Examen

1 -Définition du workflow gitFlow : GitFlow est une extension Git qui permet de gérer le versionning d'un projet ,au lieu d'une seule branch principale ,il utilise deux branche pour sauvegarder l'historique du projet.la branche master t develop,

La branche master elle permet de stocker l'historique des versions et la branche develop c'est la branche d'integration des différentes fonctionnalités dévloppées.
Ensuite y'a 3 branches qui sont : features,release,et hotfix.

On L'utilise pour le versionning d'un projet,surtout quand on est plusieurs à travailler sur un projet,il nous permet d'avoir une tracabilités du code et de collaborer tous ensemble au projet,il permet également de structurer les processus du dévloppement.

2- Avantage du Wokflow gitFlow :

- Chaque fonctionnalité a sa propre branche de développement
- Deux branch dites principal 'master' et develop
- Il Fournit un contrôle de version complet et détaillé parceque les fusions sont regroupées
- Le code de la branche master reste très propre et organisé
- Organiser le travail.
- Avoir une traçabilité du code.
- Il fournit une aide et une sortie de ligne de commande excellentes

3- Inconvénient du Workflow gitFlow :

- il contient de nombreuses branches avec de nombreuses règles compliquées

- la charge de travail de maintenance est importante pour les versions publiées
-

4- les branche :

- **Feature** : cette branche est dérivée de la branch develop ,elle est utilisée pour développer des différentes fonctionnalités,cette branche ne communiquent pas directement avec la branch principale, une fois les fonctionnalités terminées elle peuvent alors être fusionnées avec la branche develop

- **Hotfix** : appelée aussi branch « maintenance »c'est une branche qui est utilisée pour corriger les bugs sur la branch master,elle ressemble au branches release et features sauf qu'elle est basé sur la branch master et non sur la branch develop .

- **Release** : cette branch permet de relier la branche Develop et la branche Maste,elle est crée uniquement lorsque la branche de developpment est complète, quand beaucoups de fonctionnalités on été developpée sur la branch develop,on effectue un release de la branche master,elle est merger avec la branche master une fois que les test sont finis.

- **Develop** : cette branch est dérivée de la branche master , elle contient l'historique complet du code vu qu'elle recoit toutes les modification effectuées,elle permet d'integrer les différentes fonctionnalités prévues pour la version à venir,c'est également sur cette branche que les développeur fusionnent les branches de fonctionnalités.cette branche doit également restée stable.

- **Master** : c'est la branche principale ou la branche par défaut ,c'est elle qui contient la version finale du code qui est prêt au être déployé en production,elle se déplace automatiquement pour pointer vers la dernier commit effectué,on développe pas sur cette branche directement, on pase par la branch develop.

5- On se pose sur la branche Develop et on crée le tag

```
$ git tag -a <tag_name> -m "message"
$ git push –tags
```

6- On stash les modifications qu'on est en train de faire et on balance sur la branche master

```
git stash
```

-On commence par créer la branche du hotfix :

```
git checkout -b hotfix-x.x.x master
```

Ensuite,on fait la correction

```
git commit -m "fix probleme"
```

Ensuite on commit dans les branches master et develop

```
git checkout master
git merge --no-ff hotfix-x.x.x
git tag -a x.x.x -am "message"
git checkout develop
git merge --no-ff hotfix-x.x.x
```

Ensuite on supprime la branche du hotfix

```
git branch -d hotfix-x.x.x
```

Enfin on push

```
git push origin master
git push origin develop
```

git push –tags

7- les commandes a executer : git merge release prod

8- Git stash : cette commande permet de stocker d'une manière temporaire les modifications apportées à une copie du travail pour faire un autre travail et revenir plus tard pour les appliquer.

On peut l'utiliser quand on travaille sur un projet et on veut changer de branch mais qu'on veut pas commit un travail à moitié fini,cette commande permet donc de changer de branch sans valider la branch actuelle.

Pour annuler un git Stash :

git stash apply

git stash pop.