TradeGospel

Assignment: Infinite Scroll Image Blog with Background Notifications

Role: Frontend Developer (TradeGospel)
Time Limit: 24 hours

Introduction

You are required to build a clean, responsive image-blog UI where users can upload multiple images and view them in a colossal/masonry-style feed with infinite scrolling, similar to modern visual platforms. This assignment evaluates your skills in UI/UX, state management, component architecture, and smooth dynamic rendering.

As a bonus, you will also implement a background task listener that receives simulated server notifications and displays them in the UI.
You may refer to YouTube or documentation for learning, but you must not use AI tools (e.g., ChatGPT, GitHub Copilot) to generate the code.

Objective

Build a single-page web app where:

1. Users can upload multiple images.
2. Images appear in a colossal/masonry-style blog layout.
3. The feed supports infinite scrolling.
4. Bonus: The app handles a background task that receives a notification from a mock server and displays it in the UI.

Core Requirements

1. Tech Stack

- Use Next.js (React.js also allowed).
- Use CSS / SCSS / Tailwind / plain CSS.
- Do not use heavy UI libraries (Material UI, Ant Design, etc.).

2. Image Upload

- Provide a section where the user can upload multiple images.
- Show previews before the final “Publish / Post.”
- On clicking Publish, images should be added to the main feed.

3. Blog-style Image Feed

Display the uploaded images in a Pinterest-style masonry layout.

Each post should include:

- The image
- A dummy title (e.g., “Post #1”) or caption
- A timestamp

Layout must be fully responsive for mobile and desktop.

4. Infinite Scroll

Implement infinite scroll such that:

- A fixed number of posts load initially.
- More posts load automatically as the user scrolls.

You may simulate additional posts by duplicating existing ones or preloading mock data.

5. Data Handling

You may store data:

- In component state, and/or
- In localStorage (optional but preferred)

No backend is required, but write code cleanly as if the project will scale.

Bonus Task: Background Job & Notifications

1. Simulated Background Task

Create a simulation where the “server” sends notifications in the background using:

- `setTimeout()`
- `setInterval()`
- Or any async function returning a notification after a delay

2. Notification Handling

When a notification is received:

- Show a non-blocking toast, popup, or badge counter
- OR update a notification icon

The UI must remain fully responsive and smooth.

3. Background Behavior

- The background process starts when the app loads.
- Clean up intervals or listeners when components unmount.

- Simulate 2–3 random notifications like:
  “New image processing complete.”

Evaluation Criteria

1. Code Quality

- Structure & organization
- Readability
- Correct use of React hooks

2. UI / UX Quality

- Clean design
- Smooth infinite scroll
- Beautiful masonry layout
- Responsive behavior

3. Performance

- Smooth scrolling without lag
- Efficient rendering
- Correct cleanup of intervals/timeouts

4. Understanding of Asynchronous Logic

- How background tasks are simulated
- How notifications are handled

5. Ownership & Explanation

- You may learn from YouTube/docs
- No AI-generated code is allowed
- Be prepared to explain:
    - Project structure
    - Infinite scroll logic
    - Background notification flow

Deliverables

1. GitHub Repository Link

README must include:

- Installation steps
- Running instructions

- Any assumptions made

2. Short Explanation Note

(5–10 lines) including:

- Your approach
- Tools used
- Improvements you would make with more time