**React Forms Exercises with Router Integration**

Here are some exercises focused on building **React forms** with **React Router** integration. These exercises will help you practice creating forms, handling user inputs, managing form state, and navigating between different components using routing.

**Exercise 1: Simple Contact Form**

**Objective**: Create a contact form that captures user input and displays a confirmation message upon submission.

**Requirements:**

1. Create a form with the following fields:
   * Name
   * Email
   * Message
2. Validate that **Name** and **Email** are required.
3. After submission, redirect to a **Thank You** page using React Router.

**Routes:**

1. /contact - Displays the Contact Form.
2. /thank-you - Displays a Thank You message after form submission.

**Exercise 2: User Registration Form with Profile View**

**Objective**: Create a user registration form that captures user details and displays the submitted information on a profile page.

**Requirements:**

1. Create a registration form with the following fields:
   * Username
   * Password
   * Email
2. Validate that all fields are required.
3. Upon successful registration, redirect the user to a profile page showing the submitted details.

**Routes:**

1. /register - Displays the Registration Form.
2. /profile - Displays the Profile page with user information.

**Project Structure:**

/src

/components

RegistrationForm.js

ProfilePage.js

/css

RegistrationForm.css

ProfilePage.css

App.js

index.js

**Exercise 3: Multi-Step Job Application Form**

**Objective**: Create a **multi-step job application form** where users fill in their personal information, education details, and work experience. After completing all steps, the user should see a summary of the application and confirm submission.

**Requirements:**

1. **Step 1: Personal Information**
   * Full Name (Required)
   * Email (Required)
   * Phone Number (Required)
2. **Step 2: Education**
   * Degree (Required)
   * University (Required)
   * Graduation Year (Required, must be a valid year)
3. **Step 3: Work Experience**
   * Company Name
   * Position
   * Years of Experience (Must be a positive number)
4. **Step 4: Review & Submit**
   * Display all the entered information.
   * Allow users to edit any section before final submission.
   * After submission, redirect to a **Thank You** page.

**Routes:**

1. /apply/personal - Displays the Personal Information form.
2. /apply/education - Displays the Education form.
3. /apply/experience - Displays the Work Experience form.
4. /apply/review - Displays the Review & Submit page.
5. /apply/thank-you - Displays the Thank You message after form submission.

**Exercise 4: Multi-Step Registration Form with Confirmation**

**Objective:**

Create a multi-step user registration form that captures details such as personal information, contact details, and account settings. After submission, display a confirmation page showing the submitted data.

**Requirements:**

1. **Step 1:** Capture the user's personal information (name, age, gender).
2. **Step 2:** Capture contact information (email, phone).
3. **Step 3:** Account settings (username, password).
4. After all steps are completed, display a confirmation page showing the submitted information.
5. Use React Router to handle navigation between the steps.
6. Display appropriate validation messages for each step.

**Project Structure:**

/src

/components

Step1.js

Step2.js

Step3.js

Confirmation.js

ProgressBar.js

/css

Step1.css

Step2.css

Step3.css

Confirmation.css

ProgressBar.css

App.js

index.js

**Exercise 5: Job Application Form**

**Objective:**

Create a Job Application Form that captures user information and displays the submitted details on a confirmation page.

**Requirements:**

1. Create a form with the following fields:
   * Full Name
   * Email
   * Phone Number
   * Position Applied For
   * Resume (file upload)
2. Validate that Full Name, Email, and Position Applied For are required.
3. Upon successful submission, redirect the user to a confirmation page that displays the submitted information.

**Routes:**

1. /job-application - Displays the Job Application Form.
2. /application-confirmation - Displays a confirmation page with the submitted application details.

**Components:**

1. **JobApplicationForm.js** - The form where users enter their job application details.
2. **ConfirmationPage.js** - Displays a confirmation page with the submitted data.
3. **ProgressBar.js** - A simple progress bar that shows the current step in the application process.
4. **Navbar.js** - A simple navigation bar for accessing the form and confirmation page.
5. **Footer.js** - A footer component with basic copyright information.

**Exercise 6: Task Management Application**

**Objective:**

Create a Task Management Application that allows users to add tasks, mark them as completed, and view all their tasks in a list.

**Requirements:**

1. Create a form with the following fields:
   * Task Name
   * Due Date
   * Priority Level (Low, Medium, High)
2. Display a list of tasks below the form with the ability to:
   * Mark tasks as completed.
   * Delete tasks.
3. Task list should be filtered by priority levels (e.g., Low, Medium, High).

**Routes:**

1. /task-manager - Displays the Task Manager form and task list.
2. /task-detail - Displays the details of a selected task.

**Components:**

1. **TaskForm.js** - A form for adding new tasks.
2. **TaskList.js** - Displays the list of tasks and handles task actions like marking as completed and deleting.
3. **TaskItem.js** - Represents each individual task in the list.
4. **TaskDetail.js** - Shows the details of a selected task.
5. **Navbar.js** - A navigation bar for navigating between pages.

**Exercise 7:**

Create a simple Blog application with multiple components to display blog posts, allow adding new posts, and view post details.

**Features:**

1. **HomePage** - Display a list of blog posts.
2. **PostPage** - Display the details of a single post.
3. **AddPostPage** - A page with a form to add a new post.
4. **Navbar** - A navigation bar for the application.
5. **PostCard** - A component that displays a blog post preview in the list.

**Exercise 8**

**Objective:**

Create a To-Do List application with multiple components such as adding tasks, viewing tasks, and marking tasks as completed. This will involve routing, forms, and dynamic content rendering.

**Features:**

1. **App**: Main application file with routing.
2. **Navbar**: A navigation bar with links to Home and About pages.
3. **HomePage**: Displays the list of tasks and allows for task creation.
4. **TaskList**: Displays a list of tasks.
5. **TaskItem**: Displays each task with options to mark as completed or delete.
6. **AddTaskForm**: A form to add a new task.
7. **TaskFilter**: Allows users to filter tasks based on their status (all, completed, or pending).
8. **TaskCounter**: Displays the count of tasks (all, completed, and pending).
9. **AboutPage**: Displays information about the app.
10. **Footer**: Displays a footer with basic information.

**Exercise 9:**

**Objective:**

Create a Blog Application that allows users to view, add, and edit posts.

**Features:**

1. **App**: Main application with routing.
2. **Navbar**: A navigation bar with links to Home and About pages.
3. **HomePage**: Displays a list of blog posts.
4. **PostItem**: Displays each blog post in the list.
5. **PostDetails**: Displays the full content of a single post when clicked.
6. **AddPostForm**: Form to add a new blog post.
7. **EditPostForm**: Form to edit an existing blog post.
8. **PostCounter**: Displays the total number of posts.
9. **AboutPage**: Displays information about the blog.
10. **Footer**: Displays basic footer information.

**Exercise 10:**

**Objective:**

Create a **Task Manager Application** where users can manage tasks by adding, editing, and deleting them.

**Features:**

1. **App**: Main application with routing.
2. **Navbar**: A navigation bar with links to Home, Add Task, and About pages.
3. **HomePage**: Displays a list of tasks.
4. **TaskItem**: Displays each task in the list with options to edit and delete.
5. **AddTaskForm**: Form to add a new task.
6. **EditTaskForm**: Form to edit an existing task.
7. **TaskCounter**: Displays the total number of tasks.
8. **AboutPage**: Displays information about the Task Manager.
9. **Footer**: Displays footer information.
10. **TaskList**: A container to display the list of tasks.