Assignment 2

Admin Dashboard

Respected Sir/ Mam,

I have chosen Assignment 2, Admin Dashboard, as my assignment. I have used **react-bootstrap-table-next** for creating the table as I had some experience with it in my previous internship. I started by creating the required columns and fetching data for them. **react-bootstrap-table-next** has a selectRow feature which automatically creates the first row as checkboxes and if the header checkbox is selected/deselected, all the checkboxes are selected/deselected

Furthermore, I have implemented **pagination** and **search functionality** as per the instructions provided.

For the edit and delete buttons, I have used **free icons**. The shown icons are pro in **Font Awesome**, so I have used free icons.

Additionally, I have also created a **drop-down to select the number of rows to appear on each page** as per user convenience.

The edit and delete buttons are working as per instructions and are editing and deleting the rows in memory.

I have allowed a row to be edited or deleted only if the checkbox of that row is selected. It means a row can only be edited or deleted if it is selected, otherwise not.

If a user tries to edit or delete an unselected row, an alert is shown.

I have used sweetalert2 which makes alerts visually stunning.

The page number count and row count are also displaying at the bottom of the table.

Major Problems I faced

1)The selectedRows, which is keeping track of ids of selected Rows in the table, was becoming [] inside edit and delete onclick functions. The issue I was experiencing is due to closures in JavaScript. The handleSingleDelete function was creating a closure over the selectedRows state at the time of the button rendering. When I was clicking the button, the handleSingleDelete function was executing with the selectedRows state that was present at the time the button was rendered, not the current state at the time of the button click.

To solve this issue, I used the useRef hook to store the selectedRows state. The useRef hook creates a mutable object with a current property. This object persists for the lifetime of the component and does not cause a re-render when its current property changes.

2)When editing the Name, Email, and Role values of a selected row, the focus on the textbox was going out after each character entered. I searched and found that “The issue of the input

field losing focus after each character is typed is a common problem in React when the input field is part of a component that gets re-rendered upon each state change". In my code, handleRowChange function was causing the component to re-render each time I was typing a character. This is because I was updating the state within this function. In React, state updates are asynchronous and cause the component to re-render.

To solve this problem, I have used the useCallback hook to memoize the handleRowChange function. This hook returns a memoized version of the function that only changes if one of the dependencies has changed. In this case, the dependency is data. This means handleRowChange will be the same function across re-renders unless data changes.

However, just using useCallback still won't solve the problem completely because state updates are asynchronous. When I type a character, handleRowChange is called, data state is updated, and the component re-renders. But, the new value of data may not be available immediately when you're typing the next character, causing the input field to lose focus.

To fix this, I have used a useRef to store the current value of the input field and update it synchronously. React provides the useRef hook for this purpose.