# Contribution Guideline

## Git workflow

### First time setup

[Fork the repo if you are outside collaborator https://github.com/iluminar/goodwork. ]

1. Clone the project to your local pc

``` bash

$ git clone git@github.com:iluminar/goodwork.git

```

for outside collaborator

``` bash

$ git clone git@github.com:{yourusername}/goodwork.git

```

2. Configure remotes:

``` bash

$ cd goodwork

$ git remote add origin git@github.com:iluminar/goodwork.git

```

3. Setup application

#### Using docker

> Docker & Docker Compose must be installed

a) Copy `.env` from `.env.example`

```

cp .env.example .env

```

b) Define environment variable

Set app port, db credentials and mail driver credentials in `.env` file. Also provide `SSL\_CERT\_DOMAIN`(your domain) and `SSL\_CERT\_DIR`(directory of your ssl certificates file) values for ssl setup.(Not needed for local development)

c) Run the installer

```bash

sudo chmod +x install.sh

./install.sh local

```

> To stop server: `./cmd stop` & to start server `./cmd start`

Login Credentials for Admin User

`email: admin@example.com`

`password: secret`

#### Install Manualy

``` bash

$ git checkout dev

$ git pull origin dev

$ composer install

$ cp .env.example .env

$ php artisan key:generate

$ php artisan telescope:install

$ npm install

$ php artisan migrate

```

### Regular workflow

1. Create a branch for new issue from `dev` branch latest commit (the branch name should be as “issue[number]”. E.g. – “issue7”):

``` bash

$ git checkout -b issue[number]

```

2. Develop on issue[number] branch, but Do not merge the upstream master with your development branch!!

3. Run test to ensure all tests are passed before committing

4. Commit changes to issue[number] branch:

``` bash

$ git add .

$ git commit -m "commit message"

```

5. Push branch to Github:

``` bash

$ git push origin issue[number]

```

6. Repeat steps 2-5 till development is complete.

7. Fetch upstream changes that were done by other contributors:

``` bash

$ git fetch origin

```

8. Update local `dev` branch:

``` bash

$ git checkout dev

$ git pull origin dev

```

9. Rebase issue[number] branch on top of the upstream `dev`:

``` bash

$ git checkout issue[number]

$ git rebase dev

```

10. In the process of the rebase, it may discover conflicts. In that case it will stop and allow you to fix the conflicts. After fixing conflicts,run all test to ensure nothing breaks and then just run:

``` bash

$ git add .

$ git rebase --continue

```

11. Push branch to GitHub (force changes to your issue branch, it is not a common branch). Rebasing means recreation of commits, so noway to push without force. NEVER force to common branch.:

``` bash

$ git push origin issue[number] --force

```

12. Send a Pull Request.

Attention: Please, recheck that in your pull request you send only your changes, and no other changes!!

Check it by command:

``` bash

$ git diff issue[number] upstream/dev

```

13. Remove issue branch after successful pull request:

``` bash

$ git branch -D issue[number]

$ git push origin :issue[number]

```

\*\*N.B:\*\*

\* Never push to `master` or `dev` branch

\* Work on `feature` branch

\* Don't name your local working branch `feature`, rather give it name of your task id e.g `issue7`

\* Test before you push. Do not push half-done work.

\* Create a pull request in `dev` branch

\* Code will be reviewed by peers, if everything seems right then code will be merged

### Commit and messages

\* Commit message should be brief but descriptive enough to convey the message of what has been done in this commit, example

> Add delete user option in the dashboard

Or like this

> Replace jQuery onReady listener with plain JS; fixes #13

\* Add issue number in final commit message when the task is complete

## Coding style

Goodwork follows the [PSR-2](https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-2-coding-style-guide.md) coding standard and the [PSR-4](https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-4-autoloader.md) autoloading standard and StandardJS style for Javascript file [StandardJS](https://standardjs.com/rules.html).

## Project workflow

\* All the task will be created as a issue with appropriate label and milestone

\* All the task will also be available in project board of the repository

\* By default all task will be at `Ideas` column

\* You can self-assign yourself to a task or task can be assigned to you

\* When a member start a task, the card should be moved to `In Progress` column

\* When someone test a task other than the original contributor after task is complete it should be placed in `In Review`

\* When a task is reviewed by a peer the task should be placed in `Done` column

\* Only one member should work in a single task