Miebach

🧪 Full-Stack Take-Home Challenge

Project Task Management App

(Project • Staffing • Phases • Tasks • Budget • Utilization • Invoicing)

🕓 Timeline

You have **up to 5 calendar days** to complete this challenge.

This exercise is designed to take approximately around **5 hours** of focused work. We don't expect production-grade code or pixel-perfect UI – we care about **data modeling**, **business logic**, and **clarity**.

Optional extensions are available if you want to go further. These are **not required** to pass but are a great way to stand out.

🎯 Challenge Overview

You will build a **lightweight project management tool** centered on the **task level**, but reflecting a realistic hierarchy:

Project → Project Staffing → Project Phases → Tasks → Task Assignments → Budget Tracking & Utilization → Invoicing

- **Managers** set up projects and define **staffing forecasts** (roles, rates, hours based on the contract) to establish a **baseline budget**.
- **Projects** are broken down into phases, which contain tasks.
- **Managers** assign team members to tasks with their rates (fixed at the position level).
- **Contributors** log hours against their assigned tasks.
- The **staffing forecast is dynamically adjusted** as actual hours are logged, so that budget consumption reflects both **planned vs actual** usage.
- Managers track **budget consumption by task, phase, and project**, monitor **people's utilization and availability**, and generate invoices for billing periods.

You may use **any technology stack**.

🧰 Suggested Data Model

You can adapt this to your stack. A minimal schema might look like:

users(id, name, email, role) = *manager / contributor*
projects(id, name, client_name, start_date, end_date)
project_staffing(id, project_id, user_id, role_name, hourly_rate, forecast_hours)
project_phases(id, project_id, name, start_date, end_date)
tasks(id, phase_id, title, description, start_date, end_date, status, budget)
task_assignments(id, task_id, user_id, hourly_rate)
time_entries(id, task_id, user_id, work_date, hours, is_billable)
invoices(id, project_id, client_name, period_start, period_end, total_amount)

📌 MVP Scope (Required)

1. Authentication & Roles

- Seed at least **one Manager** and **2 Contributors**.
- Implement basic authentication (or mock auth).
- Role-based views:
  - **Manager**: create projects, staff projects, define phases & tasks, assign tasks, track budget & utilization, generate invoices.
  - **Contributor**: view assigned tasks and log hours.

---

2. Project Setup

- Manager creates a Project with:
  - name, client_name, start_date, end_date
- The system generates a unique project_id (and task IDs automatically when tasks are created).

Miebach

3. Project Staffing (Forecast)

- Manager adds **project-level staffing entries** to define the **forecasted budget** and to block the resources.
  Each staffing line includes:
  - Contributor (user)
  - Role name (e.g., “Consultant”, “PM”)
  - Hourly rate
  - Forecasted hours

Example:

Name	Role	Rate	Forecast Hours	Forecast Budget
Alice	Consultant	100	50	$5,000
Bob	Analyst	80	30	$2,400

Total forecast budget: **$7,400**
This represents **baseline staffing & cost** at project level.

---

4. Project Phases & Tasks

- **Phases** are defined at the **start of the project** as part of the planning process.
  - Manager creates one or more **Phases** with:
    - name, start_date, end_date
  - These phases remain fixed throughout the project lifecycle and represent major delivery milestones or workstreams (e.g., Design, Development, Testing).
- **Tasks** are added **dynamically throughout the project**.
  - Under each phase, the Manager can **add new tasks at any time** to reflect evolving scope and work breakdown.
  - Each Task includes:
    - title, description, status, start_date, end_date, due_date, budget

- The **due date** represents the planned delivery deadline for that task and can be used to monitor delays or overdue work.

---

5. Task Assignments

- Once a task is created, the **Manager assigns one or more contributors** to it.
- Each task assignment includes:
  - the **user**
  - their **hourly rate** (inherited from project staffing / role)
- Multiple contributors can be assigned to the same task.
- Tasks must be **assigned before contributors can log time** against them.
- Managers can update task assignments if responsibilities change over time.

---

6. Time Logging

- Contributors log hours against their assigned tasks
- Each time entry includes:
  - date, hours, and optionally is_billable.
- The system aggregates **actual hours** and costs at the task level.

---

7. Budget Consumption & Utilization Follow-Up

◆ **Dynamic Staffing Adjustment**

As contributors log hours on tasks, the **forecasted staffing hours** are adjusted downward, and the **actual costs** accumulate.

- **Consumed budget** = logged hours × hourly rate.
- **Remaining budget** = forecast budget − consumed.
  This allows a **hybrid view** where **staffing forecast and actuals** are blended dynamically.

◆ **Budget Tracking Levels**

Track budget vs. actual at multiple levels:

- **Task**: actual_hours × rate vs. task budget
- **Phase**: Sum of all tasks in the phase
- **Project**: Total of all phases compared to **project staffing forecast**

Example:

Level	Forecast	Actual	Remaining
Task A	$2,000	$1,400	$600
Phase 1	$4,000	$2,800	$1,200
Project ABC	$7,400	$5,600	$1,800

📅 Resource Availability & Utilization

Managers should be able to **visualize people's availability and utilization** based on the staffing plan and logged time:

- Display each person's **staffed hours per period** (e.g., per week) in a **calendar or timeline view**.
- Overlay **actual hours logged** to calculate **utilization %**:
- utilization = actual_logged_hours / staffed_forecast_hours
- It should be possible to view this **by person** and **by project**.

---

8. Invoicing

- Manager selects a **Project**, **Client**, and **Billing Period**.
- The system generates an invoice table that sums **billable hours × rate** for tasks within the period.

Example:

Task	Phase	Hours	Rate	Amount
Website Redesign	Design	14	100	1,400
Data Migration Script	Dev Phase	10	80	800

Total: $2,200

Miebach

✅ Acceptance Scenario

1. Manager creates **Project ABC** and staffs:
    - Alice @ $100/hr for 3 days per week for 10 weeks
    - Bob @ $80/hr for 2 days per week for 8 weeks
2. Manager creates **Phase "Design"**, then Task "Website Redesign" with a $2,000 budget.
3. Alice is assigned to the task at $100/hr.
4. Alice logs 8h on Monday and 6h on Tuesday.
5. Manager sees:
    - Task budget: $2,000; consumed $1,400; remaining $600
    - Phase budget: $2,000; consumed $1,400
    - Project forecast: $7,400; actual $1,400; remaining $6,000
    - Alice's staffing forecast reduced by 14h; utilization shows 14/50 = 28% for that period.
6. Manager generates an invoice for that week showing 14h × $100 = $1,400.

🧪 Optional Extensions

Pick one or more if you have time:

- **Billable vs. Non-Billable Time**
- **Dashboard** (project/phase/task progress bars, utilization heatmaps)
- **Document Interaction**
    - 📤 PDF/CSV export of invoices or budget/utilization reports
    - 📥 CSV/Excel import to bulk-create staffing or tasks
- **Contributor Self-Registration**

- **Advanced Reporting** (weekly burn per contributor, forecast vs actual curves)

📦 Deliverables

- A **GitHub repository** with your complete solution.
- The solution can run in **any environment you choose** — local machine, containerized, or hosted — as long as it can be **successfully demonstrated during the interview**.
  - If using a local setup, provide clear and minimal setup steps (e.g., npm install && npm start, docker-compose up, etc.).
  - If hosted (e.g., on a free platform or your own server), use the demo URL and any credentials.
- Your **README** must include:
  - Step-by-step **setup instructions** for running the app
  - **Seeded credentials** (at least 1 Manager and 1 Contributor)
  - A description of any **optional extensions** you implemented
  - A brief note on **design choices, assumptions, or trade-offs**

👩‍💻 Interview Day (1.5 hours)

1. **Introduction and quick office tour** (15min)
2. **Demo (15 min)** – Walk through the acceptance scenario.
3. **Live Extension / scenario exercice (30 min)** – Implement or design a small change together
4. **Q&A & Code Walkthrough (30 min)** – Explain data model & budget/invoice/utilization logic.

Miebach

📝 Evaluation Criteria

Area	Signals
Requirements & Correctness	Project → Staffing → Phases → Tasks flow works correctly
Data Modeling	Proper hierarchical relationships
Code Quality	Clean, modular, understandable
Budget & Utilization Logic	Accurate forecast vs actual + utilization tracking
UX & Usability	Clear flows for both roles
Extensions	Thoughtful optional features
Documentation	Clear README, assumptions explained

🧠 Tips

- Keep **tasks** as the unit of execution and invoicing.
- Focus on **forecast vs. actual** budget logic.
- Show at least a **basic utilization calendar or table**.
- Choose the stack you're most comfortable with.
- Finish the MVP before tackling extras.

✅ **MVP** = Project + Staffing Forecast + Phases + Tasks + Assignment + Time Logging + Budget Tracking + Utilization + Invoicing
✨ **Extensions** = Dashboard, document import/export, billable toggles, reporting
🕐 **Time** = Up to 5 days (~5 focused hours expected)