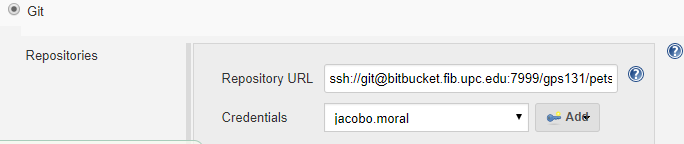
GPS - Grup 131

# Integració Contínua amb Jenkins

Amb la utilització de Jenkins, treballarem amb una pipeline, la qual estarà formada per un conjunt de tasques (jobs) els quals s’executaran, un darrere l’altre en realitzar-se alguna acció establerta, per tal d’automatitzar en certa manera el fet de realitzar unes quantes tasques les quals té sentit que es facin juntes per tal de mantenir en bon estat el nostre projecte.

Per començar, hem creat un nou “job” per fer tota la part del *Build* del nostre projecte. Per configurar-lo, hem posat la URL del ssh del nostre projecte del bitbucket com a URL com a repositori a Jenkins, indicant la branca sobre la qual volíem treballar, a més d’usar les credencials d’un dels membres del grup, per tal de poder realitzar la compilació havent verificat que la realitza un membre del nostre equip.

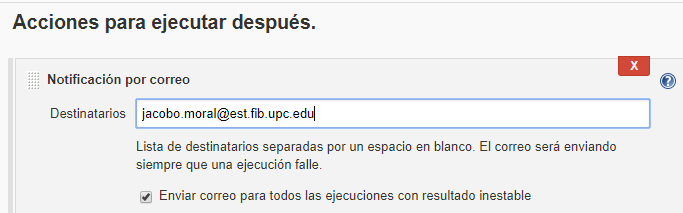


Com a build triggers, amb els que s’indiquen una sèrie de condicions sobre les quals, si es compleixen, realitzarem el build del nostre projecte, hem marcat el de poll SCM i un altre, el qual s’encarrega de fer build cada cop que es realitza un push del nostre projecte al bitbucket.



Com a accions “post-build”, que són totes aquelles accions que no es faran fins que ja s’hagi fet un build del nostre projecte, hem indicat que per cada build realitzat el qual deixi el nostre repositori com a “inestable”, és a dir, que el nostre projecte doni alguna mena d’error, ens enviï un mail per tal de notificar-nos.

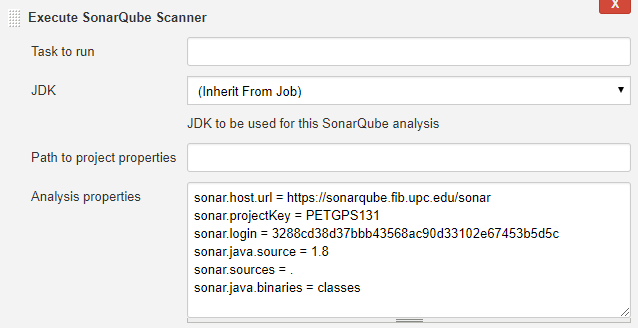




Ara, un cop tot funciona correctament, procedirem a la part de la realització dels *tests*, de manera que un cop modificat el build.gradle (per canviar l’encoding i així evitar certs problemes de format) tan sols haurem d’invocar-lo, i d’aquesta manera es realitzarà la comanda de comprovació de les nostres proves.



Un cop feta aquesta tasca, haurem de fer nou job, de l’apartat de *deploy*, el qual integrarà la nostra pipeline amb SonarQube. En aquest, el que hem fet és activar un plugin el qual permet a Jenkins connectar-se fàcilment a SonarQube, de manera que s’executi el codi en aquest, per tal de poder actualitzar l’estat del nostre projecte i així poder veure el nombre d’errors d’aquests, bugs, vulnerabilities… Entre d’altres. La configuració d’aquest és molt simple, ja que tan sols hem afegit l'URL del servei, la versió i el token d’un dels usuaris que té accés al projecte per tal d’autenticar-nos. A més, caldrà instal·lar un escànner per tal que realitzi les comprovacions pertinents a SonarQube. Un cop fet això, caldrà executar aquest escàner i fixar les propietats d’aquest, i així és com aconseguirem fer la integració completa d’aquest servei a Jenkins.





Com a últim pas, hem fet la part extra de la pràctica, amb la qual utilitzem l’eina Heroku, que ens permet fer un release del nostre projecte, és a dir, llençar la nostra aplicació per tal que estigui disponible des de internet.

El primer que hem fet ha sigut instal·lar l’aplicació, però mitjançant una altra comanda de les que se’ns deia, ja que no funcionava. Hem hagut d’usar *#brew install heroku/brew/heroku*.

Després l’hem configurat des de terminal, utilitzant les següents comandes:

*# heroku create petstore-gpsxyz --buildpack heroku/gradle*,

*# heroku config:set GRADLE\_TASK="build --x test"* i

*# git:remote --ssh-git --app petstore-alumne*.

Tot i que en un principi no ens funcionava, després de diverses modificacions que hem realitzat amb l’ajut d’internet, hem aconseguit configurar el heroku.

Alguna d’aquestes modificacions important per tal de fer-lo funcionar han sigut unes línies del fitxer *build.grade*, i hem creat dos nous fitxers, *procfile* i *production.properties*.

Un cop fet això, hem creat al server les taules mitjançant el sgbd MySQL i amb les credencials donades per Heroku.

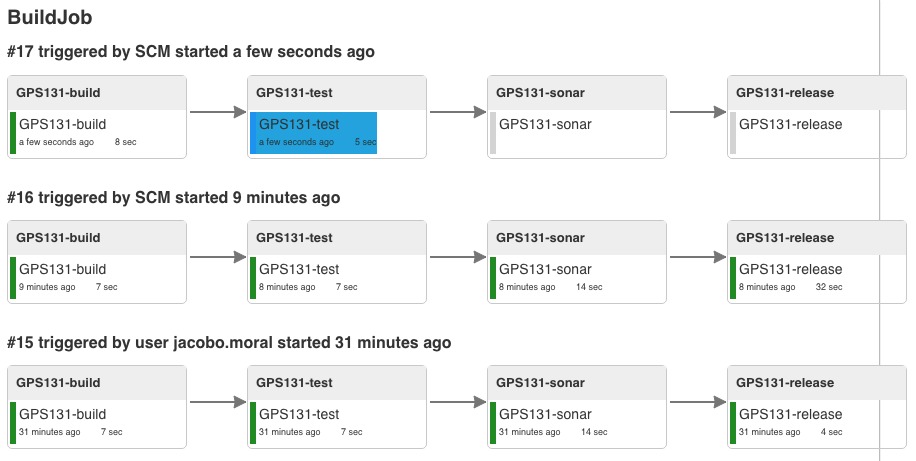
Ja autenticats, la web ja estava operativa al server de Heroku, i només ens faltava fer un nou *job* al Jenkins per tal d’automatitzar el release del projecte un cop ha passat els tests i el deploy.

Per això hem agafat les private key i public key ja utilitzades per linkar Jenkins i BitBucket.

Amb tota la feina realitzada fins ara, al fer un push es realitzaran tots els jobs descrits anteriorment, i cada cop que es realitza un d’ells, al menú principal del Jenkins podrem veure en tot moment si aquestes execucions s’han aconseguit realitzar correctament, o si hi ha hagut cap mena de problema o error, així com un historial per poder veure com ha sigut el transcurs de les execucions anteriors.

A més, hem creat dues vistes de pipeline al Jenkins, per poder observar el pipeline sencer i veure a temps real el seu funcionament.

A la primera imatge podem veure un *Delivery Pipeline.*



A les següents imatges un pipeline durant l’execució del deploy al SonarQube i un pipeline un cop finalitzada