Contents

1. Contents

[Node JS set up : 2](#_Toc110258060)

[CREATING REACT APP 2](#_Toc110258061)

[To Run Unit Test 3](#_Toc110258062)

[BEST PRACTICES TO BE FOLLOWED: 4](#_Toc110258063)

[ASSETS: 4](#_Toc110258064)

[USER STORIES: 4](#_Toc110258065)

[HOME PAGE IMPLEMENTATIONS 4](#_Toc110258066)

[PRODUCT PAGE IMPLEMENTATIONS 5](#_Toc110258067)

[FEEDBACK IMPLEMENTATIONS 7](#_Toc110258068)

[ABOUT-US PAGE IMPLEMENTATIONS 8](#_Toc110258069)

[CONTACT-US PAGE IMPLEMENTATIONS 10](#_Toc110258070)

[Unit Test: 12](#_Toc110258071)

[Expected output 12](#_Toc110258072)

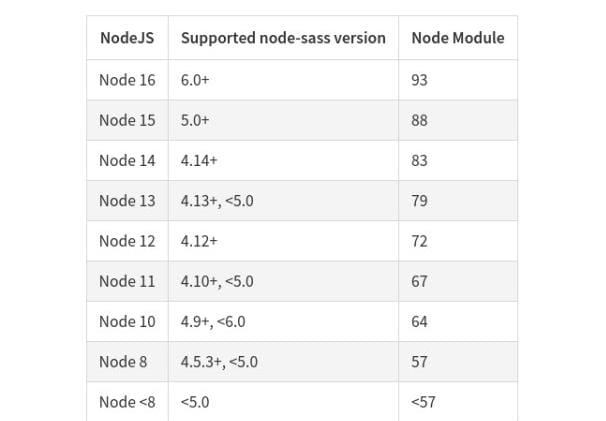
# Node JS set up :

1. Install Node version: 12 and above
2. Open your terminal and verify if node npm is installed by using the below command

**$node -v and $npm -v**

1. Open up the code editor after creating the react app given in the below section

Supported node-saas version



# CREATING REACT APP

1. Check whether node is up and running as per above version and module details
2. Create react project using the below commands

**npx create-react-app appname**

**cd appname**

**npm start**

1. Review your project structure. Check whether all the npm modules and json files are generated
2. Go to above the location of app created ( Example : **cd D:\foldername\app-name**)

**npm install -g json-server** (You could see similar messages like below)

>npm install -g json-server

added 179 packages, and audited 180 packages in 41s

13 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

Create db.json file with the json data from this github repository.

<https://github.com/arunprabu/plain-ui-react-course-feb-2023/blob/master/youtube-react-app/db.json>

Then,

**json-server --watch db.json**

(You could see similar messages like below)

\{^\_^}/ hi!

Done

Resources

http://localhost:5000/products

http://localhost:5000/categories

http://localhost:5000/contactData

http://localhost:5000/getInTouchData

Home

http://localhost:5000

Type s + enter at any time to create a snapshot of the database

Watching...

The server is running in **localhost:5000**

**Do not disturb the above Terminal/CMD window.**

Let us ensure the client/front-end is ready.

Open new terminal/CMD window with location of skyway-app (Refer Step 4)

**npm install** (You could see similar messages like below)

added 2048 packages, and audited 2049 packages in 7m

156 packages are looking for funding

run `npm fund` for details

10 moderate severity vulnerabilities

To address all issues (including breaking changes), run:

npm audit fix --force

Run `npm audit` for details.

npm start

now client/front-end is running in localhost:3000 port (opens the application )

http://localhost:3000/

# To Run Unit Test

To run unit test , write unit testcases and run the following command.

n**pm run test**

NOTE:

Images and Contents for the project is provided in the assets. Just ensure, all the requirements are implemented.

# BEST PRACTICES TO BE FOLLOWED:

* Keep component creation as minimum. Reuse components wherever necessary
* Code should be testable. Hence always write clean code
* Define Proper file structure
* Prefer pure functions and immutable data wherever necessary
* Keep modules independent wherever applicable
* Follow appropriate naming of Props

# ASSETS:



# USER STORIES:

## HOME PAGE IMPLEMENTATIONS

|  |  |
| --- | --- |
| Story no | U01 |
| Title | Home Page Implementation |
| Description | Design a page as given in the expected output. Logo and designs are customisable |
| Requirement | 1. Split the Home page into following sections  * Navbar * Menu * Image and Contents for Carousel   **Implement Navbar & Carousel Functionality**  Menu with Icons should be available for below listed options   * Logo with Name * HOME * PRODUCTS * ABOUT US * CONTACT   **Routing URL for this page should be designed as below :**  (URL:/#/) |
| Acceptance Criteria | When we click Logo or Home, it should show Home Page. When we click ABOUT US/CONTACT US and PRODUCTS, it should show ABOUT US, PRODUCTS and CONTACT US respectively with its contents.  Mandatory Criteria,   * Write Unit test for all the code |
| Assets Information   * Screenshots for Expected Output * Images * Icons * Stylesheet (You are supposed to add the text as per each user story. Also, you are free to add and modify the styles as per your customisation. This is just sample recommendation) * Fonts | **FOLLOW BELOW FOLDER STRUCTURE TO MODIFY/ADD NAVBAR Title And Image(logo):**   * src\components\Navbar\Navbar.js * Add the image for logo in public/assets/images/logo.png from the assets zip file. * Load footer social media icons from font-awesome.   **EXPECTED OUTPUT:**    **Note: Ignore the Cart button** |

## PRODUCT PAGE IMPLEMENTATIONS

|  |  |
| --- | --- |
| Story no | U02 |
| Title | Product Page Implementation |
| Description | Design a page by listing Products available under clothing. Expected output is given as sample |
| Requirement | * 1. When navigating to the products page, products with clothing category should be listed along with images for the product and price details  * 1. Write grouping logic based on the categories and display under each category.   2. When clicking ‘All’ from ‘category sidebar’ the URL should be localhost:3000/#/products/   3. When clicking ‘Men’ from ‘category sidebar’ the URL should be localhost:3000/#/products?category=men   4. When clicking ‘Women’ from ‘category sidebar’ the URL should be localhost:3000/#/products?category=women   5. When clicking ‘Kids’ from ‘category sidebar’ the URL should be localhost:3000/#/products?category=kids   6. Upon category clicked load only products that belong to the clicked category from the backend.   7. Sort by Option should show only **Price – Low to High** and **Price - High to Low.**   **Routing URL for this page should be designed as below :**  (URL: /#/products/) |
| Acceptance Criteria | Once user click on the Products Title they will be redirected to Products details page.  **Mandatory Criteria,**   * Write Unit test for all the code |
| Assets Information   * Screenshots for Expected Output * Images * Icons * Stylesheet (You are supposed to add the text as per each user story. Also, you are free to add and modify the styles as per your customisation. This is just sample recommendation) * Fonts | **FOLLOW BELOW FOLDER STRUCTURE TO ADD IMAGE AND TITLE:**   * src\pages\ProductsPage\ProductsPage.js * Also Create a component – to display categories such as All, Men, Women, Kids in sidebar in the appropriate project directory. * Load products and categories also from REST API Backend.   **EXPECTED OUTPUT :** |

## PRODUCT DETAILS PAGE IMPLEMENTATIONS

|  |  |
| --- | --- |
| Story no | U03 |
| Title | Product Details Page Implementation |
| Description | Design a page to display product details. Expected output is given as sample |
| Requirement | * 1. When navigating to the product details page, the UI should show necessary product specific information   2. Add the product title and description   3. Work with id as URL parameter to make the URL dynamic for every product detail page.   **Routing URL for this page should be designed as below:**  (URL:/#/products/1/ and /#/products/2/ and /#/products/3/ and so on) |
| Acceptance Criteria | Once user click on the Write a Review, bootstrap modal should appear with form to enter name, email, phone, review comments and rating.  **Mandatory Criteria,**   * Write Unit test for all the code |
| Assets Information   * Screenshots for Expected Output * Images * Icons * Stylesheet (You are supposed to add the text as per each user story. Also, you are free to add and modify the styles as per your customisation. This is just sample recommendation) * Fonts | **FOLLOW BELOW FOLDER STRUCTURE TO ADD IMAGE AND TITLE:**   * Load product details from REST API Backend.   **EXPECTED OUTPUT:** |

## REVIEW PAGE IMPLEMENTATIONS

|  |  |
| --- | --- |
| Story no | U04 |
| Title | Review Page Implementation |
| Description | Design a page to provide reviews by the customer by providing ratings to the products |
| Requirement | 1. This review form should be displayed in product page. The customer can enter feedback on each product 2. Whenever any customer submits more than one review for a product, throw custom exception as “Review Already Submitted”. 3. All the review submitted by the user should be displayed under Total Reviews.   **Note: The forms can be customised as per your own design. Routing URL for this page should be designed as below:**  (URL:/#/products/1) |
| Acceptance Criteria | User can provide ratings to product and submit. If ratings are already submitted throw the exception.  **Mandatory Criteria,**   * Write Unit test for all the code |
| Assets Information   * Screenshots for Expected Output * Images * Icons * Stylesheet (You are supposed to add the text as per each user story. Also, you are free to add and modify the styles as per your customisation. This is just sample recommendation) * Fonts | **F****ollow the below folder structure to add Feedback for the products:**   1. Submit the feedback thru the modal. Save the review comments on the back-end side by sending PUT request over http.   **EXPECTED OUTPUT:** |

## ABOUT-US PAGE IMPLEMENTATIONS

|  |  |
| --- | --- |
| Story no | U04 |
| Title | About US Page Implementation |
| Description | Design a page to provide information about the Clothing Shop |
| Requirement | 1. While clicking on About us page, the expected output provided should be appear. You can create your own them and icons. 2. About US – Some tagline with title and detailed description 3. Implement Nested Routing here.   **Routing URL for this page should be designed as below:**  (URL: /#/about-us/ and /#/about-us/history )  **TEXT Contents for pages:**  **===>About Us Tagline**  “We have the capabilities and experience to deliver the products you need to move forward.”  **===>The History**  The story of Spark Clothing started in Bangalore in the year 2000 with wholesale of hosiery and woven garments, catering to the needs of various wholesalers and retailers across India. Bangalore textiles industry stands at USD 4.2 billion in 2012 and is globally famous for its hosiery products. The City has more than 5,000 garment manufacturing units and job work units, and is one of the most organised processing and finishing garment clusters in India. The firm came into existence in the year 2005 as a Sole Proprietorship concern in Bangalore, Tamil Nadu. In the year 2007 it started as family enterprise with a tiny shop selling Men’s apparel at much lesser price than market and expanded to full men’s wear during next six months. The journey continued and in 2011 launched its first exclusively brand store in Bangalore with bigger retailing space for Men’s wear, Women’s wear and Kids clothing.  Spark Clothings is engaged into multiple business activities which include manufacturing, supplying and trading in a wide array of clothing products which include Women’s Wear, Polo T-Shirt, Girls T-Shirt and many more. The range offered is well acknowledged for their availability in stylish pattern, perfect finish and comfort in wearing. Products are developed after careful attention and hard work of Spark team. It aspires to render quality striven products which guarantee cheerfulness of end customer. |
| Acceptance Criteria | When clicking on About Us page, the expected output should appear.  **Mandatory Criteria,**   * Write Unit test for all the code |
| Assets Information   * Screenshots for Expected Output * Images * Icons * Stylesheet (You are supposed to add the text as per each user story. Also, you are free to add and modify the styles as per your customisation. This is just sample recommendation) * Fonts | **EXPECTED OUTPUT :**  UI can be as per your wish. Have two URL’s.  (/#/about-us/ and /#/about-us/history)  Implement Nested Routing for this feature. |

## CONTACT-US PAGE IMPLEMENTATIONS

|  |  |
| --- | --- |
| Story no | U05 |
| Title | Contact US Page Implementation |
| Description | Design a page to provide information about the Contact Details |
| Requirement | While clicking on Contact us page, the below things should be appear. You can create your own them and icons.   * Address * Phone * Email * Logo * Navigation to Home, Products, About Us and Contact US  1. Add Title and description, Address, Phone and Email in json file. 2. Sample Data provided. You are free to modify the data as per your own values   "contactData": {  "address": "280 ParK Avenue Z,Cross cut Complex, Bangalore, India",  "phone": [  "(91) 987 654 3210",  "(91) 987 654 3211"  ],  "email": "contact@sparkclothing.com"  },   1. Create a GET In Touch Form with the below values 2. After adding get in touch details it will be stored in products json file   public/assets/data/ <<products json file>>  You can view the get in data in getInTouchData array  "getInTouchData": [  {  "name": "aswin",  "email": "aswin210@gmail.com",  "message": "Hi",  "id": 1  }  ]  **Routing URL for this page should be designed as below :**  (URL:/#/contact-us/) |
| Acceptance Criteria | When clicking on Contact us page, the expected output should appear.  **Mandatory Criteria,**   * Write Unit test for all the code |
| Assets Information   * Screenshots for Expected Output * Images * Icons * Stylesheet (You are supposed to add the text as per each user story. Also, you are free to add and modify the styles as per your customisation. This is just sample recommendation) * Fonts | **Follow the below folder structure to add header in Contact Page:**   * Save the form data on the back-end side. * Fetch the above address phone and email by sending REST API call over GET method. * Submit the Form data to REST API over POST Method.     Fetch the above address phone and email by sending REST API call over GET method.  Submit the Form data to REST API over POST Method. |

# Unit Test:

* All React components should have minimum **80%** code. Ensure that **Lines and Statements** code coverage individual should not be less than **80%**.
* All components must have individual spec file associated with component file.
* Run below command to generate coverage report:

**ng test –code-coverage**

## Expected output

on application root folder inside **coverage** folder index.html as coverage report should generate which reflects required % as mentioned above as sample

Graphical user interface, application, table

Description automatically generated

Responsive Requirements:

* Application should be responsive which means the screen layout will adjust on mobile and Tab screen without element overlap
* Application should run on mobile browsers (ios and Android) (Chrome at least)
* On running on mobile browser or tab screen should not have horizontal scrollbar
* Application layout should follow **Mobile first approach**
* Application layout should adjust as free flow (Left to Right) additional element wrap downside.
* In mobile screen single column layout
* More than 600 2 columns and after 700px as desktop layout.