SUPERPROPS DOCUMENTATION

How to Configure

After downloading the file from themeforest , You will find SuperProps.zip file. Unzip the SuperProps.zip and run this command ,

If you want to run these template on Next Js server , then

1. yarn on SuperProps folder.
2. yarn web SuperProps folder .

Then, please go to address **localhost:3000** on your browser and You will find agency landing page.

Similarly,

localhost:3000/app [for App Landing Page]

localhost:3000/saas [for Saas Landing Page]

localhost:3000/saasmodern [for SAAS Modern Landing Page]

localhost:3000/saasclassic [for SAAS Classic Landing Page]

localhost:3000/hosting [for Hosting Landing Page]

localhost:3000/portfolio [for Portfolio Landing Page]

localhost:3000/ride [for Ride Sharing Landing Page]

If you want to run these template on Gatsby Js server , then

1. yarn on SuperProps folder.
2. yarn gatsby-dev SuperProps folder .

Then , please go to address **localhost:8000** on your browser and You will find agency landing page .

Similarly fo the others Landing Pages .

Stack We Used

1. Lerna (A tool for managing JavaScript projects with multiple packages. https://lernajs.io).
2. Yarn Workspace.
3. React Js and Next Js.
4. Gatsby Js.
5. Styled System and Styled Components
6. Firebase Deployment.

Folder Structure

Go to **superProps - > packages** folder. After entering to packages folder, You will find some other folders.

1. common
2. functions
3. landing
4. landing-gatsby .

common

The whole work is in **packages -> common** folder. After entering to common folder, go to common->src and You will find,

1. assets
2. components
3. containers
4. contexts
5. data
6. theme

Now, we will discuss about each and every folders and their tasks .

assets

In assets folder , you will find the **css** folder . In css folder, you will find all of the common styles needed for the template . You will also find the image folder , where all of the images are kept on the basis of the specific landing page .

components

If you are familiar with react or create react app architecture , then we are familiar with components

. Components are reusable codes that you will use throughout your project . Here in components folder , we wrote some custom components which are used in our landing pages . We have done some basic style with the styled components (https://www.styled-components.com/).

Under the ***packages -> reusecore*** folder , you will find some basic components like Text, Heading, Image, Input etc . We have written these components to make the developer's life easy. By using these basic components, you can write custom components according to your need. In the ***common -> components*** folder, we have done the same things. We have written some custom components for our landing pages by using these reusecore components.

containers

Under the ***common -> containers*** folder, we have written all of our codes part by part . Suppose , we have 4 landing pages .

1. Agency.
2. App.
3. Hosting .
4. Saas.
5. Portfolio
6. Ride etc...

In each folder, we have all the sections of that landing page part by part like Navbar, BannerSection, FeatureSection, TestimonialSection, Footer etc . This structure is done for perfect understanding of each and every section . We have also a custom style.js file like (agency.style.js , app.style.js etc) for common styles of that specific landing page.

We have used two libraries to style our Components.

1. styled components . https://www.styled-components.com/docs
2. styled-system . See their doc from here *https://github.com/jxnblk/styled-system*

theme

In the **common -> theme** folder , you will also find some folders like

1. Agency.
2. App
3. Hosting
4. Saas etc….

In each folder , you will find the three .js files colors.js : in this file, you can keep all of the custom colors for your specific landing pages. customeVariant.js : For writing custom variants index.js : all of the style props.

We have used styled system for this folder structure (https://github.com/jxnblk/styled-system) . You can also follow this article for clearing your concept https://varun.ca/styled-system/.

data

In the data folder, you will find the specific data that are used on that specific landing page .

Landing

In this folder , you will find the pages folder of next js and next configuration files. If you want to change something in next js folder , change here .

pages

As we have used next.js , we have a script to your package.json like this:

```
{
“scripts": {
"dev": "next",
"build": "next build",
"start": "next start”}
}
```

After that, the file-system is the main API. Every .js file becomes a route that gets automatically processed and rendered.

Please have a look at this link https://nextjs.org/docs/ for a quick look. You will understand the basic things so quickly.

As we have four landing pages named agency, app, hosting, saas, so we have four js files named agency.js, app.js, hosting.js , saas.js .

In these main js files, we have imported all the codes written in the **common -> containers** folder step by step. Every .js file becomes a route that gets automatically processed and rendered in next.js, so you will find our...

agency landing page at http://localhost:3000/agency
app landing page at http://localhost:3000/app
hosting landing page at http://localhost:3000/hosting
saas landing page at http://localhost:3000/saas

Here, in the index.js file , we have assigned the agency landing page .

next.config.js

We have used some plugins for better performance and optimization .

1. next-optimized-images
2. next-fonts
3. next-css

Landing-Gatsby

In this folder , you will find the pages folder of gatsby js and gatsby configuration files. If you want to change something in gatsby js folder , change here .

gatsby-config.js

According to gatsby configurations , we gave support for styled components and web fonts for our project. The other settings are as same as gatsby default configurations . You can follow the config documentations of gatsby Js https://www.gatsbyjs.org/docs/customization/#customization .

Icons

We have used custom flat icon. If you want to see our icon list then you need to go to **pages->icons.js** folder. After running yarn web command then go to your browser and write http://localhost:3000/icons and hit enter.

We have also used react icon kits to support large variety of icons . You can check out through this link https://wmira.github.io/react-icons-kit/

Ready For Firebase

1. keep the static folder [css, images, fonts] same both in root public/static folder & landing/static folder

2. Always keep the package.json file same/synched among inside landing & functions folder

Firebase Deployment

1. Keep Sync the package.json file both inside functions and landing folder

2. Keep Sync the landing > static & root Public > static folder [if firebase cli isn't installed on you machine, then follow 3 & 4]

3. install firebase-tools from **npm globally npm i -g firebase-tools**

4. login into firebase from command line/terminal. See the doc of that package in npm. Now,Remove all the node_modules, yarn.lock, package-lock.json, DS_Store etc...[for cleanup, not mandatory]. If you want to change your project name from default ,Go to .firebaserc file and change the name of the default.

5. run **yarn** or **npm i** on App root

6. run **npm i** inside **packages > functions** folder. [do not run yarn in there!]

7. run **yarn build** or **npm run build** on App root.

8. run **firebase serve** to check as a local server.

9. run **firebase deploy** to host the App on firebase.