Assignment 3: TrackFlow – Lightweight CRM & Operations Process Automation App

Objective

Build a basic web application that helps an organization manage:

- Sales leads lifecycle: capture → qualify → close
- Order lifecycle: receive → build → dispatch → track

This tool should allow teams to **define, track, and manage workflows** related to both CRM and internal operational processes.

Part 1: Core Features (Must Have)

🧲 Lead Management

- Add new leads (name, contact, company, product interest)
- Assign lead stage: New, Contacted, Qualified, Proposal Sent, Won, Lost
- Follow-up reminder system (date-based)
- View leads in Kanban or list view

📦 Order Management

- Link closed leads to an "Order"
- Define stages: Order Received → In Development → Ready to Dispatch → Dispatched
- Update order status manually
- Store dispatch details (courier, tracking number)

📊 Dashboard

- Total leads, open leads, conversions
- Orders in various stages
- Weekly follow-ups pending

Part 2: Bonus Features (Nice to Have)

🔔 Reminders / Notifications

- Set date-based reminders for follow-up or order updates
- Email or in-app pop-ups for upcoming actions

📁 Document Upload

- Upload related documents for leads or orders (e.g., invoices, proposal PDFs)

🤖 Automation Rules (Basic)

- Example: When a lead is marked "Won", automatically create an Order entry
- Example: If order is dispatched, send confirmation email to client

📱 Responsive Design

- Mobile-optimized view

Tools & Stack Suggestions

- **Backend**: Python (FastAPI/Django), Node.js, or Firebase
- **Frontend**: React, Vue, or plain HTML/CSS/JS
- **Database**: SQLite, PostgreSQL, Firebase, or Supabase
- **Deployment**: Vercel, Railway, Netlify, Render
- **Optional Libraries**: FullCalendar (for follow-ups), Chart.js (for dashboard)

Deliverables

1. Live Web App (hosted)
2. GitHub Repo with:
    - Clean source code
    - `README.md` (setup, features, architecture)
3. Demo Video (3–5 min)
4. Optional: Flowchart showing lead/order lifecycle

Evaluation Criteria

Criteria	Weightage
Lead & Order Tracking Flow	30%
UI/UX & Data Entry	20%
Backend Structure & DB Design	20%
Dashboard & Usability	15%
Bonus Features	15%

7-Day Execution Plan – TrackFlow CRM Web App

Day 1: Planning & Setup

Objective: Understand the app's structure, choose tech stack, and plan database schema.

- Define core features: Lead Management, Order Workflow
- Choose tech stack (e.g., React + FastAPI + PostgreSQL/Supabase)
- Create a project repo and initialize backend + frontend
- Design DB schema:
    - `Leads`: id, name, contact, stage, follow_up_date, notes
    - `Orders`: id, lead_id, status, dispatch_date, tracking_info

Deliverables:

- Tech stack selected
- Project skeleton on GitHub
- DB schema documented
- Basic README

Day 2: Lead Management - Backend APIs

Objective: Implement backend logic for lead lifecycle.

- Create models and APIs for:
    - Add new lead
    - Update lead stage
    - Get leads (filtered by stage/date)
- Enable CORS for frontend use
- Test APIs using Postman or Swagger

Deliverables:

- Working backend API for leads
- Sample data in DB
- Basic test logs for CRUD operations

Day 3: Lead Management - Frontend

Objective: Build UI for adding and viewing leads.

- Create "Add Lead" form with:
    - Name, contact, company, stage, follow-up date
- Display lead list in table or Kanban-style cards
- Add form validation and submission flow

Deliverables:

- Frontend UI for lead form
- Kanban/List view for leads
- Integration with backend APIs

Day 4: Order Workflow - Backend + Linkage

Objective: Build backend models & endpoints for order tracking.

- Create order creation, update, get APIs
- Link orders to `lead_id` (one-to-many)
- Define stages: Received, In Development, Ready to Dispatch, Dispatched
- Add test data and connect DB logic

Deliverables:

- Backend logic & API for order lifecycle
- Orders tied to leads

Day 5: Order Workflow - Frontend UI

Objective: Create UI for order entry and status updates.

- Allow selecting a lead (from dropdown or list)
- Show current status and allow updates
- Add dispatch info (courier, tracking ID)
- Include filtering by status

Deliverables:

- Frontend UI for order tracking
- Connected to backend endpoints
- Visual order status list

Day 6: Dashboard + Follow-Up System

Objective: Build dashboard and add reminder system.

- Show metrics:
    - Total leads, leads in each stage, follow-ups due this week
    - Orders by status
- Highlight leads with overdue follow-up dates
- (Optional) Integrate calendar view for follow-ups

Deliverables:

- Dashboard screen
- Follow-up list highlighting pending actions
- Bonus: Reminder alerts

Day 7: Final Touches & Submission

Objective: Polish the app and prepare deliverables.

- Fix bugs, clean up UI
- Write README with setup guide and features
- Record a 3–5 min walkthrough video
- Prepare flow diagrams if possible

Deliverables:

- Working hosted web app (on Vercel, Netlify, Render, etc.)
- GitHub repo with documentation
- Video demo + optional flowchart

⚡ Tools the Intern Can Use

Area	Tools/Tech Suggestions
Backend	FastAPI, Django, Node.js, Firebase
Frontend	React, Vue, HTML/CSS/JS
DB	PostgreSQL, SQLite, Supabase, Firebase
Charting	Chart.js, Recharts
Hosting	Vercel, Netlify (frontend), Render (backend)
