**IceBreaker Product Development Plan**

**1. Overview**

**Purpose**

Deliver IceBreaker’s MVP and beta around **spontaneous, ephemeral, proximity-based connections** — fast, modular, privacy-first, and ready for vibe-driven iteration.

**MVP Core Objectives**

* Real-time, interest-based matching.
* Ephemeral, secure chats that vanish after proximity ends.
* Validate usability, engagement, and privacy perceptions through beta testing.
* Maintain modular, swappable architecture.

**2. Core Features**

* **Authentication:** Supabase Auth with OAuth providers (Google, Apple, Email).
* **Interest Import & Matching:** Users import/select interests; matches based purely on proximity and shared interests.
* **Proximity Discovery:** Turf.js + Supabase Edge Functions for real-time, geospatial scanning.
* **Ephemeral Chat:** Supabase Realtime; chats auto-expire after proximity loss.
* **Privacy & Safety:** Visibility toggle, Panic button, End-to-End encryption, Anonymous matching.

**3. UI/UX Vision**

* **Aesthetic:** Subdued retro-tech theme with scan-lines and typewriter text animations.
* **Key Screens:** Welcome & onboarding, match discovery, ephemeral chat, settings & safety controls.
* **Accessibility:** WCAG 2.1 compliance, high-contrast option planned for future releases.

**4. Technology Stack**

* **Frontend:** Next.js 14 (App Router), TailwindCSS, Expo (mobile future).
* **Backend:** Supabase Postgres (Auth, Database, Realtime, Edge Functions).
* **Hosting:** Vercel for Web; Expo for Mobile.
* **Push Notifications:** Expo Notifications (mobile) and Web Push (desktop PWA).
* **Payments:** RevenueCat for subscription and in-app purchase management.
* **Analytics:** PostHog and Vercel Analytics.
* **CI/CD:** GitHub Actions for build/test/deploy automation.

**5. Security Requirements**

* End-to-End encrypted chats.
* GDPR/CCPA compliant data handling.
* No persistent message storage.
* Panic button for immediate emergency alerting.
* Role-Based Access Control (RBAC) for backend and admin functions.

**6. Testing & QA Strategy**

* **Unit Testing:** Core feature logic validation.
* **Integration Testing:** API communications between frontend, backend, database.
* **End-to-End Testing:** Full flows (sign-up, match, chat, exit).
* **Security Testing:** OWASP standard vulnerability scans.
* **Performance Testing:** Simulated multi-user loads.

Tools: Jest, Mocha/Chai, Cypress, Postman, OWASP ZAP, Locust.

**7. Deployment & Operations**

* Three environments: Development, Staging, Production.
* Feature flag toggling via environment configurations.
* Automated deployment using GitHub Actions.
* Monitoring via Vercel and PostHog with real-time alerting.

**8. Beta Testing & Feedback Plan**

* Invite-only beta for 18+ users.
* In-app feedback and structured surveys.
* Private community group (Slack or Discord) for real-time tester insights.
* Success metrics: Match rate, Chat initiation rate, Session duration, User comfort score.

**9. Repository & Folder Structure**

Recommended IceBreaker GitHub Repository Layout:

/frontend

/app

/components

/hooks

/contexts

/pages (for Next.js routing)

/public

/styles

next.config.js

/backend

/functions

matchUser.js

sendMessage.js

panicAlert.js

/schemas

userProfile.sql

matchRecord.sql

/edge-functions

proximityScan.js

/infrastructure

/ci-cd

github-actions.yml

/env

.env.development

.env.staging

.env.production

/docs

/architecture

/api-contracts

/feature-specs

**Key Points:**

* Clear separation of frontend, backend, infrastructure, and documentation.
* Environment configs managed through .env files.
* All API contracts and service logic modularized from day one.

**10. API Entity Definitions (Word-friendly, no code)**

**User Entity:**

* Unique Identifier (UUID)
* Display Name (optional real name, otherwise randomized username)
* Avatar (optional)
* Interests (array of strings)
* Visibility Status (boolean)
* Geolocation (lat/long, temporary and not stored persistently)

**Match Entity:**

* Match Identifier (UUID)
* User 1 Identifier
* User 2 Identifier
* Shared Interests (array)
* Match Timestamp
* Match Expiry Timestamp (based on proximity break)

**Chat Message Entity:**

* Message Identifier (UUID)
* Sender Identifier
* Receiver Identifier
* Message Content (encrypted text)
* Timestamp
* Auto-delete Trigger (linked to match expiry)

**11. 90-Day Roadmap (Vibe-Coding Milestones)**

| **Timeline** | **Milestone** | **Details** |
| --- | --- | --- |
| Weeks 1–2 | Core Environment Setup | GitHub repo initialized, Supabase instance set up, Vercel hosting connected |
| Weeks 3–6 | Core Feature Development | User authentication, profile management, interest import, proximity detection |
| Weeks 7–9 | Ephemeral Chat System | Real-time chat, WebSocket integration, auto-expire logic |
| Weeks 10–12 | Privacy, Safety, Analytics | Visibility toggle, Panic button, basic PostHog event tracking |
| Week 13 | Internal Alpha Test | Closed team testing, performance tuning, initial bug fixing |
| Week 14–15 | Beta Test Launch | Invite-only user testing, feedback capture |
| Week 16 | First Iteration Sprint | Implement critical feedback, polish UX, prepare for scaled public launch |