To design and implement a web application using Angular or React that allows users to view a list of profiles and interactively explore the addresses of each profile on a map, follow these steps:

Framework Selection: Choose either Angular or React for the project. ​ Both frameworks are suitable, but for this example, let's assume we are using React. ​

Project Setup:

Initialize a new React project using Create React App.

Install necessary dependencies such as React Router for navigation, Axios for API calls, and a map library like react-google-maps or react-map-gl for map integration.

Profile Display:

Create a ProfileList component to fetch and display a list of profiles.

Each profile should include essential information such as the person's name, photograph, and a brief description. ​Interactive Mapping:

Create a MapComponent that integrates with Google Maps or Mapbox. ​

Use markers to display addresses on the map. ​

Summary Integration:

Add a "Summary" button next to each profile in the ProfileList. ​

On clicking the "Summary" button, the MapComponent should display with a marker indicating the precise address of the selected profile. ​

Map Services Integration:

Utilize Google Maps or Mapbox for the map services. ​

Set up markers and ensure addresses are correctly rendered on the map. ​

User-Friendly Experience:

Ensure smooth navigation and interaction within the application. ​

Implement loading indicators to provide feedback during data fetching or map rendering. ​

Profile Data Management:

Create an admin panel or dashboard for administrators to add, edit, or delete profiles. ​

Use forms with validation to manage profile data efficiently. ​

Search and Filter Functionality:

Implement search and filter options to allow users to find profiles based on criteria like name or location. ​

Responsive Design:

Ensure the application is responsive and mobile-friendly. ​

Use CSS frameworks like Bootstrap or Material-UI to achieve responsiveness. ​

Error Handling

Implement robust error handling to manage issues like invalid addresses or failed map service requests gracefully. ​

Loading Indicators:

Include loading indicators or progress bars to inform users when data is being fetched or maps are being rendered. ​

Profile Details:

Create a separate ProfileDetails view to provide more in-depth information about each profile when a user clicks on a profile card. ​

Example Repository Structure:

Example Code Snippets:

ProfileList.js:

ProfileCard.js:

MapComponent.js:

AdminPanel.js:

ProfileDetails.js:-By following these steps and using the provided code snippets, you can create a fully functional web application that meets the requirements outlined in the case study. ​