

Hello Jumpstart, I am Lucas Manuel Faner, an applicant for the Front-End Engineer (REMOTE) job.

I tried to keep the guide as comprehensible as possible, while adding shortcuts for a better experience.

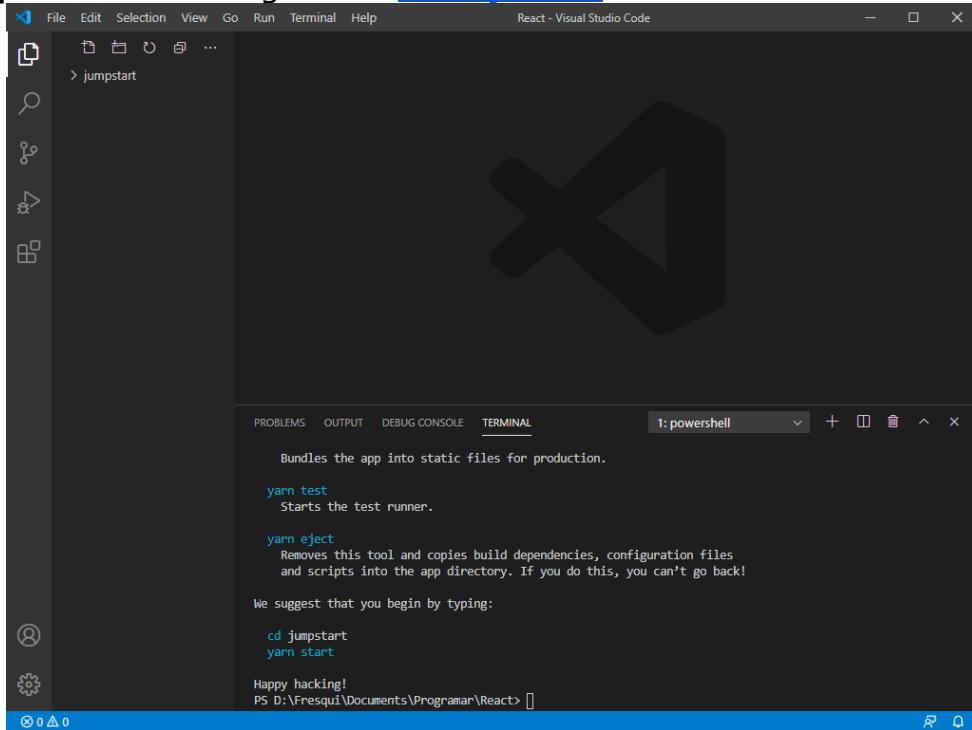
If I can provide you with any further information, please let me know.

Table of Contents:

1. [Introduction](#)
2. [Creating the Main Rectangle](#)
 - a. [Adding the Icon](#)
 - b. [Adding Title and Text](#)
 - c. [Adding the Button](#)
3. [Interactive Object](#)
 - a. [States](#)
 - b. [onClick](#)
 - c. [Creating the Smaller Rectangle](#)
4. [Cleaning Code](#)
5. [Result](#)
6. [Additional Notes](#)

1. Introduction

I began creating a React project with Visual Studio Code's integrated terminal, using the command **npx create-react-app jumpstart** according to the [react.js docs](#).

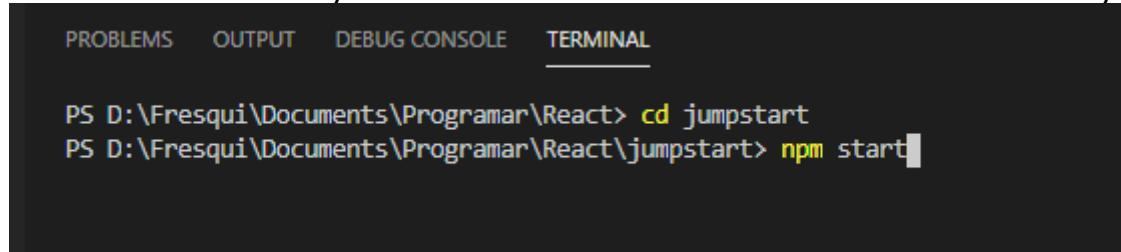


The screenshot shows the Visual Studio Code interface with a dark theme. On the left is the Explorer sidebar, which has a folder icon and shows a single item named 'jumpstart'. The main workspace is mostly black, featuring the large Microsoft logo. At the bottom, the Terminal tab is active, displaying the following text:

```
Bundles the app into static files for production.  
yarn test  
Starts the test runner.  
yarn eject  
Removes this tool and copies build dependencies, configuration files  
and scripts into the app directory. If you do this, you can't go back!  
We suggest that you begin by typing:  
cd jumpstart  
yarn start  
Happy hacking!
```

The status bar at the bottom indicates the path: PS D:\Fresqui\Documents\Programar\React>.

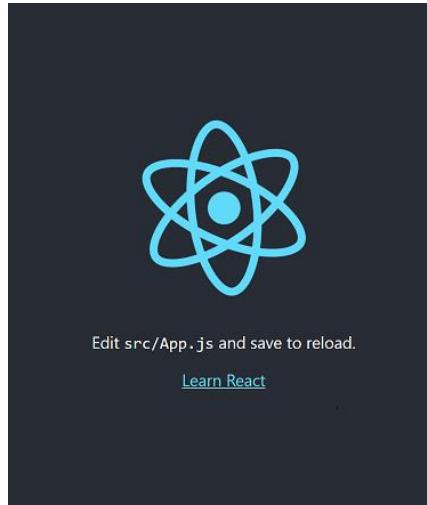
then switch directory and start to see if it has been created correctly.



The screenshot shows the terminal window with the following text:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL  
  
PS D:\Fresqui\Documents\Programar\React> cd jumpstart  
PS D:\Fresqui\Documents\Programar\React\jumpstart> npm start
```

and here it is.



2. Creating the Main Rectangle

Following the design provided,

Corporate Bylaws

An internal document that details the operating rules for the corporation and are typically adopted at the organizational meeting of the board of directors.

[View document](#)

I feel like I could start from creating the main rectangle to contain the **elements** (icon, title text, body text and button). To do so, I had to erase unused code, such as the default template for a fresh started react app, while creating a new “**Rectangle24**” CSS class for the mentioned rectangle. (It will be renamed eventually)

```
File Edit Selection View Go Run Terminal Help
jumpstart > src > JS App.js > App
1 import './App.css';
2
3 function App() {
4   return (
5     <div className="App">
6       <header className="App-header">
7         <p>
8           Edit <code>src/App.js</code> and save to reload.
9         </p>
10        <a
11          className="App-link"
12          href="https://reactjs.org"
13          target="_blank"
14          rel="noopener noreferrer"
15        >
16          Learn React
17        </a>
18      </header>
19    </div>
20  );
21
22
23 export default App;
```

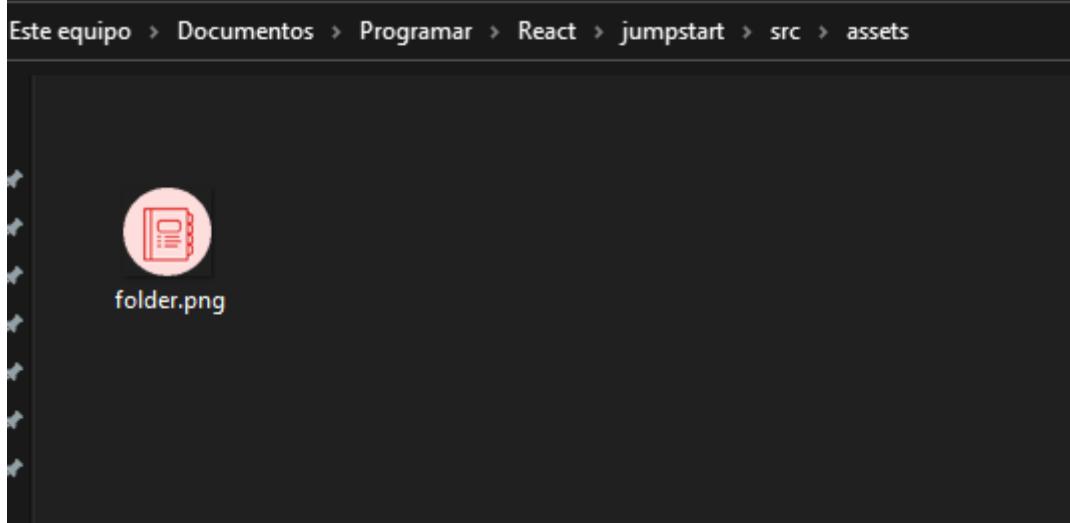
Now I go to the new app directory and create an **assets** folder within the "**src**" folder of the project.

```
JS App.js X JS App.test.js # App.css
jumpstart > src > JS App.js > App
1 import './App.css';
2 import folder from './assets/folder.png'
3
4 function App() {
5   return (
6     <div className="App">
7       <body className="Rectangle24">
8         <img src={folder} alt="folder" className="Folder" />
9       </body>
10      </div>
11    );
12 }
13
14 export default App;
15
```

```
9
10 .Folder {
11   align-self: center;
12   margin-top: 25px;
13   width: 44px;
14   height: 44px;
15   left: 178px;
16   top: 137px;
17 }
18
```

| Este equipo > Documentos > Programar > React > jumpstart > src | | | | |
|--|---------------|-----------------------|---------------------|--------|
| | Nombre | Fecha de modificación | Tipo | Tamaño |
| ido | assets | 12/11/2020 10:44 | Carpeta de archivos | |
| tos | App.css | 12/11/2020 10:26 | Documento de ho... | 1 KB |
| | App.js | 12/11/2020 10:44 | Archivo JavaScript | 1 KB |
| | App.test.js | 12/11/2020 10:04 | Archivo JavaScript | 1 KB |
| | index.css | 12/11/2020 10:04 | Documento de ho... | 1 KB |
| | index.js | 12/11/2020 10:13 | Archivo JavaScript | 1 KB |
| | setupTests.js | 12/11/2020 10:04 | Archivo JavaScript | 1 KB |

This is where I will save every image that I will use in the project.

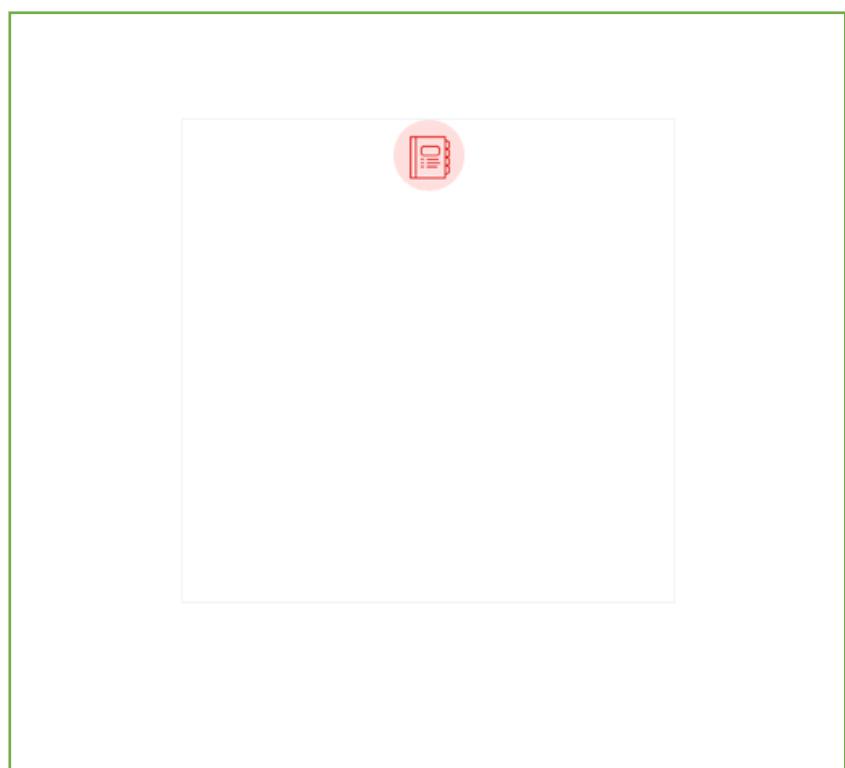


For guidance, I followed the dimensions that [Figma](#) had provided me in px. Sometimes I did not have the exact measures, but I found a workaround to solve it, which I mention in additional notes.

The diagram illustrates a card component with the following dimensions and features:

- Outer Dimensions:** 130px width and 131px height.
- Icon Area:** A central square icon with a red border and a blue background, containing a white document icon. It has a bounding box of 44x44.
- Text Area:** The title "Corporate Bylaws" is centered below the icon.
- Text Content:** A detailed description of Corporate Bylaws follows the title.
- Call-to-Action:** A purple-outlined button labeled "View document" is located at the bottom left.
- Coordinate Labels:** Numerical labels (25, 130, 131, 44x44, 231) are placed around the icon and text area.

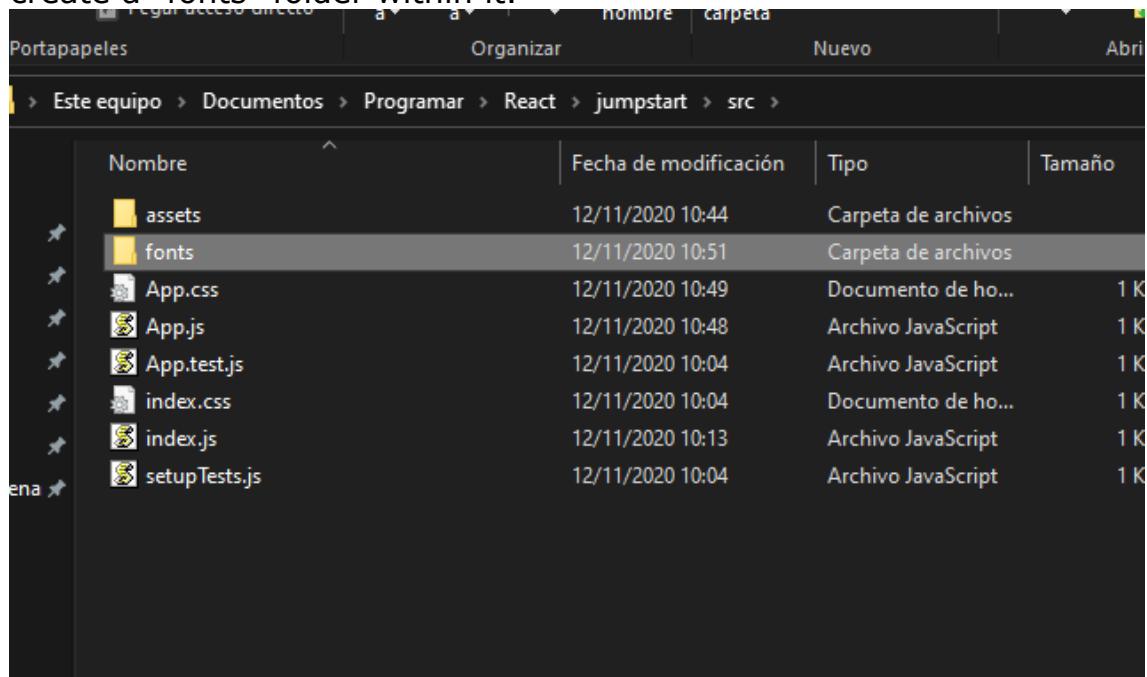
The result is as follows:



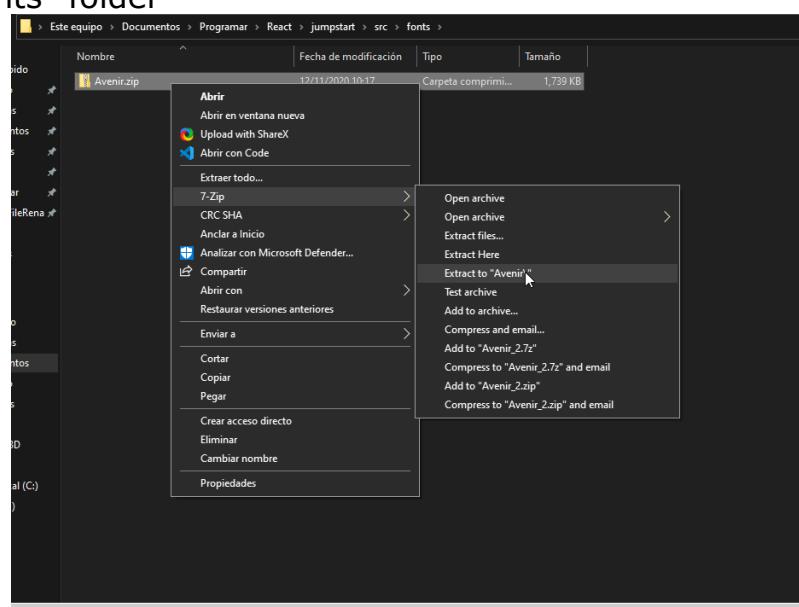
A white box with a grey border in a white background. It looks messy so I will add the background color to work better.
It must be added within the **index.css** file.

```
jumpstart > src > # index.css > body
1   body {
2     background-color: #E5E5E5;
```

Okay, now that the Icon works, I will go back to the **src** folder and create a "fonts" folder within it.



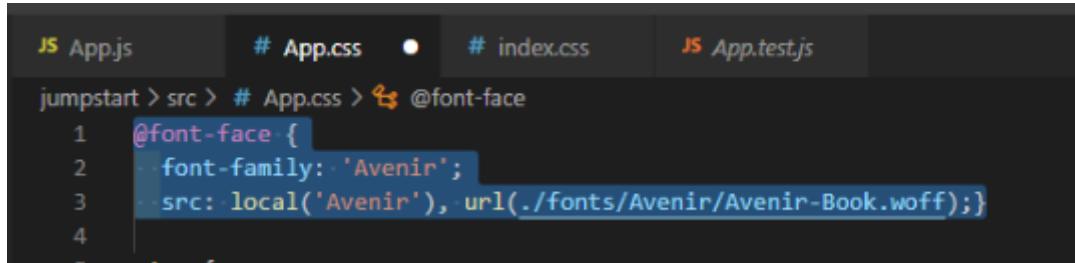
Download the font provided by [Jumpstart](#) and extract it in a folder inside "fonts" folder



equipo > Documentos > Programar > React > jumpstart > src > fonts >

| Nombre | Fecha de modificación | Tipo |
|------------|-----------------------|---------------------|
| Avenir | 12/11/2020 10:54 | Carpeta de archivos |
| Avenir.zip | 12/11/2020 10:17 | Carpeta comprimida |

Within **app.css** we will install the new font, following the directory in which it was extracted.

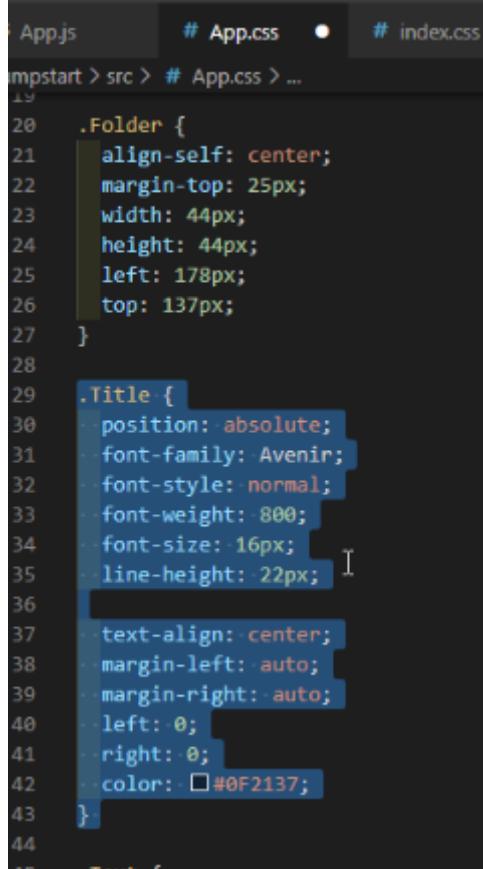


```

JS App.js      # App.css • # index.css JS App.test.js
jumpstart > src > # App.css > @font-face
1  @font-face {
2    font-family: 'Avenir';
3    src: local('Avenir'), url(./fonts/Avenir/Avenir-Book.woff);
4

```

and then insert it to a style:



```

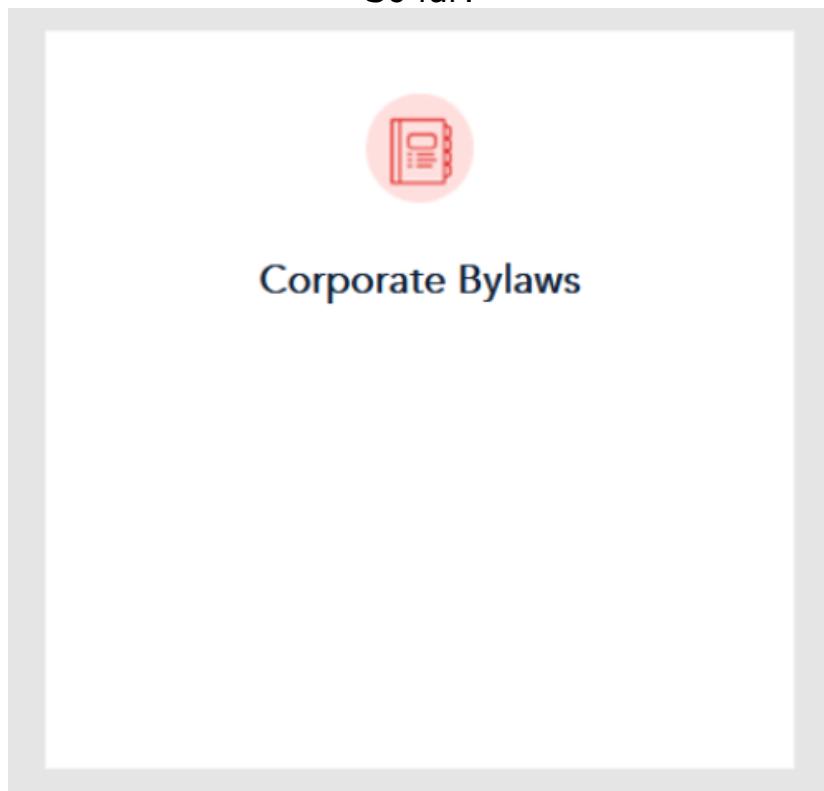
App.js      # App.css • # index.css
jumpstart > src > # App.css > ...
19
20 .Folder {
21   align-self: center;
22   margin-top: 25px;
23   width: 44px;
24   height: 44px;
25   left: 178px;
26   top: 137px;
27 }
28
29 .Title {
30   position: absolute;
31   font-family: Avenir;
32   font-style: normal;
33   font-weight: 800;
34   font-size: 16px;
35   line-height: 22px;
36
37   text-align: center;
38   margin-left: auto;
39   margin-right: auto;
40   left: 0;
41   right: 0;
42   color: #0F2137;
43 }
44

```

Now we can add the **.Title** class to a HTML paragraph.

```
function App() {  
  return (  
    <div className="App">  
      <body className="Rectangle24">  
        <img src={folder} alt="folder" className="Folder" />  
        <p className="Title">Corporate Bylaws</p>  
      </body>  
    </div>  
  );  
}
```

So far:



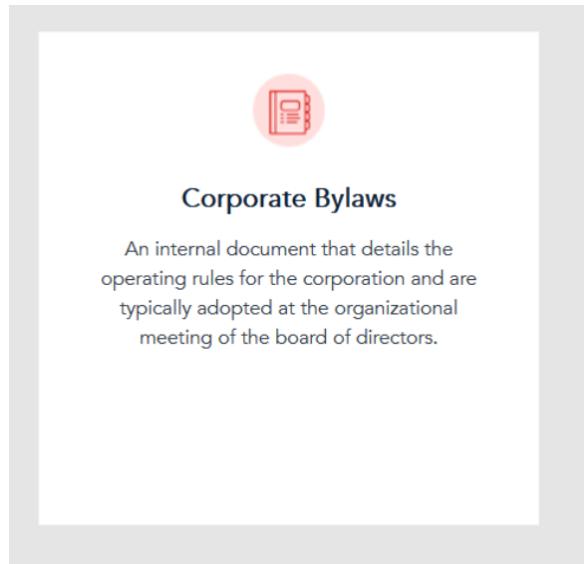
Now it is time for the text body:

```
function App() {  
  return (  
    <div className="App">  
      <body className="Rectangle24">  
        <img src={folder} alt="folder" className="Folder" />  
        <p className="Title">Corporate Bylaws</p>  
        <p className="Text">An internal document that details the operating rules for the corporation and are typically adopted at the organizational meeting of the board of directors.</p>  
      </body>  
    </div>  
  );  
}
```

And its class:

```
.Text {  
    position: absolute;  
    width: 243px;  
    height: 72px;  
    left: 79px;  
    top: 110px;  
  
    font-family: Avenir;  
    font-style: normal;  
    font-weight: normal;  
    font-size: 12px;  
    line-height: 150%;  
  
    margin-left: auto;  
    margin-right: auto;  
    left: 0;  
    right: 0;  
  
    text-align: center;  
  
    color: #343D48;  
}
```

Result:



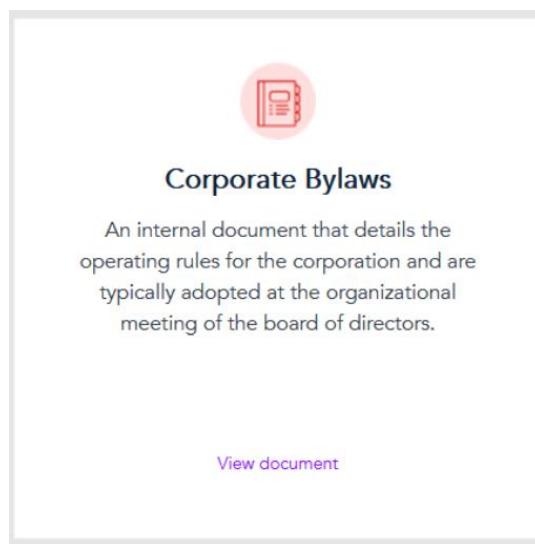
Now for the View Document button, I will create a paragraph with a class:

```
function App() {
  return (
    <div className="App">
      <body className="Rectangle24">
        <img src={folder} alt="folder" className="Folder" />
        <p className="Title">Corporate Bylaws</p>
        <p className="Text">An internal document that details
          <div style={{marginTop: 176}}>
            <p className="ViewDocument">View document</p>
          </div>
        </body>
      </div>
    );
}

.ViewDocument {
  font-family: Avenir;
  font-style: normal;
  font-weight: 500;
  font-size: 10px;
  line-height: 14px;

  color: #8C14FC;
}
```

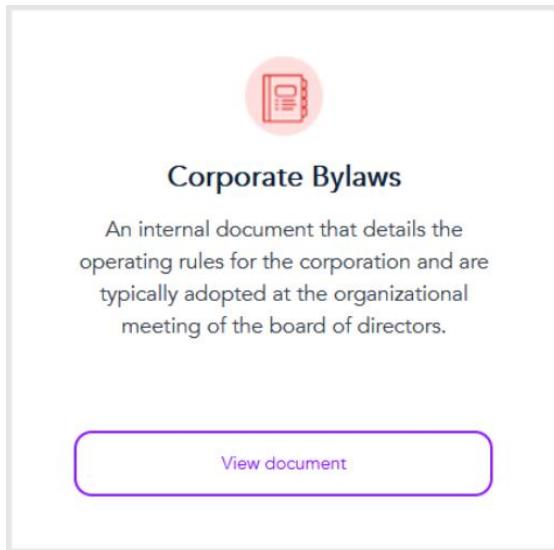
Result:



Now for the border of the button:

```
.Border {  
    border-style: solid;  
    border-color: #8c14fc;  
    color: #8c14fc;  
    border-width: 1.5px;  
    margin: 35px;  
    border-radius: 10px;  
}  
  
<div style={{marginTop: -176}}>  
  <div className="Border">  
    <p className="ViewDocument">View document</p>  
  </div>  
</div>
```

And here it is:



For the rectangle below the *Main Rectangle*, I will create a rectangle with the same width as the main one, and a height of 4px according to Figma once again.

```
.MainRectangleShadow {  
    width: 305px;  
    height: 4px;  
    background: #FEDEDD;  
    margin-top: 320px;  
    margin-left: 20px  
}
```

And put the code below the </body> of the main rectangle.

```
</body>  
<div className="MainRectangleShadow"></div>
```



Corporate Bylaws

An internal document that details the operating rules for the corporation and are typically adopted at the organizational meeting of the board of directors.

[View document](#)

Title was looking a little messy, so I added a new font:

```
@font-face {  
    font-family: 'Avenir-medium';  
    src: local('Avenir-medium'), url(./fonts/Avenir/Avenir-Medium.woff);  
}
```

And I added it to the class:

```
.Title {  
    position: absolute;  
    font-family: Avenir-medium;  
    font-weight: 800;  
    font-size: 18px;  
    line-height: 22px;  
    top: 65px;
```



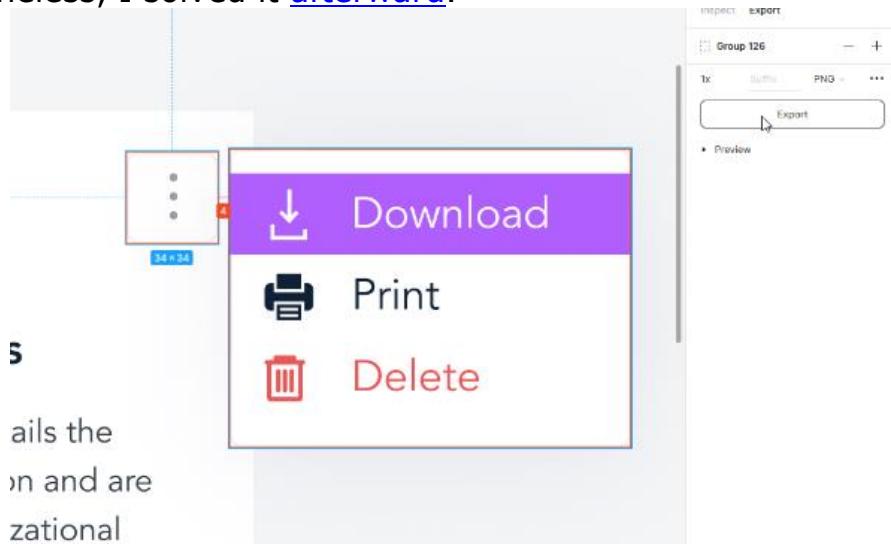
Corporate Bylaws

An internal document that details the operating rules for the corporation and are typically adopted at the organizational meeting of the board of directors.

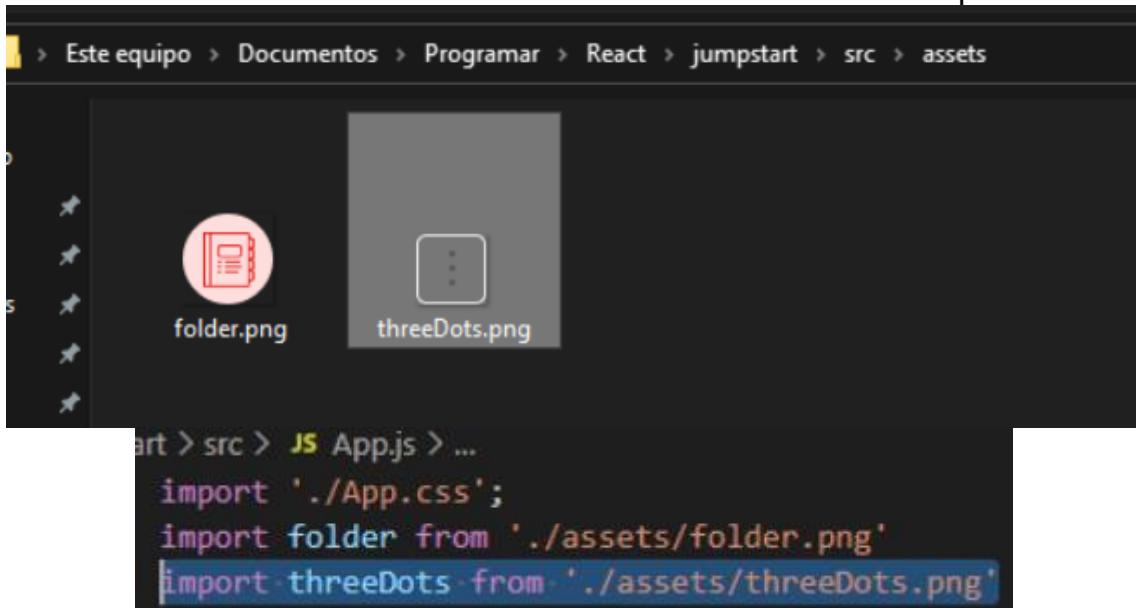
[View document](#)

3. Interactive Object

Now for the *three dots button* (...) I discovered a feature Figma had to export as PNG. Sadly, it becomes pixelated. Nevertheless, I solved it [afterward](#).



I saved the PNG file to the **assets** folder within **src** and imported it.



To make the *three dots button* show when the mouse was hovered on the *main rectangle*, I had to add **states** to my function, and the **CSS** class.

```
function App() {
  const [isShown, setIsShown] = useState(false);
  return (
    <div>
```

```

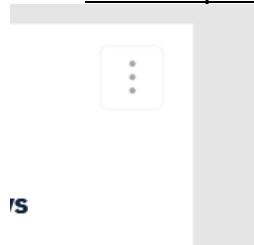
        <body className="Rectangle24" onMouseEnter={() => setIsShown(true)} onMouseLeave={() => setIsShown(false)}>
          {isShown && (
            <div>
              <img src={threeDots} alt="threeDots" className="threeDots"/>
            </div>
          )}
        
```

```

.threeDots {
  position: fixed;
  left: 276px;
  top: 31px;
  height: 34px;
  width: 34px;
}

```

It now shows as soon as the mouse pointer is on the rectangle



Now I needed to add an **onClick** event for the image to bring the new rectangle

```

          <img src={threeDots} alt="threeDots" className="threeDots" onClick={() => setDotsMenu(true)}/>
        </div>
      
```

For the program to work correctly, I must add a new state to the function

```

function App() {
  const [isShown, setIsShown] = useState(false);
  const [DotsMenu, setDotsMenu] = useState(false);
  return (
    ...
  );
}

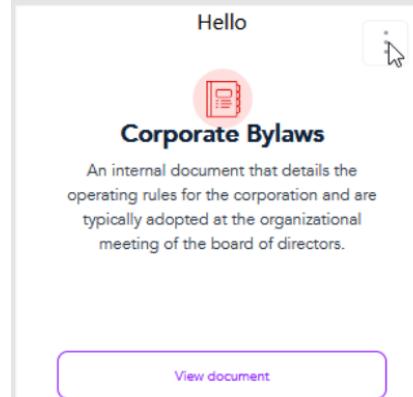
```

Before I continue, I must check if everything is in order so far. To do so, I add a Hello paragraph with the **if (&&) conditioner** in React.

```

function App() {
  const [isShown, setIsShown] = useState(false);
  const [DotsMenu, setDotsMenu] = useState(false);
  return (
    <div className="App">
      <body className="Rectangle24" onMouseEnter={() => setIsShown(true)} onMouseLeave={() => setIsShown(false)}>
        {isShown && (
          <div>
            <img src={threeDots} alt="threeDots" className="threeDots" onClick={() => setDotsMenu(true)}/>
            {DotsMenu && (<div>Hello</div>)}
          </div>
        )}
      
```

And the script is working:



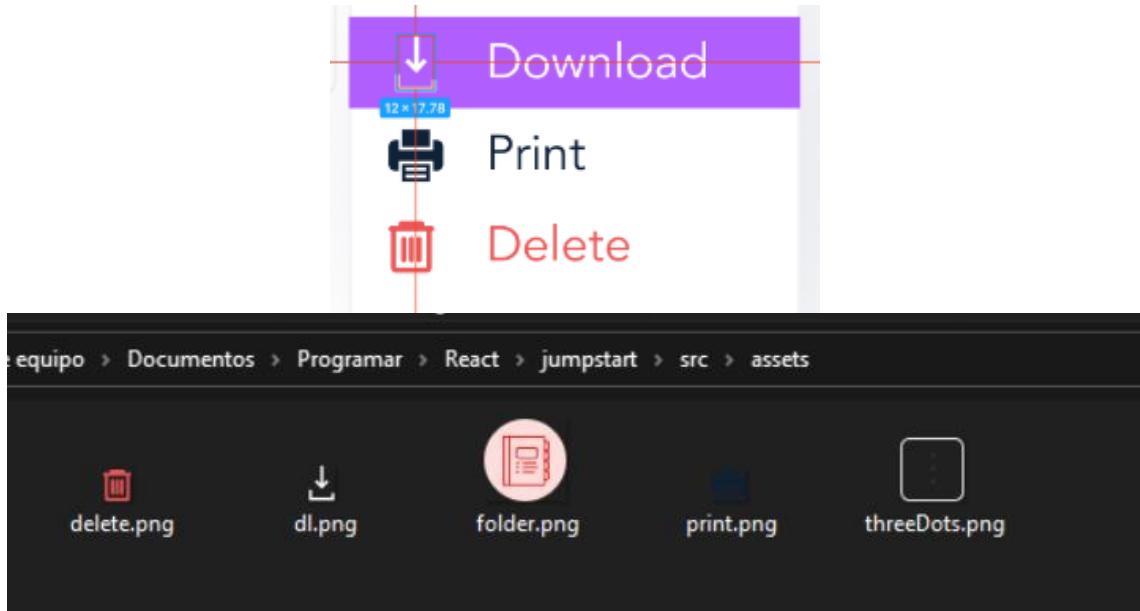
Great! Now I will create the new rectangle, being smaller than the main one, following the dimensions provided.

```
.RectangleDots {  
  position: fixed;  
  width: 148px;  
  height: 110px;  
  left: 313px;  
  top: 31px;  
  background: #FFFFFF;  
  border: 1px solid #F1F4F6;  
  border-radius: 8px;  
  box-sizing: border-box;  
}
```

I will now make the menu close when the mouse pointer leaves both the *main rectangle* and the small new rectangle.

```
{isShown && (<div>  
  <img src={threeDots} alt="threeDots" className="MainRectangleThreeDotsButton" onClick={() => setDotsMenu(true)}/>  
  {DotsMenu && (<div className="DotsMenu" onMouseLeave={() => setDotsMenu(false)}>
```

Repeating the [process used before](#) to download a PNG image from Figma, I downloaded the small rectangle icons, saving them to the **assets** folder once again.



And of course, imported them.

```
import dl from './assets/dl.png'
import del from './assets/delete.png'
import print from './assets/print.png'
```

Now the classes:

```
.DotsMenuImage {
  float: left;
  margin-top: 6px;
  margin-top: 6.22px;
  margin-left: 16px;
}

.DotsMenuPurpleBackground {
  height: 30px;
  background: #B05EFD;
  box-sizing: border-box;
}

.DotsMenuPurpleBackgroundText {
  font-family: Avenir;
  font-style: normal;
  font-weight: normal;
  font-size: 16px;
  line-height: 30px;
  color: #ffffff;
  display: flex;
  align-items: center;
  margin-left: 45px;
  margin-top: 10px;
}

.DotsMenuPrintText {
  font-family: Avenir;
  font-style: normal;
  font-weight: normal;
  font-size: 16px;
  line-height: 30px;
  color: #00F213;
  display: flex;
  align-items: center;
  margin-left: 45px;
  margin-top: 10px;
}

.DotsMenuDownloadImage {
  float: left;
  margin-top: 6px;
  margin-top: 6.22px;
  margin-left: 16px;
}

.DotsMenuPrintImage {
  float: left;
  margin-top: 6px;
  margin-top: 6.22px;
  margin-left: 16px;
}

.DotsMenuDeleteText {
  font-family: Avenir;
  font-style: normal;
  font-weight: normal;
  font-size: 16px;
  line-height: 30px;
  color: #EB5757;
  display: flex;
  align-items: center;
  margin-left: 45px;
  margin-top: 10px;
}

.DotsMenuDeleteImage {
  float: left;
  margin-top: 6px;
  margin-top: 6.22px;
  margin-left: 16px;
}
```

And the HTML code:

```
<div className="DotsMenuPurpleBackground">
  <img src={dl} alt="dl" className="DotsMenuDownloadImage"/></div>
  <p className="DotsMenuPurpleBackgroundText">Download</p></div>

<div className="DotsMenuNoBackground">
  <img src={print} alt="print" className="DotsMenuPrintImage"/></div>
  <p className="DotsMenuPrintText">Print</p></div>

<div className="DotsMenuNoBackground">
  <img src={del} alt="del" className="DotsMenuDeleteImage"/></div>
  <p className="DotsMenuDeleteText">Delete</p></div>
</div>
```

A little messy, but close to the end:

The screenshot shows a document card for "Corporate Bylaws". The card includes a thumbnail icon, a title, a description, and a "View document" button. A context menu is open on the right side of the card, containing three items: "Download" (purple background), "Print" (blue background), and "Delete" (red background). The "Download" item is currently selected.

Corporate Bylaws

An internal document that details the operating rules for the corporation and are typically adopted at the organizational meeting of the board of directors.

[View document](#)

Download

Print

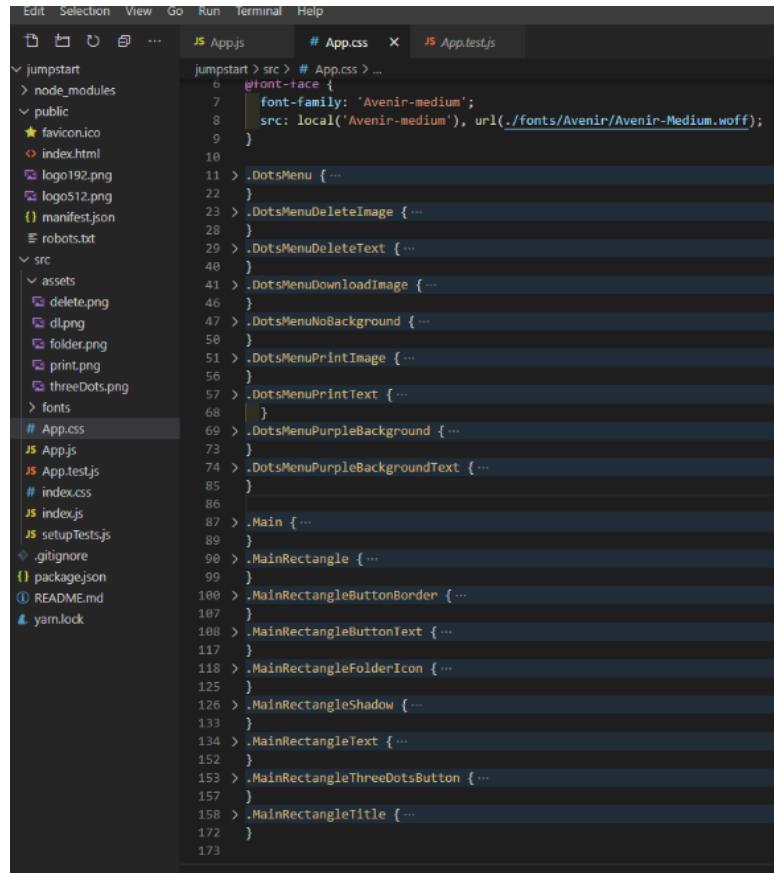
Delete

4. Cleaning Code

Classes were disorganized as well as code. It was time to begin sorting alphabetically and cleaning code.

```
jumpstart > src > # App.css > .Main
  1  @font-face {
  2    font-family: 'Avenir';
  3    src: local('Avenir'), url(./fonts/Avenir/Avenir-Book.woff);
  4  }
  5
  6  @font-face {
  7    font-family: 'Avenir-medium';
  8    src: local('Avenir-medium'), url(./fonts/Avenir/Avenir-Medium.woff);
  9  }
 10
 11 > .Main [...]
 12 []
 13
 14
 15 > .MainRectangle {...}
 16 }
 17
 18 > .DotsRectangle {...}
 19 }
 20
 21 .Folder {
 22   align-self: center;
 23   margin-top: 25px;
 24   width: 44px;
 25   height: 44px;
 26   left: 178px;
 27   top: 25px;
 28 }
 29
 30 .Title {
 31   position: absolute;
 32   font-family: Avenir-medium;
 33   font-weight: 800;
 34 }
```

Sorted:



The screenshot shows a code editor interface with three tabs: "App.js", "# App.css", and "App.test.js". The "# App.css" tab is active, displaying the following sorted CSS code:

```
jumpstart > src > # App.css > ...
  1  @font-face {
  2    font-family: 'Avenir-medium';
  3    src: local('Avenir-medium'), url(./fonts/Avenir/Avenir-Medium.woff);
  4  }
  5
  6  @font-face {
  7    font-family: 'Avenir';
  8    src: local('Avenir'), url(./fonts/Avenir/Avenir-Book.woff);
  9  }
 10
 11 > .DotsMenu {...}
 12 }
 13 > .DotsMenuDeleteImage {...}
 14 }
 15 > .DotsMenuDeleteText {...}
 16 }
 17 > .DotsMenuDownloadImage {...}
 18 }
 19 > .DotsMenuNoBackground {...}
 20 }
 21 > .DotsMenuPrintImage {...}
 22 }
 23 > .DotsMenuPrintText {...}
 24 }
 25 > .DotsMenuPurpleBackground {...}
 26 }
 27 > .Main {...}
 28 }
 29 > .MainRectangle {...}
 30 }
 31 > .MainRectangleButtonBorder {...}
 32 }
 33 > .MainRectangleButtonText {...}
 34 }
 35 > .MainRectangleFolderIcon {...}
 36 }
 37 > .MainRectangleShadow {...}
 38 }
 39 > .MainRectangleText {...}
 40 }
 41 > .MainRectangleThreeDotsButton {...}
 42 }
 43 > .MainRectangleTitle {...}
 44 }
```

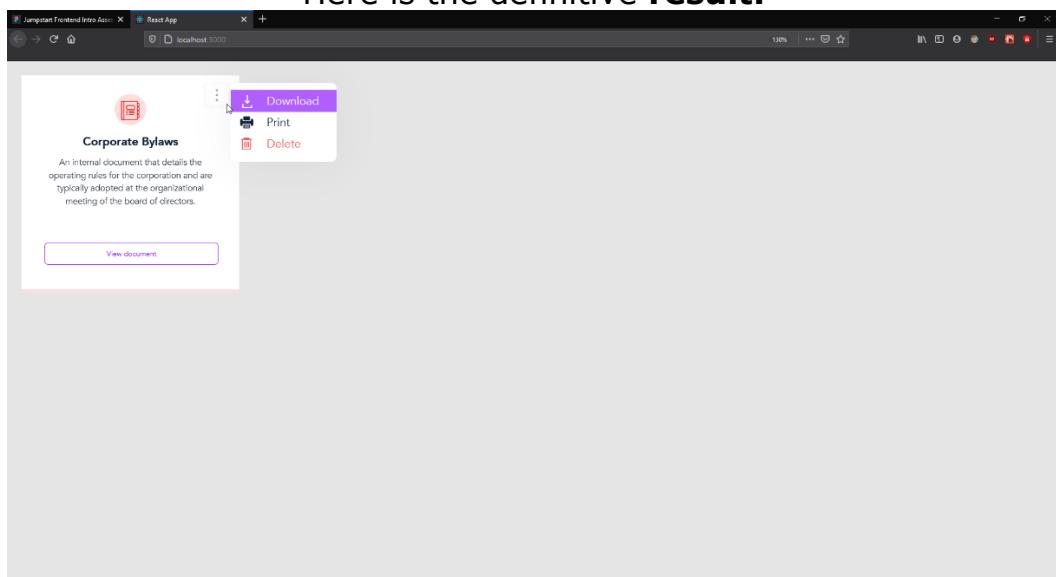
Cleaner HTML/JS now with commentary:

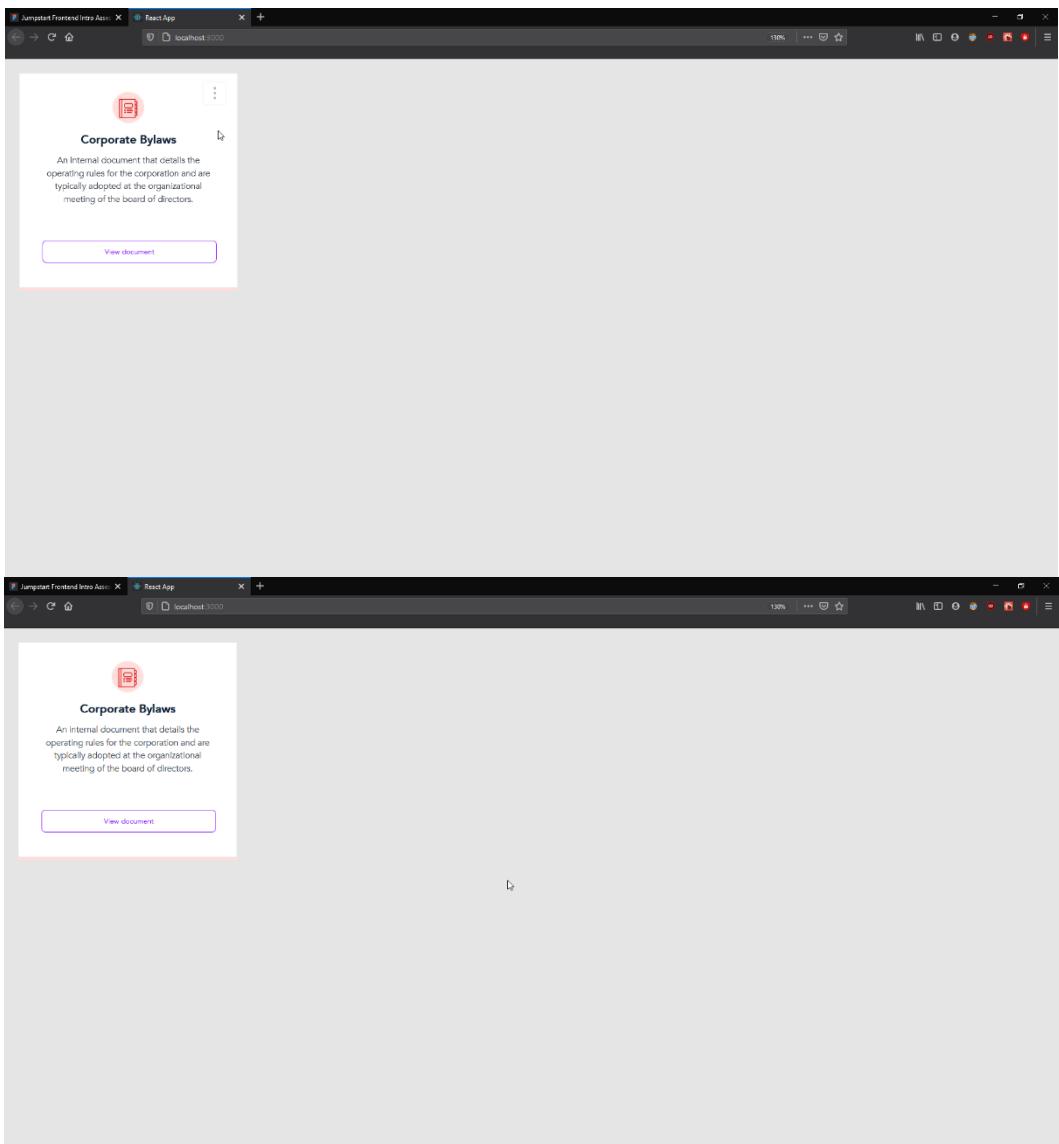
```
11  function App() {
12   /* STATES */
13   const [isShown, setIsShown] = useState(false);
14   const [dotsMenu, setDotsMenu] = useState(false);
15   return (
16     <div className="Main">
17       <body className="MainRectangle" onMouseEnter={() => setIsShown(true)} onMouseLeave={() => setIsShown(false)}>
18         <div style={{display: "flex", flex: 1, gap: 10}}> /* OPTIONS BUTTON / DOTS BUTTON */
19           {isShown && <div>
20             <img alt="threeDots" className="MainRectangleThreeDotsButton" onClick={() => setDotsMenu(true)}/>
21             {dotsMenu && <div className="DotsMenu">
22               /* DOWNLOAD */
23               <div className="DotsMenuPurpleBackground">
24                 <img alt="d1" className="DotsMenuDownloadImage"/>
25                 <p className="DotsMenuPurpleBackgroundText">Download</p>
26               </div>
27               /* PRINT */
28               <div className="DotsMenuNoBackground">
29                 <img alt="print" className="DotsMenuPrintImage"/>
30                 <p className="DotsMenuPrintText">Print</p>
31               </div>
32               /* DELETE */
33               <div className="DotsMenuNoBackground">
34                 <img alt="del" className="DotsMenuDeleteImage"/>
35                 <p className="DotsMenuDeleteText">Delete</p>
36               </div>
37             </div>
38           )
39         )
40       <div> /* MAIN RECTANGLE & COMPS */
41         <img alt="folder" className="MainRectangleFolderIcon" />
42         <p>Corporate Bylaws</p>
43         <p>An internal document that details the operating rules for the corporation and are typically adopted at the organizational meeting of the</p>
44         <div style={{marginTop: 160,}}>
45           <div className="MainRectangleButtonBorder">
46             <p className="MainRectangleButtonText">View document</p>
47           </div>
48         </div>
49       </div>
50     </body>
51   </div>
52 );
53 }
```

Followed by a couple CSS code tweaks to improve visuals and formatting, deleted some extra HTML/JS code as well as useless files.

5. Result

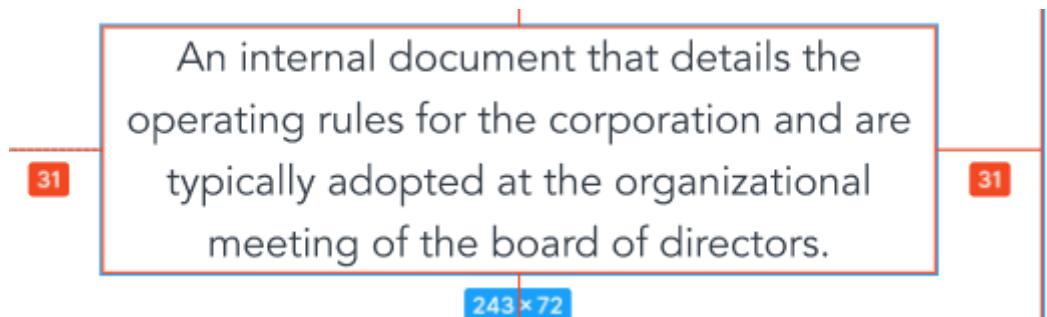
Here is the definitive **result**:



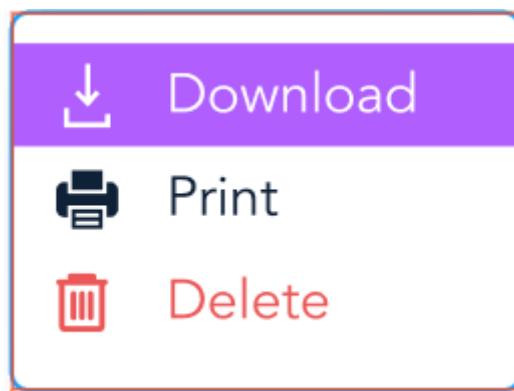


ADITIONAL NOTES

Sometimes I had to think outside of the box, because the representations were not as accurate as expected (e.g. the *View Document border* didn't specify its size or margins, so I assumed it would follow the same line as the *body text container* had.



This happened to me as well when I saw the *three dot options menu icons* (2) I calculated it by downloading the image at 100% size and using the ruler provided by [Adobe Photoshop](#), to estimate the px value.



To fix the pixelated icons I asked my girlfriend, who is a Graphic Designer, to re-draw those icons for exclusive use within this exercise. I then resized them within the code to sharpen the way they look.

```
.DotsMenuDeleteImage {  
    float: left;  
    margin-top: 6px;  
    margin-bottom: 6.22px;  
    margin-left: 14px;  
    width: 15px;  
    height: 17px;  
}  
.DotsMenuDeleteText { ... }  
.DotsMenuDownloadImage {  
    float: left;  
    margin-top: 6px;  
    margin-top: 6.22px;  
    margin-left: 16px;  
    width: 14px;  
    height: 19px;  
}  
.DotsMenuNoBackground { ... }  
.DotsMenuPrintImage {  
    float: left;  
    margin-top: 6px;  
    margin-top: 6.22px;  
    margin-left: 13px;  
    width: 18px;  
    height: 17px;  
}
```

