PRD Document – Campus Connect: AI -Driven Community and Study Co-Ordination

Executive Summary

When students try to organize academic and social life on campus they struggle because communication is fragmented across noisy platforms and group formation is manual. Our product—an AI-Driven Study Group Matchmaker—automatically finds and forms balanced study groups by ingesting course, schedule, and preference signals, reducing friction, lowering academic stress, and increasing on-campus participation. The primary business goal is to validate product–market fit by achieving rapid adoption among target cohorts and demonstrating measurable improvements in study group formation speed and retention.

Target User & Persona

Persona: Anjali, the Freshman
Age / Year / Program: 18, First-year undergraduate, Computer Science
Campus Life Snapshot: Lives in a dorm, active in 1 cultural club, attends 5 courses and weekly lab sessions.
Motivations: Wants to do well academically, make friends, and not miss out on study sessions or campus events.
Tech Habits: Uses WhatsApp and Discord heavily, checks email occasionally, prefers mobile-first experiences, signs into university services via single sign-on (SSO). Comfortable with short app-driven workflows and calendar integrations.
Pain Points: Doesn't know which classmates are at the same pace; study group invites are scattered across WhatsApp and Discord; manual coordination means groups are unbalanced (skill gaps or scheduling conflicts); she misses helpful sessions and often studies alone.
Expectations from a solution: Wants a lightweight, privacy-respecting app that plugs into course roster / calendar, suggests 3–5 well-matched peers, creates a trial group session, and allows quick messaging or voice/video meetups.

Solution Overview (MVP)

Core idea (MVP): A mobile/web app that uses a lightweight AI matchmaking engine to form 3–6 member study groups for a given course/topic, based on profile, course enrollment, schedule availability, topic proficiency, and study style.
Minimal feature set to validate problem–solution fit:

1. **SSO-based sign-up & course import** (university SSO or manual course code entry). Minimal friction for onboarding.
2. **Short profile + preferences form (1–2 min):** course(s), topics needing help, topics they can teach, preferred study times, preferred group size, study style (deep-dive / quick-revision), anonymity preference for initial match.
3. **AI Matchmaker engine (MVP algorithm):**
   - Inputs: course list, schedule windows, topic tags (self-reported), recent LMS activity indicator (optional), friends/blocked list.

- Process: lightweight embedding of topic tags + rule-based constraints (schedule overlaps, max 2 experts per group, skill diversity), greedy clustering to produce candidate groups.
- Output: ranked list of 3–5 candidate groups with match score and rationale (e.g., "2 peers available Wed 6–8pm; mixed skill levels; one peer strong in recursion").

4. **In-app group preview & instant-join flow:** Students preview group, accept/decline invitations, and join a temporary group chat. If accepted, group session is created with a suggested meeting time and calendar invite.
5. **Trial session + feedback capture:** After a first meeting (or 72 hours after join), prompt short feedback: helpful? meet again? This drives iterative matching improvements.
6. **Basic integrations:** Calendar add (Google/Outlook), optional copy-to-WhatsApp/Discord invite link. Minimal analytics dashboard for admins (adoption and match stats).

Why this is minimal & sufficient: These features are enough to test whether targeted students (like Anjali) can discover relevant peers faster and form balanced groups without heavy integration work or full-blown messaging persistence.

Acceptance Criteria

For each acceptance story we provide 2–3 testable conditions.

1) Onboarding & Course Import

As Anjali, I want to sign up with my university account and import my courses so that I don't have to manually find classmates.

- AC1: SSO sign-up completes and imports at least one course from university roster or accepts a valid manual course code.
- AC2: Profile + preferences screen is completed within 2 minutes on 90% of test users.
- AC3: When SSO is not available, manual entry still allows the user to be discoverable by course code within 1 hour of submission.

2) Find Matches

As Anjali, I want the app to suggest study groups for my course so that I can join peers who match my schedule and topics.

- AC1: For 80% of active course cohorts (>= 20 participants), the system returns at least one candidate group per user with >= 3 members.
- AC2: Each suggested group displays match rationale (schedule overlap + topic fit) and a match score between 0–100.
- AC3: The average time from "request matches" to "displayed suggestions" is under 5 seconds for 95% of requests.

3) Join & Trial Session

As Anjali, I want to join a suggested group and quickly schedule a trial study session so we can test the chemistry.

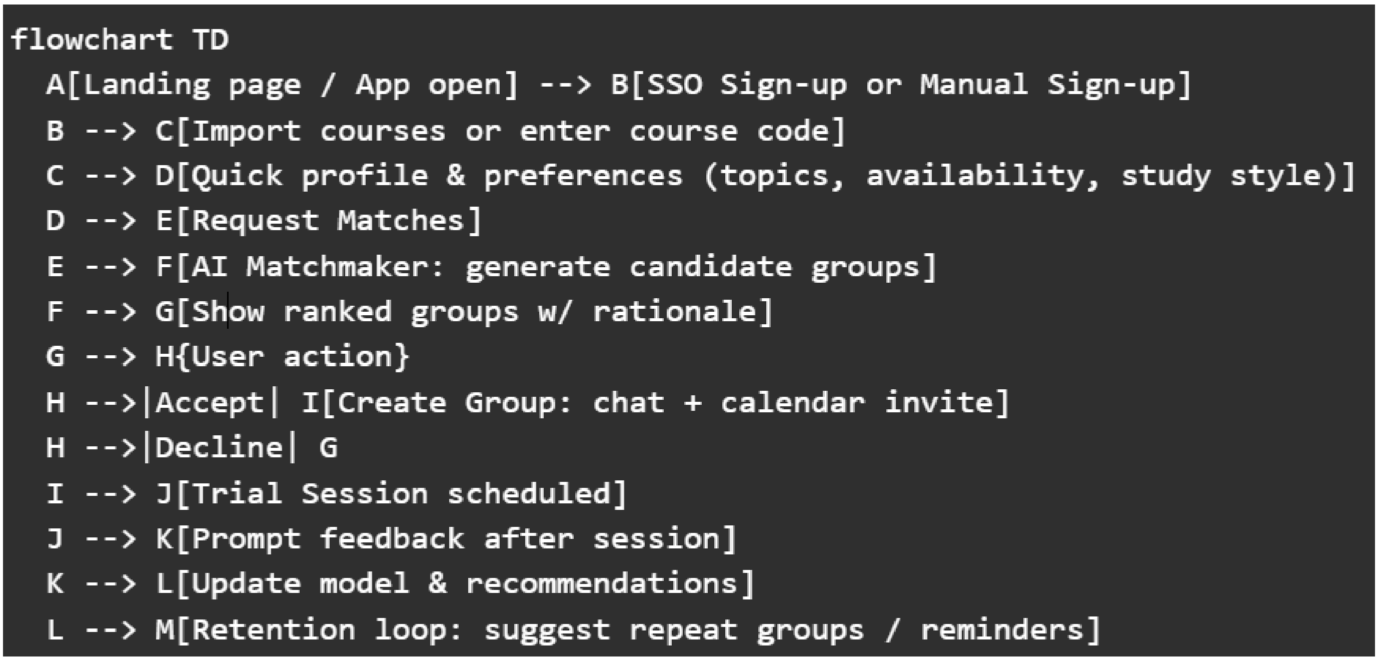
- AC1: When a user accepts a suggested group, a group chat is created and at least one proposed meeting time is auto-populated based on common availability.
- AC2: Calendar invite can be added to Google or Outlook in a single click.
- AC3: After the session window (72 hours) feedback is prompted and at least 60% of joined users respond in pilot test.

4) Privacy & Control

As Anjali, I want control over who sees my profile and to opt out of being discovered so that I feel safe using the platform.

- AC1: User can set discoverability to Public / Course-only / Hidden at any time; changes propagate within 60 minutes.
- AC2: Users can block specific peers; blocked peers are never suggested in future matches.
- AC3: Data usage and retention policy is shown during onboarding and acceptance is logged.

User Flow Diagram (Core MVP Flow)



```
flowchart TD
  A[Landing page / App open] --> B[SSO Sign-up or Manual Sign-up]
  B --> C[Import courses or enter course code]
  C --> D[Quick profile & preferences (topics, availability, study style)]
  D --> E[Request Matches]
  E --> F[AI Matchmaker: generate candidate groups]
  F --> G[Show ranked groups w/ rationale]
  G --> H{User action}
  H -->|Accept| I[Create Group: chat + calendar invite]
  H -->|Decline| G
  I --> J[Trial Session scheduled]
  J --> K[Prompt feedback after session]
  K --> L[Update model & recommendations]
  L --> M[Retention loop: suggest repeat groups / reminders]
```

Notes: The flow assumes optional branches for manual course entry, connecting calendar, and exporting invites to existing chat apps.

Technical Requirements (focus on AI component)

AI Inputs (expected):

- **Primary:** Course identifier (roster or code), topics needing help (tag list), topics able-to-teach, preferred study times (time windows), timezone.

- **Secondary (optional for MVP but highly recommended):** LMS engagement signal (last activity timestamp / assignment submissions), past study group ratings, calendar busy/free events (read-only), friend/blocked user lists.
- **Privacy & consent signals:** discoverability flag, data retention consent.

AI Outputs (expected):

- Ranked candidate groups for a specific course/topic with: group member IDs (anonymized if needed), match score, top matching signals (schedule, topic overlap, skill diversity), suggested meeting time.
- Per-user recommended actions ("Invite 1 more beginner", "Reschedule to Tue 7pm") and predicted match confidence.

Core AI components & algorithm choices (MVP):

1. **Feature vector construction:** One-hot or small dense embeddings for topics + normalized availability vectors + scalar skill level. Keep embedding dimension small (e.g., 32–64) for speed.
2. **Clustering/ranking:** Use a fast greedy clustering or approximate nearest neighbor (ANN) + diversity constraints (e.g., ensure at least one peer with higher mastery for each group). A rule-based post-filter ensures schedule overlap and size constraints.
3. **Explainability:** Generate short natural language rationale for matches using a template-based approach ("X available Wed; Y strong in topic Z").
4. **Feedback loop:** Capture post-session feedback to weight future match scoring and surface repeat partners.

Core AI components & algorithm choices (MVP):

1. **Feature vector construction:** One-hot or small dense embeddings for topics + normalized availability vectors + scalar skill level. Keep embedding dimension small (e.g., 32–64) for speed.
2. **Clustering/ranking:** Use a fast greedy clustering or approximate nearest neighbor (ANN) + diversity constraints (e.g., ensure at least one peer with higher mastery for each group). A rule-based post-filter ensures schedule overlap and size constraints.
3. **Explainability:** Generate short natural language rationale for matches using a template-based approach ("X available Wed; Y strong in topic Z").
4. **Feedback loop:** Capture post-session feedback to weight future match scoring and surface repeat partners.

Infra & stack (MVP-level):

- **Frontend:** React (web) + React Native or PWA for mobile.
- **Backend:** Node.js/Python microservice for match logic.
- **Database:** Postgres for relational data; Redis for session/cache.
- **ML infra:** Lightweight model served via REST (FastAPI or Flask), embeddings stored in an ANN index (e.g., FAISS or Milvus).

- **Messaging/real-time:** WebSocket or Sendbird/Stream for in-app chat; allow exportable invite links for WhatsApp/Discord.
- **Scheduler & calendar integration:** Google Calendar & Microsoft Graph APIs (OAuth flows).

Required third-party APIs & integrations:

- **University SSO (SAML/OAuth/Shibboleth)** — mandatory to lower friction.
- **LMS API (Canvas, Blackboard) — optional** to ingest engagement signals.
- **Calendar APIs (Google Calendar, Microsoft Graph)** — to propose meeting times.
- **Messaging connectors:** Sendbird / Stream for in-app chat; or deep-link support for WhatsApp/Discord invites.
- **Authentication & privacy tools:** standard OAuth 2.0 flows and consent logging; GDPR/FERPA considerations.

Security & privacy considerations:

- Follow FERPA guidance on student data; limit sensitive data storage and provide export/delete flows.
- Data minimization: only request course rosters and availability needed for matching.
- Anonymized suggestions: optionally mask full names until both parties accept a match.

Launch Success Metrics (KPIs) — mapped to AARRR funnel

Acquisition

- **KPI 1 — Campus Sign-up Rate:** % of invited students (email or SSO prompt) who complete onboarding and import at least one course in the first 14 days. **Target:** 20% in pilot cohort.

Activation

- **KPI 2 — Match Acceptance Rate:** % of users who receive at least one candidate group and accept/join a group within 48 hours. **Target:** >= 35%.

Retention

- **KPI 3 — Weekly Active Users (WAU) among onboarded users:** % of onboarded users returning and using match/feedback flows at least once per week. **Target:** 25% at week 4.

Referral / Virality

- **KPI 4 — Invite Rate:** Average number of outbound invites (shareable links to WhatsApp/Discord/Email) per activated user within first 7 days. **Target:** 1.5 invites/user.