American Impact Review

THEORETICAL ARTICLE

A Local-First Architecture for Privacy-Preserving Personalization in iOS Health and Fitness Applications

Tomasz Kubiak*

Welltech, Krakow, Poland

* Corresponding author: tm.kubiak.dev@gmail.com

OPEN ACCESS

Citation:
Tomasz Kubiak (2026). A Local-First Architecture for Privacy-Preserving Personalization in iOS Health and Fitness Applications. *Am. Impact Rev.* 10.66308/air.e2026058

Received: January 24, 2026
Accepted: February 17, 2026
Published: February 21, 2026

DOI:
10.66308/air.e2026058

ISSN: 3071-124X

Abstract

Health and fitness applications increasingly adapt goals, reminders, and recovery flows from behavioral and physiological signals. On iOS, these signals may be adjacent to health data even when a product is marketed as wellness rather than medical software. The privacy problem is therefore not only a consent or notice problem; it is an architecture problem concerning where raw signals are transformed, stored, linked, and exported. This article proposes the Local-First Personalization Envelope (LFPE), a conceptual architecture for privacy-preserving personalization in iOS health and fitness applications. Using a design-science-oriented synthesis of mobile health privacy research, privacy engineering theory, EU and U.S. governance, and Apple platform requirements, the paper maps legal and platform principles to app-level controls. LFPE places raw health-related inputs, local feature extraction, and personalization decisions inside the device trust boundary where feasible. Remote feedback is limited to tightly scoped aggregate metrics, differentially private telemetry, or federated updates that pass a disclosure-threshold review. The paper contributes a regulatory-to-engineering matrix, a disclosure-threshold protocol, and a personalization ladder that assigns controls to local rules, compact on-device models, aggregate telemetry, federated learning, and cloud personalization. The framework does not claim legal compliance or empirical utility without implementation evidence. It provides a structured starting point for iOS teams that need personalization without treating raw health-related data as ordinary analytics telemetry.

Keywords: privacy-preserving personalization, iOS, mobile health, fitness applications, HealthKit, privacy by design, contextual integrity, federated learning

1. Introduction

Mobile health and fitness applications increasingly use personalization to adjust goals, reminder timing, interface order, and recovery prompts after onboarding. In a walking or wearable-companion product, a fixed target can be inappropriate when recent activity, sleep, or recovery signals indicate fatigue or routine disruption. These adaptations are commercially and clinically adjacent: many products avoid medical claims but still process signals that can reveal physical condition, daily schedule, menstrual cycle context, medication routines, or recovery periods.

The core risk is not simply that an app uses sensitive data. It is that personalization often moves those data into ordinary product analytics. A common implementation pattern logs events with identifiers, routes them through mobile analytics or attribution software development kits (SDKs), enriches a profile on a backend, and returns a decision to the device. Empirical analyses of mHealth apps have found insecure transmissions, third-party sharing, and significant divergence between privacy policies and observed data flows [1,2]. Software-engineering research also indicates that mobile privacy work has emphasized leak detection and policy analysis more than requirements-level and architecture-level methods [3].

Current governance does not remove this engineering gap. The General Data Protection Regulation (GDPR) imposes duties such as purpose limitation, data minimization, security, and data protection by design and by default [4]. In the United States, the Health Insurance Portability and Accountability Act (HIPAA) applies only in defined covered-entity and business-associate contexts, while the Federal Trade Commission (FTC), Food and Drug Administration (FDA), and U.S. Department of Health and Human Services (HHS) guidance may still matter for consumer health apps depending on claims, data sources, and organizational roles [5-7]. On iOS, Apple adds a platform governance layer through HealthKit permissions, App Privacy Details, privacy manifests, and required-reason API declarations [8-12]. These instruments constrain design, but they do not specify the exact boundary between local adaptation and remote profiling.

This paper addresses that boundary. The research question is: how can an iOS health or fitness application personalize its interface while minimizing export of raw health-related signals and keeping the remaining data flows auditable? The answer proposed here is the Local-First Personalization Envelope (LFPE), a component model that separates raw signals, local features, personalization decisions, and optional feedback channels. The contribution is practical rather than doctrinal. LFPE translates privacy-by-design and contextual-integrity principles into feature-level engineering gates before implementation.

The paper makes four contributions. First, it defines LFPE as an iOS-oriented architecture for local-first personalization. Second, it introduces a regulatory-to-engineering matrix that connects EU, U.S., and Apple platform requirements with technical controls. Third, it proposes a disclosure-threshold protocol for deciding when data export is acceptable. Fourth, it presents a personalization ladder that assigns controls to local rules, on-device models, aggregate telemetry, federated learning, and cloud personalization. The framework is conceptual and requires prototype evaluation before operational deployment.

2. Background and related work

2.1. Mobile health privacy and the limits of policy-centered protection

Empirical studies show that privacy failures in mHealth applications often result from ordinary implementation choices rather than unusual attacks. Papageorgiou et al. [1] analyzed popular mHealth applications with static and dynamic techniques and reported systematic weaknesses in security and privacy practice. Tangari et al. [2] conducted a large cross-sectional analysis and found serious problems in data-sharing behavior and in the correspondence between privacy policies and observed data flows. For personalization architecture, the relevant implication is narrow but important: event collection, SDK integration, network routing, and retention rules can expose health-related context even when no explicit clinical record is uploaded.

User research also limits the protective value of notice and consent. Schroeder et al. [13] found that mature adults' willingness to use mHealth applications depended on trust, perceived control, and uncertainty about post-collection data use. Acquisti et al. [14] explain why privacy choices do not follow a stable rational-choice model: preferences shift with framing, context, and timing. A user may authorize step-count access for a fitness feature without understanding that a derived activity state can later be linked to attribution SDKs, persistent identifiers, or server-side experimentation.

Software-engineering research reaches the same unresolved layer. Ebrahimi et al. [3] map mobile app privacy research into categories such as privacy policies, requirements, user perspectives, and leak detection; requirements-level and architecture-level work remain comparatively underdeveloped. Rezaee et al. [15] move closer to implementation by identifying criteria and countermeasures for mHealth developers. The remaining gap is architectural: criteria must become component boundaries, data schemas, dependency controls, and release gates for iOS personalization features.

2.2. Privacy by design, contextual integrity, and privacy engineering

Privacy by design is useful only when it is bound to engineering artifacts. Cavoukian's formulation emphasizes privacy as the default setting, privacy embedded into design, full-lifecycle security, and visibility [16]. The GDPR gives this principle legal force through data protection by design and by default [4]. ENISA guidance similarly treats privacy engineering as a set of strategies, technical building blocks, and repeatable design choices rather than a post hoc compliance exercise [17,18].

Contextual integrity explains why technical protection alone is insufficient. Nissenbaum [19] argues that privacy depends on appropriate information flows within a social context, including the actors, data attributes, and transmission principles involved. A step-count-derived state used locally to suppress an aggressive reminder can fit the health and fitness context. The same state exported to a marketing endpoint, combined with a persistent identifier, and retained for cross-app profiling changes the recipient and transmission principle. LFPE therefore treats the data recipient and export purpose as architectural variables, not merely legal-notice fields.

The NIST Privacy Framework supplies a complementary organizational vocabulary for identifying, governing, controlling, communicating, and protecting privacy risks [20]. Bednar et al. [21] show why that translation matters inside engineering teams: developers may treat privacy as a vague legal demand unless responsibilities are linked to concrete design decisions. A usable personalization architecture should therefore be visible in tickets, code review, SDK review, privacy manifests, and release governance.

2.3. On-device personalization, federated learning, and differential privacy

Not every personalization function requires the same technical method. Local deterministic rules can adjust step goals, suppress reminders, or restore streaks from recent on-device states. Compact on-device models can rank content or select reminder timing without exporting raw records. For many health and fitness use cases, the product objective is immediate adaptation for one user, not remote profile enrichment.

Federated learning extends local computation by allowing devices to compute updates while a server aggregates those updates instead of collecting raw training data [22]. Federated evaluation has also been used to assess on-device personalization strategies without centralized logging of sensitive user data [23]. Differential privacy provides a formal method for limiting the effect that one individual's data can have on aggregate outputs [24].

These methods reduce some risks but do not remove purpose limitation. Federated updates may leak information through high-dimensional gradients, small cohorts, eligibility rules, or weak aggregation. Differentially private telemetry may still be excessive if the schema collects events unrelated to the user's fitness relationship. The privacy architecture must therefore start with minimization and local execution; statistical and cryptographic techniques should reinforce that boundary rather than compensate for broad collection.

2.4. Regulatory and platform governance as engineering inputs

The EU and U.S. regimes place different but overlapping pressure on mobile personalization. The GDPR is horizontal and principle-based, with recurring engineering implications for lawful basis, transparency, purpose limitation, data minimization, storage limitation, security, and data protection by design [4]. The EU Artificial

Intelligence Act adds risk-based obligations for AI systems, although ordinary wellness personalization does not automatically become high-risk unless the function's intended use and effects fall within regulated categories [25].

The U.S. regime is more fragmented. HIPAA coverage depends on the covered entity, business associate, and protected health information relationship; many direct-to-consumer wellness and fitness apps sit outside that structure. The FTC's Health Breach Notification Rule reaches vendors of personal health records and related entities outside HIPAA coverage, and its 2024 amendments clarified relevance to health apps and similar technologies [5]. FDA oversight depends on whether a function is a medical device function and on the risk of harm if the function fails [6,26]. The FTC Mobile Health Apps Interactive Tool reflects this multi-agency environment by routing developers through HIPAA, FTC, FDA, and related questions [27].

Apple's rules are not public law, but they shape iOS implementation. HealthKit is based on fine-grained user permission and privacy constraints [8,9]. App Privacy Details require developers to disclose data collection and tracking practices in App Store Connect [10]. Privacy manifest files and required-reason API declarations move part of the accountability into app dependencies and APIs that could otherwise be used for fingerprinting or tracking [11,12,28]. For iOS teams, privacy architecture is therefore also supply-chain and release-management work.

3. Method

3.1. Design objective and unit of analysis

This work uses a design-science-oriented conceptual method. The design objective is to specify an architecture that enables personalization in iOS health and fitness applications while keeping raw health-related signals local where feasible and making any export decision explicit. The unit of analysis is a personalization feature that reads a signal, transforms it, stores it, exports it, or discards it.

The method is not an empirical evaluation and not a systematic literature review. It develops an implementable conceptual model by synthesizing failure modes reported in mHealth privacy research, privacy-engineering principles, regulatory constraints, and iOS platform requirements. Product utility, computational cost, legal compliance, and formal privacy loss are left for future prototype evaluation.

3.2. Source base and synthesis procedure

The source base has four layers. The first layer consists of empirical and review studies on mHealth leakage, over-sharing, policy gaps, and user trust [1-3,13,15]. The second layer consists of privacy theory and engineering literature, including privacy by design, contextual integrity, ENISA guidance, the NIST Privacy Framework, and work on the gap between privacy principles and engineering practice [16-21]. The third layer covers computation techniques relevant to local personalization, federated learning, federated evaluation, and differential privacy [22-24]. The fourth layer consists of official EU, U.S., and Apple documents that affect health and fitness applications on iOS [4-12,25-28].

The synthesis began from recurrent failure points: over-collection, transformation of raw signals into sensitive inferences, export through analytics dependencies, unclear retention, and inconsistency between platform disclosures and implementation. For each failure point, a candidate control was assigned at the lowest feasible architectural layer: permission gating, local feature extraction, storage separation, network export review, SDK inventory, privacy manifest review, retention limits, or aggregate/federated feedback controls.

The resulting LFPE model was checked against three criteria. First, each component had to correspond to an engineering decision that an iOS team could implement or review. Second, the model had to preserve useful personalization without assuming server-side profiling as the default path. Third, the model had to make regulatory and platform requirements auditable before implementation, rather than leaving them as privacy-notice edits after development.

4. Results

4.1. The Local-First Personalization Envelope

LFPE places the device trust boundary around the personalization core (Fig. 1). User permission and context govern access to HealthKit, sensors, and app events. Raw records are then converted into lower-resolution local features before they reach personalization logic. A timestamped step sequence, for example, can become a local ”goal lag today” state; a sleep-window signal can become a coarse readiness category. The interface receives a decision, not the underlying history.

The personalization core can be a rule set or a compact on-device model. The architecture does not require a particular algorithm. Its premise is that raw health-related data should not cross the network boundary merely because the interface adapts. Local interaction events may feed back into the local feature extractor and personalization core, but diagnostic logging of feature vectors should be disabled unless a separate disclosure threshold is satisfied.

The governance and audit layer is part of the architecture rather than an end-stage documentation task. It contains the SDK inventory, retention rules, privacy manifests, required-reason API declarations, HealthKit permission wording, and export policy. If a dependency, backend schema, or release artifact contradicts the declared personalization boundary, the feature has an architecture defect rather than a paperwork defect.

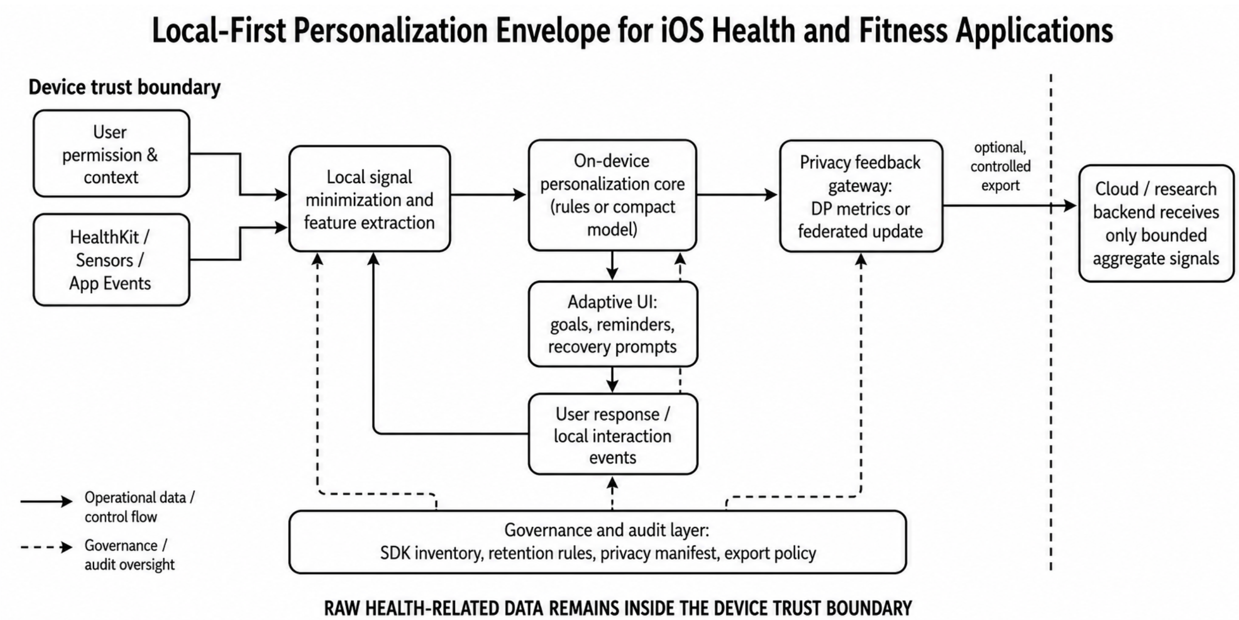

Figure 1. Local-First Personalization Envelope for iOS health and fitness applications.

The feedback gateway is intentionally narrow. When product learning cannot be achieved locally, the backend may receive a coarse aggregate, a differentially private metric, or a federated update. The gateway is not a general analytics outlet. Its function is to support bounded improvement without requiring upload of a user’s activity history, recovery pattern, or other raw health-related timeline.

4.2. Regulatory-to-engineering matrix

Table 1 translates selected EU, U.S., and Apple platform pressures into architecture-level controls. It is a design aid, not legal advice. Its purpose is to prevent teams from treating legal classification as the only trigger for privacy-preserving design.

Table 1. Regulatory and platform requirements translated into architecture-level controls.

Dimension	European approach	U.S. approach	Architectural consequence
Regulatory structure	The GDPR is horizontal and principle-based, with duties such as purpose limitation, minimization, security, and data protection by design/default.	The U.S. regime is sectoral: HIPAA, FTC authority, the Health Breach Notification Rule, FDA device-software policy, and state privacy laws may apply depending on the actor and intended use.	Implement a baseline privacy architecture that is stronger than the narrowest legal trigger; then add jurisdiction- and product-specific obligations.
Health and wellness data	Health data may receive special-category protection when it relates to health status or reveals health information.	HIPAA applies only in defined covered-entity or business-associate contexts; FTC and state rules may still reach consumer health apps outside HIPAA.	Do not route health-related signals through ordinary analytics pathways; use local derivation and purpose-specific export thresholds.
AI and personalization	The AI Act adds risk-based duties where a function qualifies as an AI system within regulated categories.	FDA oversight depends on intended medical-device function; FTC authority can address deceptive or unfair data practices.	Separate wellness personalization from diagnostic or treatment functions; document intended use and user-facing claims.
Transparency and consent	GDPR transparency and rights require meaningful explanation of purposes, recipients, retention, and user choices.	U.S. requirements are fragmented, but FTC practice can penalize misleading privacy representations and unauthorized health-data disclosure.	Create a purpose contract for each personalization feature and align it with in-app permission wording, privacy labels, and backend schemas.
Platform governance	EU rules interact with app-store and platform constraints but remain legally independent.	U.S. distribution still depends on platform rules in addition to legal requirements.	Maintain privacy manifests, SDK inventory, required-reason API declarations, and HealthKit permission review as release artifacts.

Note. GDPR = General Data Protection Regulation; HIPAA = Health Insurance Portability and Accountability Act; FTC = Federal Trade Commission; FDA = Food and Drug Administration.

The matrix does not imply that EU and U.S. products must be architecturally incompatible. A stronger shared baseline can keep raw health-related personalization local, define the purpose early, limit recipients, and make dependency behavior auditable. Jurisdiction-specific overlays can then address lawful basis, notice language, breach duties, device-software classification, and documentation depth.

4.3. Disclosure-threshold protocol

In many product workflows, the privacy decision is made implicitly when a team chooses the easiest data flow. The disclosure-threshold protocol reverses that order (Fig. 2). The team first defines the personalization function, classifies the signal, tests whether a lower-resolution local feature is sufficient, chooses the data boundary, and verifies the audit trail before implementation.

The protocol distinguishes four cases. Low-value, low-exposure display logic can remain a local UI adaptation. High-value functions that require sensitive signals should normally become local recommendations, with raw signals kept on device. High-value functions that require population-level learning may use aggregate governance through coarse metrics, differential privacy, or federated updates. Low-value functions with high privacy exposure should be rejected or redesigned.

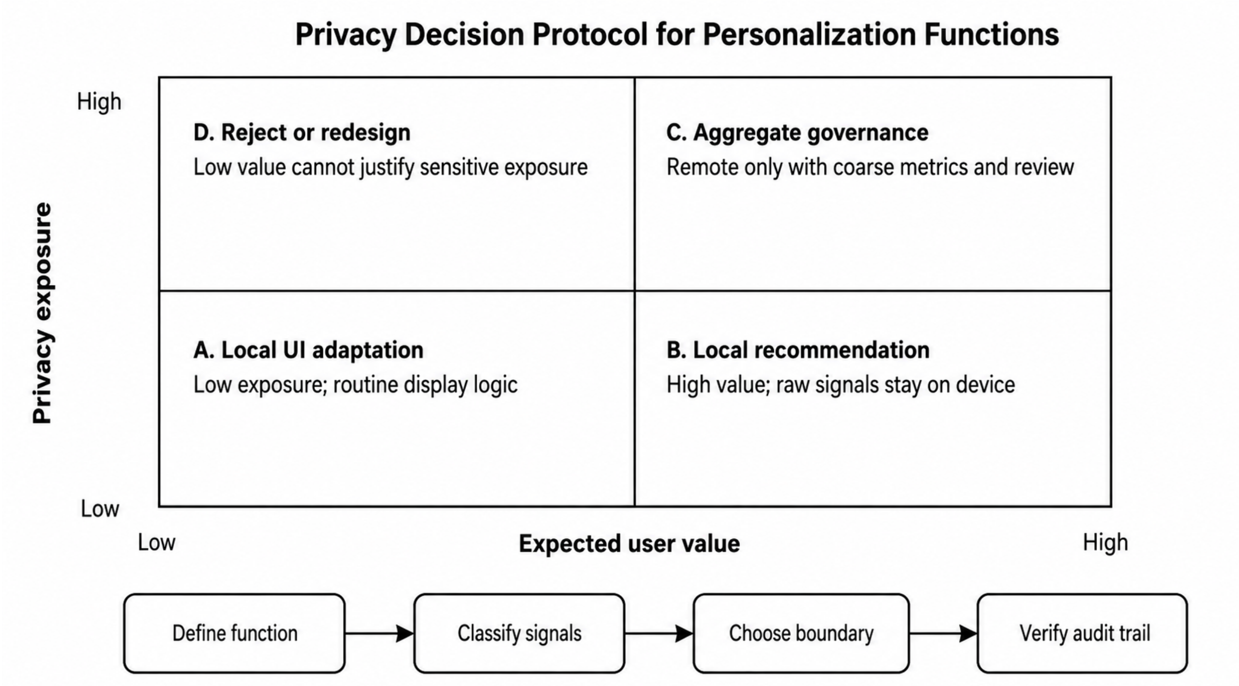

Figure 2. Disclosure-threshold decision protocol for personalization functions.

The protocol is conservative because health and fitness personalization can expose sensitive context through apparently routine inferences. A requirement such as ”make onboarding smarter” is insufficient because it does not specify signal type, recipient, resolution, or retention. A more precise requirement such as ”avoid a hard training reminder after a low-activity day” can often be implemented from a derived local state without uploading timestamped activity history.

4.4. Personalization ladder and control assignment

Table 2 assigns controls to increasingly disclosure-intensive personalization techniques. The ladder is intended to prevent premature use of complex remote methods when simpler local mechanisms are sufficient.

Table 2. Personalization ladder for privacy-preserving iOS health and fitness applications.

Technique	Data boundary	Suitable use	Residual risk	Required control
Local deterministic rules	Raw signals remain on device; rules use derived states.	Goal adjustment, reminder suppression, streak recovery prompts.	Low; rules may still encode sensitive categories if poorly designed.	Document rule purpose; avoid hidden categories; keep logs local.
Compact on-device model	Model inference occurs on device; raw records are not exported.	Adaptive reminder timing, local ranking of tips, interface sequencing.	Medium; model artifacts and features may reveal patterns if synced or logged.	Protect model state; minimize features; disable diagnostic export of feature vectors.
Differentially private aggregate telemetry	Only coarse, noisy aggregate measurements leave the device.	Feature health metrics, population-level failure detection, non-personalized product learning.	Medium; utility depends on schema design and privacy budget.	Set privacy budget; use coarse events; prohibit identifier linkage.
Federated update	Training data remain local; updates are aggregated externally.	Population model improvement where local-only tuning is insufficient.	Medium to high; updates can leak information without secure aggregation and privacy controls.	Use secure aggregation, differential privacy where appropriate, eligibility limits, and update audits.
Cloud personalization	Signals or features are exported to a backend for individualized decisioning.	Only where device execution is infeasible or a legally required service workflow demands it.	High; creates recipient, retention, breach, and secondary-use risks.	Require explicit purpose, minimal fields, contractual controls, retention limits, and security review.

The ladder deliberately places local deterministic rules before local models. For a goal or reminder feature, rules are often easier to inspect, test, and constrain. Federated learning and differential privacy become useful when the team can show that local execution alone cannot answer a legitimate product-learning question. Individualized cloud personalization remains available but should be treated as an exception requiring necessity, proportionality, retention, dependency, and security review.

5. Discussion

LFPE makes privacy-by-design operational by starting with the data boundary. Legal principles do not tell an engineer what to approve in a pull request. A component model can. In the proposed architecture, the team gates access to the source signal, derives only the feature needed for the adaptation, keeps the decision on the device where feasible, and makes dependency behavior auditable. This addresses the implementation gap identified in privacy-engineering literature [17,21].

The most important design question is whether the information flow still fits the relationship between the user and the fitness product. A local activity state used to adjust a goal can remain within that relationship. The same state sent to a marketing endpoint or linked to a persistent identifier changes the context. Recipient, retention, and transmission principle therefore belong in feature design review, not only in a privacy notice.

For multimarket teams, the matrix argues against tuning privacy architecture to the narrowest legal classification. A wellness app may fall outside HIPAA but still handle sensitive consumer health information. A feature may fall outside FDA oversight and still create harmful inferences. An AI component may fall outside high-risk categories and still require transparency and control. A local-first baseline with jurisdiction-specific overlays is more robust than a permissive baseline that changes only when a legal trigger is unavoidable.

For iOS engineering practice, LFPE changes routine artifacts. A feature ticket should state the personalization purpose, signal source, local feature, data boundary, retention rule, and export justification before coding begins. Code review should reject raw health-related signals crossing module or network boundaries without necessity. Release review should reconcile the SDK inventory, HealthKit permission wording, App Privacy Details, privacy manifests, required-reason API declarations, and backend schemas. Disagreement among these artifacts signals an architectural inconsistency.

Technical privacy methods should be used with the same discipline. Differential privacy and federated learning are valuable when configured correctly and used for appropriate goals. They do not justify vague collection. A noisy aggregate may still be illegitimate if it measures behavior unrelated to the user's health and fitness relationship. A federated update may still be risky if cohorts are small, updates are high-dimensional, or eligibility rules expose sensitive states. Architectural minimization should therefore precede cryptographic or statistical mitigation.

6. Limitations and future work

LFPE remains conceptual. It has not been implemented in a production iOS application, and the paper does not measure user acceptance, retention effects, device resource cost, legal compliance, or formal privacy loss. The source base is selective rather than systematic. It draws on influential empirical, theoretical, regulatory, and platform documents, but it does not rank all available evidence or claim exhaustive coverage of privacy-preserving personalization literature.

The framework also does not solve adversarial inference by itself. Keeping raw signals local can reduce disclosure, but unsafe code, weak authentication, insecure storage, malicious dependencies, or poorly designed model updates can still expose sensitive information. Threat modeling, static and dynamic analysis, SDK traffic inspection, secure aggregation review, and privacy-budget accounting remain necessary before deployment.

Future work should build a reference iOS module that implements LFPE for concrete features such as adaptive step goals, reminder suppression, and recovery prompts. The prototype should measure CPU, memory, battery, storage, and latency costs; inspect SDK and network traffic; compare rules with compact on-device models; and test user comprehension of privacy explanations. Where aggregate or federated feedback is justified, future work should evaluate privacy loss, cohort size, eligibility rules, update frequency, and model utility. Legal review should then map the implementation to concrete roles such as controller, processor, covered entity, or business associate.

7. Conclusions

Privacy-preserving personalization in iOS health and fitness applications should not start from the assumption that raw signals must be exported. LFPE proposes the opposite default: raw health-related inputs remain inside the device trust boundary where feasible; local features support rules or compact on-device models; and remote feedback is limited to bounded aggregate, differentially private, or federated mechanisms that pass a disclosure-threshold review.

The practical conclusion is that privacy has to be built into the personalization path itself. EU, U.S., and Apple platform requirements differ, but they can be translated into a shared engineering baseline: define the purpose, reduce signal resolution, prefer local execution, limit recipients, constrain retention, and make dependencies auditable. Jurisdiction-specific obligations can then be layered on top.

The recommended implementation sequence is deliberately conservative. Start with a local rule. Move to a local model only when a rule cannot support the feature. Use aggregate or federated feedback only for product learning that cannot be achieved on the device. Reserve individualized cloud personalization for cases where the team can document necessity and controls. The next step is empirical validation through a working iOS reference implementation with threat modeling, SDK traffic inspection, performance measurement, and user testing.

Highlights

- A local-first envelope is proposed for iOS health and fitness personalization.
- Raw health-related signals remain on device where local adaptation is feasible.
- A disclosure-threshold protocol makes export an exception to be justified.
- EU, U.S., and Apple platform duties are translated into engineering controls.

Declarations

Funding. No external funding was reported for this work.

Declaration of competing interest. The author's industry affiliation is disclosed on the title page. No additional competing interests were reported.

Ethics approval. Not applicable. The manuscript reports a conceptual architecture and does not describe research involving human participants, animals, or identifiable personal data.

Data availability. No datasets were generated or analyzed for this conceptual study.

References

1 A. Papageorgiou, M. Strigkos, E. Politou, E. Alepis, A. Solanas, C. Patsakis, Security and privacy analysis of mobile health applications: the alarming state of practice, IEEE Access 6 (2018) 9390-9403. https://doi.org/10.1109/ACCESS.2018.2799522.

2 G. Tangari, M. Ikram, K. Ijaz, M.A. Kaafar, S. Berkovsky, Mobile health and privacy: cross-sectional study, BMJ 373 (2021) n1248. https://doi.org/10.1136/bmj.n1248.

3 F. Ebrahimi, M. Tushev, A. Mahmoud, Mobile app privacy in software engineering research: a systematic mapping study, Inf. Softw. Technol. 133 (2021) 106466. https://doi.org/10.1016/j.infsof.2020.106466.

4 European Parliament and Council, Regulation (EU) 2016/679 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, Official Journal of the European Union, 2016. https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng.

5 Federal Trade Commission, Health Breach Notification Rule, Federal Register 89 (2024) 47028. https://www.federalregister.gov/documents/2024/05/30/2024-10855/health-breach-notification-rule.

6 Food and Drug Administration, Policy for Device Software Functions and Mobile Medical Applications: Guidance for Industry and Food and Drug Administration Staff, 2022. https://www.fda.gov/regulatory-information/search-fda-guidance-documents/policy-device-software-functions-and-mobile-medical-applications.

7 U.S. Department of Health and Human Services, Resources for mobile health apps developers, 2026. https://www.hhs.gov/hipaa/for-professionals/special-topics/health-apps/index.html (accessed 30 June 2026).

8 Apple Inc., Learn how the Health app and HealthKit protect your privacy, Apple Privacy White Paper, 2023. https://www.apple.com/privacy/docs/Health_Privacy_White_Paper_May_2023.pdf.

9 Apple Inc., Protecting user privacy, HealthKit documentation, n.d. https://developer.apple.com/documentation/healthkit/protecting-user-privacy (accessed 30 June 2026).

10 Apple Inc., App privacy details on the App Store, Apple Developer, n.d. https://developer.apple.com/app-store/app-privacy-details/ (accessed 30 June 2026).

11 Apple Inc., Privacy manifest files, Apple Developer Documentation, n.d. https://developer.apple.com/documentation/bundleresources/privacy-manifest-files (accessed 30 June 2026).

12 Apple Inc., Describing use of required reason API, Apple Developer Documentation, n.d. https://developer.apple.com/documentation/bundleresources/describing-use-of-required-reason-api (accessed 30 June 2026).

13 T. Schroeder, M. Haug, H. Gewald, Data privacy concerns using mHealth apps and smart speakers: comparative interview study among mature adults, JMIR Form. Res. 6 (2022) e28025. https://doi.org/10.2196/28025.

14 A. Acquisti, L. Brandimarte, G. Loewenstein, Privacy and human behavior in the age of information, Science 347 (2015) 509-514. https://doi.org/10.1126/science.aaa1465.

15 R. Rezaee, M. Khashayar, S. Saeedinezhad, M. Nasiri, S. Zare, Critical criteria and countermeasures for mobile health developers to ensure mobile health privacy and security: mixed methods study, JMIR mHealth uHealth 11 (2023) e39055. https://doi.org/10.2196/39055.

16 A. Cavoukian, Privacy by Design: The 7 Foundational Principles, Information and Privacy Commissioner of Ontario, 2011. https://www.ipc.on.ca/en/media/1826/download?attachment=.

17 European Union Agency for Cybersecurity, Privacy and data protection by design: from policy to engineering, ENISA, 2014. https://www.enisa.europa.eu/publications/privacy-and-data-protection-by-design.

18 European Union Agency for Cybersecurity, Engineering personal data sharing, ENISA, 2023. https://www.enisa.europa.eu/publications/engineering-personal-data-sharing.

19 H. Nissenbaum, Privacy as contextual integrity, Wash. Law Rev. 79 (2004) 119-158. https://digitalcommons.law.uw.edu/wlr/vol79/iss1/10/.

20 National Institute of Standards and Technology, NIST Privacy Framework: A Tool for Improving Privacy Through Enterprise Risk Management, Version 1.0, 2020. https://www.nist.gov/privacy-framework.

21 K. Bednar, S. Spiekermann, M. Langheinrich, Engineering privacy by design: are engineers ready to live up to the challenge? The Information Society 35 (2019) 122-142. https://doi.org/10.1080/01972243.2019.1583296.

22 H.B. McMahan, E. Moore, D. Ramage, S. Hampson, B. Aguera y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Proceedings of AISTATS 2017, PMLR 54, 2017, pp. 1273-1282. https://proceedings.mlr.press/v54/mcmahan17a.html.

23 K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, D. Ramage, Federated evaluation of on-device personalization, arXiv:1910.10252, 2019. https://arxiv.org/abs/1910.10252.

24 C. Dwork, A. Roth, The algorithmic foundations of differential privacy, Found. Trends Theor. Comput. Sci. 9 (2014) 211-407. https://doi.org/10.1561/0400000042.

25 European Parliament and Council, Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence, Official Journal of the European Union, 2024. https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng.

26 J. Shuren, B. Patel, S. Gottlieb, FDA regulation of mobile medical apps, JAMA 320 (2018) 337-338. https://doi.org/10.1001/jama.2018.8832.

27 Federal Trade Commission, Mobile Health Apps Interactive Tool, n.d. https://www.ftc.gov/business-guidance/resources/mobile-health-apps-interactive-tool (accessed 30 June 2026).

28 Apple Inc., Reminder: privacy requirement for app submissions starts May 1, Apple Developer News, 26 April 2024. https://developer.apple.com/news/?id=pvszzano.