

Fixing Issues & Wrapping up TextUtils | Complete React Course in Hindi #19

Earlier, we created our Textutils application. To check out our app, visit mytextutils.com. Today, we are going to enhance our TextUtils application by fixing some issues.

Here are some of the issues in our TextUtils application:

1. Word Counting Error
2. No Mobile friendly buttons
3. No Relevant About Text
4. The layout is shifting when an alert appears (while clicking a button)
5. The cursor is invisible in Dark mode
6. Expected Reading Time Error
7. No SEO optimization
8. Buttons are still working when the textbox is blank.
9. No SSL due to which the 'copy Text' button isn't working
10. Deselecting the Copied Text

So let's start our application in localhost and begin solving these issues one by one:

1. Word Counting Error

We have created the function of Counting words in our Textform.js. The `split()` method is used to convert the string into an array. In our case, We are splitting the string by a space and then counting the length of that array, which comes out to be one. Therefore, a blank space is being counted as a word.

Solution: The quick way of fixing this error is by removing the empty strings from the split array. We can easily do so by using the filter method in our Text.split function as described below:

```
{text.split(" ").filter((element)=>{return element.length!==0}).length}
```

If the arrow function returns the true value for an element, then only it would be allowed to stay in the split array. The `filter()` does not execute the function for empty array elements.

Our Error has been Successfully Resolved.

2. Adding Mobile-Friendly Buttons



Figure 1.1: No Mobile-Friendly Buttons

The above error is occurring as there is no margin for the buttons in the Y direction. To solve this issue add some margin to the buttons, available in "textform.js", in the Y direction. Our Error has been **successfully** resolved.

3. Enhancing "about.js"

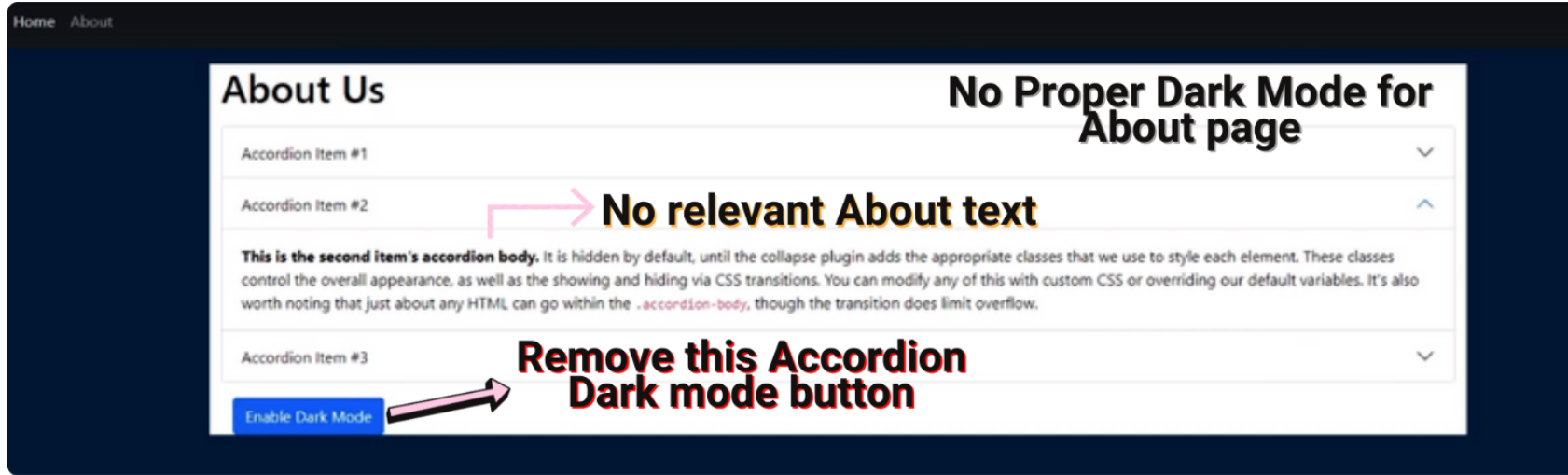


Figure 1.2: Unfavourable About Page

We would remove the Dark mode button of the Accordion and also the ToggleStyle function assigned to it from about.js. We are adding our desired text in Accordion and editing it in our own way.

Adding a proper Dark Mode to about.js

To add dark mode to our about.js, follow the below steps.

Step 1: Firstly, Pass Mode in About component of App.js as

```
<About mode={mode}/>
```

Step 2: We would create a new 'myStyle' variable which would provide our desired design to the About page, on enabling the dark mode.

```
let myStyle = {
  color: props.mode === 'dark'? 'white': '#042743',
  backgroundColor: props.mode === 'dark'? 'rgb(36 74 104)': 'white'
}
```

Step 3: We have used conditional operators to set color while changing modes. After that, we would pass this "myStyle" to the Accordion component, and We would only pass the color to the heading because we would like to differentiate the heading from the rest of the component. You can use any of your desired colors to style the components.

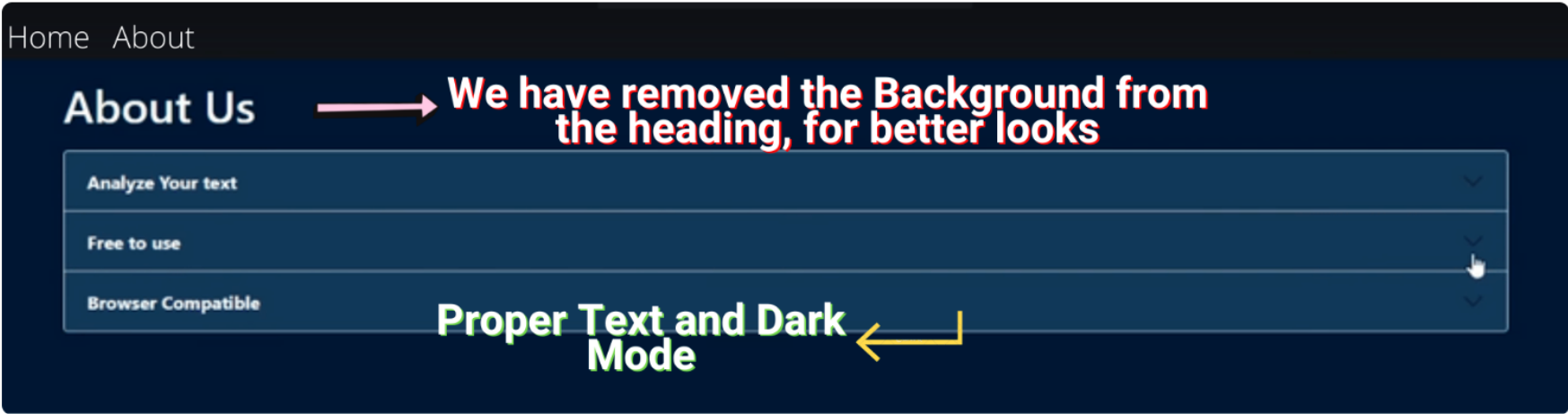


Figure 1.3: About Page of our Application
We have successfully enhanced our About Page

4. Layout Shifting Issue

The layout of our application is shifting when an alert is shown. Remember that the layout shift must be minimum for your application, as this is one of the ranking factors from the SEO point of view.

Solution: To resolve the issue, go to your "alert.js", and inside the alert component, we would wrap the alert inside a div container having a specific height.

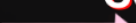

```
return (  
  <div style={{height: '50px'}}>  Set height of the container  
    {props.alert && <div className={`alert alert-${props.alert.type}  
      alert-dismissible fade show`} role="alert">  
      <strong>{capitalize(props.alert.type)}</strong>: {props.alert.msg}  
    </div>  
  </div>  Wrap Alert inside Div container  
)
```

Figure 1.4: Solving CLS issue

As a result, the Alert component will have a specific height for its display, and hence it would not shift the layout.

Our CLS Error has been **successfully** resolved.

5. Cursor Invisibility in Dark Mode

Our cursor is becoming invisible on hovering in the textbox due to its grey color. To solve this problem, simply change the color of the "textarea" from grey to blue. We can easily do that by going to the textarea in "textform.js" and editing its existing style.

```
<h1>{props.heading}</h1>
<div className="mb-3">
  <textarea className="form-control" value={text} onChange=
  {handleOnChange} style={{backgroundColor: props.mode==='dark'?
  '#13466e': 'white', color: props.mode==='dark'? 'white': '#042743'}}
  id="myBox" rows="8"></textarea>
</div>
```

Changing Color of the Textbox from grey to #13466e (blue Color)

Figure 1.5: Solving Cursor Invisibility

Our cursor is visible in the dark mode.

6. Expected Reading Time Error

Our application will be showing the expected time to read the provided text, but if the Textbox is blank then also our Time counter is showing an expected time to read the text. To fix this bug in our application, we would use the filter method as we have used to solve the counting word error.

```
<p>{0.008 * text.split(" ").filter((element)=>{return element.length!==0}).length} Minute rea
```

Our Reading Time Error has been **successfully** resolved.

7. Fixing for SEO

SEO is one of the factors which helps our website to rank on Google. To make our application SEO optimized we would make the following changes to our Index.html and Textform.js:

In index.html:

Meta Description: It provides a brief sketch of the content of the website to the Google crawler. Therefore, it becomes utterly important to add specific keywords related to our content in the meta description.

Title: It specifies the Content of our specific web pages. Hence, we have changed the title of our application by adding some relevant Keywords to it.

```
<meta
  name="description"
  content="TextUtils is a word counter and a character counting utility which can be used to manipulate your text in the
  way you want. You can remove extra spaces, copy the manipulated text as well as convert your text from Uppercase to
  lowercase and lowercase to Uppercase" />
<link rel="apple-touch-icon" href="%PUBLIC_URL%/favicon-16x16.png" />
<link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
<!-- Bootstrap CSS -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/css/bootstrap.min.css" rel="stylesheet"
  integrity="sha384-KyZXEAg3QhQLMpG8r+8fhAXLRk2vvoC2f3B09zVXn8CA5QIVfZ0J3BCsw2P0p/We" crossorigin="anonymous">
<title>TextUtils - Word counter | character counter | lowercase to uppercase | uppercase to lowercase</title>
</head>
```

Meta Description of Our Application

Title of Our Application

Figure 1.6: Adding Meta Description and Title

In textform.js:

Editing Headings: It plays a major role in the SEO optimization of our website as it makes it easier for a User and Crawler to navigate through the page. We will be passing a new heading, having relevant keywords related to our application, in props.heading from app.js.

```
<Switch>
  { /* /users --> Component 1
    /users/home --> Component 2 */ }
  <Route exact path="/about">
    <About mode={mode} />
  </Route>
  <Route exact path="/">
    <TextForm showAlert={showAlert} heading="TextUtils - Word Counter, Character Counter,
    Remove extra spaces" mode={mode} />
  </Route>
</Switch>
```

Passing New heading to props.heading, in Textform.js

Figure 1.7: Passing New Heading

We have optimized our app for search engines.

8. Disable button

When our Textbox is empty, but our buttons are working and performing their functions, which seems like a bug to our application,

Solution: To overcome this issue, we would add the below code to every button.

```
disabled={text.length===0}
```

This code means to disable the buttons when the length of the text is 0. That is, the textbox is empty. Error has been **successfully** resolved.

9. Enabling HTTPS

The copy text button of TextUtils isn't working as we don't have https. To enable https for our application, simply run the following commands in Git Bash or any other terminal.

1. Install Certbot

```
sudo apt-get install certbot
```

2. Certbot Nginx

```
apt-get install python3-certbot-nginx
```

3. Verifying the Syntax and Restarting Nginx

```
nginx -t && nginx -s reload
```

4. Obtain the SSL certificate

```
sudo certbot --nginx -d mytextutils.com -d www.mytextutils.com
```

Make sure to run this command for your domain. We have used it for our mytextutils.com domain.

5. After that, fill up some details like entering your Email and select your favorable options.

HTTPS has been Enabled for the myTextutils website.

10. Deselecting the copied text



Figure 1.8: Selected Copy Text Issue

Whenever we copy the text, it results in selecting the entire text.

Solution: To overcome this issue, we will edit our `handleCopy` function, available in "textform.js".

Add the below code in `handleCopy` function:

```
document.getSelection().removeAllRanges();
```

Our issue has been **successfully** resolved!

Deploying the Application

Create your Build folder by using the `npm run Build` command and drag and drop the build folder in Filezilla as we have done in the previous tutorial, and restart your nginx. All changes will be reflected in our application.

Check out our application by clicking [here](#).