Product Requirements Document (PRD): EmoHabit

Version: 1.0 **Date:** October 26, 2025 **Platform:** iOS & Android (React Native + Expo) **Backend:** Python FastAPI + AI Agent

1. Project Overview

EmoHabit is not just a habit tracker; it is an **AI-powered Routine Architect**. It helps users design realistic days, tracks their consistency, and uses "Radical Honesty" to hold them accountable. The core differentiator is "Emo"—a character that acts as a silent observer and guide, helping users optimize their lives without constant chatting.

Brand Identity:

- **Colors:** Black (#000000), White (#FFFFFF), Orange (#FF5722).
- **Vibe:** Minimalist, Bold, Dark Mode Default.
- **Mascot:** "Emo" (White silhouette, black background).

---

2. User Authentication & Onboarding

2.1 Splash Screen (The "Instagram" Effect)

- **Visual:** Black background. Center screen shows the **Emo Logo** (White).
- **Footer:** "From [Your Name/Company]" appears at the bottom in grey text.
- **Behavior:** Lasts 1-2 seconds while the app loads data.
- **Auto-Login:** If a user has logged in previously, bypass auth screens and go straight to the Dashboard.

2.2 Authentication

- **Methods:**
    1. **Sign in with Apple** (Mandatory for iOS release).
    2. **Sign in with Google** (Standard for Android/iOS).
    3. **Email/Password** (Custom account creation).
- **Technical:** Use Firebase Auth or Supabase (integrated with FastAPI) to handle tokens.

2.3 The "Low-Friction" Onboarding

- **Step 1: Introduction:** A clean screen where Emo introduces the concept.
    - *Copy:* "I am Emo. I help you build routines that actually stick."
- **Step 2: The Goal Dump:**
    - Input field: "What are your current main focus areas?" (e.g., Coding, Fitness, Sleep).
    - *Constraint:* Do not ask for name/age/weight yet. Keep it strictly about *goals* to reduce friction.

---

3. Core Feature: The Routine Architect (AI)

3.1 The "Brain Dump" (Input)

- **User Action:** User types a rough plan for the day/week.
    - *Example:* "I want to code for 4 hours, go to the gym, and read."
- **AI Processing (FastAPI):** The backend receives this text.

3.2 The "Smart Adjustment" (Reasoning)

- **Logic:** The AI analyzes the input against health parameters.
    - *Scenario:* If user schedules 20 hours of work, AI flags it.
    - *Scenario:* If user forgets sleep, AI suggests it.
- **Output:** Emo presents a **Modified Plan**.
    - *UI:* "I've drafted your schedule. I added an 8-hour sleep block because you need rest to code effectively. Is this okay?"
- **Approval:** User clicks "Confirm Routine" to lock it in.

3.3 The "Prepare for Tomorrow" Reminder

- **Trigger:** At 8:00 PM (configurable), Emo sends a notification.
- **Message:** "Tomorrow is a new game. Plan your routine now to win the morning."
- **Action:** Opens the "Brain Dump" screen for the next day.

---

4. Core Feature: Tracking & "Focus Mode"

4.1 Execution Mode

- **UI:** A Timeline View. The current task is highlighted (e.g., "Reading: 2:00 PM - 3:00 PM").
- **User Action:** User taps **"Start"** on the task.

4.2 The Distraction Detector (Logic)

- **Constraint:** On iOS, we cannot see *which* app the user opens (Sandboxing). We can only see if they *left* EmoHabit.
- **Scenario A: "Offline Task" (e.g., Reading, Gym, Sleeping)**
    - If User minimizes EmoHabit → **Flag as Suspicious.**
    - *Reasoning:* You don't need your phone to read a book.
- **Scenario B: "Online Task" (e.g., Watch Tutorial, Research on Instagram)**
    - If User minimizes EmoHabit → **Flag as Safe/Neutral.**
    - *Reasoning:* They left the app to do the task.

4.3 The "Radical Honesty" Check

- If a user flagged as "Suspicious" returns to the app:
    - **Emo Prompt:** "You left the app during 'Reading'. Be honest—were you reading or scrolling?"
    - **Options:** "I was distracted" vs. "I was working."
    - *Data:* This feeds the "Time Wasted" analytic.

---

5. Analytics & Reporting

5.1 Daily/Weekly Report

- **Visual:** Bar charts using the Orange/Black theme.
- **Metrics:**
    1. **Completion Rate:** % of planned tasks finished.
    2. **Time Wasted:** Calculated based on "Distracted" flags from Section 4.3.
    3. **Screen Time Context:** "You spent 2 hours on your phone during 'Deep Work' blocks."

5.2 The "Wasted Potential" Stat

- Emo calculates the gap between "Planned" and "Actual."
- *Message:* "You planned 4 hours of coding, but wasted 45 minutes. That's 5 hours of lost progress this week."

---

6. Technical Specifications

6.1 Frontend (React Native + Expo)

- **App State Management:** Use `AppState` API to detect when the app goes to the background (minimized).
- **Notifications:** `expo-notifications` for reminders.
- **Storage:** `AsyncStorage` for keeping the user logged in locally.

6.2 Backend (FastAPI + Python)

- **AI Engine:** Groq API (GPT-4o or similar).
- **Structured Data:** Use **Pydantic** to force the AI to return clean JSON for the schedule (Start Time, End Time, Activity Type).
- **Database:** PostgreSQL or SQLite (for storing user logs and routines).

---

7. UI Design Guidelines (For AI Generators)

Use this prompt to generate your UI screens:

> **Project:** EmoHabit **Style:** High-contrast Dark Mode (OLED Black #000000). **Accent:** Vibrant Safety Orange (#FF5722). **Typography:** San Francisco (iOS) / Roboto (Android) - Bold weights. **Key Screen:** "Active Routine View" **Description:** A vertical timeline of tasks. The current task is expanded. **Character:** Include a white vector silhouette of a character (Emo) pointing up, placed near the header. **Feedback:** Success state = Green/Orange glow. Failure/Distraction = Grey/Red dimming.

---

8. Next Immediate Action

Since you are the sole developer, here is your **step-by-step execution plan**:

1. **Setup:** Initialize the Expo project and the FastAPI backend.
2. **Phase 1 (The Brain):** Write the Python script that takes a string ("I want to sleep and code") and returns a JSON schedule. **Do this first.**
3. **Phase 2 (The UI):** Build the Login screen and the Timeline view in React Native.
4. **Phase 3 (The Link):** Connect the frontend to the backend so the app displays the AI's plan.
5. **Phase 4 (The Tracker):** Implement the `AppState` logic to detect when the user minimizes the app.