**React Test**

**Time Period:** 6 Days

**Test Completion Bonus:**  2500 - 5000 INR/- ( based on test result and terms of joining)

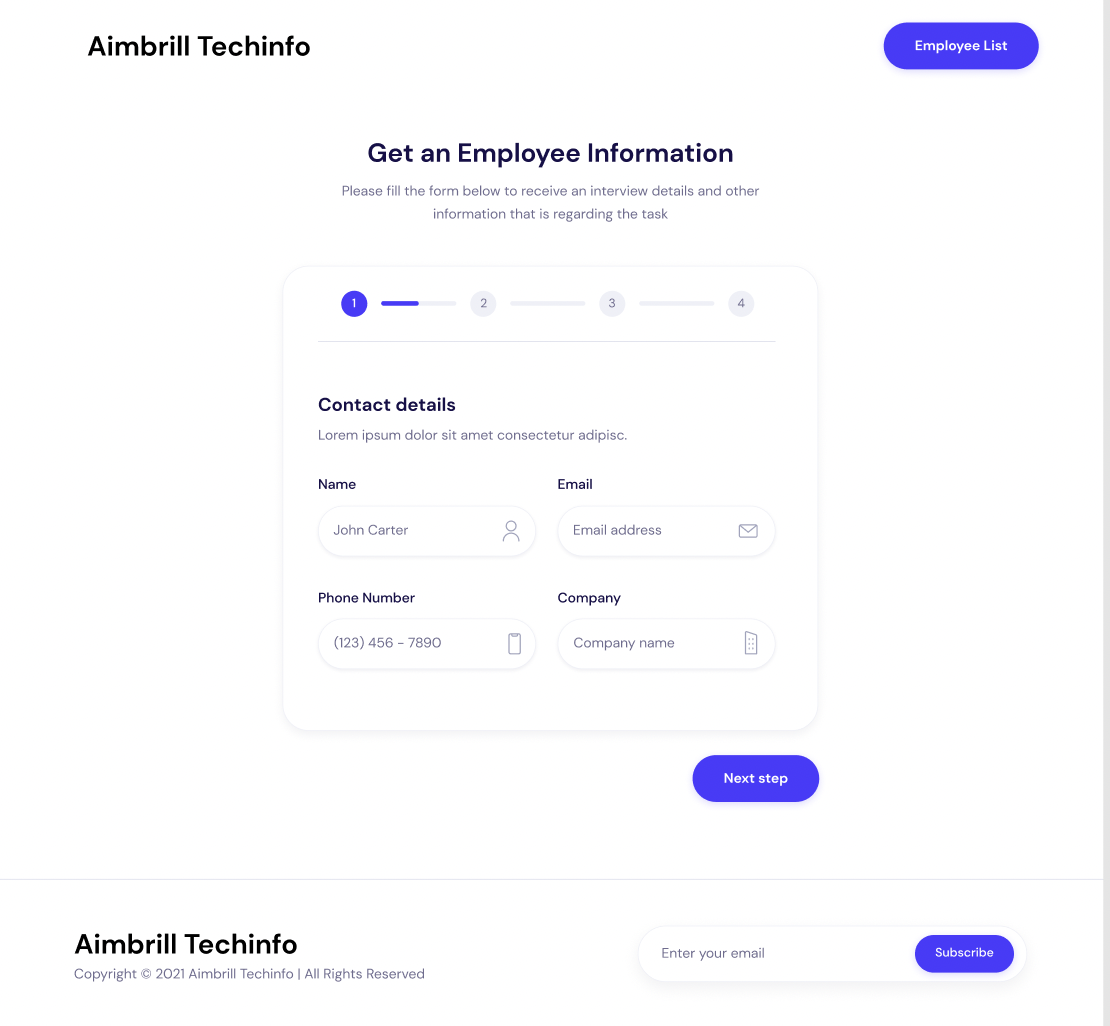
**Joining Bonus:**  Based on Performance and Market Standard (10000 - 20000 INR based on test result and terms)

**Submission:**  push code on github. If you are aware about the deployment then deploy it and send a URL and write a mail on [career@aimbrill.com](mailto:career@aimbrill.com) Add CC: [hr@aimbrill.com](mailto:hr@aimbrill.com) and [dharmik@aimbrill.com](mailto:dharmik@aimbrill.com) and share a github repo and final task screen recording.

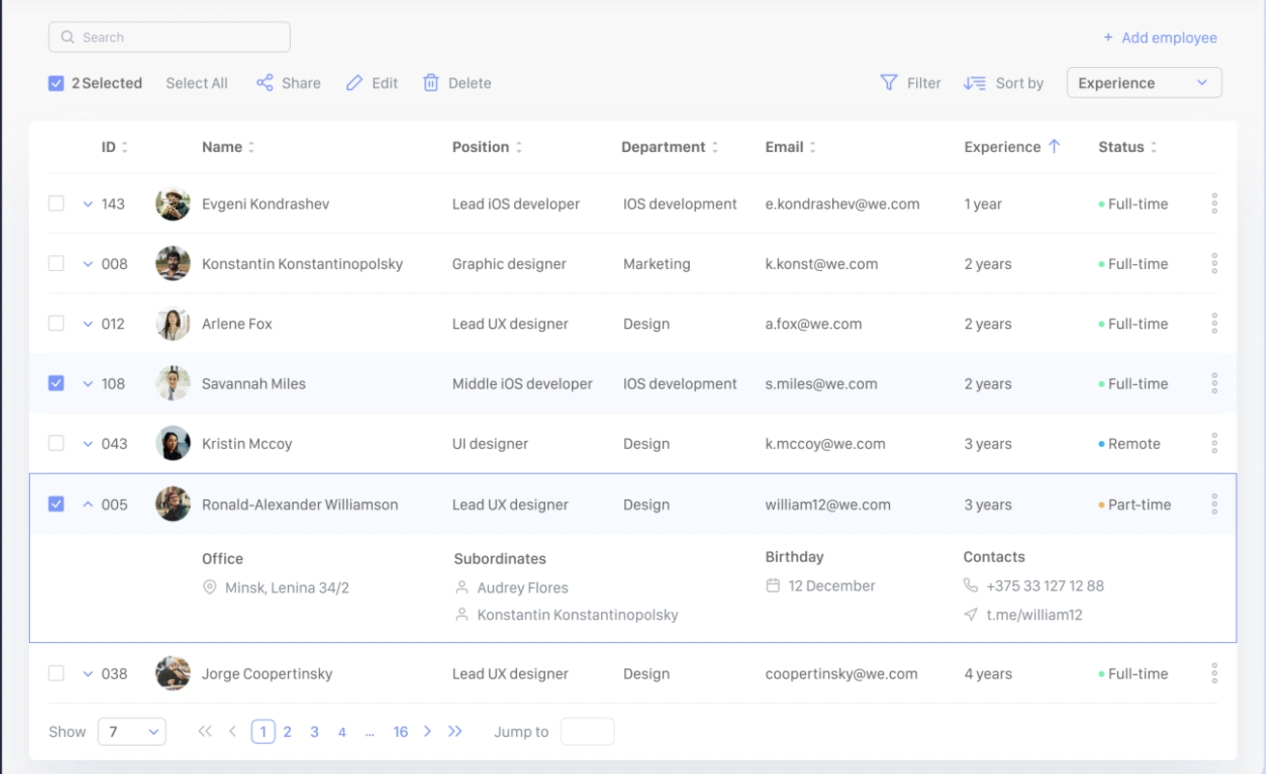
**Requirement:**

Build a multi-step form wizard using React.js. The form wizard should guide users through a series of steps to collect and submit information and store it in redux toolkit and display it in the record list table.

**Reference Images for Multi Step form:**



**Employee List Design Reference:**



-> No need for search, pagination, or selection. but if you can do it. it will consider as a extra points and bonus as well

**Specifications:**

1. The form wizard should consist of multiple steps, with each step representing a different section of the form.

2. Display a progress bar or indicator to show the user's current progress in completing the form.

3. Implement navigation buttons (e.g., "Next" and "Previous") to allow users to move forward and backward between steps.

4. Perform form validation at each step to ensure that all required fields are filled out correctly before proceeding to the next step.

5. Include various types of form fields such as text inputs, dropdowns, checkboxes, and radio buttons.

6. Implement conditional rendering to show or hide certain form fields based on user selections or input.

7. Provide real-time feedback to users when they interact with form fields (e.g., displaying error messages or validation hints).

8. Allow users to review and edit their entered information before submitting the form.

9. Upon submission, display a summary of the entered data for the user to review before final submission.

10. Implement proper form submission handling, such as making an API call to store the form data or displaying a success message.

**Additional Logics: ( Bonus Points )**

1. Implement logic to dynamically generate form steps and fields based on a configuration object. This allows for easy customization and scalability of the form wizard.

2. Implement conditional logic to enable or disable navigation buttons based on the current step and the validity of the entered data.

3. Implement custom validation logic for specific fields, such as validating email addresses, phone numbers, or complex input patterns.

4. Add logic to persist the entered data when users navigate between steps.

5. Implement logic to dynamically calculate or suggest values for certain fields based on user input or external data sources. example: ask input from user about the form fields and stepper form values

6. Add logic to handle complex dependencies between form fields, such as showing additional fields or sections based on specific selections.

**Another Bonus (optional):**

- Implement data persistence to allow users to save their progress and resume the form later.

- Add animations and transitions to enhance the user experience during form navigation and interactions.

- Implement multi-language support for the form wizard.

- Implement client-side form validation using a library like Formik or Yup.

- Add logic to display progress indicators or completion percentages for each step based on the user's completion status.

- Implement logic to handle multi-page form flows, where the user can jump to a specific step or navigate back to a previous step.

**Evaluation Criteria:**

1. Code organization and component structure.

2. Proper usage of React.js concepts like state, props, and lifecycle methods.

3. Implementation of form validation and handling of user input.

4. User interface design, including layout, styling, and feedback mechanisms.

5. Implementation of form navigation and progress tracking.

6. Code readability, maintainability, and best practices.