Role: Full Stack Developer (MERN)
Timeline: 72 Hours (3 Days)
Output: GitHub Repository + Deployment Link (Vercel/Render)

🚀 The Context

At BoloForms, our core promise is reliability. When a user places a signature field on a legal contract, it **must** appear in that exact location on the final PDF.

The Problem: Web browsers (Frontend) and PDF Files (Backend) speak different languages.

- Browsers use **CSS Pixels** relative to the Top-Left.
- PDFs use **Points (72 DPI)** relative to the Bottom-Left.
- Screens vary in size (Mobile vs. Desktop), but the PDF is static.

Your Mission: Build a prototype "Signature Injection Engine" that bridges this gap flawlessly.

🛠️ Functional Requirements

1. The Frontend (The "Responsive" Editor)

[illegible]

Tech: React.js, PDF.js (or react-pdf)

- **PDF Viewer:** Render a sample PDF (use any standard A4 PDF).
- **Drag & Drop:** Allow the user to drag the given fields onto the PDF.
    - Text Box, Signature field, image box, Date selector and Radio should be there for the user to drag & drop.
- **Resize:** Allow the user to resize this box.
- **The "Responsiveness" Constraint:**
    - If I place a box on the PDF while on a large Desktop screen, and then switch to "Mobile View" (Chrome DevTools), the box must **stay visually anchored** to the correct paragraph on the PDF.
- The user should be able to sign this document in a viewer and use the dropped fields.

2. The Backend (The "Burn-In" Engine)

Tech: Node.js, Express, `pdf-lib` (or similar), MongoDB

- **Endpoint:** `POST /sign-pdf`
- **Payload:** Should receive the PDF ID, the Signature Image (Base64), and the Coordinate Object from the frontend.
- **Processing:**
    - The server must overlay the signature image onto the PDF.

    - **The "Aspect Ratio" Constraint:** If the user draws a *square* box but uploads a *wide* signature, the image must be **contained** within the box (centered) without stretching/distorting the image.
- **Output:** Return a URL to the newly signed PDF.

3. The Security Layer (Audit Trail)

- Before signing: Calculate the **SHA-256 Hash** of the original PDF.
- After signing: Calculate the hash of the final PDF.
- Store these hashes in MongoDB to prove the document history.

📝 Deliverables

1. **Source Code:** A public GitHub repository.
2. **Live Demo:** Deployed on Vercel/Netlify (Frontend) and Render/Railway (Backend).
3. **Video Walkthrough (Important):** A 3-minute Loom/Screen recording where you:
    - Demo the feature.
    - **Show us the Code:** specifically the function where you calculate the coordinates. **Explain your math.**

NOTES

1. Please don't use AI code we'll know 🙂
2. In our company we do use AI everywhere but to check your capabilities & logical reasoning we refrain from using tools in this process.
3. Make any assumptions you want to take.
4. Check out https://signature.boloforms.com/ for understanding the product well.

Best Of Luck