This **Developer Brief** is a professional summary designed to be sent to a software agency or a freelance developer. It strips away the "pitch" and focuses on the **logic, APIs, and constraints** they need to know to give you an accurate cost and timeline estimate.

Developer Brief: Project BeaconMatch (MVP)

1. Executive Summary

BeaconMatch is a cross-platform mobile application (iOS/Android) designed for hyper-local disaster response. The core value proposition is **offline peer-to-peer resource matching**. The app must allow users to request aid and offer resources using a "swipe-based" UI (Tinder-style) without relying on centralized servers or cellular data.

2. Core Technical Pillar: The Mesh

The developer must implement a **Bluetooth Low Energy (BLE) / Wi-Fi Direct Mesh Network**.

- **Requirement:** Asynchronous "Store-and-Forward" messaging.
- **Suggested SDKs:** Bridgefy, Google Nearby Connections, or Multipeer Connectivity (iOS).
- **Constraint:** The app must be able to transmit small JSON packets (Lat/Long, User ID, Asset Category) across at least 3 "hops" between devices.

3. Key Functional Modules

Module A: Dual-Mode UI/UX

- **The Toggle:** A global state switch between "Requester" and "Provider."
- **The Stack:** A swipeable card interface.
    - *Input:* Simple tags (Medical, Food, Tool).
    - *Output:* Distance-sorted cards based on local mesh data.

- **Survival Mode:** A low-power CSS/Style theme (Pure #000000 background) that minimizes CPU/GPU load.

Module B: Offline Geospatial Logic

- **Requirement:** No reliance on external map APIs (Google/Apple Maps) during Disaster Mode.
- **Feature: The Compass Bridge.** Calculate the bearing and distance between two sets of GPS coordinates (retrieved from the phone's hardware) and display a directional needle.
- **Offline Caching:** Capability to download and store small vector map tiles for a 10km radius during initial setup.

Module C: The "Pulse" Manager

- **Logic:** A background service that manages the radio duty cycle.
- **Cycle:** 15 seconds of active discovery / 5 minutes of sleep.
- **Battery API:** The app should monitor battery percentage and automatically increase the "Sleep" interval if the battery drops below 15%.

Module D: Local Handshake (Verification)

- **Mechanism:** Offline QR Code generation and scanning.
- **Logic:** Scanning a peer's QR code must trigger a "Status: Resolved" packet that propagates through the mesh to clear that specific ID from the neighbourhood stack.

4. Technical Stack Preferences

- **Framework:** Flutter or React Native (for code sharing across platforms).
- **Database:** NoSQL (e.g., Hive or WatermelonDB) for fast local-first syncing.
- **Security:** AES-256 encryption for all mesh-transmitted data.

---

5. Deliverables for MVP Phase 1

1. **Proof of Connectivity:** Two phones exchanging a "Need Card" with Airplane Mode ON.
2. **The Swipe Engine:** Functional UI that updates a local "Mission List."
3. **The Compass Guide:** Accurate directional pointing between two peer nodes.

How to use this document:

Send this to a developer and ask:

1. *"Do you have experience with Bluetooth Mesh or the Bridgefy SDK?"*

2. *"What would be the estimated 'Man-Hours' to build this MVP?"*
3. *"Which framework (Flutter vs. React Native) would you recommend for the 'Pulse' background service?"*