Development Cycle

1. Start Your Dev server

Now that you have everything set up, you should start your application development server provided by your UI framework or bundler (assuming you're using one, of course).

> **NOTE**
> Every framework has its own development tooling. It is outside of the scope of this document to cover them all or stay up to date.

2. Start Tauri Development Window

npm Yarn pnpm Cargo

```
$ cargo tauri dev
```

The first time you run this command, the Rust package manager takes several minutes to download and build all the required packages. Since they are cached, subsequent builds are much faster, as only your code needs rebuilding.

Once Rust has finished building, the webview opens, displaying your web app. You can make changes to your web app, and if your tooling enables it, the webview should update automatically, just like a browser. When you make changes to your Rust files, they are rebuilt automatically, and your app automatically restarts.

> **ABOUT CARGO.TOML AND SOURCE CONTROL**
> In your project repository, you SHOULD commit the "src-tauri/Cargo.lock" along with the "src-tauri/Cargo.toml" to git because Cargo uses the lockfile to provide deterministic builds. As a result, it is recommended that all applications check in their Cargo.lock. You SHOULD NOT commit the "src-tauri/target" folder or any of its contents.

Edit this page

Last updated on ***Jul 16, 2022***