CALIFORNIA STATE UNIVERSITY
BAKERSFIELD

Kinetiq

Rajveer Khosa, Matthew Quezada,
Moises Gonzalez, and Oscar Rocha

Overview & Motivation

What We're Building

- Mobile app that suggests next lift weight and estimates PR date.
- Shows a clear "why this weight" based on your recent sets.

Why it matters

- Lifters guess loads → stalls, misses, injury risk.
- Logging apps don't explain changes; templates aren't personal.

How it works

- Compute e1RM (estimated one-rep max) from each set (uses weight, reps, RPE)
- Track true strength with a Kalman estimate (with confidence)
- Convert target e1RM → next load (plate-aware, capped jumps)

Scope for this course

- iOS app (Swift/SwiftUI) + SQL backend.
- Dashboard for Nutrition, Workouts, PR evaluation

How It Works & Benefits

Core Idea

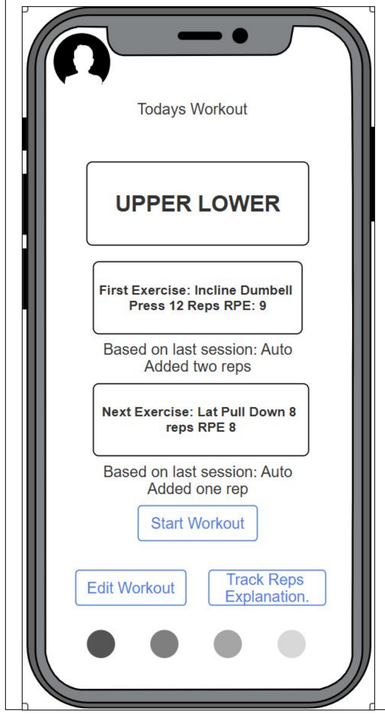
- Compute e1RM from each set (weight, reps, RIR).
- Track true strength with a Kalman estimate (+ confidence).
- Convert target e1RM → next load (plate-aware, capped jumps).

Why is This Better

- **Explainable**: shows why a load is suggested.
- **Personalized**: adapts per lift, per user.
- **Private & fast**: on-device math; works offline.
- **Safer progress**: caps prevent reckless jumps.

Key Outcomes

- More sets in target RIR (1–3).
- Fewer failed attempts.
- PR ETA that improves as you train.



Build & Team

Tech

- **App**: SwiftUI + Swift Charts (on-device calc: e1RM, EMA, Kalman).
- **Backend**: Plain SQL (T-SQL/SQL Server) via small REST bridge.
- **Sync**: JSON with URLSession; local cache; CSV export.
- **Privacy**: On-device by default; opt-in sync.

Success Measures

- % sets in target RIR.
- PR ETA accuracy.
- Fail rate ↓; user adherence ↑

Team Roles

- **Rajveer: iOS App/UI**: SwiftUI screens, charts, offline cache, accessibility.
- **Matthew: ML/Algorithms**: e1RM/EMA funcs, Kalman, load picker, PR ETA.
- **Moises: Backend/SQL**: schema, prepared queries, REST endpoints, auth.
- **Oscar: QA & Data**: synthetic data, test plans, telemetry, docs & poster.

CALIFORNIA STATE UNIVERSITY
BAKERSFIELD

Kinetiq

Rajveer Khosa, Matthew Quezada, Moises Gonzalez, and Oscar Rocha

Motivation & Background

Why we're interested

We all lift and an all inclusive app would be helpful.

Why it fits a senior project

End-to-end system (app, API, ML, deploy).
Clear metrics
Real users, privacy, and testing requirements.

Concept background

Progressive overload: increase load/reps/sets over time.
RIR/RPE: aim 1–3 RIR for strength work.
e1RM: estimate max from submax sets; track trend.

Existing solutions & gaps

Logging apps: good logs, weak/opaque load picks.
Programs templates, not per-user, per-set.
Gap: per-user probabilistic strength tracking with transparent "why this weight" and PR ETA.

Contributions & System Design

What's novel (our contributions)

Bayesian/Kalman strength model with uncertainty.
Target-RIR load picker (plate-aware) with rationale.
PR ETA with confidence bands.
Stall/Deload detector from trends + fatigue signals.

Core components

Logging UI (sets, reps, weight, RIR).
e1RM + EMA trend per lift.
Kalman estimator (+ variance).
Next-load recommendation + "Why this weight."
PR forecast; stall/deload actions.
Auth, offline cache, CSV export.

Ranked features

Critical: logging • e1RM/EMA • Kalman • recommender • rationale • auth/export.
High: PR forecast • stall/deload • templates • charts.
Stretch: contextual bandit • coach/share view • watch companion • health data import.

Work split (smallest parts → owners)

Matthew (ML): e1RM/EMA funcs, Kalman, load picker, PR ETA, stall/deload rules, unit tests.
Moises (Backend): Postgres schema, Supabase RLS, /log/set, /recommendation, /forecast, CI.
Rajveer (App): SvelteKit UI, logging flow, charts, offline/sync, plate math, accessibility.
Oscar (QA/PM): synthetic data, telemetry dashboard, test plans, docs, user study.

Methods, Resources & Timeline

Methods (how it works)

Per-set e1RM: e1RM = weight × (1 + (reps + RIR)/30)
Session best e1RM → Kalman update → mean & variance (true strength).
Target next e1RM (+0.2–1.0%) → convert to load for planned reps & target RIR.
Snap to plate math; cap jumps (≤5%/session, ≤10%/week); show confidence.

Materials / Tech stack

Frontend: SvelteKit + Capacitor — fast web→mobile
Backend: Supabase (Postgres, Auth, Storage, RLS) — simple, secure analytics.
Charts: Chart.js/Recharts — clear trend & confidence visuals.
ML proto: Python (NumPy/Pandas) → port to TypeScript — lightweight, no GPU.
CI/CD: GitHub Actions; optional PostHog for product analytics.
Data: synthetic generator + 5–15 pilot lifters.
Hardware: none (phones only); optional micro-plates for demo.