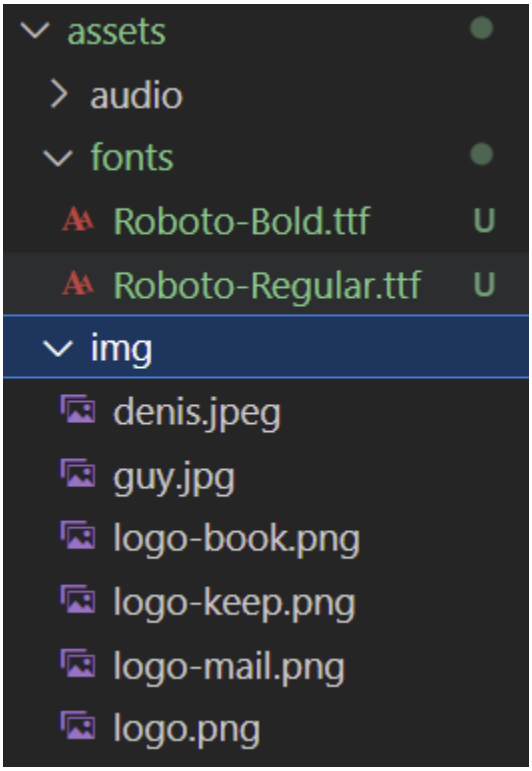
GitHub Pages - Paths

1. Place your photo and font files in the correct directory within your project.



2. Use relative paths instead of absolute paths when referencing your photo and font files. This means starting the path from the current directory rather than from the root directory of your computer.

```
@font-face {
  font-family: Roboto-bold;
  src: url(../../fonts/Roboto-Bold.ttf);
}
```

3. Use **src** for images and **href** for fonts in HTML to reference the files. Use the **url()** function in CSS to reference images and fonts.

4. Make sure that your file names and directory names are spelled correctly and that your photo and font files are actually included in your project.

5. Avoid using a leading forward slash **/** in your file paths. Using a leading slash will tell the browser to look for the file path from the root directory of your domain, which may not be what you want. Instead, use a relative file path that starts from the current directory. (Or you can use '. /')

```
  <link rel="stylesheet" href="css/styles.css" />
  <title>Appsus</title>
</head>
<body>
  <div id="app"></div>
  <script src="lib/recorder.js"></script>
  <script src="lib/vue.js"></script>
  <script src="lib/vue-router.js"></script>
  <script type="module" src="js/main.js"></script>
```

6. Finally, make sure to name your main HTML file **index.html**. This is the default filename that most web servers look for when a user visits your website's root directory, so it's a good convention to follow. If you don't name your main HTML file **index.html**, you'll need to specify the filename in the URL when you want to access it.

If you still have trouble getting your photos and fonts to display on GitHub Pages, you can use your browser's developer tools to inspect the elements and see if the path to the file is correct. You may need to adjust your file path or directory structure to make it work.