miniwe

SENIOR FULL-STACK ENGINEER · 20 ЛЕТ В TYPESCRIPT/NODE.JS

Закрываю задачи, которые блокируют продукт: финансовая точность, архитектура под нагрузку, надёжные интеграции. Работаю там, где инженеру дают ownership, а не кусок тикета.

📍 Москва, Россия
miniwe@mail.ru
t.me/miniwe
github.com/miniwe

КЕЙСЫ — ЧТО УМЕЮ РЕШАТЬ

Деньги не должны теряться

В пул-расчётах на N победителей обнаружился классический враг — `number` в JavaScript: бесконечная дробь, накопленная погрешность. Не баг — **систематическая ненадёжность**.
Решение — в архитектуре: утилита на `decimal.js` как доменный примитив, `decimal(12,4)` на уровне БД, форматирование строго в UI. Атомарные транзакции: активация и списание — либо оба, либо никакой. Финансовая точность стала свойством системы, а не договорённостью команды.

TypeScript · NestJS · PostgreSQL · decimal.js · TypeORM

HTTP-таймаут не должен рушить бизнес

Расчёт большого тура падал при превышении таймаута. Состояние — несогласованное, запуск — вручную, снова и снова.
Решение — вынос расчёта туда, где HTTP не участвует: **BullMQ + Redis**, idempotent-дизайн, dead-letter queue. Массовые расчёты стали наблюдаемы и восстанавливаемы.
Принцип: следующий инженер понимает решение без звонка предыдущему.

BullMQ · Redis · NestJS · TypeScript

Провайдер меняет схему в пятницу вечером

Спортивный провайдер изменил структуру ответа — упала половина обработчиков, дежурство в выходные.
Ответ — слой `Raw→Adapter→DTO→Domain`: исходные JSON логируются, ручной ввод — полноценный fallback с теми же контрактами. Изменения провайдера изолированы в адаптере.

Zod · NestJS · PostgreSQL · Structured Logging

Чужие данные не должны утекать

Multi-tenant системы с изоляцией на уровне запросов: JWT + RBAC (USER / ADMIN / DEVELOPER), автоконвертация внешних ID во внутренние, tenant-aware middleware.
Случайный `findAll()` без фильтра tenant — не инцидент безопасности. **Защита встроена в слой**, а не в дисциплину разработчика.

JWT · RBAC · OAuth2/OIDC · NestJS · TypeORM

КАК РАБОТАЮ

СТЕК

API / БИЗНЕС-ЛОГИКА
Node.js · TypeScript · NestJS (Fastify)

ДАННЫЕ
PostgreSQL · TypeORM

ASYNC / ОЧЕРЕДИ
BullMQ · Redis

КОНТРАКТЫ
Zod → DTO → Entity

АУТЕНТИФИКАЦИЯ
JWT · RBAC · OAuth2/OIDC

ФРОНТЕНД
React · RTK Query · FSD

ИНФРАСТРУКТУРА
Docker · Docker Compose

AI-ИНТЕГРАЦИИ
OpenAI API · LLM-пайплайны (JS/Python)

ИЩУ

- Реальный ownership — не кусок тикета
- Качество решений важнее скорости производства
- Через год — уметь больше, чем сейчас

НЕ МОЁ

- Бюрократия без результата
- «Старший» только по стажу, без влияния

ОБРАЗОВАНИЕ

Baku State University, 2000
Прикладная математика · Программист

— **Ownership от спецификации до production.** Помогаю выбрать, что строить и где нельзя накапливать долг.

— **Технические спецификации** как основа кода и AI-инструментов. Хороший промпт = хороший brief.

— **Продуктовый контекст.** Знаю, какую гипотезу проверяет релиз — это уменьшает расстояние между проблемой и решением.

— **Code review как диалог** — объясняю trade-off, не «перепиши».

— **Неопределённость — тоже задача.** Фиксирую trade-off письменно, разделяю «нормально сейчас» и «временное с датой ревизии».

Senior Full-Stack Engineer · Москва · Май 2026

miniwe@mail.ru · t.me/miniwe · github.com/miniwe