Image Text Composer - Adomate Coding Assignment

Objective
The goal of this coding challenge is to design and implement a **desktop-only, single-page** image editing tool that enables users to upload a PNG image and overlay it with fully customizable text. The app should feel polished, intuitive, and performant, demonstrating your ability to combine **front-end engineering skills, clean architecture, and thoughtful UX design** into a cohesive, production-ready experience.

At its core, the app should allow a user to:

1. Upload a PNG image, with the editor canvas automatically matching the image's aspect ratio.
2. Add one or more **text layers** on top of the image, each of which can be styled, moved, resized, and rotated independently.
3. Edit key text properties such as font family, size, weight, color, opacity, alignment, and multi-line text content.
4. Reorder layers in the stacking order, giving control over what appears in front or behind.
5. Export the final design as a PNG image with the text rendered on top, preserving the original image dimensions.

While the functional requirements are straightforward, this challenge is as much about **how** you implement the solution as it is about meeting the basic feature list. You should aim for a **snappy, predictable editing experience** where interactions such as dragging, resizing, and rotating feel smooth and precise.

The app should be built with **Next.js and TypeScript**. You may use established, open-source libraries (e.g., Fabric.js, Konva) to handle canvas interactions, but be prepared to explain your choices in the README. No paid APIs are allowed. All Google Fonts should be available for selection (see Google Fonts API).

The **Bonus Points** section in this brief contains suggestions for how you can show off your skills and further improve the solution. This is where you can shine and demonstrate your technical creativity, going beyond the initial assignment if you still have time and energy. The bullet points in that section are simply ideas, feel free to propose your own improvements.

This is a **7-day challenge** starting at **8:00 a.m. CET** and finishing on **Monday, 18th of August at 8:00 a.m. (CET)**. All solutions must be submitted by the deadline along with the deliverables outlined in this brief below to lucas@adomate.com with the subject line:
Image Text Composer – [Your Full Name]

Tech & Constraints

- **Framework:** Next.js + TypeScript (desktop only app)
- **Rendering:** HTML5 Canvas (you may use Fabric.js, Konva, or other open-source libraries)
- **Fonts:** All **Google Fonts** available via Google Fonts API
- **Assets:** Image upload **PNG only**
- **Export: PNG** with text overlay (no scaling, keep original image dimensions)
- **Hosting:** Public demo on Vercel (no login)
- **No paid APIs**

Core Requirements

- Upload a background **PNG**; the canvas should match the uploaded image's aspect ratio and maintain original dimensions in export
- Possibility to add **multiple text layers** with editing capabilities for:
    - Font family (all Google Fonts available)
    - Font size
    - Font weight
    - Color (solid)
    - Opacity
    - Alignment (left, center, right)
    - **Multi-line editing** (support for multiple lines within the same text box)
- Transform text layers: drag, resize with handles, rotate
- **Layer management:** reorder layers
- **Canvas UX:** snap-to center (vertical & horizontal), nudge with arrow keys
- **Undo/Redo:** at least 20 steps with a visible history indicator
- **Autosave:** automatically save the current design to the browser's localStorage so it is restored after page refresh or reopening
- **Reset button:** clear the saved design from localStorage and return the editor to a blank state

Export

- Export the final design as a **PNG** with the text overlay and original image dimensions

(Optional) Bonus Points

Some ideas, but feel free to propose your own improvements:

- Custom font upload (TTF/OTF/WOFF)
- Multi-select with group transforms
- Smart spacing hints between selected layers
- Ability to edit line-height, letter-spacing
- Lock / unlock layers, duplicate layers

- Text shadow (customizable color, blur, and offset)
- Warp or curved text along a path

Non-Goals

The following points will not be considered:

- Mobile/touch UI
- Collaboration or multi-user editing
- Non-PNG import/export formats

Deliverables

Your submission must include:
1. Public GitHub Repository, including:

- All source code
- `README.md` with:
    - Setup and run instructions
    - Concise description of the architecture
    - Technology choices and trade-offs
    - Which bonus points are implemented
    - Known limitations

2. A working, deployed version of the app (e.g., on Vercel, Netlify) that we can access and test via a public URL.
3. (*Optional)* A **<= 3-minute** video walkthrough of the app

Evaluation Criteria

We'll assess your submission based on:

- **Functionality:** Do the core features work as described? Is it robust?
- **UX Design & Performance:** Is it a smooth, intuitive, responsive editor experience?
- **Code Quality:** Clean, modular, commented codebase with clear state management?
- **Creativity:** Did you go beyond the minimum in thoughtful or impressive ways?