

Name- Sankalp Gupta

Roll No.- 102103784



Implementation Report: News Dashboard Project



Project Overview:

The News Dashboard is a React + Material UI-based application that fetches and displays top news headlines, allows filtering, exports data, and includes admin-specific views. Enhancements were made to improve **user experience**, **theme consistency**, **functionality**, and **interface design**.



1. Theme Toggle Functionality



File(s): **App.jsx**, **Home.jsx**, **theme.js**

- Implemented **light/dark mode toggle** using MUI's **ThemeProvider** and **CssBaseline**.
- Added a **setMode** state in **App.jsx** and passed it down to child components.
- Built a custom theme in **getTheme()** with distinct **light blue** and **dark blue** palettes.



Benefits:

- Enables user-friendly viewing in different lighting conditions.
 - Offers consistent contrast and color harmony across themes.
-



2. Consistent Button Styling

File(s): Home.jsx

- Standardized all button styles using theme-based color variables.
- Ensured consistent `sx` styling across:
 - Export
 - View Payouts
 - Author Chart
 - Login/Logout
 - Theme Toggle

Style Sample:

js

CopyEdit

```
sx={{  
  backgroundColor: "#abc4ff",  
  color: "#fff",  
  borderRadius: 2,  
  textTransform: "none"  
}}
```

Benefits:

- Visual consistency across all components.
 - Easy to maintain and theme-compliant styling.
-

3. Dynamic Article Filtering and Sorting

File(s): Home.jsx

- Implemented client-side filtering:
 - Keyword (`query`)


- Date range (**from**, **to**)
- Sort by (**publishedAt**, **popularity**, **relevancy**)
- Applied validations:
 - 'To' date must be after 'From'
 - Sorting is preserved post-filter

Benefits:

- Gives users precise control over what content they view.
 - Improved filtering logic enhances performance and UX.
-

4. CSV Export Functionality

 **File(s):** **exportUtils.jsx**, **Home.jsx**

- Added a utility to **export filtered articles to CSV**.
- Triggered via a dedicated “ Export to CSV” button.

Benefits:

- Enables offline analysis or backup of fetched news data.
-

5. Admin-Specific Navigation Logic

 **File(s):** **Home.jsx**, **AuthContext.js**, **App.jsx**

- Checked for **user?.isAdmin** before navigating to **/payouts**.
- Displayed alert if access is denied to non-admin users.

Benefits:

- Ensures proper access control for sensitive data.
-

✓ 6. News Data Visualization

 **File(s):** `AuthorChart.jsx`

- Passed filtered article data via route state to the Author Chart page.
- Used to display aggregated author contributions.

 **Benefits:**

- Helps visualize article distribution by authors in a chart format.
-

✓ 7. Responsive Layout with Material UI

 **File(s):** `Home.jsx`, global component files

- Used `Container`, `Grid`, `Stack`, and `Paper` for responsive layout.
- Ensured mobile-first and adaptive rendering.

 **Benefits:**

- Smooth performance and consistent look across all devices.
-

✓ 8. Authentication Integration

 **File(s):** `AuthContext.js`, `App.jsx`, `Login.jsx`, `Home.jsx`



- User context managed using `useAuth()` hook.
- Components render conditionally based on auth state.

Benefits:

- Dynamic rendering of login/logout UI.
 - Secures protected routes and features.
-

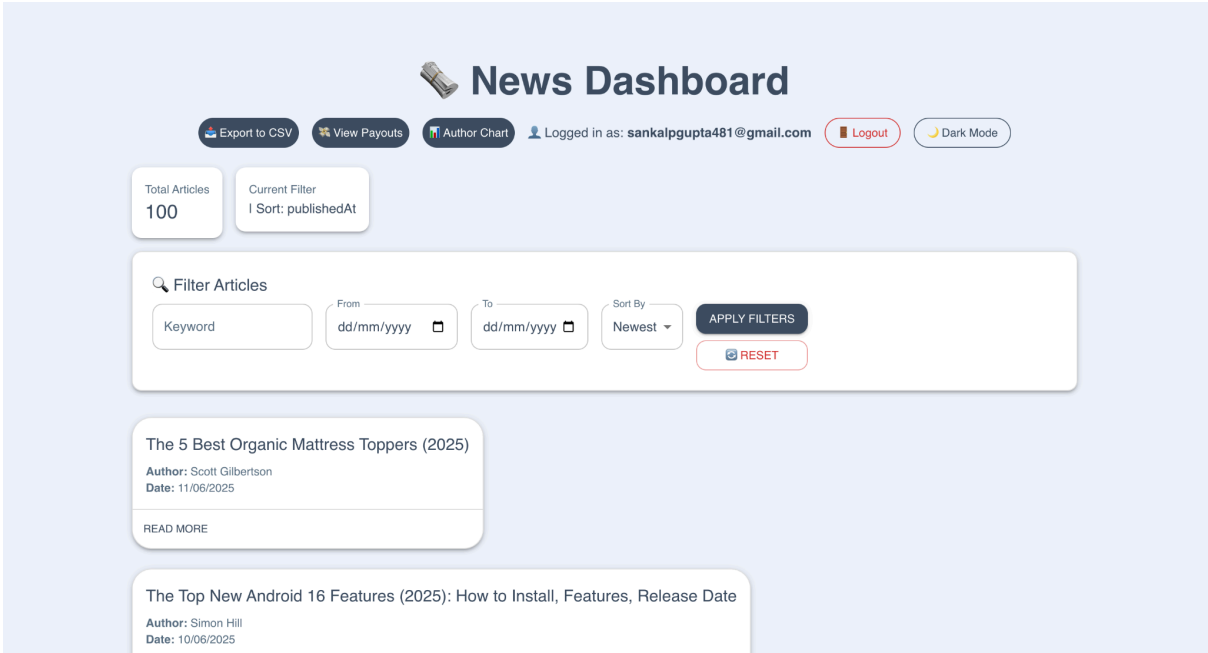
9. Error Handling and Loading States

File(s): **Home.jsx**

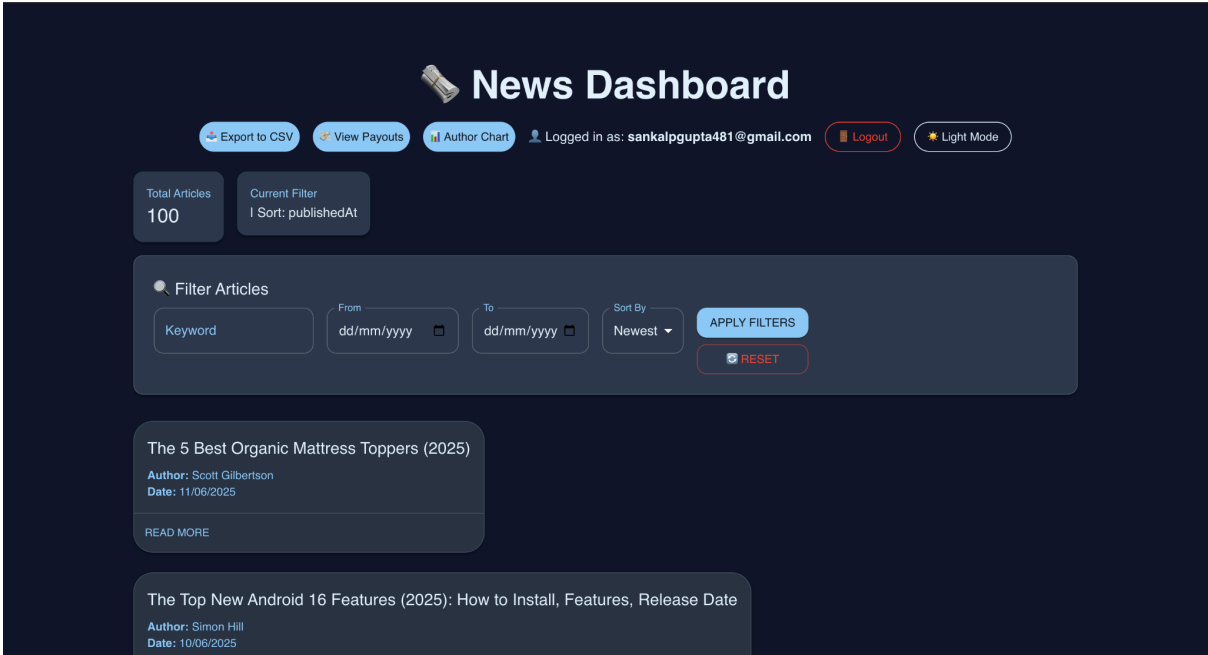
- Displayed loading indicators () and error messages () gracefully.
 - Handled API failures and empty results smoothly.
-

10. Code Cleanliness and Modularity

- Separated logic across reusable files:
 - `services/newsService.js` for API
 - `components/exportUtils.jsx` for CSV
 - `context/AuthContext.js` for auth
-



Light Mode



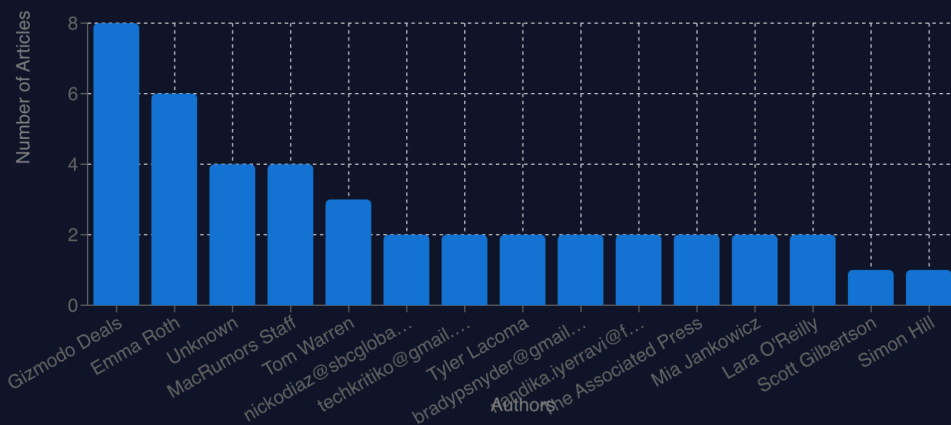
Dark Mode

🌸 Article Payouts

Author	# Articles	Payout Rate (\$)	Total (\$)
Scott Gilbertson	1	<input type="text" value="100"/>	100.00
Simon Hill	1	<input type="text" value="200"/>	200.00
Gizmodo Deals	8	<input type="text" value="300"/>	2400.00
Unknown	4	<input type="text" value="400"/>	1600.00
Germain Lussier	1	<input type="text" value="0"/>	0.00
Brittany Vincent	1	<input type="text" value="0"/>	0.00
Joe Tilleli	1	<input type="text" value="0"/>	0.00

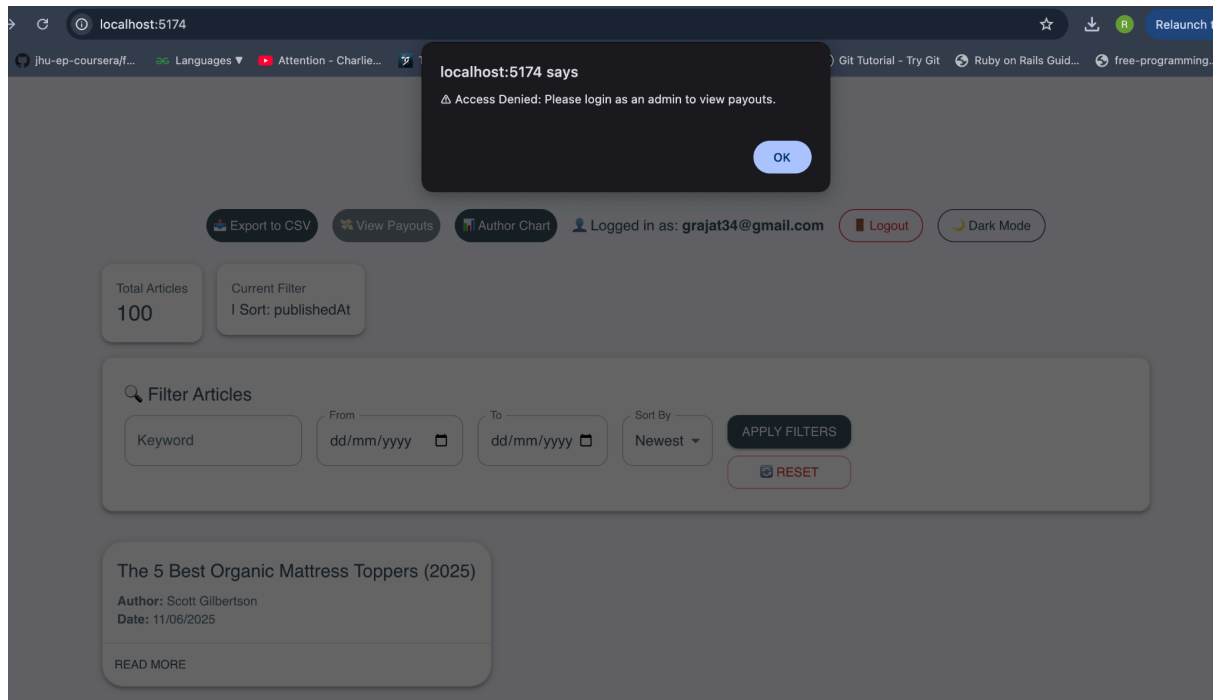
Payouts Admin Ascess

🧠 Author-wise Article Chart



← BACK TO HOME

Author Wise Chart



Restricting Admin Access

Setup Instructions

Important:

The `node_modules` folder has been **intentionally excluded** from this zipped project to reduce the file size below 25 MB for submission purposes.

This folder contains all the project dependencies and will be automatically restored by running the steps below.

Steps to Run the Project Locally

Follow the steps below to properly install and run the project on your local machine:

1. Install Node.js (if not already installed)

Make sure Node.js (which includes `npm`) is installed on your system.

- Download from: <https://nodejs.org/>
- Recommended version: **Node.js 18.x or later**

To check if it's installed, run:

```
bash
CopyEdit
node -v
npm -v
```

2. Install Project Dependencies

Open your terminal/command prompt, navigate to the root folder of this project, and run:

```
bash
CopyEdit
npm install
```

This command will read from the `package.json` and download all required packages into a new `node_modules` folder.

3. Start the Development Server

Once the dependencies are installed, start the Vite development server using:

```
bash
CopyEdit
npm run dev
```

This will launch the app locally, usually at:

👉 <http://localhost:5173/> (or as specified in the terminal)

4. Build for Production (Optional)

To generate a production-ready build:

```
bash
CopyEdit
npm run build
```

The output will be in the `dist` folder.

