**FEEDBACK**

**Challenges & Solutions:**

* **Challenge**: Ensuring that the dashboard is responsive across various devices can be time-consuming.
* **Solution**: Used Material-UI's responsive grid system and breakpoints to create a responsive design. Tested the application on different devices and screen sizes to fine-tune the layout.
* **Challenge**: Integrating with a backend API to fetch and update tasks can be challenging, especially handling errors and asynchronous operations.
* **Solution**: Implemented robust error handling for API requests using “try-catch”, used asynchronous functions with “async/await”.

**Security & Optimization:**

* **Authentication**: If our application involves user accounts and authentication, we can implement secure authentication mechanisms, such as JWT (JSON Web Tokens) or OAuth, to protect user data.
* **Authorization**: We can implement authorization rules to ensure that users can only access and modify their own tasks or tasks they have permission to access.
* **Client-Side Caching**: We can implement client-side caching for frequently used data to reduce unnecessary API calls and improve performance and this can be done using GraphQL.
* **Over Fetch & Under Fetch:** Using GraphQL can also help in solving problems of over fetch and under fetch in API calls.
* **Lazy Loading**: Lazy load images and components to reduce the initial bundle size and improve page load performance.