Dynamic Tabs Component - Project Documentation

## Overview

The **Dynamic Tabs Component** is a React-based UI element that allows users to **add, remove, and switch between tabs dynamically**. Each tab displays an image fetched from an API. The component ensures a smooth user experience by always maintaining at least one active tab.

**Features**

✅ **Add New Tabs** – Users can dynamically add tabs.

✅ **Switch Between Tabs** – Clicking a tab displays its content.

✅ **Remove Tabs** – Users can remove tabs, ensuring at least one tab remains.

✅ **Active Tab Highlighting** – The UI clearly indicates the active tab.

✅ **Image Fetching** – Tabs display a random image from an API.

✅ **Smooth Transitions** – CSS animations for a clean user experience.

## Technology Stack

* **React (Vite)** – Modern frontend framework for a fast UI.
* **React Hooks** (useState, useEffect) – Manages tabs and state.
* **API Integration** – Fetches random images for tab content.
* **CSS Styling** – Provides a visually appealing interface.

## Installation & Setup

1. **Clone the Repository**
2. **Install Dependencies**

npm install

1. **Run the Application**

npm run dev

1. **Open in Browser** Visit http://localhost:5173/ (or the provided Vite dev server link).

## Project Structure

/src

│── components

│ ├── DynamicTabs.js # Main Tabs Component

│── App.js # Main Application File

│── styles.css # Styling for Tabs Component

## Functionality

**State Management**

* tabs – Stores an array of tab objects (id, title, imageUrl).
* activeTab – Stores the **ID of the currently active tab**.
* fetchRandomImage() – Fetches a random image from an external API.

**User Interactions**

* **Clicking "Add Tab"**: Creates a new tab with an API-fetched image.
* **Clicking a Tab**: Switches the active tab.
* **Clicking "×" on a Tab**: Removes the tab (ensuring one remains).

## API Usage

The component fetches images from the api-ninjas.com API:

GET https://api.api-ninjas.com/v1/randomimage?category=nature

Headers:

{

"X-Api-Key": "YOUR\_API\_KEY",

"Accept": "image/jpg"

}

**CSS Styling**

The component uses **flexbox for layout**, **smooth transitions**, and a **glassmorphism effect** for aesthetics.

* **Active Tab Styling**: The currently selected tab has a darker background.
* **Tab Buttons**: Styled for better visibility and interaction.
* **Tab Content Area**: Displays images responsively.

## Test Cases

**✅ Test Case 1: Add a New Tab**

**Steps:** Click "Add Tab" button. **Expected Result:** A new tab appears with a fetched image.

**✅ Test Case 2: Switch Tabs**

**Steps:** Click on an existing tab. **Expected Result:** The tab content updates with the selected tab's image.

**✅ Test Case 3: Remove a Tab**

**Steps:** Click the close (×) button on a tab. **Expected Result:** The tab disappears, and another tab becomes active.

## Constraints & Best Practices

⚡ **Ensure at least one tab remains.**

⚡ **Use useState for dynamic updates.**

⚡ **Ensure API calls handle errors gracefully.**

⚡ **Optimize UI responsiveness with flexbox/grid layouts.**

## Future Enhancements

✨ **Fetch images from a user-selected category**.

✨ **Persist tabs across page reloads using local storage**.

✨ **Add animations for smoother transitions**.

✨ **Allow users to rename tabs dynamically**.

**Conclusion**

The **Dynamic Tabs Component** is an efficient and visually appealing solution for managing multiple tabs dynamically. It provides a seamless user experience with real-time API integration and clean UI interactions.