Escola Superior Tecnologia de Castelo Branco

Computação em Nuvem

Cloud 4: Plataforma como Serviço (PaaS)

Trabalho Prático: Introdução ao Git

Docente: Gabriel Beck dos Santos
Discente: João Miguel Ramiro da Fonseca

Exercício 1

O Git é um sistema de controlo de versão, de código aberto e livre, desenvolvido para gerir grandes ou pequenos projetos de forma eficiente, sendo possível gerir ficheiros de texto, imagem, entre outros.

O controlo de versão é um sistema que permite gravar mudanças efetuadas num ficheiro ou conjunto de ficheiros ao longo do tempo, permitindo recuperar versões antigas.

Exercício 2

Existem diversos sitemas de controlo de versão, como seja o CVS, Subversion, Perforce, Bazaar, entre outros. No entanto importa referir que estes sistemas se diferenciam do Git, uma vez que o Git pensa os dados como uma série de snapshots de um sistema de ficheiros em miniatura, enquanto os outros sistemas pensam os dados como um conjunto de ficheiros e de alterações feitas a cada ficheiro ao longo do tempo, ou seja, *delta-based version control*.

Exercício 3

A utilização de um sistema de controlo de versão despoleta algumas vantagens. Podemos considerar uma delas o ambiente colaborativo, pensemos que temos 3 pessoas a trabalhar num mesmo projeto, e utilizam uma pasta partilhada onde colocam o ficheiro em que estão a trabalhar, isto irá provocar uma enorme quantidade de ficheiros, com nomes semelhantes e imensas versões, gerando ineficiência e baixa produtividade devido à má organização e ficando altamente susceptivel a ser acedido e modificado por um membro que não o deveria ter feito. Com um SCV cada pessoa pode trabalhar livremente, em qualquer ficheiro em qualquer altura, e no fim o SCV permite juntar todas as versões e formar um único ficheiro *clean*.

O SCV permite efetuar um claro armazenamento de versões, entendendo o ficheiro como um um único, assim existe apenas uma versão no disco, aquela em que o desenvolvedor está a trabalhar. Por outro lado, o desenvolvedor pode criar vias alternativas ao *branch* principal, e no fim juntar tudo num único ficheiro. Desta forma a organização de ficheiros é mais rápida e limpa, cada equipa pode trabalhar independentemente da outra, numa mesma pasta, sem correr o risco de se comprometerem o trabalho de uma e de outra.

Outra grande vantagem é o restauro de versões antigas, enquanto numa pasta sem o SCV, se o utilizador apagar o ficheiro e o apagar do *Recycle Bin*, perde o seu tempo e eventualmente parte do trabalho da equipa. Com o SCV cria-se uma camada adicional de proteção, assim mesmo se a situação enunciada ocorrer, o utilizador pode recorrer ao SCV e recuperar determinada versão ou versões especificas. No fundo isto leva-nos a outra vantagem que é o de permitir recuperar facilmente versões antigas, o que pode ser necessário se a versão atual em que o utilizador está a trabalhar se revelar um insucesso.

Por último e como complemento das anteriores está a organização do trabalho, que leva a acréscimos de produtividade. Sempre que um utilizador cria uma nova versão o SCV vai lhe pedir que crie uma pequena descripção daquela alteração, este pequeno texto vai ser de extrema utilidade quer em recuperação de versões, que em perceber os passos que foram dados ao longo do projeto, permitindo a um novo elemente, entender rapidamente todo o desenvolvimento conceptual do projeto.

Exercício 4

Utilizando o Git CLI para iniciar o serviço numa dada pasta o utilizador deve com o comando *cd* mover-se para a pasta onde quer que o Git inicie para a gerir e em seguida iniciar o serviço com o comando: *git init* . Isto levará a que o serviço incie, sendo criada a *branch* master à qual podem ser adicionados ficheiros e em seguida criação de versões.

Exercício 5

Em primeiro lugar é necessário adicionar o ficheiro ou conjunto de ficheiros ao Git. Como tal pode ser usado o comando *git add .* , se pretendermos adicionar todos os documentos na pasta em que estamos a trabalhar ou *git add nome-doc* , após esta acção teremos de começar a criar as versões, isto é alcançado com o comando *git commit -m "descrição da alteração/versão"* . Por forma a verificar se foi de facto criada uma nova versão no Git introduz-se o comando *git log*.

Exercício 6

O comando *checkout* permite-nos trocar de *branch*, isto é útil se começarmos um trabalho na *branch* master que é digamos o corpo/tronco principal e necessitarmos ou quisermos criar ramificações auxiliares deste tronco principal. Além disso este comando perimite-nos trocar de *branch* em qualquer momento e para aquela que desejarmos.

Exercício 7

Tal como explicado ligeiramente no exercício anterior uma *branch* é criada quando um utilizador pretende criar ramificações do corpo ou da linha principal do projeto, trabalhando desta forma independente da linha principal ou de outras ramificações, permitindo que não interfira nem seja interferido por outros utilizadores a trabalhar noutras *branch*. Esta é a digamos que a funcionalidade mais importante do Git, uma vez que comparada com outros SCV, esta operação é muito mais leve, o que faz com que no Git esta operação seja quase instântanea, um Git Branch é resumidamente um apontador para um snapshot das alterações que se vão efetuando.

Exercício 8

Podemos dizer que existem três plataformas principais que suportam o uso de Git na internet, sendo elas o GitHub, o Bitbucket e o GitLab.

Comçando pelo GitHub, que é possivelmente a plataforma mais conhecida e com maior utilização com mais de 56 milhões de utilizadores, 3 milhões de organizações e 72% do Fortune 50. Em termos de preçário, o GitHub tem alguma variedade de oferta, existindo 4 escalões, um deles grátis, bem como planos educacionais e não lucrativos. Este repositório suporta quer repositório públicos, quer privados, incluindo na modalidade grátis.

O Bitbucket é uma plataforma orientada para equipas profissionais, abrangendo 1 milhão de equipas e 10 milhões de uitlizadores. Quanto ao preço o bitbucket disponibiliza três escalões, sendo um

deles grátis. Oferece ainda preços especais para equipas superiores a 101 elementos. Aquando da criaçáo de um repositório Bitbucket cloud é possível especificar se o mesmo deve ser público ou privado.

Por último apresenta-se o GitLab. Esta plataforma apresenta quatro escalões de preço, sendo um dele grátis, contudo no escalão grátis os minutos CI/CD são mais reduzidos se comparados com o GitHub por exemplo. Possui 30 milhões de utilizadores. Enquanto o GitHub é bastante usado para produção e código-fonte aberto, o GitLab é muito utilizado para produção de código-fonte fechado. Tal como as outras duas plataformas suporta repositórios privados.