Folder Structure

> assets/styles
- All project-related CSS files will be placed here.

> assets/fonts
- All fonts related to the project will be placed here.

> assets/images
- Images that are being used in this project, will be placed here.

> components
- Your components will be placed here.
- You have to create a folder for your page and place components of this page into this new folder.

> containers
- This folder is for future development.

> data
- Common variables which are static should be placed in appropriate files here

> modules
- Utility classes or functions need to be placed here.

> pages
- This folder is for the general pages

> redux/action
- All your API call actions will be placed here and grouped in their respective folder names.
- For all action calls, use common functions placed in modules/api.js
- Folders in this directory need to follow the same naming convention as done in the actions folder.

> redux/types
- Types for redux will be placed in this folder.
- Folders in this directory need to follow the same naming convention as done in the actions folder.

> redux/reducer
- reducer related files will be placed in this folder
- Folders in this directory need to follow the same naming convention as done in the actions folder.

> shared
- Your common HOC components will be placed here.

Notes - Code Standard

# Class components should not be used if not needed.

# Usage of redux in functional components should be with useDispatch & useSelector.

# Common components should have PropTypes defined for better scalability

# You must install the packages below for VS CODE plugin. We recommend using VS Code for your development process. But, if you are using any other IDE or editor, you will have to find similar packages for that.
> Code Spell Checker

- For spelling mistakes: We do not want to have any spelling errors in any file. Even your file names should not have any spelling errors in them.

# Naming convention
- Keep using the same naming convention in files and variables which are being used.
- Preferable: Camel Cases

# Before making your PR, you must have to take the latest code from the source branch.

# Even if you do not have to merge your code, push your code daily and take the latest code from the source branch.

# Refrain from using window and document variables. You are only allowed to use these variables if you do not have any other solutions available.

# Add proper conditions to your code. Make sure that, if BE does not provide you with any param or wrong format of that param, the current page should not break. To do this, try to use - Optional Chaining.

# Do not make any edits in style.css, as we are importing this from the HTML repository. If you do any changes to this file, there is a huge chance that your changes will be overwritten by doing this. To overcome this, we have added a custom.css file. You will be using this custom.css file for your CSS rules.

# In case of conflicts, do not blindly discard incoming changes. Check your current and incoming changes and after that, decide on which to keep or remove or both needs to be kept. If you are not sure, try to contact the person who had updated the code on the conflicted file. To identify this, use VS Code plugin - GitLens.

# Strings like tokens and other configuration variables which should not be there in GIT will be placed in the .env file at the root level. If you add new keys in the .env file, do not forget to add those new keys in the .env.sample file, so others can add those keys in their .env file.

# Do not bypass commit prettier in any way. It is intended to keep all code in one format standard.

# If someplace, user click has any action, make sure it has “cursor: pointer;” CSS applied. If it does not have, add this class - “cursor-pointer” to that element.

# Imports in your component should be grouped and follow the below order. Each group must have a blank line break for easy identification.
- packages
- components/pages/containers
- HOC
- actions
- modules
- assets
- any other file imports
- variable declarations

# Make sure we are showing some indication that we are fetching data with API.

# If any popup is being opened, that popup should close on outside click if not specifically has been told or mentioned. If the popup has a form, user filled data should be kept until that form has been submitted or data cleared.

# Any page should not have multiple popups opened at any single time if not specifically has been told or mentioned to have them open.

# Try to use Formik and Yup for form and validation handling.

# Do not install unnecessary packages.

# Make sure that there are no unused variables and functions left in your code. Also, the same goes for the console.*. Only keep your console if you need to debug on the server with that. Once done, do not forget to remove them.

# There should not be any commented code. If you need some code for backup, keep that in your local. If you need some code for future purposes make sure that you add proper comments for that part. It would be best to use Better Comments which is available for VS Code. Like this -

# To use VS Code with absolute imports:
https://medium.com/@fpastorelima/using-absolute-imports-with-vscode-and-create-react-app-dec6ba0da7ed

Notes - Deployment in Sub-Directory

1) package.json file changes

```
"homepage": "."
```

2) env file changes

```
REACT_APP_PUBLIC_URL = {SITE_URL}
PUBLIC_URL = {SITE_URL}
REACT_APP_BASE_NAME = /{SUBDIRECTORY_NAME}
```

```
REACT_APP_PUBLIC_URL = http://13.228.53.106/okletsplay/
PUBLIC_URL = http://13.228.53.106/okletsplay/
REACT_APP_BASE_NAME = /okletsplay
```

3) In router add base name

```
<Router basename={process.env.REACT_APP_BASE_NAME}></Router>
```

4) .htaccess file

```
Options -MultiViews
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /PROJECT_DIR_NAME/
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-l
RewriteRule . /PROJECT_DIR_NAME/index.html [L]
</IfModule>
```

```
Options -MultiViews
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /okletsplay/
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-l
RewriteRule . /okletsplay/index.html [L]
</IfModule>
```

Notes - Code Enhancement

> Remove dependencies which are not being used

```
npx depcheck
```

> Detect all files which are not being used by detecting if those files have been imported or not

```
npx unimported
```