**Why Git?**

Git is a distributed code version repository where we get the benefit of keeping the feature, dev, rnd, production changes separately and get the confidence of keeping the code on a centralized server clean. All conflicts resolve locally.

The main branches.

* features
* develop
* qa
* release (or main)

1. Feature branch should be used for any particular feature or bug. Ideally the name should be trello id so we can track for which story this was done. Once local dev testing is done and develop is confident, a merge request should be raised to merge that feature branch into develop.
2. Develop branch is the branch where all the feature branches will get merged and it will be deployed on the dev server. The approver should do the code review and either merge or ask for changes. Once approver is satisfied, he should merge the feature branch into develop and delete the feature branch (Option is there when approving merge request)
3. If a release is planned then develop should be merged to qa branch and this will be deployed on qa server where testing will be done. Whatever bugs come during qa they will be fixed and merged to both develop and qa.
4. Once qa branch changes reflects a stable build it will be merged to release branch and tag with release number will be created. Changes denoted by that tag will be deployed on the production server. If any production issue comes then changes will be happened by creating a feature branch from release branch and changes will be merged to release

**Some Tips**

1. When creating a feature from develop branch

* git checkout -b myfeature develop

1. Create merge request after myfeature changes are done into develop
2. Merge myfeature into develop (direct if you wish to do without merge request)

* git merge –no-ff myfeature

1. Delete myfeature

* git branch -d myfeature

**CI / CD**

**CI / CD Steps (implemented in script)**

1. Select the staging based on the branch (development, qa or production)
2. Setup the private key for SSH
3. SSH to correct server based on the staging (develop, qa or production)
4. Pull the latest code from correct branch
5. Stop docker containers
6. Remove docker images
7. Create new docker images
8. Run the containers
9. **Gitlab pipeline from develop branch to dev server (AWS)**

Branch: develop

Stage: develop

Server: [ubuntu@ec2-13-232-5-49.ap-south-1.compute.amazonaws.com](mailto:ubuntu@ec2-13-232-5-49.ap-south-1.compute.amazonaws.com)

1. **Gitlab pipeline from qa branch to qa server (AWS)**

Branch: qa

Stage: test

Server: [ubuntu@ec2-15-207-94-37.ap-south-1.compute.amazonaws.com](mailto:ubuntu@ec2-15-207-94-37.ap-south-1.compute.amazonaws.com)

1. **Gitlab pipeline from release (main) tag to prod server (AWS)**

Branch: master

Stage: production

Server: [ubuntu@ec2-13-233-126-200.ap-south-1.compute.amazonaws.com](mailto:ubuntu@ec2-13-233-126-200.ap-south-1.compute.amazonaws.com)

* Refer to the file [**.gitlab-ci.yml**](https://gitlab.com/gautam_icc/10xtd_website/-/blob/master/.gitlab-ci.yml) in the repository for more details.
* Gitlab account used on server to pull the latest code is (deployer / PitsVWA5sxzxQgVhsfr7)