Key Features

1. **Onboarding**
   A questionnaire with 38 questions to gather information on physical condition, goals, and experience.
2. **Registration**
   Through Google, Email, or Apple ID.
3. **Paywall**
   Monetization through subscription (payment integration: Stripe or other).
4. **Types of Personal Trainers (PT)**
   - Option #1: AI Only
   - Option #2: Human Trainer Only
   - Option #3: Hybrid (AI + Trainer)
5. **Chat with PT**
   - AI + Human (at the beginning)
   - Ability to pin messages in the chat
   - Pin messages to the main page
   - Save messages to bookmarks
6. **AI/Trainer Deadline Features**
   - Creation of workout plans based on onboarding: within 24 hours
   - User photo analysis: within 24 hours
   - Technique check via video: within 24 hours
   - Exercise replacement due to pain: within 5 minutes
   - Exercise replacement without equipment: within 5 minutes
   - Adaptation for hotel gym: within 5 minutes
   - Weekly report analysis: within 24 hours
   - Monthly report analysis and adaptation: within 24 hours
   - Push notifications about replies from the coach
7. **Workout Module**
   - Description, illustrations, YouTube link to exercises
   - Sync with Apple Health
   - Push notification for workout completion (even outside the app)
8. **App Pages**
   - Main Page: Fitness journal, weekly plan, progress, technique
   - Training Program: By day with descriptions, muscle groups, and alternatives
   - Learn: Blog with videos, images, and links to PTs
   - Progress Page:
     - Photos, dates, measurements, comments
     - Recommendations for uploading photos
     - Sync with journal and chats
     - ToDo / Done view
9. **Nice to Have**
   - Voice input
   - Voice communication
   - Full sync between chat and user interface
   - All chat data transferred to Workout/Progress sections

Deep Thoughts

1. **UI Tech Stack**
    - **React Native**
      JavaScript/TypeScript skills, a large ecosystem of community libraries, and near-native performance. Its drawbacks include occasional performance bottlenecks on complex UIs
      (*used by Facebook, Walmart, etc.*)

    - **Flutter (Dart-based)**
      Excellent UI performance, rich out-of-box widgets, and better performance on animations. However, the team must learn Dart and Flutter idioms; the pool of Flutter developers is smaller than for JS/React

    - **.NET MAUI/Xamarin**
      Smaller binary sizes (e.g. ~9 MB vs 54.5 MB for React Native). MAUI's community is growing but still smaller than JS. React Native startups faster to start with faster builds, whereas .NET MAUI apps are smaller and use native controls

    - **Ionic/Capacitor**
      Fast to prototype but less suited if smooth scrolling or heavy graphics are needed. Similarly, a Progressive Web App could cover all devices but has limited access to native APIs (health sensors, device permissions) and isn't ideal for an App Store presence. Usually gives lower performance and "less native" UI feel

   In summary, React Native will maximize code reuse and team productivity given the React background. It has a large, mature ecosystem and ongoing investment by Meta. Flutter offers future-proof performance and UI flexibility but requires Dart. .NET MAUI is attractive if deep .NET integration matters, but may slow the team. Considering maintenance and hiring, React Native's popularity (with large NPM library support) and our team's skills, React Native is the pragmatic MVP choice

2. **AI Services for Workouts and Chat**
   LLM will generate a workout plan. Top LLMs - OpenAI's GPT, Anthropic's Claude, Google's Gemini, DeepSeek (?supported in the USA?).
   DeepSeek-V3 at $0.27 input/$1.10 output per 1M tokens
   Google Gemini-2.0-Flash-001 at $0.10 input/$0.40 output per 1M tokens

   Workout plan generation typically involves moderate content lengths, falling into the "Short Article" category with 400-600 words requiring 520-900 tokens. A comprehensive workout plan with exercise descriptions, sets, reps, and nutritional guidance would likely consume 800-1,500 output tokens per generation. Input prompts containing user data (age, weight, fitness goals, equipment availability) typically range from 50-200 tokens.

   Using the provided pricing data, a single workout plan generation would cost

approximately $0.0005-0.002 using DeepSeek-V3, $0.0008-0.003 using Gemini-2.0-Flash, or $0.004-0.018 using GPT-4o. For applications expecting 1,000 workout plan generations daily, monthly costs would range from $15-60 for budget options to $120-540 for premium models

.Net integration:
Provider-specific SDKs offer enhanced functionality and simplified integration. OpenAI provides an official .NET SDK supporting async operations and streaming responses. Google's Vertex AI SDK includes .NET bindings for Gemini models. Anthropic offers unofficial community SDKs for Claude integration. These SDKs often include built-in error handling, authentication management, and response parsing capabilities.

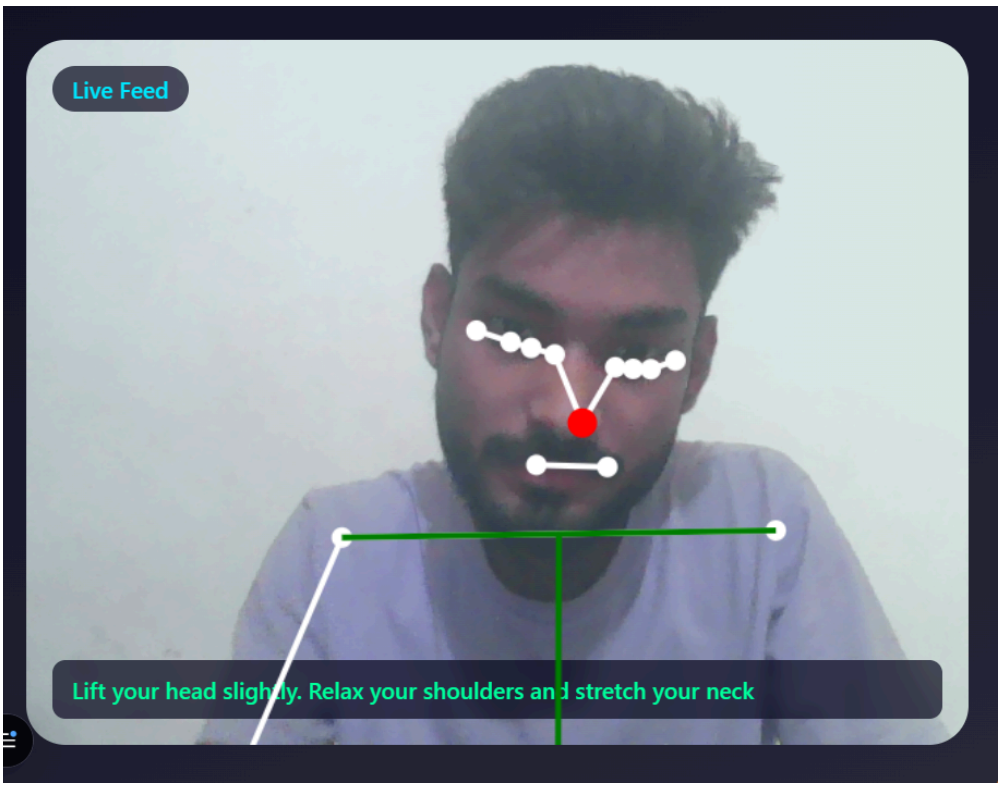
IDEA: Can we combain RPA somehow? 1 paid subscription and through RPA access different chats. PROS: Direct memory management + cost-efficient. CONS: Not sure it is a valid approach

Recommendation: (Implement A/B testing for a good decision)
DeepSeek-V3 - optimal choice for basic workout plan generation, offering exceptional cost-effectiveness at $0.27 input/$1.10 output with adequate performance (1,368 Arena Score)
Google Gemini-2.0-Flash provides the best balance of cost and performance for most applications, with superior pricing ($0.10 input/$0.40 output) and strong context handling capabilities (1,000k tokens)

3. **Pose Detection, image/video analysis**
   Ask Petro how they developed

   There are a lot of services that can track your movement, so not sure it should be difficult to implement. BIGGER CHALLENGE - make recommendations on a fly

- **KaiaHealth** DEMO
  “Motion Coach” solution that uses phone camera to give real-time form feedback, but it is a closed product (no public API)
- **AWS Rekognition, Google Cloud Vision**
  do not output human pose skeletons out-of-the-box
- Google **MediaPipe Pose Landmarker** DEMO
  can detect 33 body keypoints in images/videos. It has client-side versions for Android/iOS/Web to identify body joints and analyze posture
- **Apple’s Vision/ARKit pose detection** DEMO
- **MoveNet** or **OpenPose** (via TensorFlow) DEMO
- **MediaPipe**
  Live demo
  GitHub Repo



- **Possible Prompt Strategy (in case of LLM)**
  You are a certified fitness coach with 10 years of experience.
  User profile: {Age:25, Weight:70kg, Goal:Muscle gain}
  Current exercise: {Exercise:Squat, Set:3/5}
  Recent form feedback: {KneeValgus:2.5°}

  Analyze the attached form data and:
  1. Identify 3 technique improvements
  2. Suggest form correction exercises
  3. Output JSON format

Recommendation/Idea to verify: MediaPipe or MoveNet - most practical, low latency, no cloud dependency

4. **Prompt Engineering and Context Management**
   Since LLM APIs are stateless, we must manage conversation context ourselves. Good prompting is critical: use a clear system prompt (e.g. “You are a fitness coach…”), then append user inputs. To maintain memory, one approach is Buffer memory – simply appending the last N exchanges to the prompt.

   ? Vector db to store and retrieve relevant facts for each prompt(LangChain .NET)

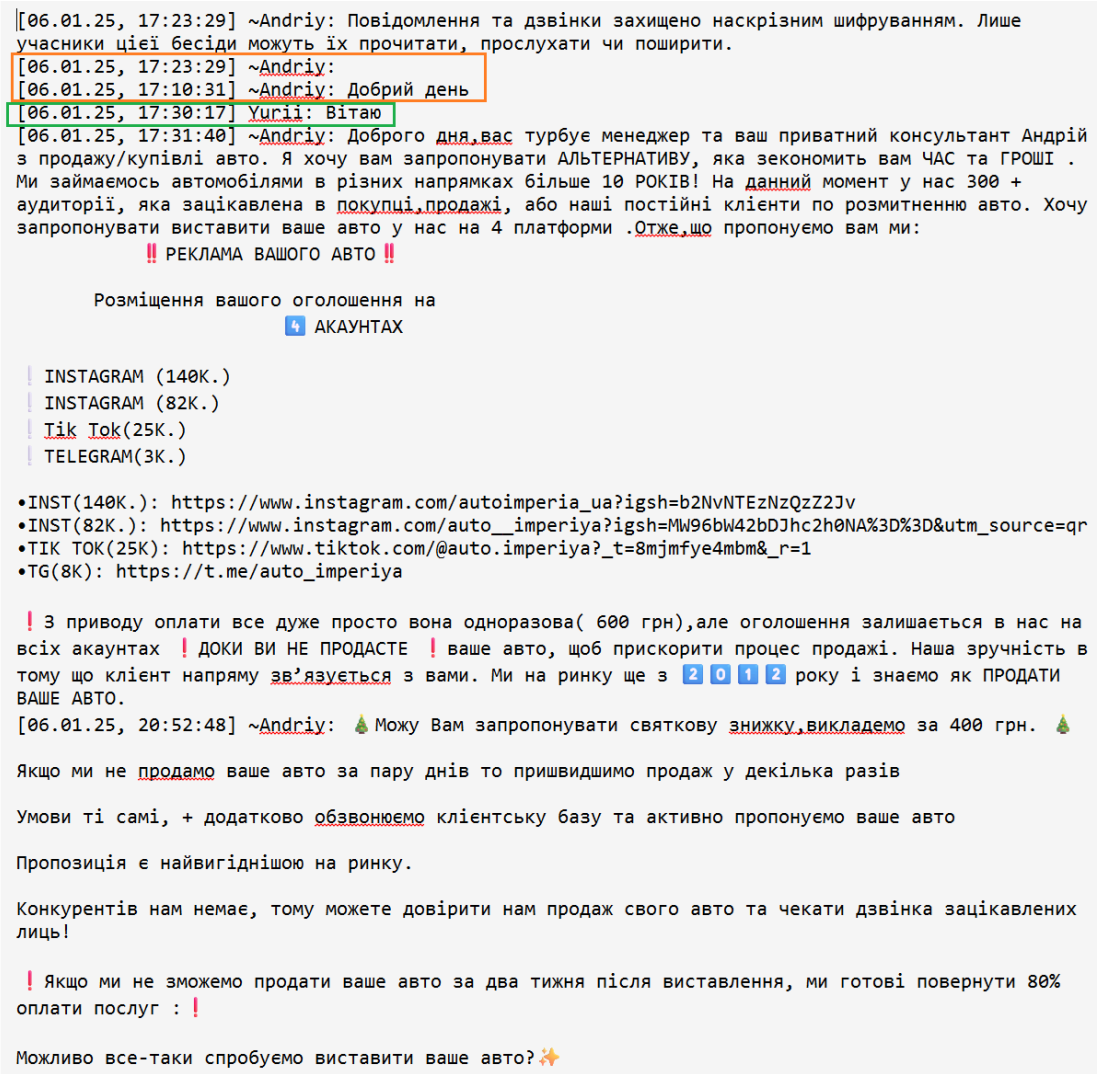
   Prepare multiple prompt templates with placeholders for dynamic content.

5. **WhatsApp Integration**
   To reach users on WhatsApp, we need a WhatsApp Business API provider. Two common paths: **Twilio API for WhatsApp** or **360dialog (Meta's official BSP)**. Both require a registered business and a WhatsApp-approved number. Twilio offers a unified API, charging about $0.005 per message plus WhatsApp's conversation fees. 360dialog charges a monthly fee $50/month, plus WhatsApp's per-conversation rates. Twilio includes all messaging (SMS, WhatsApp) in one account and bills per message, which can suit varied use. For pure WhatsApp usage, 360dialog is often cheaper at scale (flat monthly + lower per-message cost)

   We can handle user's response through webhook and store in our db.

   MIGRATION: WhatsApp allows manual export of a chat thread to a text file, but there is no official API to pull a user's entire message history. WhatsApp built-in Export feature, but we should manually do it. [RPA can help]. In such case we should create a parsing algorithm to update the in-app chat. High-level vision steps: We should go through each chat -> export chat -> paste to our system -> parse it properly -> retrieve user by phone -> add to db -> remove chat or send a message *Migrated to app*.
   Using RPA, parsing the exported text (dates, senders, messages) into a database would be complex. *Unreliable and hard to scale*

   PITFALLS
   User is registered with a different number
   Export does not include audio/media or those files are base64 - difficult to parse - can cause problems
   Potentially wrong number generation or instead of number WhatsApp can take username - impossible to find a user in our system - need to clarify
   …etc



[06.01.25, 17:23:29] ~Andriy: Повідомлення та дзвінки захищено наскрізним шифруванням. Лише учасники цієї бесіди можуть їх прочитати, прослухати чи поширити.
[06.01.25, 17:23:29] ~Andriy:
[06.01.25, 17:10:31] ~Andriy: Добрий день
[06.01.25, 17:30:17] Yurii: Вітаю
[06.01.25, 17:31:40] ~Andriy: Доброго дня,вас турбує менеджер та ваш приватний консультант Андрій з продажу/купівлі авто. Я хочу вам запропонувати АЛЬТЕРНАТИВУ, яка зекономить вам ЧАС та ГРОШІ . Ми займаємось автомобілями в різних напрямках більше 10 РОКІВ! На данний момент у нас 300 + аудиторії, яка зацікавлена в покупці,продажі, або наші постійні клієнти по розмитненню авто. Хочу запропонувати виставити ваше авто у нас на 4 платформи .Отже,що пропонуємо вам ми:
‼️ РЕКЛАМА ВАШОГО АВТО ‼️

Розміщення вашого оголошення на
4️⃣ АКАУНТАХ

INSTAGRAM (140K.)
INSTAGRAM (82K.)
Tik Tok(25K.)
TELEGRAM(3K.)

•INST(140K.): https://www.instagram.com/autoimperia_ua?igsh=b2NvNTEzNzQzZ2Jv
•INST(82K.): https://www.instagram.com/auto__imperiya?igsh=MW96bW42bDJhc2h0NA%3D%3D&utm_source=qr
•TIK TOK(25K): https://www.tiktok.com/@auto.imperiya?_t=8mjmfye4mbm&_r=1
•TG(8K): https://t.me/auto_imperiya

❗З приводу оплати все дуже просто вона одноразова(600 грн),але оголошення залишається в нас на всіх акаунтах ❗ДОКИ ВИ НЕ ПРОДАСТЕ ❗ваше авто, щоб прискорити процес продажі. Наша зручність в тому що клієнт напряму зв’язується з вами. Ми на ринку ще з 2️⃣0️⃣1️⃣2️⃣ року і знаємо як ПРОДАТИ ВАШЕ АВТО.
[06.01.25, 20:52:48] ~Andriy: 🎄Можу Вам запропонувати святкову знижку,викладемо за 400 грн. 🎄

Якщо ми не продамо ваше авто за пару днів то пришвидшимо продаж у декілька разів

Умови ті самі, + додатково обзвонюємо клієнтську базу та активно пропонуємо ваше авто

Пропозиція є найвигіднішою на ринку.

Конкурентів нам немає, тому можете довірити нам продаж свого авто та чекати дзвінка зацікавлених лиць!

❗Якщо ми не зможемо продати ваше авто за два тижня після виставлення, ми готові повернути 80% оплати послуг :❗

Можливо все-таки спробуємо виставити ваше авто?✨

6. **In-app chat**
   - **Sendbird** DEMO
   Provides ready-made UI kits and supports text, images, video, audio, threads, etc. Sendbird is enterprise-grade with end-to-end encryption, GDPR/data-residency options, and data export tools.
   Pricing can be high for large user bases
   - **Stream Chat** DEMO DEMO2
   Similar to Sendbird, GDPR-friendly infrastructure
   - **DIY solutions**
   A full in-app chat is more work: implement real-time messaging, notifications, message storage, etc.

7. **Compare usability WhatsApp and in-app chat**
   A/B testing. Many users already prefer WhatsApp

8. **Apple HealthKit and Google Fit**
   For iOS devices, HealthKit is the system health data store. In React Native to read/write HealthKit data we can use modules like rn-apple-healthkit or @ovalmoney/react-native-fitness(supports both HealthKit and Google Fit access)

To use HealthKit, you must enable the HealthKit capability in Xcode (add the HealthKit entitlement to your app's entitlements file). At runtime, you request permission for each type of data (steps, heart rate, workouts, etc.). Per Apple guidelines, *users must explicitly grant permission for each data type*, and if denied, your app simply sees no data (it cannot detect the denial)

One important restriction: apps using HealthKit must have a privacy policy and can only use the data for health-related purposes. Apple's Health & Fitness guidelines state that "*third-party apps must explain why they are requesting access to your Health app data*" and "*each app is also required to have a privacy policy that describes its use of health data*". Also, HealthKit data should not be used for advertising or sold. For Android/Google Fit, similar rules apply

9. **App Store & Google Play Deployment**
    - **iOS (App Store)**
      You need an Apple Developer account ($99/yr). In Xcode, configure signing (Certificates, Identifiers, Profiles) for your app ID. Enable capabilities like HealthKit in the App ID and entitlements if used. The binary (.ipa) is built via Xcode archives. You must have an App Store listing with app icon/screenshots and a privacy policy URL. Specifically for fitness/health apps using HealthKit, Apple requires a clear privacy policy. According to Apple, *all HealthKit-enabled apps must have a privacy policy explaining health data usage*.
      Also follow the **App Store Guidelines**: fitness apps must not make unproven medical claims, must respect user privacy, and any subscription (e.g. premium workout plans) must use Apple's In-App Purchase (IAP) system. Failure to use IAP for digital services risks rejection. [IAP - Needs to clarify]

    - **Google Play (Android)**
      Create a Google Play Console account ($25 one-time). Configure an Android keystore to sign your app. Use Android Studio or a CI system (Bitrise, App Center) to build an .aab. The Play listing also needs store graphics and a privacy policy. Google's **User Data policy** requires you to explain how you use fitness data. If requesting permissions like BODY_SENSORS or Google Fit data, ensure you follow the *permission declaration form rules*.
      Subscriptions must use Google Play Billing for in-app digital content. Google also enforces 15% fee on first $1M (same as Apple), etc.

   In both stores, consider the **Digital Services Act** (EU) and age ratings (likely low for fitness). Apple and Google also enforce **PSD2/SCA** compliance indirectly via Apple Pay / Google Pay (they handle 2FA). For non-AppStore payments (if any), integrate 3D-Secure for EU cards.

10. **HealthKit and App Store Policies**
    Apple has specific rules for health data: besides the privacy policy, the user must have control over data. We must only request HealthKit data we need. As the Apple Health privacy page states, "*Each app is also required to have a privacy policy that describes its use of health data*". During app review, be transparent: in the

description and review notes, explain how HealthKit data improves the app (e.g. "*We use step count and heart rate to adjust workout intensity*"). In sum, follow Apple's HealthKit guidelines precisely: enable the entitlement, ask explicit consent, and honor data privacy. For Google Fit, similar rules exist (declare fitness data use and store a privacy policy).

11. **Payment & Billing**
    - **Stripe**
      Widely used (2.9% + $0.30 per charge in US). Supports subscriptions (Stripe Billing), has great developer tools, and handles EU SCA and VAT (via tax APIs). With Stripe, you manage all billing logic and taxes yourself
    - **Braintree (PayPal)**
      Similar fees (~2.9% + $0.30) to Stripe. Integrates PayPal/Apple Pay easily. Also does not refund the original fee on refunds. It's a solid choice if you want PayPal. Both Stripe and Braintree charge $15 for chargebacks
    - **Apple/Google In-App Subscriptions**
      Mandatory for any digital content sold through the app. Apple takes ~17–30% commission (17% in many regions now, or 10% for annual subscriptions/Small Biz), plus a 3% payment processing fee. Google's Play fees are similar (30% or 15% after year one for subscriptions). Using IAP simplifies EU VAT (they collect it), and SCA is handled by Apple/Google. The downsides are the high commission and rigid refund policies.

      Apple's App Store Guidelines historically mandate that any digital content or feature unlocked in-app must use Apple's In-App Purchase (IAP) system. This rule (section 3.1.1) forbids directing users to external buying mechanisms for digital goods. That means fitness plans or subscription content delivered in the app must go through IAP, and Apple takes a 30% cut

      Stripe explicitly notes: physical goods/services can use Stripe in-app, but "sales of digital products/content… must use Apple's in-app purchases"

In summary: For MVP, starting with **Stripe** is simplest for web payment & subscriptions, but **App Store/Play** IAP will be required for in-app purchases

12. Payment scenarios
    - **In-app purchase (IAP)**
      The most straightforward compliant method is to use Apple/Google's in-app billing for any paid fitness plans or subscriptions. This is required for digital content but incurs store fees. It integrates smoothly (users pay with their Apple/Google account) and ensures the app won't be rejected.

    - **Web-only payment**
      Alternatively, we can ask users to sign up or subscribe on the website (using Stripe/Braintree) and then let them log into the app. For example, the app could be free to download, but require a login tied to a paid web account. Apple's new rules now allow an app to include a link/button to an external

purchase page, but Google still discourages directing users out of app for content purchases unless they fall under a specific program. ** Some apps use web checkout for subscriptions (paying Stripe ~2.9% fee) and only use the app as content reader.

- **Hybrid model**
  We might combine approaches. For instance, offer a free trial or basic plan via IAP (ensuring compliance), and upsell premium plans on the website (accessed via an external link). Or sell related physical goods (e.g. branded equipment) via Stripe in-app, while using IAP for the workout content. Each strategy has trade-offs: linking to the web can reduce fees but adds friction and potential review risk.

  *** *We could offer, for example, a web-only sign-up option, BUT, ensure it doesn't violate the latest store policies (as they've recently changed).*

13. **Data Privacy (GDPR/HIPAA)**
    GDPR: If serving EU users, the app must comply. This means giving users control over personal data (clear privacy policy, ability to delete account/data, etc.). Implement functions to let the user export or delete their data. On the backend, GDPR requires lawful basis (consent) to process fitness/health data, and possibly a Data Protection Officer/Impact Assessment for sensitive health data. Use HTTPS everywhere, store data encrypted at rest, and limit retention. For example, we might timestamp and remove old chat logs. For analytics, ensure anonymization or use tools with EU data hosting.

    HIPAA: If the app collects Protected Health Information (PHI) (identifiable health data), and we operate in the US, HIPAA applies. As soon as we store anything like medical conditions or GPS workout logs tied to a person, treat it as PHI. This means using HIPAA-eligible services: e.g. AWS (with BAA), Azure (with BAA), or Google Cloud's HIPAA offerings. We must sign Business Associate Agreements (BAAs) with the cloud providers, and encrypt data. For chat and AI, note that any health info we feed to an LLM must be carefully handled (e.g. avoid sending directly PHI to OpenAI, or anonymize). Tools: Amazon's Comprehend Medical is HIPAA-eligible for text extraction, but not needed for chat; HIPAA compliance means additional security (access logs, audit, staff training).

    * **BAA - Business Associate Agreement**, a legal contract under HIPAA regulations, or the Buy American Act, which dictates the origin of products purchased by the US government

14. **Coach Model Strategy**
    AI-only allows the fastest rollout (no need to hire coaches). An AI can immediately generate generic plans and answer questions. However, purely AI coaching may lack trust and nuance.
    Human-only (real coaches) ensures quality and motivation but scales slowly (each coach can only handle few clients and the cost is high). Generation/review will be longer.

Hybrid: AI generation + coach should validate what AI created - that will create higher trust/value to customers

Suggestion: ukrainian coaches should be cheaper than the US ones, so we can try to make a translation eng ←→ urk - should be OR we can try setting up the AI agent that will create a workout plan and than other LLM should verify it correctness

15. **Team size**
    Example roles:
    1 Frontend Mobile Engineer - React Native
    1 Full-stack - backend focus - ability to help on UI
    1 Backend with AI/ML experience - to integrate LLM and other AI services - ON DEMAND
    —
    1 DevOps later to deploy properly

16. **Feature Priority + timeline**
    - **Authentication**
      Web or mobile? Custom JWT vs Auth0/other service?

      Timing
      - UI: ~ 3 days
        Auth screens, session storage, logout, profile edit
      - APi ~3-7 days
        Custom JWT/Auth0 respectively

    - **Onboarding**
      Onboarding should capture the most essential user inputs (for example: age, gender, height/weight, fitness goal, current activity level, experience, available equipment, and any medical conditions). In practice, only ~7–10 of the 38 questions are needed initially (covering personal info, goals, and major constraints). The rest (detailed lifestyle or preference questions) can be postponed

      Can be modified from admin panel
      Timing
      - UI: ~ 3-4 days
        Questionnaire flow, validation, progress indicator, summary screen
      - APi: ~ 3 days
        Models, endpoints, input validation, storage

    - **Workout Plan Generation**
      Generate a weekly plan via LLM API calls

      Timing
      - Prompt Design & Testing LLM: ~3days
        Define prompt, examples, A/B, refinement

    - APi integration: ~4days
      APi client, management, endpoint to request plan, parse and persist LLM response to a needed structure
    - UI: ~3days
      Plan generation, loading, UI components

- **Progress Tracking**
  Log completed workouts and view basic stats

  Timing
    - UI: ~3-4days
      Check-off interface, progress dashboard, charts
    - APi: ~2-3days
      Endpoints to record, fetch, validate, generate stats

- **Blog**
  Timing
    - UI: ~ 2-3 days
    - APi: ~1-2days
    - Admin: ~1-2days

- **Settings**
  Timing
    - UI: ~2-4 days
      Settings screen, edit profile, preferences, other toggles
    - APi: ~1-2days

- **Admin Tool**
  Web or mobile?

  Timing
    - UI: ~3-4days
      CRUD for workout templates, dashboard to view key metrics(user count, active subs)
    - APi: ~2-3days
      Workout template management, user lookup, flagging

- **Chat integration**
  Timing
    - WhatsApp integration
        - Twilio webhook setup: ~ 2 days
        - APi handler, message store: ~2days
        - UI to navigate to WhatsApp: 1day
    - In-App Chat
        - Investigation: 1day
        - Implementation: depends - 7+ days

- **Payment**
  Timing
  - Stripe through Web ~ 1-2weeks
  - In-App Purchase - Apple/Google - Clarify - 1-2 weeks

- **HealthKit/Google Fit Integration**
  Timing
  - Need to clarify. ~ 5-7 days

- **Buffer & Overhead**
  Timing
  - Testing, bug fixes: ~ 2weeks
  - Project setup, architecture, db creation, deployment: ~5-7days
  - App Store/Play Market review cycles: ~1-2weeks of waiting
  - + ADDITIONAL PITFALLS

Ideas

- Nutrition
- Find people nearby - like Tinder for gym