SKXXX - Git Workflow

Purpose - Standardize the development process to ensure master is always Production quality

Policy

Overview of the Git Workflow

```
1. $ git fetch upstream
2. $ git merge upstream/master
3. $ git checkout -b proj/sks-reports
3a. $ git add (files)
4. $ git commit -S
5. $ git checkout master
6. $ git merge --no-ff --verify-signatures -S proj/sks-reports
7. $ git push origin master
8. $ git branch -d proj/sks-reports
```

1. $ git fetch upstream

Imagine a large project was devised and assigned to you. Before you even begin anything, make sure your local copy of the remote repository is up-to-date.

2. $ git merge upstream/master

Once the local copy of your remote repository contains the updates, you can merge the contents into your local master branch.

3. $ git checkout -b proj/sks-reports

Create a new branch immediately after your local master is in sync with upstream master. All changes from here on out will remain in the proj/sks-reports branch.

4. $ git commit -S

Once you are done with your changes, you will be committing with a signed GPG key. After entering a standardized Git commit message, it will prompt you for a password.

5. $ git checkout master

After proj/sks-reports is completed, you will have to prepare to merge it back into your local master branch.

6. $ git merge --no-ff --verify-signatures -S proj/sks-reports

Merging proj/sks-reports back into your local master branch with the recursive strategy. At the same time, the GPG signature will be verified.

7. $ git push origin master

Push the reports into your remote origin master so a Pull Request can be initiated

8. $ git branch -d proj/sks-reports

Assuming proj/sks-reports works flawlessly, you can delete the local branch.