# Contributing

We welcome contributions to jrnl, whether it's through reporting bugs, improving the documentation, testing releases, engaging in discussion on features and bugs, or writing code.

## Table of Contents

\* [Code of Conduct](#code-of-conduct)

\* [Reporting Bugs](#reporting-bugs)

\* [Editing Documentation](#editing-documentation)

\* [Testing](#testing)

\* [Submitting feature requests and ideas](#submitting-feature-requests-and-ideas)

\* [Developing](#developing)

## Code of Conduct

Before starting, please read the [Code of Conduct](CODE\_OF\_CONDUCT.md).

## Reporting Bugs

Please report bugs by [opening a new issue](https://github.com/jrnl-org/jrnl/issues/new/choose) and describing it as well as possible. Many bugs are specific to a particular operating system and Python version, so please include that information!

## Editing Documentation

If you find a typo or a mistake in the docs, please fix it right away and send a pull request.

To edit the documentation, edit the `docs/\*.md` files on the \*\*develop\*\* branch. You can see the result if you run `make html` (or `poetry run mkdocs serve` if on Windows) inside the project's root directory, then navigating your browser to [locahost:8000](http://localhost:8000).

The `gh-pages` branch is automatically maintained and generated after your changes are merged. You should never have to edit that branch.

### Recipes and external editors

If you'd like to share a jrnl command line trick that you find useful, or advice on how to integrate a particular external editor, you may find it worthwile to add it to the ["Recipes" section](docs/recipes.md).

## Testing

Much of the work of maintaining jrnl involves testing rather than coding.

The nature of jrnl means we deal with extremely sensitive data, and can't risk data loss. While jrnl does have a comprehensive automated testing suite, user testing is crucial to mitigating this risk.

### Prereleases

[Prereleases are deployed through PyPi much like normal releases](https://pypi.org/project/jrnl/#history). You can use [pipx](https://pypi.org/project/pipx/) to fetch them and test them. See the [changelog](CHANGELOG.md) for information on what has changed with each release.

### Pull requests

If you are comfortable enough with git, feel free to fetch particular [pull requests](https://github.com/jrnl-org/jrnl/pulls), test them yourself, and report back your findings. Bonus points if you can add a screencast of how the new feature works.

### Confirm bug reports

There are always [open bugs among our GitHub issues](https://github.com/jrnl-org/jrnl/issues?q=is%3Aissue+is%3Aopen+label%3Abug) and many are specific to a particular OS, Python version, or jrnl version. A simple comment like "Confirmed on jrnl v2.2, MacOS 10.15, Python 3.8.1" would be extremely helpful in tracking down bugs.

### Automate tests

See the develop section below for information on how to contribute new automated tests.

## Submitting feature requests and ideas

If you have a feature request or idea for jrnl, please [open a new issue](https://github.com/jrnl-org/jrnl/issues/new/choose) and describe the goal of the feature, and any relevant use cases. We'll discuss the issue with you, and decide if it's a good fit for the project.

When discussing new features, please keep in mind our design goals. jrnl strives to do one thing well. To us, that means:

\* be \_slim\_

\* have a simple interface

\* avoid duplicating functionality

## Developing

### Getting your environment set up

You will need to install [poetry](https://poetry.eustace.io/) to develop jrnl. It will take care of all of the project's other dependencies.

### Understanding the branches

jrnl uses two primary branches:

\* `develop` - for ongoing development

\* `release` - for releases

In general, pull requests should be made on the `develop` branch.

### Common development commands

You can find an inventory of commands in the `makefile`. \\*nix users can run the commands by typing `make` followed by the name of the command; however, Windows users will need to type out the commands directly, or install a third-party make tool such as [GNU Make for Windows](http://gnuwin32.sourceforge.net/packages/make.htm).

A typical development workflow includes:

\* Running tests: `make test`

\* Running the source in a virtual environment:

\* `poetry install`

\* `poetry shell`

\* `jrnl` (with or without arguments as necessary)

\* Linting the code to standardize its style: `make lint`

### Updating automated tests

When resolving bugs or adding new functionality, please add tests to prevent that functionality from breaking in the future. If you notice any functionality that isn't covered in the tests, feel free to submit a test-only pull request as well.

For integration testing, jrnl uses [behave](https://behave.readthedocs.io/) tests, which are all in the `features` folder.

Many tests can be created by only editing `feature` files with the same format as other tests. For more complicated functionality, you may need to implement steps in `features/steps` which are then executed by your tests in the `feature` files.

Starting in 2020, jrnl is also using [pytest](https://docs.pytest.org/) for unit tests. These tests are in the `tests` folder.

### Submitting pull requests

When you're ready, feel free to submit a pull request (PR). The jrnl maintainers generally review the pull requests every two weeks, but the continuous integration pipeline will run on automated tests on it within a matter of minutes and will report back any issues it has found with your code across a variety of environments.

The pull request template contains a checklist full of housekeeping items. Please fill them out as necessary when you submit.

If a pull request contains failing tests, it probably will not be reviewed, and it definitely will not be approved. However, if you need help resolving a failing test, please mention that in your PR.

### Finding things to work on

You can search the [jrnl GitHub issues](https://github.com/jrnl-org/jrnl/issues) by [label](https://github.com/jrnl-org/jrnl/labels) for things to work on. Here are some labels worth searching:

\* [critical](https://github.com/jrnl-org/jrnl/labels/critical)

\* [help wanted](https://github.com/jrnl-org/jrnl/labels/help%20wanted)

\* [bug](https://github.com/jrnl-org/jrnl/labels/bug)

\* [enhancement](https://github.com/jrnl-org/jrnl/labels/enhancement)

### A note for new programmers and programmers new to python

Although jrnl has grown quite a bit since its inception, the overall complexity (for an end-user program) is fairly low, and we hope you'll find the code easy enough to understand.

If you have a question, please don't hesitate to ask! Python is known for its welcoming community and openness to novice programmers, so feel free to fork the code and play around with it! If you create something you want to share with us, please create a pull request. We never expect pull requests to be perfect, idiomatic, instantly mergeable code. We can work through it together!