Casha App: Remote API Sync Strategy

Final Sync Strategy for Casha App

ADD TRANSACTION FLOW (AI-DRIVEN)

---

1. User types natural language message.

2. Send message to Remote API (AI processes it).

3. Receive a structured TransactionModel from API.

4. Save transaction to CoreData.

5. UI reloads from CoreData (reactive update).

Pros:

- CoreData remains source of truth.

- Ensures accurate AI parsing before saving.

- Clean separation of concerns.

LIST TRANSACTIONS FLOW (LOCAL-FIRST + SYNC)

---

1. Fetch and display from CoreData immediately.

2. In the background, fetch latest from Remote API.

3. Compare (diff) remote vs local.

4. Merge new/updated transactions into CoreData.

5. UI updates automatically via Published.

Pros:

- Fast initial load.

- Offline support.

- Eventually consistent with remote.

Sync Logic (Example):

For each remote transaction:

- If not in CoreData -> insert.

- If exists -> update if changed.

Sync Triggers:

- On app launch.

- After adding new transaction.

- Optionally: periodic sync (e.g. background task).

Final Decision:

Add Transaction -> Remote First (AI -> CoreData -> UI)

List Transactions -> CoreData First (UI -> Background Sync with Remote)

This hybrid approach gives best UX and consistency.