RivalStats

Executive Feasibility Snapshot

Assessment of Firebase-backed account architecture, cross-device rivalry history, and the transition from deferred online scope to justified product core

Product concept

RivalStats represents the broader online direction that was deliberately rejected in narrower products such as FlipStats and WorldFrame. Instead of adding cloud scope to a single-game utility, this concept makes persistent online history the core product value: tracking games, friends, sessions, rankings, and head-to-head performance over time.
The current iOS codebase combines SwiftUI and SwiftData with Firebase Authentication and Cloud Firestore collections for per-user friends, games, and sessions.

Executive decision

Proceed with Firebase-backed account architecture, per-user cloud persistence, and local on-device caching. Keep public social features and live collaborative match rooms out of the initial release to preserve a controlled v1 scope. Reason: RivalStats gains real value from persistent identity, reusable friend
libraries, cross-game history, and long-term rivalry analysis.

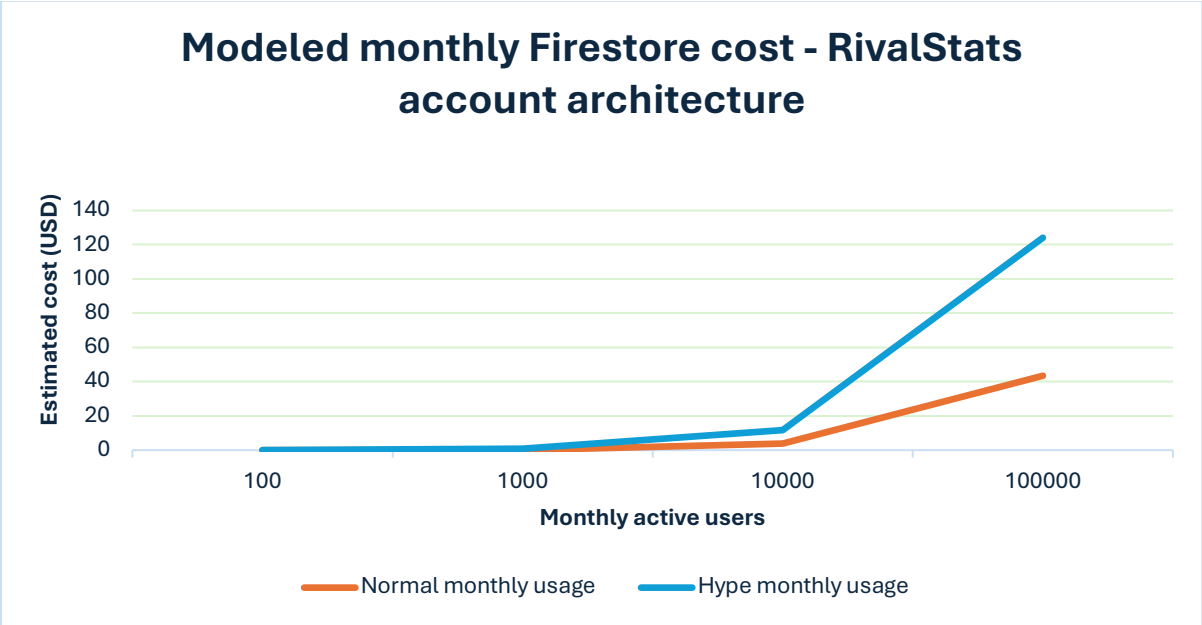
Local-only tracker
Single-device history and no sign-in. Assessment: faster to ship, but underpowered for the broader rivalry concept.

Firebase account architecture
Email sign-in, cloud-backed libraries, SwiftData caching, and cumulative rivalry analytics. Assessment: right-sized online scope for v1.

Upper stress test
$124.11 modeled monthly Firestore cost at 100,000 monthly active users in the hype case.

Cost-growth view

Current Firestore benchmark shown below; the chart models the monthly document-operation cost of RivalStats' account-based sync layer under normal and heavier usage patterns. Read volume grows mainly with collection size and how often users reopen the app.



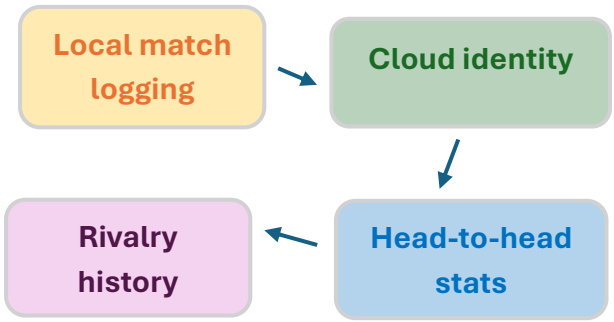
Note: management illustration based on simplified Cloud Firestore operations. The model excludes storage, bandwidth, negotiated discounts, Firebase Authentication overage beyond the no-cost tier, and unusual listener churn.

Benchmark model and management interpretation

Illustrative assumptions

- Signed-in single-user model; no shared public profiles or live multi-user rooms in v1.
- Normal case: 12 active app days per month; about 15 friends, 8 games, and 90 synced sessions; about 18 new sessions logged.
- Hype case: 20 active app days; about 24 friends, 10 games, and 160 synced sessions; about 40 new sessions logged.
- Benchmark uses current public Firestore document pricing plus free-tier quotas of 50k reads/day, 20k writes/day, and 20k deletes/day.

Why online scope fits



Cloud sync keeps friend libraries, game definitions, and cumulative sessions stable across devices. It also powers rankings and head-to-head summaries without requiring live multiplayer infrastructure. Here, Firebase supports the core product.

Decision framework

Monthly cost benchmark, scope-risk interpretation, and management recommendation.

Normal benchmark

Monthly active users	Firestore
100	$0
1.000	$0
10.000	$3.80
100.000	$43.47

Hype benchmark

Monthly active users	Firestore
100	$0
1.000	$0.76
10.000	$11.61
100.000	$124.11

Annualized Firestore view at 100,000 monthly active users: about $521.64 per year in the normal benchmark and about $1,489.32 per year in the hype benchmark, before storage and transfer costs. Firebase Authentication is not included in the table.

Risk and management interpretation

Risk area	Management interpretation
Product scope inflation	Keep v1 centered on private account data, friend and game libraries, sessions, rankings, and head-to-head stats. Result: clearer identity and controlled scope.
Read-volume growth from history sync	Use lean documents, practical retention, and SwiftData caching for responsive UI. Result: useful history with manageable cloud cost.
Auth + privacy responsibility	Use Firebase Auth, per-user partitioning, account deletion, and clear privacy handling. Result: lower privacy and data-governance risk.
Future social / public expansion	Defer public leaderboards, invites, and live multi-user features. Result: the first release keeps the right online depth.

Management recommendation

RivalStats should proceed as an account-based iOS app built on SwiftUI, SwiftData, Firebase Authentication, and Cloud Firestore. In this product, online scope is justified because persistent identity, reusable friend and game libraries, and long-term rivalry history are core to the concept rather than optional decoration.
The recommended posture is cloud-backed private stats first, with broader social, sharing, or live network features deferred until user demand and product maturity support them.