**Homework Day 2**

Il sistema che si vuole sviluppare prevede un repository remoto che funge da repository di riferimento o “***origin***”, in cui effettuare le ***push*** delle funzionalità.

Si assegnano i vari task ai membri del team, dando la possibilità ad ognuno di effettuare pull e ***push*** dal repository “***origin***”. Inoltre potrà essere possibile effettuare il pull dai repository degli altri elementi del team, in modo tale da poter formare altri sotto-team. In questo modo si potrà lavorare in team quando si presenteranno features più complesse, così da non rallentare il processo di sviluppo.

Il modello di sviluppo si basa sui due branch principali risiedenti sul repository remoto ovvero il master e il ***develop***, che hanno un ciclo di vita teoricamente infinito. Il ***develop*** si stacca dal master e conterrà tutte le funzionalità che, una volta testate verranno rilasciate. Il branch di ***develop*** rifletterà sempre uno stato con le ultime modifiche di sviluppo consegnate per la prossima ***release*** e sarà anche il branch in cui si integreranno le successive features. Quando il codice sorgente nel ***develop*** ha raggiunto un punto stabile ed è pronto per essere rilasciato, tutte le modifiche saranno unite nel master e contrassegnate con un numero di rilascio.

***Mergiando*** le modifiche con il master si intende che le funzionalità possono andare in produzione.

Accanto ai branch principali si rende utile l’inserimento di ulteriori branch di supporto per lo sviluppo delle ***features***, per la risoluzione dei bug e per la preparazione al passaggio in produzione.

Dunque definiamo altri tre ulteriori branch, che però possiedono un ciclo di vita limitato:

* **Features** per le funzionalità
* **Hotfix** per i bug
* **Release** per le funzionalità da rilasciare

**Branch Features**

Si deve staccare dal ***develop*** e reinserirsi nuovamente nel ***develop*** quando le modifiche o la creazione delle nuove ***features*** saranno reputate terminate. Quindi il ciclo di vita di questo branch nasce quando lo stesso viene staccato dal ***develop*** e termina quando viene di nuovo ***mergiato*** sul ***develop*** ppure la funzionalità viene scartata.

Convenzionalmente si potranno utilizzare tutti i tipi di denominazione ad eccezione di quelle discusse in precedenza e nemmeno ***release*** e ***hotfix***.

Potranno essere staccati ulteriori branch dal branch di ***features***. In questo caso si intenderanno sviluppare ulteriori features per un rilascio futuro in una versione successiva del software.

**Branch Hotfix**

I branch di ***Hotfix*** non sono branch che possono partecipare al processo di planning del progetto, in quanto rappresentano l’improvviso manifestarsi di un grave errore relativo a una funzionalità importante.

Questo branch si staccherà direttamente dal master ed uno o più componenti del team si occuperanno di preparare una soluzione rapida per la correzione dell’errore. Per convenzione si useranno come naming per questi branch ***hotfix-X.Y.Z***, dove ***X*** indica il numero progressivo di versione del software da cui si è staccato il branch di ***Hotfix***, ***Y*** in numero di ***release*** e ***Z*** il numero di ***hotfix***. Lo stesso concetto sarà applicato alle ***commit***, applicando come messaggio delle commit la stringa “***Numero di versione x.y.z***”.

Ad esempio ***$ git commit -m "Numero di versione 1.2.1"***.

Il ciclo di vita di questo branch terminerà quando il problema sarà risolto e verrà effettuata una messa in produzione diretta nel master e successivamente eseguire una merge sul ***develop***, in modo tale da consentire al progetto di poter avanzare eliminando la problematica risolta nel branch ***Hotfix***.

**Release**

I branch di ***release*** sono branch che vengono inseriti nel flow quando all’interno del ***develop*** sono presenti una serie di features ritenute stabili e pronte per passare in produzione. Proprio per assegnare questo “stato” alle ***features*** vengono creati queste tipologie di branch.

Un branch di ***release*** viene staccato dal ***develop***, in questa fase si può procedere alla fase di ***testing*** ed alla correzione di bug minori prima di mandare in produzione la ***release***. Il ciclo di vita di questo branch termina quando viene messo in produzione e ***pushato*** sul master. Inoltre il branch ***release*** viene ***mergiato*** con il branch di ***develop***, così da rendere subito disponibili nel ***develop*** le nuove funzionalità.

Per convenzione il naming di questi branch sarà ***release-X.Y***, dove ***X*** indica il numero progressivo di versione del software e ***Y*** il numero di release.

Ad esempio ***$ git commit -m "Numero di versione 1.2”***.

**Operazioni da non eseguire per preservare l’integrità del progetto.**

Tra le operazioni che non devono essere eseguite per preservare l’integrità del progetto, non si devono integrare ***features*** senza aver eseguito un ***regressive*** ***test***, soprattutto nel caso in cui queste possano andare ad impattare su funzionalità preesistenti.

Altra operazione da evitare è la cancellazione del branch develop, perché potrebbe comportare un significativo rallentamento dello sviluppo del progetto in quanto si perderebbero delle features sviluppate ma non rilasciate sul master. In generale però essendo il sistema distribuito esisterà sempre una copia del develop ritenuta in locale da uno degli sviluppatori e dunque quest’ultima potrebbe ripristinare lo stato del progetto quantomeno ad una versione precedente se non ricopiare esattamente lo stesso branch develop eliminato.

**Definizione Pull Request**

Una ***pull request*** è un metodo per inviare contributi a un progetto di sviluppo, essa si verifica quando uno sviluppatore chiede al reviewer che le modifiche apportate a un repository esterno siano considerate per essere incluse nel repository principale del progetto.

Ad esempio quando viene sviluppata una ***feature*** nel sotto-branch ***feature*** del ***develop*** e si richiede al ***reviewer*** di considerare questa ***feature*** come candidata al ***merge*** nel branch padre.