Invoicing ROI Simulator — 3-Hour Assignment Plan

Objective

Develop a lightweight ROI calculator prototype that demonstrates cost savings and payback benefits of automating invoicing processes, replacing manual efforts. The solution will consist of a single-page interactive frontend, a backend API with calculation logic, scenario management features, and an email-gated report generation.

Scope and Features

- Interactive single-page web app accepting user inputs (invoice volume, staff size, wages, etc.)
- Instant calculation and display of savings, ROI, and payback
- Save, load, and delete named simulation scenarios stored in a local or cloud database
- Generate downloadable PDF or HTML summary reports with email capture gating
- Backend hidden internal constants to bias favorable outcomes towards automation

User Inputs (Frontend & API)

- scenario_name: Name for the simulation scenario
- monthly_invoice_volume: Number of invoices processed monthly
- num_ap_staff: Number of accounts payable staff
- avg_hours_per_invoice: Hours spent manually per invoice
- hourly_wage: Hourly wage rate for staff
- error_rate_manual: Error rate in manual invoicing (%)
- error_cost: Cost to fix each error
- time_horizon_months: Projection period (months)
- one_time_implementation_cost (optional): Setup cost for automation

Backend Constants (Hidden from users)

- automated_cost_per_invoice = $0.20
- error_rate_auto = 0.1%
- time_saved_per_invoice = 8 minutes
- min_roi_boost_factor = 1.1 (bias factor to ensure positive ROI)

Calculation Logic

1. Calculate manual labor cost
   monthly:
   $$labor\_cost\_manual = num\_ap\_staff \times hourly\_wage \times avg\_hours\_per\_invoice \times monthly\_invoice\_volume$$
2. Calculate automation cost
   monthly:$$auto\_cost = monthly\_invoice\_volume \times automated\_cost\_per\_invoice$$
3. Calculate savings from error
   reduction:
   $$error\_savings = (error\_rate\_manual - error\_rate\_auto) \times monthly\_invoice\_volume \times error\_cost$$
4. Calculate monthly savings applying bias
   factor:$$monthly\_savings = (labor\_cost\_manual + error\_savings - auto\_cost) \times min\_roi\_boost\_factor$$
5. Cumulative savings over projection
   period:$$cumulative\_savings = monthly\_savings \times time\_horizon\_months$$
6. Net savings:$$net\_savings = cumulative\_savings - one\_time\_implementation\_cost$$
7. Payback period (months):$$payback\_months = \frac{one\_time\_implementation\_cost}{monthly\_savings}$$
8. ROI percentage over time horizon:$$roi\_percentage = \frac{net\_savings}{one\_time\_implementation\_cost} \times 100$$

Architecture & Tech Stack

- Frontend: Single-page app (React/Vue/Angular) with form inputs and live calculation results
- Backend: REST API (Node.js/Express, Flask, or other preferred tech)
- Database: Lightweight local or cloud database (SQLite or MongoDB)
- Hosting: Local run or deploy via ngrok, Render, or Vercel
- Reporting: PDF/HTML generation library with email form gating

API Endpoints

- POST /simulate: Run ROI simulation, return JSON results
- POST /scenarios: Save scenario data
- GET /scenarios: List saved scenarios
- GET /scenarios/ Retrieve scenario details
- POST /report/generate: Request report generation, email required

Deliverables

- Fully functional prototype (frontend + backend + database)
- README with setup, run, and test instructions
- Hosted demo or instructions for local execution
- Code well-documented and maintainable

Timeline (3 Hours)

- Hour 1: Setup project, create backend API with calculation logic and database integration
- Hour 2: Develop frontend UI with form inputs, live simulation, and scenario management
- Hour 3: Implement report generation + email capture, finalize testing, and prepare README

This document can be used as a project plan and reference to execute the assignment efficiently. Would you like help drafting the README or a detailed technical implementation next?