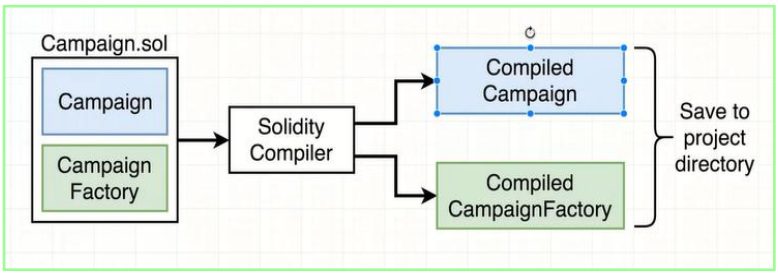
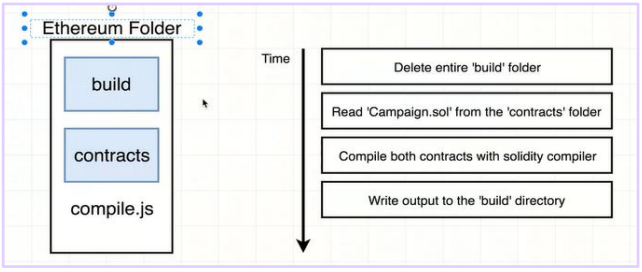




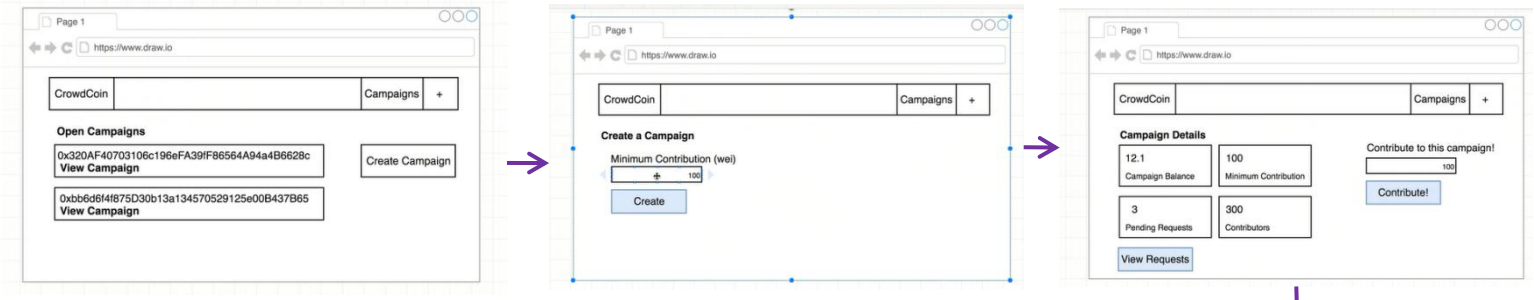

```
const addresses = await factory.methods.getDeployedCampaigns().call();
camapignAddress = addresses[0];
```

(or)

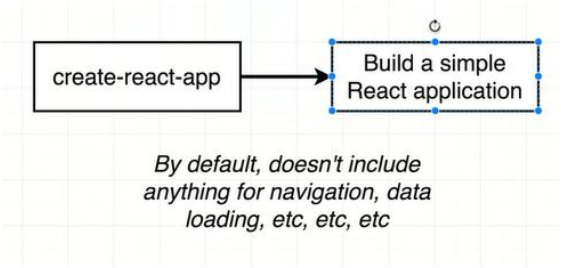
```
[camapignAddress] == await factory.methods.getDeployedCampaigns().call();
```

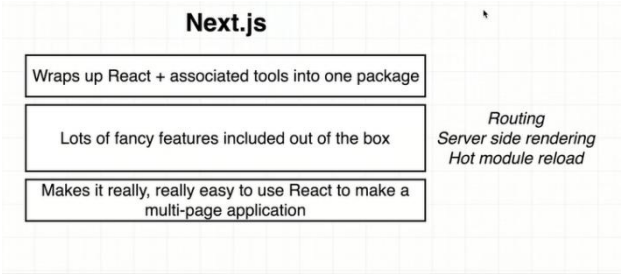
Both above are same.

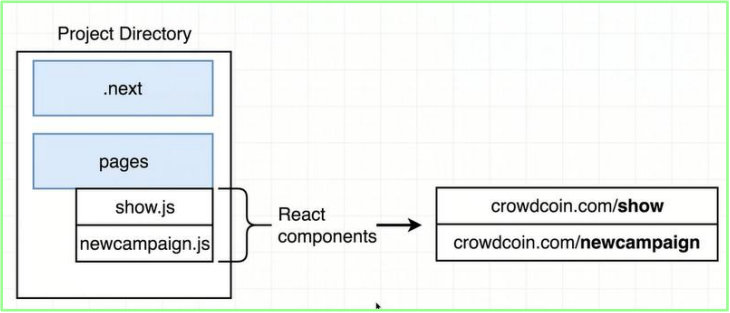
WebAPP



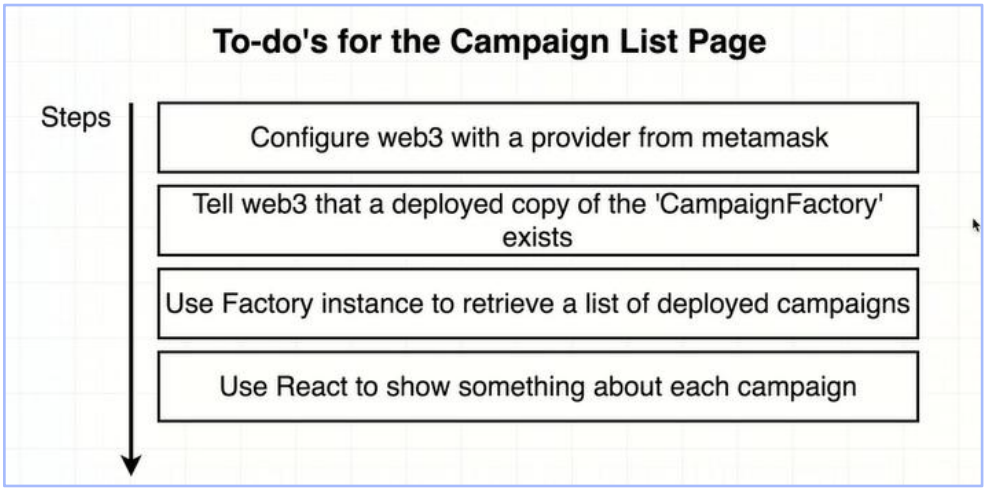
Routing	
Path	We should show...
/	List of Campaigns
/campaigns/new	Form to make a campaign
/campaigns/0x8147	Campaign details for campaign at address 0x8147
/campaigns/0x8147/requests	Requests for campaign at address 0x8147
/campaigns/0x8147/requests/new	Form to create a request for campaign at address 0x8147

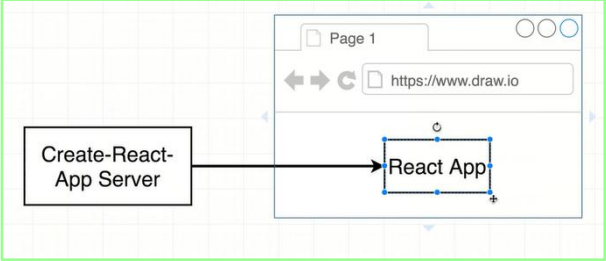




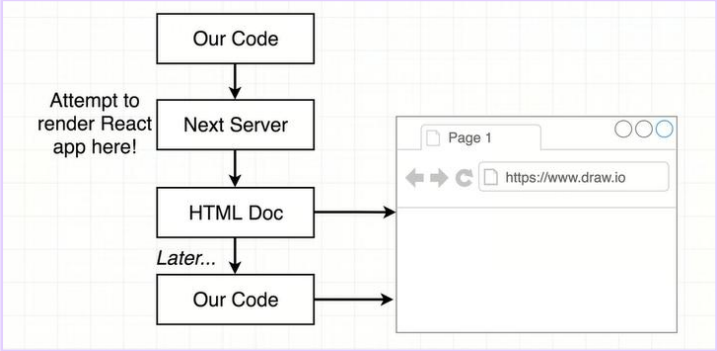


Whenever next is activated it looks for "pages" folder. Every file containing react components inside pages folder turns into a single web-page that can be visited. If we want a default root page save it as index.js. Every js file in pages folder must export a react component otherwise the nextjs will show error.

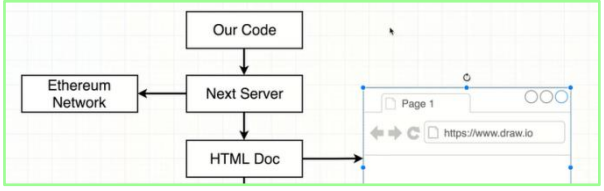




Create react app server just Dumps the entire js code/ files to a browser and browser execute the code.

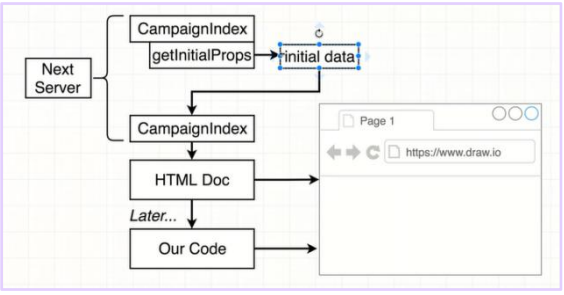


Nextjs uses server side rendereing. The nextjs server takes the react app and render the entire app itself. Next server builds HTML doc and sends it to browser. Which makes the loading on browser very very quickly. Later it sends all the js code to browser. Then the react app inside the code boots up.



As all the users don't have metatmask, we take care of all the Ethereum network stuff inside nextjs server and just pump the HTML doc to browser so that no one faces the problem.

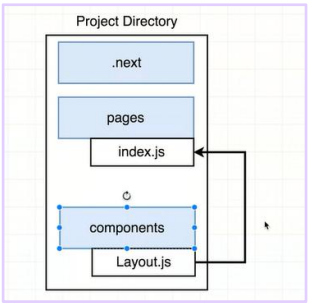
ComponentDidMount() works only in traditional react application, but it doesn't get executed by server of nextjs. getInitialProps() isrendered by the nextjs and solves the server side execution problem.



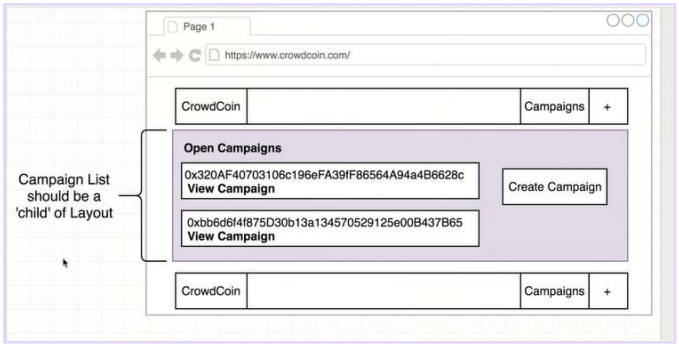
Initial data(list of campaigns) and provide it to CampaignIndex as props. And give it to HTML Doc for rendering on webapp.

Rendering is very costly operation in react.

SemanticUi-React kit => Precreated components along with styling.



We want to display header and footer on every page of the app. By default the nextjs doesnt support this notion. So create a new "components" folder and place all default components in that folder and import them in files of pages folder.



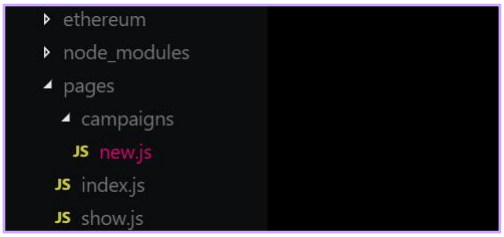

```
render() {
  return (
    <Layout>
      <div>
        <link
          rel="stylesheet"
          href="//cdnjs.cloudflare.com/ajax/libs/semant
        />
        <h3>Open Campaigns</h3>
        {this.renderCampaigns()}
        <Button content="Create Campaign" icon="add ci
      </div>
    </Layout>
```

All the JSX inside the layout tag is passed to Layout.js as props.children .

index.js | Layout.js | package.json

```
import React from 'react';

export default (props) => {
    return (
        <div>
            <h1>Im a header</h1>
            {props.children}
            <h1>Im a footer</h1>
        </div>
    );
};
```



For the nested route campaign/new,create a new subfolder inside nextjs and create a new file inside it.

```
import Head from 'next/head';
import Header from './Header';

export default (props) => {
    return (
        <Container>
            <Head>
                <link
                rel="stylesheet"
                href="//cdnjs.cloudflare.com/ajax/libs/semant
                />
            </Head>
```

Whatever tags placed inside <Head></Head>m everything will be moved to head tag of HTML doc while rendering.

<Form onSubmit={this.onSubmit} error={!!this.state.errorMessage}>

Usually React doesn't render the error component it must be specified inside the Form tag.

Nextjs by default doesn't have dynamic routing. **next-routes** is helper module to achieve the dynamic routing. Firstly create routes.js and server.js and write the configuration in those files.