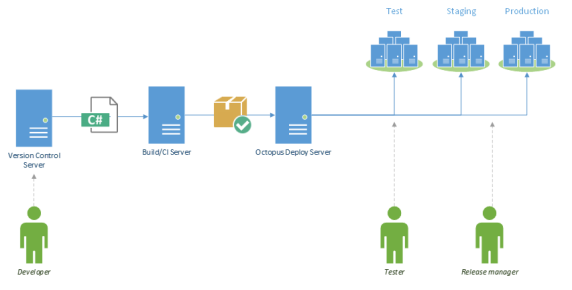
With Continuous Deployment, any updated working version of the application is **automatically pushed** to production. Continuous Deployment mandates Continuous Delivery, but the opposite is not required.



Octopus Deploy

Built for .NET developers





Build servers build. Octopus deploys

* Pulling code from your source control system
* Compiling the code
* Build-related tasks, like static code analysis
* Running unit tests and tracking code coverage over time

Octopus is great at:

* Distributing applications to all the remote machines, securely
* Environment-specific configuration, like connection strings
* Configuring IIS sites and installing Windows Services
* Doing all of the above across many machines in parallel

SQL Server Manager & SQL database on localhost

integration with [JetBrains TeamCity](https://octopus.com/teamcity) and [Microsoft Team Foundation Server](http://docs.octopusdeploy.com/display/OD/Team+Foundation+Server), and has a [command line application](http://docs.octopusdeploy.com/display/OD/Octo.exe+Command+Line) for creating and deploying releases that can be integrated with any build server.

Octopus also comes with a [comprehensive REST API](http://docs.octopusdeploy.com/display/OD/Octopus+REST+API) and a [.NET client library](http://docs.octopusdeploy.com/display/OD/Octopus.Client); anything that can be done through the UI can be done through the API. In fact, we built the UI and all of the integrations on top of that REST API.

It makes it easy to answer questions like:

* When did we last deploy to production?
* Does the version in staging match the version in production?
* Was last nights QA deployment successful?

deployment automation server

Native app for admin tasks on Octopus deploy

Until deployed to production

Tentacle Agent - easily deploy to Azure or Amazon cloud

Octopus deploy suitable for Agile teams

To compliment source control

Control test & builds unit tests

* Stored in home folder C:\Octopus artefacts, Nuget Packages & Server logs
* Service Account - local system or custom domain account
* Database - external or local - give db owner rights
* web portal http listen port - 80 & virtual directory - primary ui you & team will use to interact with octopus
* Authentication - Active directory or username/passwords stored in octopus

Package

* + Like a zip file
  + NuGet Package format
  + Metadata - id & version
  + Content files

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Environments

name & desc

e.g. Test Staging Production

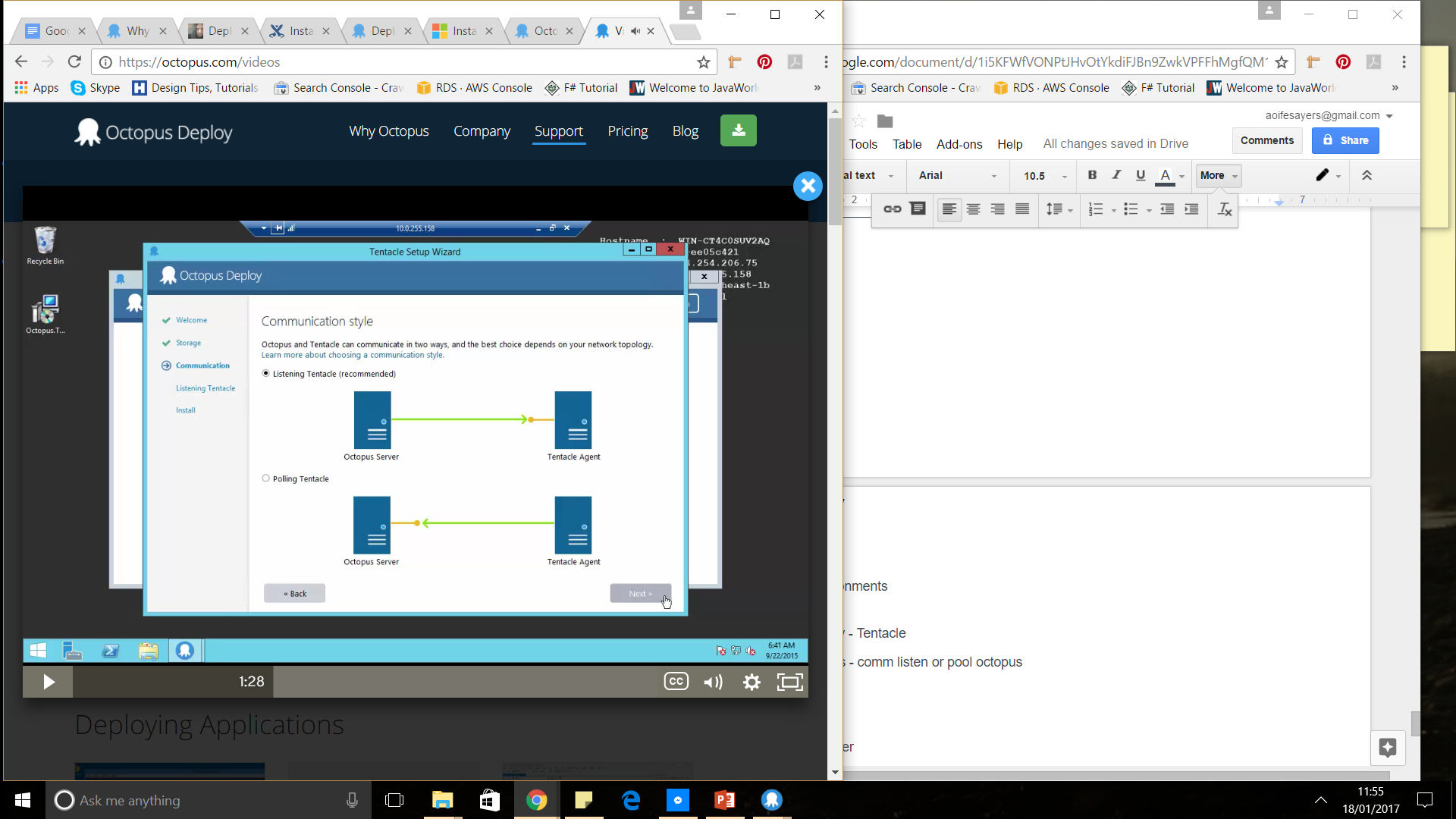
Guided Failure Mode - Octopus Deploy will prompt for user intervention if a deployment fails the environment

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

IIS **Internet Information Server**

**n extensible** [**web server**](https://en.wikipedia.org/wiki/Web_server) **created by** [**Microsoft**](https://en.wikipedia.org/wiki/Microsoft) **for use with** [**Windows NT**](https://en.wikipedia.org/wiki/Windows_NT) **family.**[**[2]**](https://en.wikipedia.org/wiki/Internet_Information_Services#cite_note-2) **IIS supports** [**HTTP**](https://en.wikipedia.org/wiki/HTTP)**,** [**HTTPS**](https://en.wikipedia.org/wiki/HTTPS)**,** [**FTP**](https://en.wikipedia.org/wiki/File_Transfer_Protocol)**,** [**FTPS**](https://en.wikipedia.org/wiki/FTPS)**,** [**SMTP**](https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol) **and** [**NNTP**](https://en.wikipedia.org/wiki/Network_News_Transfer_Protocol)**.**

Adding machines to environments



* To remotely deploy - Tentacle
* Tentacle & octopus - comm listen or pool octopus
* listening tentacle - next to no resources
  + tcp listen port 10933
  + tick add windows firewall exception
  + environments > add deployment target
  + octopus thumbprint
* polling tentacle
* tentacle msi installer

