Level 1 Concept: AI-Powered Matchmaking for Proof of Talk

1) System Architecture

This system is built for one goal: **high-quality introductions for a curated audience**, not high-volume networking.

End-to-end flow

- **Data Collection Layer**
- Mandatory data from registration: role, company, goals, availability, language, seek/offer text.
- Optional enrichment: LinkedIn (opt-in only), company website, public content.
- **Profile Builder**
- Converts raw inputs into a structured profile:
- expertise
- current_focus
- what_they_seek
- what_they_offer
- constraints (language, availability, exclusions)
- Adds confidence tags per field (high/medium/low).
- **Matching Engine**
- Step 1: hard constraint filter.
- Step 2: soft scoring for strategic fit.
- Step 3: limited discovery injection (controlled exploration).
- Step 4: quality gate (score > 65, target 3-7 matches).
- **Explanation + Output Layer**
- Produces plain-English “why this match” reasons.
- Shows only quality-gated recommendations.
- **Organizer Console**
- Add/import attendees.
- Review/export matches.
- Surface non-obvious pair/triad opportunities for concierge intros.

```
[Data Collection] -> [Profile Builder] -> [Matching Engine] -> [Explanation Layer] ->
[Attendee/Organizer Output]
```

2) Data Strategy

Source priority

- **Registration form (mandatory)**
- Most reliable consented baseline.
- Weakness: often shallow.
- **LinkedIn (optional opt-in)**
- Strong context for role history and domain depth.
- Risk: privacy sensitivity and stale profiles.
- **Company website (optional)**
- Useful for current product and positioning signals.
- Risk: marketing bias.
- **Public content (optional)**
- Useful for current thinking (talks/articles/podcasts).

- Risk: coverage is inconsistent across attendees.
- **Past Proof of Talk outcomes (optional)**
- Useful for long-term quality learning.
- Risk: can reinforce network echo chambers.

Handling messy, incomplete, and private data

- **Consent-first**: no hidden scraping.
- **Field confidence**: inferred fields are tagged and down-weighted.
- **Fallback logic**: if enrichment is missing, system still runs on mandatory fields.
- **Conflict resolution**: explicit attendee input overrides inferred data.
- **Cold start policy**: useful matches are still possible from a minimal profile (goal + seek + offer + availability).

3) Matching Logic

How the system decides who should meet

- **Hard constraints (must pass)**
- Language compatibility.
- Schedule overlap.
- Explicit exclusions / policy constraints.
- **Soft scoring (rank candidates)**
- Goal relevance (investment, partnerships, hiring, regulation, learning).
- Strategic fit (decision-making level and domain overlap).
- Value exchange clarity (what one seeks vs what the other offers).
- Feedback prior (past outcomes improve ranking confidence).

Similar vs complementary

- **Default: complementary first** for business outcomes
- investor <-> founder
- bank executive <-> compliance/infrastructure CTO
- policy leader <-> operator deploying in regulated markets
- **Limited similar matching** for peer calibration (for example, investor-to-investor only when clearly useful).

Quality policy

- Show only matches with score > 65.
- Return **3-7** recommendations when enough pass threshold.
- If fewer qualify, show fewer and flag organizer curation support.

4) Output Design

What an attendee sees

- A compact shortlist (3-7 high-confidence matches).
- Each match card includes:
- Name, role, company
- Match score
- 2-3 plain-English reasons
- Suggested next action (“Request Intro”, “Save”, “Not Relevant”)

Before / during / after event

- **Before**: personalized shortlist with reasons and schedule fit.
- **During**: nudges for open time windows and accepted intro requests.
- **After**: simple feedback capture (met / not met / follow-up planned).

Example match card (mock)

- **Elena Rossi — Head of Digital Assets, European Bank**
- **Score:** 81
- **Why this match:**
- Your focus is institutional tokenization; Elena is actively evaluating compliant custody partners.
- Your goals are complementary (partnership + deployment).
- You both have Day 1 PM availability.
- **Action:** Request Intro

5) Technical Stack

Proposed MVP stack (buildable in weeks)

- **Backend:** FastAPI (Python) for rapid API delivery and maintainability.
- **Data layer:** SQLAlchemy + SQLite for demo, Postgres for production.
- **Web UI:** Server-rendered templates for fast iteration and low complexity.
- **Matching service:** rule-based + lightweight semantic text interpretation.
- **Enrichment adapters:**
- company website fetcher,
- LinkedIn ingestion only when user opts in.

Key APIs

- GET /v1/matches/{attendee_id}
- POST /v1/feedback
- GET /v1/scenarios
- POST /v1/enrich/company
- POST /v1/enrich/linkedin

Why these choices

- **Speed:** practical MVP in weeks, not months.
- **Control:** transparent logic + explainability for high-trust users.
- **Scalability:** suitable for ~2,500 attendees with pagination, batching, and thresholded outputs.
- **Safety:** RBAC, CSRF, audit logs, consent controls, and SSRF protections already align with production expectations.