Blogitt Technical documentation

1. Technology:

This website Is made using these tools

* 1. ReactJs
  2. React Router
  3. SASS/SCSS
  4. JSON-Server
  5. Icons imported from the Material Design UI Library

1. Components:
   1. Navbar: contains the logo and 2 NavLinks for the create post page and the hompage.
   2. Sidebar
   3. Post preview: shows some of the content of the post as well as its votes and comments, you can also like or dislike the post from the preview.
   4. Add comment: a form to submit a comment to a post using the action prop in React Router.
   5. Comments: a component that renders the comment for a post using the Loader prop in React Router can add likes and dislikes to a comment using the handleVote method and can delete the comment using a handleDelete method.
   6. Post Details: show the entire content of the post as well as the ability to delete it.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Pages:

Scroll down for more information about each page

* 1. index
  2. Homepage
  3. Create a Post
  4. Post Page
  5. Alert Page

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3.a) Index:

This page serves as the main Route wrapper for all available routes and begins by creating a BrowserRouter using the createBrowserRouter() method which creates a main route

Then you create Routes for every page inside this method using

Your created elements with createRoutesFromElements()

Then we start creating the routes for all our pages using the Route

Element and nest more routes when needed, finally we return the browser router element which is the RouterProvider, This element is injected inside the App element that in turn is injected with whaterver route is present.

3.b)Homepage:(‘/’)

The homepage is part of the RootLayout page which is a parent route element that renders the navbar and a loading bar, the component begins by calling the useFetch custom hook that fetches all available posts as a post preview component from the database, then it detatches the loading bar from the dom and also shows the sidebar.

3.c) Create post:(/create)

This page lets you create a new post, you type the title of post, your name and the post’s content and hit the create button, this button fires a handle submit function that sends a post request to the server with the post content and 3 counters for the votes and comments.

3.d)Post Page:(blogs/:id)

This page shows the entire content of a post as well as the ability to delete it using the handle delete method, which first fetches all comments for this post and deletes them before finally deleting the post and returning a Deleted Alert Page the likes and dislikes are incremented and updated using the useVote custom Hook and the handler function handleVotes.

3.e) Alert Page:

This component takes the alert type, name and content as props

Then renders them to the dom with a return to homepage link, it acts as an error element.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Project Structure:

/data

/db.json : the database used for json server

/public : contains the website icon, BG image and main HTML

/scr

/components: holds the static componets of the application

/handlers: holds the handle functions of the application

/hooks: hold the custom hooks of the application

/layouts

/RootLayout: the main layout of the application

/BlogLayout: the Layout of the post Page

/loaders

/commentsLoader: convension name for the loader function that fetches the comments in the load prop of a route

/props: holds the dynamic components of the application such as a blog component or an alert

/styles:

/index.css: styles of the application

/styles: holds the SCSS flies of the application

/components: styles of different components of a page

/pages: styles of different pages

/animations.scss : holds the animations and effects

/breakpoints.scss: holds the mixins of media queries of screens sizes

/variable.scss: common props used across the all files

/index.scss: the main file that is complied by the gulpfile into css, it holds all imports of most scss files

1. definitions:

Route

Which takes the arguments (path of the page, element to be rendered also known as Layout which is always rendered regardless of what the child routes are and then you render the current child route with the Outlet element, an errorElement if an error occurred), you can also add a Layout.

It also contains a Loader porp that takes a function which runs during the render of a route, usually is used to fetch data from the database it has a default variable called parameters which gives the route parameters of the route

1. we use the comments loader to fetch the data in a route
2. We get the data using the useLoaderData hook in our component.

The action prop is a function that runs when a Form component is submitted it has a default variable called request which is the request object of the METHOD request done by the submit

1. It’s first attached to a route where the form is located
2. Then you pass it the form data using the .formData() prop in the requst object
3. The function then procceds and returns a response of your making

NavLink

Similar to a link were it prevents communication with the server and instead injects the js bundles but also when active it adds a class to the anchor tag of “active” which makes styling a bit easier

useFetch

a custom hook that fetches all available posts from the database

It takes the url of the api endpoint, as an argument and returns an object that has a loading flag, an error flag and the fetched data.

It has the loading flag set to true first and the error and data to null, if the fetch is successful the data is changed from null and the error stays the same, if not the error is changet to true.

The fetch is also attached to a signal so it aborts if the user changed the page before it loads, the flag and data state hook are updated due to the presence of a useEffect hook.

useVote

Creates 2 useState hook for liking and disliking flags and returns their getters and setters.

HandleVote

Takes 4 arguments, the data of the component (comment, post) the num of likes/dislikes, type of vote, type of data, setters for the flag hooks of useVote, it increments the num of votes and sends a PUT request to the database to update the correspoding property.