The Brief

Compensation:
We value your time and will compensate you with 150 Euros for completion of the task, regardless of the outcome.

Context:
Our platform allows mobile apps to embed vertical video experiences (think TikTok-style feeds). Mobile SDKs fetch content and configuration from our backend APIs. Content teams use our CMS to manage videos, playlists, and display rules.

We're planning a new feature: **Personalised Video Feeds**. Instead of showing the same content to every user, we want to serve different videos based on user signals (watch history, engagement, demographic hints passed from the host app).

Your Task:
Design and partially implement the backend system that would power personalised feeds.

Non-negotiable constraints

- Scale: peak **3k RPS** to the feed endpoint (avg ~600 RPS).
- Latency: **p95 < 250 ms**, **p99 < 600 ms** for a **20-item** feed.
- Freshness: new content visible in **≤60 s**; user-signal updates can lag **≤5 min**.
- Privacy: SDK sends **hashed user ID**; no raw PII leaves our VNet; user-event retention **90 days**.
- Multi-tenant: **120 tenants**, each may override global ranking weights.
- Rollout: behind a **feature flag** with a safe **kill switch** to nonpersonalized feed.

Deliverables

1. **System Design Document (primary deliverable)**
   - **Architecture diagram**: how personalisation fits API/CMS/SDK, where ranking happens, caching strategy, data flows for content and events.

    - **Data model**: minimal tables/collections for user signals and content metadata; show keys and retention.
    - **API contract**: endpoint(s) for the SDK (e.g., GET /v1/feed?limit=20), request/response schema, caching headers (ETag/Max-Age), and error semantics.
    - **CMS configuration**: how content managers set rules (e.g., editorial boosts, maturity filters, per-tenant weights).
    - **Trade-offs & decisions**: what you optimized for, what you deferred, and why - call out **reversible** steps.
    - **Rollout & observability**: flags, fallback to non-personalized, the metrics/dashboards you'd ship first (adoption, p95/99, cold-start click-through).
    - **What you'd do with more time**: e.g., bandits/ML, richer real-time signals, offline evals.
2. **Working Prototype (secondary deliverable)**
    - A basic API endpoint that demonstrates your core personalisation logic
    - Doesn't need to be production-ready - we're looking at your approach, code quality, and how you structure things
    - Use whatever language/framework you're most productive in
    - Include a README with setup instructions
3. **Brief Write-up: How You Used AI**
    - What AI tools did you use during this task (if any)?
    - What worked well? What didn't?
    - How would you think about AI assistance for your team if you joined?

What We're Evaluating

- **System thinking**: Can you design something that balances flexibility, simplicity, and real constraints?
- **Pragmatism**: Do you make sensible trade-offs for a small team with limited resources?
- **Code quality**: Is your prototype code clear, testable, and something you'd be comfortable having others build on?
- **Communication**: Can you explain technical decisions clearly in writing?
- **AI fluency**: Do you have a thoughtful, practical perspective on AI in development workflows?

What We're NOT Looking For

- A fully working, deployed system
- Complex ML recommendation algorithms
- Perfect code coverage or enterprise patterns
- Any specific technology choice - use what makes you productive