Framework workflow

Yes, that's correct! You'll develop your framework code, organize it into modules, and then bundle it into an npm package that users can install and use in their projects.

Here's a summarized workflow:

1. Develop Your Framework: Write the code for your frontend framework. Organize it into modules, components, utilities, etc., just like you would for any JavaScript project.

2. Export Your Framework: Export your framework code using Common JS (`module. exports`) or ES6 modules (`export`/`import`) so that it can be imported by other projects.

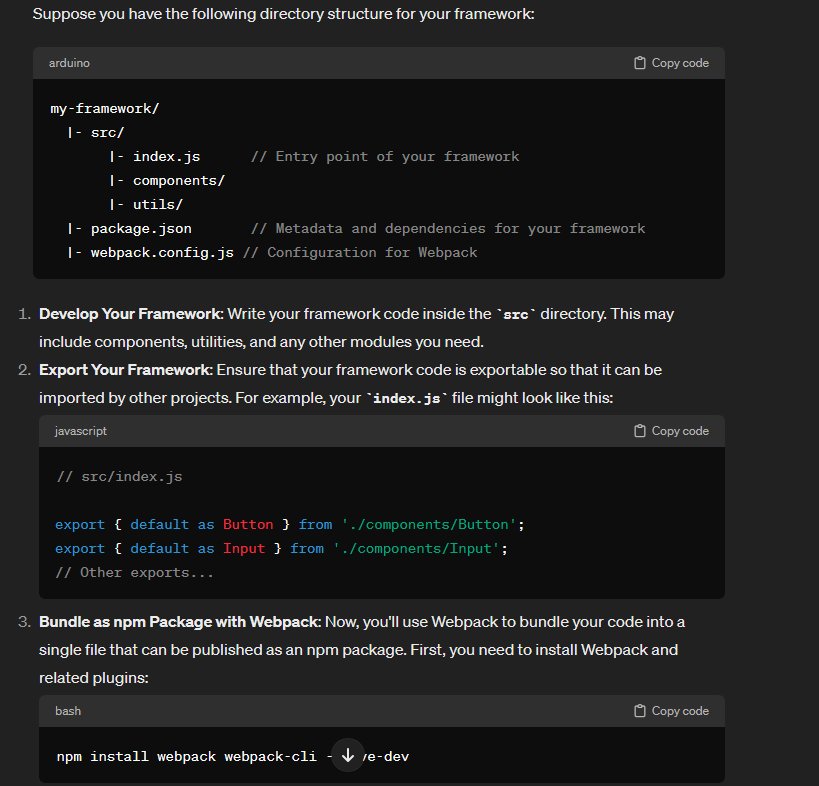
3. Bundle as npm Package: Once you have a working version of your framework, bundle it into an npm package. You can use tools like Web pack or Rollup to bundle your code into a single file that can be easily distributed and installed via npm.

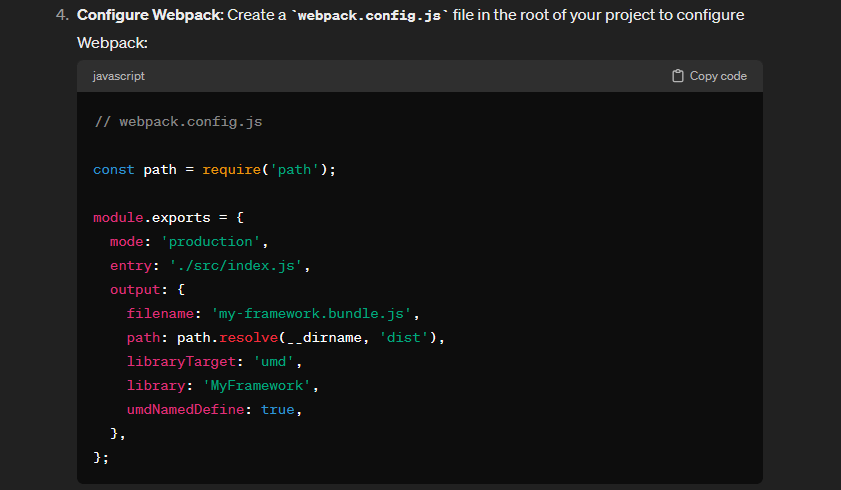
4. Publish to npm: Publish your npm package to the npm registry so that others can install it using `npm install`. You'll need to create an account on npm if you don't already have one and use the `npm publish` command to publish your package.

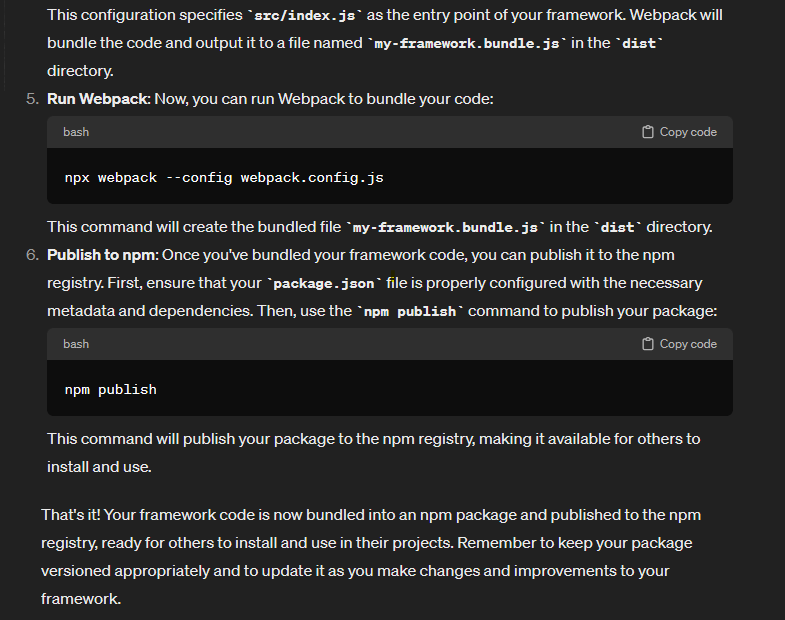
5. Create CLI Tool (Optional): Optionally, you can create a CLI tool as described earlier to streamline the process of setting up new projects with your framework.

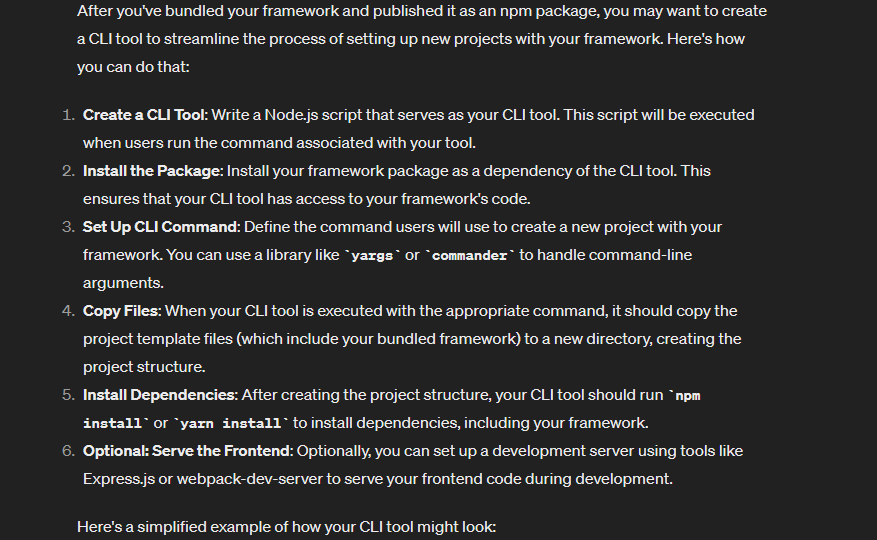
6. Documentation: Write documentation for your framework, including usage instructions, API references, examples, and any other relevant information. This will help users understand how to use your framework effectively.

By bundling your framework as an npm package, you make it easy for other developers to install and use it in their projects, which can help increase its adoption and popularity.





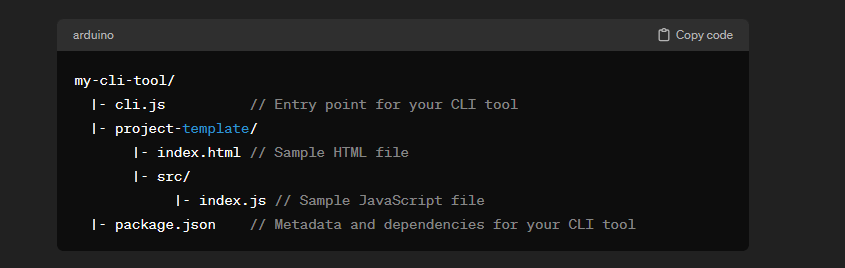




To publish your CLI tool and project template as an npm package, follow these steps:

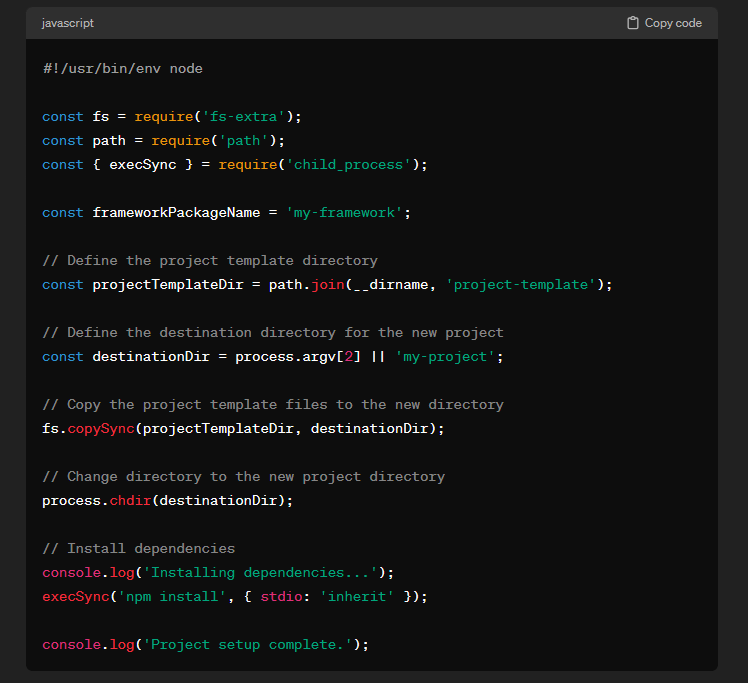
1. Create your project structure:

Set up your directory structure for your CLI tool and project template:



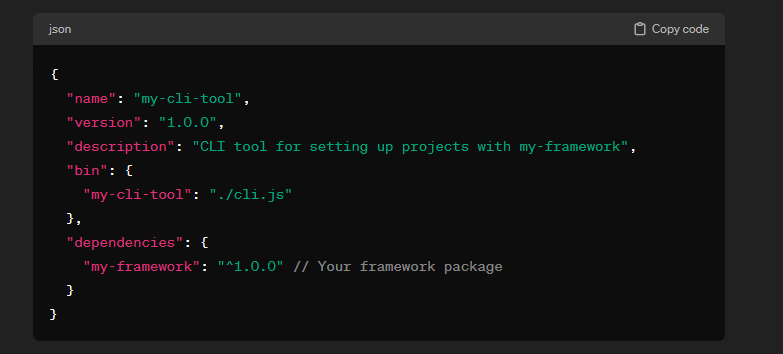
2. Write your CLI tool:

Write the logic for your CLI tool in `cli.js`. This script will be executed when users run your CLI command.



3. Write your `package.json`:

Define the metadata and dependencies for your CLI tool in `package.json`.



4. Write your project template:

Create a project template in the `project-template` directory. This directory will be copied to the destination directory when users run your CLI tool.

5. Test your CLI tool:

Test your CLI tool locally to ensure it sets up projects correctly.

6. Publish your CLI tool:

Once you're ready to publish, log in to your npm account using `npm login`, then publish your package using `npm publish`.

7. Share your CLI tool:

Share the name of your package with others so they can install it using `npm install -g my-cli-tool`. Once installed globally, they can use your CLI command (`my-cli-tool`) to create new projects with your project template and framework.

That's it! You've now published your CLI tool and project template as an npm package, allowing others to easily create projects with your framework.