# Contributing

We're glad you want to help us out and make this panel the best that it can be! We have a few simple things to follow when making changes to files and adding new features.

### Project Branches

This section mainly applies to those with read/write access to our repositories, but can be helpful for others.

The `develop` branch should always be in a runnable state, and not contain any major breaking features. For the most part, this means you will need to create `feature/` branches in order to add new functionality or change how things work. When making a feature branch, if it is referencing something in the issue tracker, please title the branch `feature/PTDL-###` where `###` is the issue number.

Moving forward all commits from contributors should be in the form of a PR, unless it is something we have previously discussed as being able to be pushed right into `develop`.

All new code should contain unit tests at a minimum (where applicable). There is a lot of uncovered code currently, so as you are doing things please be looking for places that you can write tests.

### Update the CHANGELOG

When adding something that is new, fixed, changed, or security-related for the next release you should be adding a note to the CHANGELOG. If something is changing within the same version (i.e. fixing a bug introduced but not released) it should \_not\_ go into the CHANGELOG.

### Code Guidelines

We are a `PSR-4` and `PSR-0` compliant project, so please follow those guidelines at a minimum. In addition, StyleCI runs on all of our code to ensure the formatting is standardized across everything. When a PR is made StyleCI will analyze your code and make a pull to that branch if necessary to fix any formatting issues. This project also ships with a PHP-CS configuration file and you are welcome to configure your local environment to make use of that.

All class variable declarations should be in alphabetical order, and constructor arguments should be in alphabetical order based on the classname. See the example below for how this should look, or check out any of the `app/Service` files for examples.

```php

class ProcessScheduleService

{

protected $repository;

protected $runnerService;

public function \_\_construct(RunTaskService $runnerService, ScheduleRepositoryInterface $repository)

{

$this->repository = $repository;

$this->runnerService = $runnerService;

}

```

### Responsible Disclosure

This is a fairly in-depth project and makes use of a lot of parts. We strive to keep everything as secure as possible and welcome you to take a look at the code provided in this project yourself. We do ask that you be considerate of others who are using the software and not publicly disclose security issues without contacting us first by email.

We'll make a deal with you: if you contact us by email and we fail to respond to you within a week you are welcome to publicly disclose whatever issue you have found. We understand how frustrating it is when you find something big and no one will respond to you. This holds us to a standard of providing prompt attention to any issues that arise and keeping this community safe.

If you've found what you believe is a security issue please email us at `support@pterodactyl.io`.

### Where to find Us

You can find us in a couple places online. First and foremost, we're active right here on Github. If you encounter a bug or other problems, open an issue on here for us to take a look at it. We also accept feature requests here as well.

You can also find us on [Discord](https://pterodactyl.io/discord). In the event that you need to get in contact with us privately feel free to contact us at `support@pterodactyl.io`. Try not to email us with requests for support regarding the panel, we'll probably just direct you to our Discord.