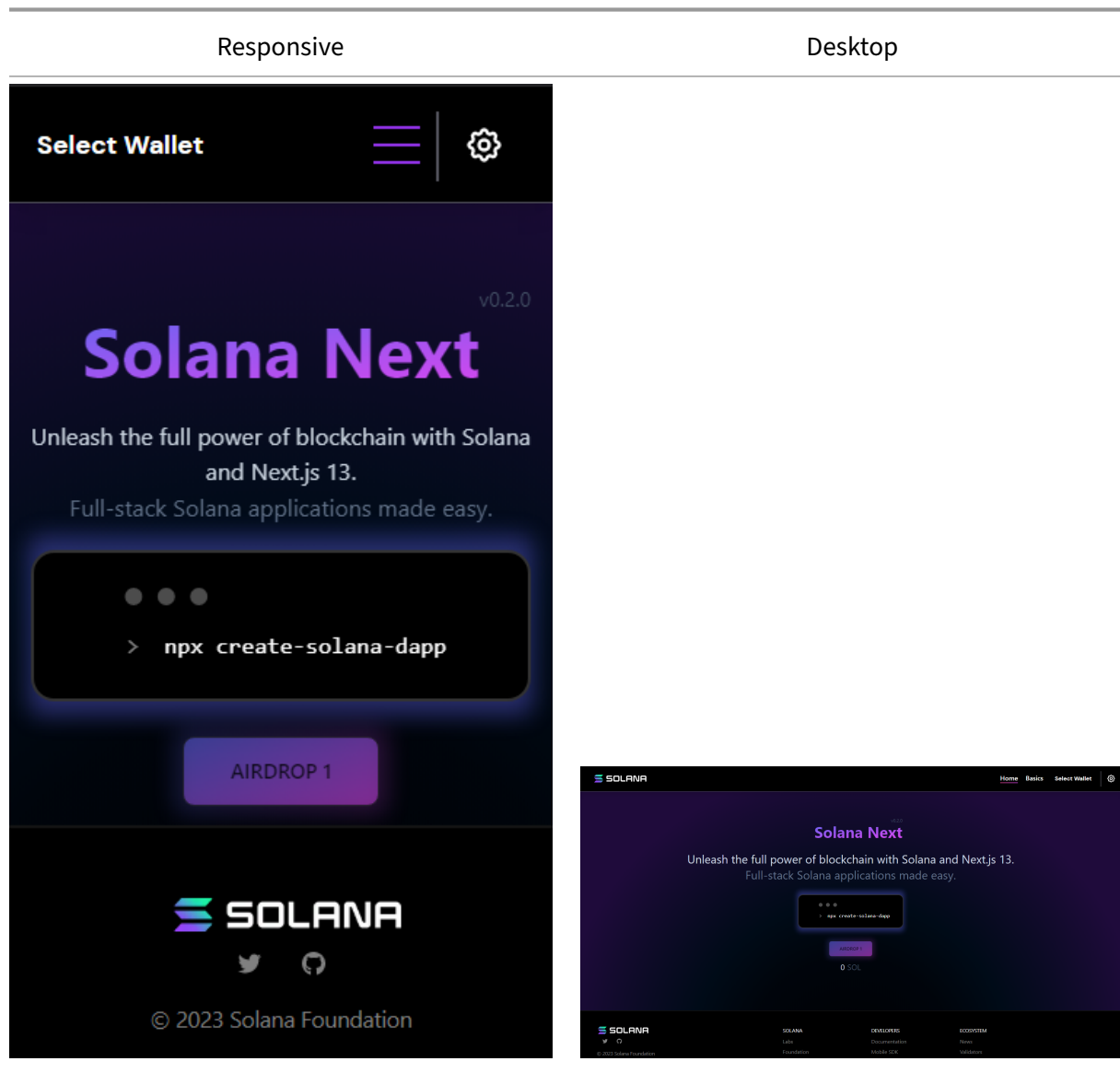

Solana dApp Scaffold Next

The Solana dApp Scaffold repos are meant to house good starting scaffolds for ecosystem developers to get up and running quickly with a front end client UI that integrates several common features found in dApps with some basic usage examples. Wallet Integration. State management. Components examples. Notifications. Setup recommendations.



Getting Started

This is a Next.js project bootstrapped with `create-next-app`.

The responsive version for wallets and wallet adapter may not function or work as expected for mobile based on plugin and wallet compatibility. For more code examples and implementations please visit the Solana Cookbook

Installation

```
1 npm install
2 # or
3 yarn install
```

Build and Run

Next, run the development server:

```
1 npm run dev
2 # or
3 yarn dev
```

Open <http://localhost:3000> with your browser to see the result.

You can start editing the page by modifying [pages/index.tsx](#). The page auto-updates as you edit the file.

API routes can be accessed on <http://localhost:3000/api/hello>. This endpoint can be edited in [pages/api/hello.ts](#).

The [pages/api](#) directory is mapped to [/api/*](#). Files in this directory are treated as API routes instead of React pages.

Features

Each Scaffold will contain at least the following features:

```
1 Wallet Integration with Auto Connec / Refresh
2
3 State Management
4
5 Components: One or more components demonstrating state management
6
7 Web3 Js: Examples of one or more uses of web3 js including a
  transaction with a connection provider
8
9 Sample navigation and page changing to demonstate state
10
```

```
11 Clean Simple Styling
12
13 Notifications (optional): Example of using a notification system
```

A Solana Components Repo will be released in the near future to house a common components library.

Structure

The scaffold project structure may vary based on the front end framework being utilized. The below is an example structure for the Next js Scaffold.

```
1 |—
2 public : publically hosted files|—
3 src : primary code folders and files|—
4   components : should house anything considered a reusable UI
5     component|—
6   contexts` : any context considered reusable and useful to many
7     compoennts that can be passed down through a component tree|—
8   hooks` : any functions that let you 'hook' into react state or
9     lifecycle features from function components|—
10  models` : any data structure that may be reused throughout the
11    project|—
12  pages` : the pages that host meta data and the intended `View` for
13    the page|—
14  stores` : stores used in state management|—
15  styles` : contain any global and reusable styles|—
16  utils` : any other functionality considered reusable code that can
17    be referenced|—
18  views` : contains the actual views of the project that include the
19    main content and components within
20 style, package, configuration, and other project files
```

Contributing

Anyone is welcome to create an issue to build, discuss or request a new feature or update to the existing code base. Please keep in mind the following when submitting an issue. We consider merging high value features that may be utilized by the majority of scaffold users. If this is not a common feature or fix, consider adding it to the component library or cookbook. Please refer to the project's architecture and style when contributing.

If submitting a feature, please reference the project structure shown above and try to follow the overall architecture and style presented in the existing scaffold.

Committing

To choose a task or make your own, do the following:

1. Add an issue for the task and assign it to yourself or comment on the issue
2. Make a draft PR referencing the issue.

The general flow for making a contribution:

1. Fork the repo on GitHub
2. Clone the project to your own machine
3. Commit changes to your own branch
4. Push your work back up to your fork
5. Submit a Pull request so that we can review your changes

NOTE: Be sure to merge the latest from “upstream” before making a pull request!

You can find tasks on the project board or create an issue and assign it to yourself.

Learn More Next Js

To learn more about Next.js, take a look at the following resources:

- [Next.js Documentation](#) - learn about Next.js features and API.
- [Learn Next.js](#) - an interactive Next.js tutorial.

You can check out the Next.js GitHub repository - your feedback and contributions are welcome!

Deploy on Vercel

The easiest way to deploy your Next.js app is to use the Vercel Platform from the creators of Next.js.

Check out our Next.js deployment documentation for more details.