Intern Assignment (10 Days)

Project: ProjectPulse – Client Feedback & Project Health Tracker

Internship Task – Full Stack Web Application

---

1. Overview

This assignment is designed to evaluate an intern's ability to build a **real-world internal system** commonly used by IT and software companies to track project progress, client satisfaction, and delivery risks.

Rather than building a standard company website, this project focuses on **project health monitoring**, **structured feedback**, and **decision-support dashboards**, reflecting real operational challenges in professional software teams.

---

2. Objective

Develop a web-based system where:

- **Clients** provide structured feedback on ongoing projects
- **Employees** submit weekly progress updates and identify risks
- **Admins** monitor overall project health and intervene early

The system must automatically calculate a **Project Health Score** based on collected data.

---

3. Tech Stack (Mandatory)

Applicants must use the following stack:

Frontend

- **Next.js**
- **Tailwind CSS**

Backend (Choose One)

- **Option A:** Express.js (REST API)
- **Option B:** Next.js Backend (API Routes / App Router APIs)

Database

- **MongoDB**

Authentication & Security

- JWT-based authentication
- Role-based authorization (Admin / Employee / Client)

Applicants must clearly mention in the README which backend option they chose.

---

4. User Roles

Admin

- Full access to the system
- Creates and manages projects
- Assigns clients and employees
- Monitors project health and risks

Employee

- Assigned to one or more projects
- Submits weekly progress check-ins
- Reports risks and blockers

Client

- Views only their assigned projects
- Submits weekly feedback
- Flags issues when dissatisfied

---

5. Core Features

5.1 Authentication & Access Control

- Login system (no public registration)
- Users created via admin seed script
- Protected routes on frontend and backend
- Strict role-based access enforcement

5.2 Project Management

Admins must be able to:

- Create projects
- Assign one client and multiple employees
- Set project start and end dates

Each project includes:

- Name and description
- Timeline
- Status: **On Track / At Risk / Critical / Completed**
- Automatically calculated **Health Score**

5.3 Weekly Check-In System (Key Feature)

Employee Weekly Check-In

Employees submit **one check-in per week per project**, including:

- Progress summary
- Blockers or challenges
- Confidence level (1–5)
- Estimated completion percentage

Client Weekly Feedback

Clients submit:

- Satisfaction rating (1–5)
- Communication clarity rating (1–5)
- Optional comments
- Option to flag an issue

5.4 Project Health Score (Logic-Based Requirement)

The system must calculate a **Health Score (0–100)** automatically using:

- Recent client satisfaction ratings
- Recent employee confidence levels
- Project progress compared to timeline
- Number of flagged issues or risks

Health interpretation:

- **80–100:** On Track
- **60–79:** At Risk
- **Below 60:** Critical

Applicants must **explain their health score logic clearly in the README**.

5.5 Risk Management

Employees can:

- Create risk items with:
    - Title
    - Severity (Low / Medium / High)
    - Mitigation plan
    - Status (Open / Resolved)

Admins can:

- View all risks across projects
- Identify high-risk projects easily

5.6 Dashboards

Client Dashboard

- List of assigned projects
- Current health status
- Last feedback submission

Employee Dashboard

- Assigned projects
- Pending weekly check-ins
- Open risks count

Admin Dashboard

- Projects grouped by health status
- Projects missing recent check-ins
- High-risk projects summary

5.7 Activity Timeline

Each project must display an activity timeline showing:

- Weekly check-ins
- Client feedback submissions
- Risk updates
- Project status changes

---

6. UI & Quality Standards

- Clean, professional UI
- Fully responsive design
- Proper loading, error, and empty states
- No hard-coded data
- Clear user feedback on actions

---

7. Deployment (Required)

Applicants **must deploy the application** and provide a **live website link**.

Deployment Requirements

- Frontend: Vercel (preferred)
- Backend:
  - Express API → Render / Railway / Fly.io
  - OR Next.js backend → same Vercel project
- Database: MongoDB Atlas

The live URL must be included in the README.

---

8. Submission Requirements

1. GitHub Repository

Repository structure:

- `/frontend` (if separate frontend)
- `/backend` (if using Express)
- OR single Next.js repo (if using Next.js backend)

Include:

- `.env.example`
- Seed script (admin + demo users + sample project)

---

2. README File (Mandatory)

The README must contain:

- Project overview
- Tech stack used
- Backend choice (Express or Next.js API)
- Setup instructions
- Demo login credentials (Admin / Employee / Client)
- Explanation of Health Score logic
- **Live website URL**
- **https://forms.gle/xWh86tdyHZ4jGBwW9 (Submit Here)**

---

3. Demo Video (Mandatory)

A **5–8 minute video** explaining:

- Project overview

- Role-based login
- Weekly check-in process
- Health score behavior
- Admin dashboard insights

Accepted formats:

- Google Drive link
- YouTube (Unlisted)

---

4. Digital Project Folder

Submit a **single Google Drive folder** containing:

- Demo video
- Additional documentation (if any)
- Deployment links (if not in README)

---

9. Suggested 10-Day Timeline (31/12/2025, 11:59pm)

---

10. Evaluation Criteria

We will assess:

- Role-based access correctness

- API and database design
- Health score logic and reasoning
- Code cleanliness and structure
- UI clarity and responsiveness
- Quality of README, live deployment, and demo video

---

11. Important Notes

- Partial completion is acceptable if core features work well
- Code quality and logic are more important than extra features
- Copied or plagiarized submissions will be rejected
