

InventisLabs Project Report

1. Idea of the Website

InventisLabs is built with a singular, powerful mission: **to save lives and protect critical infrastructure through technology.**

The website is not just a corporate portfolio; it is a digital platform for high-stakes safety solutions. It serves as the primary gateway for Governments and Large Industries to discover and implement:

- **EQ-Alert:** An Earthquake Early Warning system that gives people seconds to react before a disaster hits.
- **Structural Monitoring:** Smart sensors that watch over bridges and buildings to prevent collapses.
- **IoT Engineering:** A specialized lab designing the hardware that makes this all possible.

The "Idea" is to present complex, life-saving technology in a way that feels trustworthy, modern, and accessible to decision-makers.

2. Context of the Website

Who is this for?

The website targets a very specific, high-level audience:

- **Government Officials:** Who need to install city-wide alert systems.
- **Factory Owners:** Who need to automatically shut down dangerous machines during an earthquake.
- **Infrastructure Managers:** Who need to know if a bridge is safe to drive on.

The "Vibe":

Because the product is about safety and advanced engineering, the website's context is "**Premium Reliability.**" It uses a clean, dark-mode aesthetic with smooth animations to show that the company is cutting-edge and professional. It's designed to build confidence that "these people know what they are doing."

3. Admin Routes & Access

To keep the website dynamic, there is a hidden "Control Center" for the InventisLabs team. This allows them to post new jobs and read messages from clients without needing a programmer.

Admin Credentials (Default)

- **Username:** `admin`
- **Password:** `changeme`

(Note: These are standard defaults for the development version.)

Key Admin Areas

The admin panel is divided into simple sections:

- **Login Route (`/api/admin/login`):** The secure door to enter the control panel.
 - **Message Center:** Allows admins to read and reply to inquiries from the "Contact Us" page.
 - **Job Portal:** Allows admins to post, edit, or delete job openings on the "Careers" page.
-

4. Advantage of the Route Structure

The website is organized into specific "Routes" (pages/paths) for strategic reasons. Here is the advantage of this structure:

1. Safety & Security (Admin Routes):

- Advantage: By keeping the Admin tools on a completely separate route (`/api/admin`), we ensure that regular visitors never accidentally stumble upon sensitive controls. It acts like a "Staff Only" door in a building.

2. Clarity for Customers (Solution Routes):

- Advantage: Each major product (EQ-Alert, Structural Monitoring) has its own dedicated section. This prevents information overload. A government official looking for "Earthquake Alerts" goes straight to that route without being distracted by "Circuit Board Design."

3. Smooth Experience (User Flow):

- Advantage: The routes are designed so users can jump straight to what they need—"Home" for an overview, "Solutions" for details, and "Contact" to buy—without getting lost.
-

5. Technology Stack (Simplified)

This platform is built using a modern "MERN" stack, chosen for speed and flexibility.

- **The Interface (Frontend):**

- **React.js:** Used to build the visible website. It makes the site feel like a smooth mobile app rather than a clunky old webpage.
- **Tailwind CSS:** Handles the styling, giving us that sleek, dark, modern look effortlessly.
- **Framer Motion:** Powering the smooth animations that make the tech feel "alive."

- **The Engine (Backend):**

- **Node.js & Express:** The "brain" of the website that handles visitor requests and sends back the right information.
 - **MongoDB:** The "memory" where we store job postings and client messages.
 - **Nodemailer:** The "postman" that automatically sends emails when someone contacts us.
-

6. Code Overview

Here is a look at the code, explained simply.

A. The "Brain" (Server Setup)

This code sets up the server and tells it to listen for visitors. It also sets up the "Staff Only" door (Admin routes).

```
// server.js
const express = require("express");
const app = express();

// 1. Basic Security: Put on a "helmet" to protect against basic web attacks
app.use(require("helmet")());

// 2. The Rules: Define who can talk to the server (CORS)
app.use(require("cors")());

// 3. The Routes: Directions to different parts of the application
// If someone goes to /api/admin, send them to the Admin tools
app.use("/api/admin", require("./routes/adminRoutes"));
// If someone goes to /api/contact, handle their message
app.use("/api/contact", require("./routes/contactRoutes"));
```

```
// 4. Start the Engine: Turn the server on
app.listen(5000, () => console.log("Server is running..."));
```

B. The "Control Panel" (Admin Logic)

This is the logic used when an Admin wants to create a new Job posting. It focuses on simplicity and data safety.

```
// adminController.js
exports.createJob = async (req, res) => {
    // 1. Get the details: Read what the admin typed (Title, Salary, etc.
    const { title, location, salary } = req.body;

    // 2. Create the file: Make a new Job entry
    const newJob = new Job({
        title,
        location,
        salary,
        status: 'active' // automatically set it to active
    });

    // 3. Save it: Store it in our database "memory"
    await newJob.save();

    // 4. Confirm: Tell the admin it worked
    res.json({ success: true, message: "Job Posted!" });
};
```