# Technical Report: Rohith's Portfolio Landing Page

This report documents the React component `Landing` for Rohith's portfolio website. The component is a single-page application showcasing personal information, projects, and contact details.

1. Project Purpose:

The primary purpose of the `Landing` component is to serve as the main entry point for Rohith's online portfolio. It aims to provide a visually appealing and interactive experience for visitors, showcasing Rohith's skills, projects, and experience in a concise and engaging manner. The component uses animations and interactive elements to enhance user experience.

2. Key Modules, Classes, and Functions:

The `Landing` component utilizes several external libraries and custom components:

External Libraries:

1. `@emailjs/browser`: For handling email submissions.

2. `framer-motion`: For declarative animations.

3. `gsap` and `ScrollTrigger`: For advanced animation and scroll-based interactions.

4. `react-icons/fa`: For Font Awesome icons (FaBars, FaTimes).

Custom Components:

1. `CustomCursor`: Manages a custom mouse cursor.

2. `GooeyNav`: Implements a visually appealing navigation menu.

3. `InteractiveText`: Creates interactive text elements.

4. `SplitText`: Animates text by splitting it into individual characters.

5. `SpotlightCard`: Renders project cards with spotlight effects.

6. `TiltedCard`: Renders cards with a tilting animation.

Key Functions:

1. `sendEmail(e)`: Handles form submission and sends emails using `emailjs`.

2. `toggleMobileMenu()`: Toggles the mobile navigation menu's visibility.

3. `handleNavClick(e, href)`: Handles navigation clicks, scrolling smoothly to the target section.

3. Data Models or Entities:

The component uses two main data structures:

`navItems`: An array of objects, each representing a navigation item with a `label` and `href` property. Example: `{ label: "About Me", href: "#about" }`

`certifications`: An array of objects, each describing a certification with `text`, `link`, `image`, and `description` properties. Example: `{ text: "Google Machine Learning", link: "...", image: "/MachineLearningPreview.webp", description: "..." }`

`techStack`: Similar structure to `certifications`, holding information about technologies used, including `name`, `icon`, and `description`. Example: `{ name: "React", icon: "/React.webp", description: "Interactive front-end experiences" }`

4. Animation and Interactions:

The `Landing` component extensively leverages `framer-motion` and `gsap` for animations. Animations are triggered on component mount (`useEffect`) and on scroll (`ScrollTrigger`). Animations include:

Opacity transitions.

Position transitions (e.g., `y` for vertical movement).

Scale transitions.

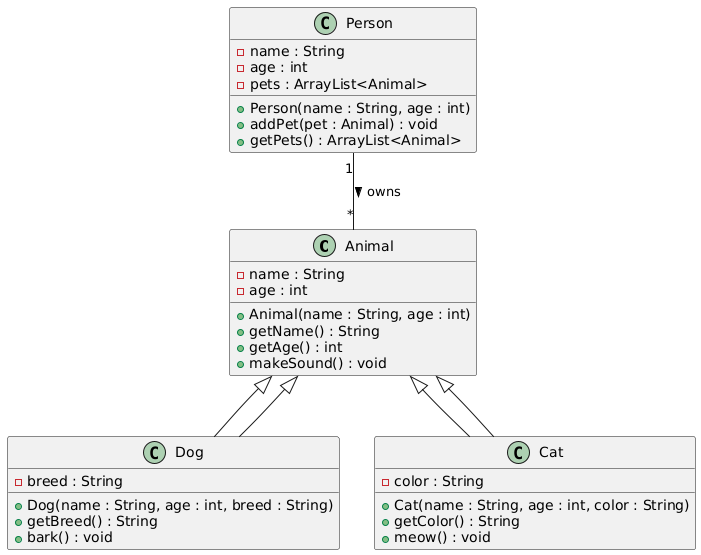
Scroll-based animations using `gsap.fromTo` and `ScrollTrigger`.

5. Conclusion:

The `Landing` component is a well-structured and highly interactive React component. Its use of animation libraries creates an engaging user experience. The modular design using custom components enhances maintainability and reusability. Further documentation of the custom components (`CustomCursor`, `GooeyNav`, `InteractiveText`, `SplitText`, `SpotlightCard`, `TiltedCard`) would enhance the overall understanding of the implementation. UML diagrams would further clarify the relationships between these components.

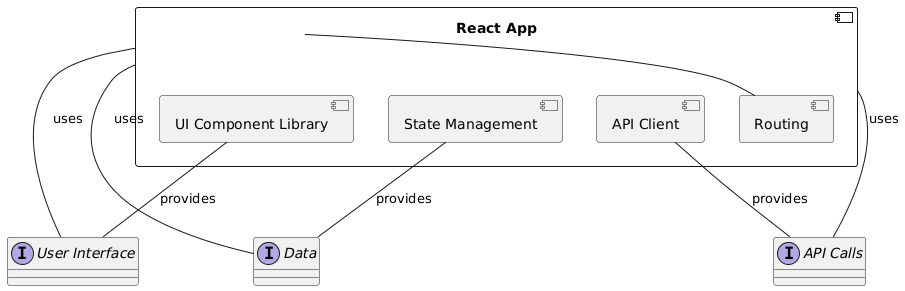
## **Class\_Diagram**

\*\* Illustrates the classes, their attributes, methods, and relationships (inheritance, association, composition) within the system.



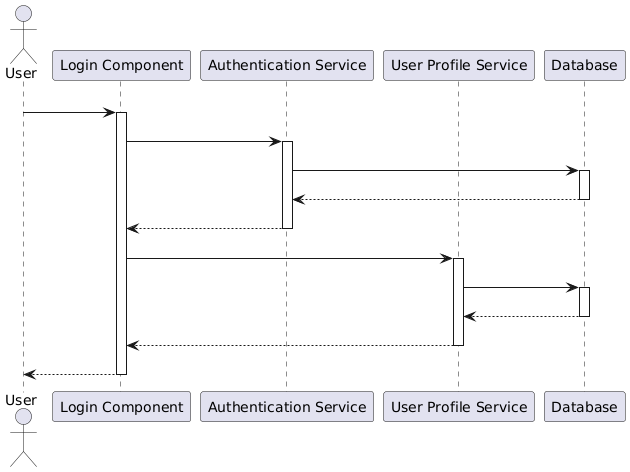
## **Component\_Diagram**

\*\* Shows the components of the React application, their dependencies, and interfaces.



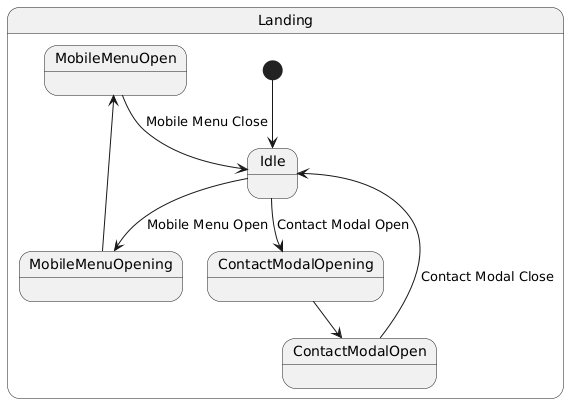
## **Sequence\_Diagram**

\*\* Depicts the interactions between objects (components, functions) over time, showing the order of message passing.



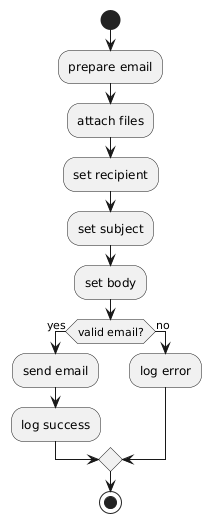
## **State\_Machine\_Diagram**

\*\* Models the different states of the `Landing` component (e.g., `mobileMenuOpen`, `isContactModalOpen`) and the transitions between them.



## **Activity\_Diagram**

\*\* Visualizes the workflow or steps involved in specific actions, such as the `sendEmail` function.



## **Package\_Diagram**

\*\* Organizes the codebase into logical packages (e.g., Effects, components) and their interdependencies.

