Task 05: HTML and CSS (Pure)

Objective:

- Design a webpage that mimics the shared theme using only HTML and CSS.
- Replace images and text to make it unique.
- Skip transitions and animations.

Duration: 1 week

Instructions:

1. **Structure**: Create a similar layout as the shared theme using HTML.
2. **Styling**: Use CSS to style the webpage to match the theme.
3. **Customization**: Replace the placeholder images and text with original content to make it unique.

Task 06: Bootstrap

Objective:

- Redesign the webpage using Bootstrap to simplify the layout and styling process.

Duration: 3 days

Instructions:

1. **Include Bootstrap**: Add the Bootstrap CSS and JavaScript files to your project.
2. **Layout**: Utilize Bootstrap's grid system to structure the layout.
3. **Components**: Use Bootstrap's pre-built components (e.g., Navbar, Cards, Buttons) to replicate the design.
4. **Customization**: Ensure the webpage maintains its unique content (images and text) from Task 05.

Task 07: Tailwind CSS

Objective:

- Redesign the webpage using Tailwind CSS for a utility-first CSS approach.

Duration: 3 days

Instructions:

1. **Include Tailwind CSS**: Add Tailwind CSS to your project.
2. **Layout and Styling**: Use Tailwind's utility classes to style the webpage.
3. **Customization**: Maintain the unique content (images and text) from Task 05.

Task 08: Next.js Conversion

Objective:

- Convert the webpage into a Next.js application for server-side rendering and other advanced features.

Time:

- 1 week

Instructions:

1. **Setup Next.js**: Create a new Next.js project.
2. **Page Components**: Convert each section of the HTML into React components.
3. **Routing**: Use Next.js routing to handle different pages if necessary.
4. **Styling**: Apply the styles using CSS, or you can reuse the Tailwind CSS setup from Task 07.
5. **Customization**: Ensure all unique content is preserved.

General Tips:

- **Planning**: Before starting each task, plan the layout and structure to ensure efficient coding.
- **Consistency**: Keep the styling consistent throughout all tasks.
- **Version Control**: Use version control (e.g., Git) to keep track of changes and progress.
- **Testing**: Test the webpage across different browsers to ensure compatibility.

This step-by-step approach will help you effectively transition from basic HTML/CSS to using frameworks and libraries, ending with a modern Next.js application. Let me know if you need more details or specific examples for each task!