Contributing to Brightlayer UI

Thank you for helping to make Brightlayer UI projects great and being a part of the Brightlayer UI community. Here are a few guidelines that will help you along the way.

Code of Conduct

Brightlayer UI has adopted the Contributor Covenant as its Code of Conduct, and we expect project participants to adhere to it. Please read the document so that you can understand what will and will not be tolerated.

Contributions

There are many ways to contribute to Brightlayer UI (code contribution is just one aspect of it). Here are some ways you can help:

- Reporting and/or fixing bugs
- Suggesting and/or implementing new features
- Adding to or improving our documentation
- Adding new code examples, components, or libraries

Your first Pull Request

If you decide to fix an issue, please check the comments in the issues section on the repository to see if somebody is working on a fix. If nobody is working on it, please leave a comment stating that you have started to work on it so other people don't accidentally duplicate your effort. If you decide to fix an issue that has not been logged, please log the issue first and leave a comment.

Branch Naming and Pull Request Workflow

We have an established convention for branch naming and commit messages following the rules below. We ask that you also adhere to these rules to maximize the likelihood of your submission being accepted.

Branch Naming Conventions:

Branch names should:

- be prefixed with `feature/` or `bug/` depending on what you're working on
- use only lowercase alphanumeric characters
- use dashes as the word separator (no spaces or underscores)
- be concise but descriptive

``` feature/my-topic-branch or bug/fix-topic-branch ```

Commit Message Conventions:

Commit messages should:

- capitalize the subject line
- **not** end the subject with a period
- use the imperative mood in the subject line, e.g., `Add capability X` instead of `Added capability X`
- be concise but descriptive

Pull Request:

Pull requests should:

- Target `dev` branch
- Have a clear and descriptive title
- Adhere to the Brightlayer UI Pull Request template
- Reference any issue(s) that are fixed (i.e., 'Fixes #XX')
- Briefly describe at a high level what is being proposed
- Use a bulleted list to highlight the key changes/additions

Sending a Pull Request

Brightlayer UI is a community with many projects, so pull requests are always welcome. If you are planning to work on something, please open an issue first (bug or feature request) to discuss it with the maintainers of Brightlayer UI. Keep pull requests small to have the best chance of getting accepted. Several small pull requests are easier to review than one large one.

> ***Pro Tip:**: Don't bundle more than one feature or bug fix per pull request.*

Get started

1. Fork the relevant Brightlayer UI repository to your GitHub account.

2. Clone the fork repository to your local machine and add upstream remote:

``` git clone https://github.com// -b cd git remote add upstream https://github.com/etn-ccis/ ```

3. Synchronize your local branch with the upstream one:

``` git pull upstream ```

4. Install the dependencies:

``` yarn install ```

5. Create a new branch:

``` git checkout -b feature/my-topic-branch ```

or

``` git checkout -b bug/my-topic-branch

```

6. Make changes and verify the code. Pull requests will only be accepted if they pass the following mandatory checks:

- the code is formatted via `yarn prettier` with the Brightlayer UI prettier configuration
- the code is linted via `yarn lint` using the Brightlayer UI ESLint configuration
- the code builds successfully via `yarn build` (where appropriate).
- the code passes all tests via `yarn test` (some repositories include additional tests that must also pass).
- the branch name, commit messages, and pull request formats adhere to the requirements described above.

7. Commit and push to your fork:

``` git push –set-upstream origin ```

8. Go to the proper Brightlayer UI repository and make a Pull Request.

The maintainers of Brightlayer UI will be notified and will review your pull request and either merge it, request changes to it, or close it with an explanation.

> *The `master` branch in Brightlayer UI repositories represents the latest released/published code. The `dev` branch is used to hold the latest updates and features that are waiting to be released.*

Get support from Brightlayer UI

Contact Us

Roadmap

Want to know where Brightlayer UI is heading or ideas on where you could contribute, take a look at the roadmap.

License

By contributing to the Brightlayer UI GitHub organization, you agree to license your contribution under the BSD-3-Clause license.