Naa Khata

Logo

THE MVP PRODUCT MODULES

1. Shop & Profile Management (The Onboarding)

Keep this as minimal as possible. Shopkeepers are busy; they don't want to fill out long forms.

- **Mobile-First Login:** OTP-based login (no passwords to remember).
- **Shop Identity:** Simple fields for Shop Name, Owner Name, and Business Type (e.g., Kirana, Medical, Dairy).
- **Business Card Generator:** A simple auto-generated digital card the owner can share on WhatsApp.

2. Customer Management (The "Khata" List)

The heart of the app. It should behave like a contact list.

- **Contact Sync:** One-tap to import customers from the phone's contact list.
- **Customer Profiles:** Name, phone number, and a "Total Balance" visible at a glance.
- **Customer Groups:** (Optional but helpful) Categorize customers (e.g., "Regulars," "Local Area," "Office Staff").

3. Core Ledger Module (Udhaar & Jama)

This replaces the pen-and-paper entries. Avoid accounting jargon like "Debit" or "Credit"; use "You Gave" (Udhaar) and "You Got" (Jama).

- **Entry Screen:** Big buttons for **"Gave" (Red)** and **"Got" (Green)**.
- **Notes & Attachments:** Allow users to type what was bought (e.g., "1kg Sugar") or take a photo of a physical bill/chit.
- **Date Selection:** Default to today, but easy to back-date an entry.
- **Auto-Calculation:** Instant update of the "Total Due" whenever a transaction is saved.

4. Reminders & Notifications

Since you want no complex payments, the focus should be on **Communication**.

- **WhatsApp/SMS Alerts:** Send a free or automated message to the customer whenever an entry is made.
- **Payment Reminders:** A "Remind" button that sends a pre-filled WhatsApp message: *"Namaste, your balance at [Shop Name] is ₹[Amount]. Please clear it at your convenience."*
- **Due Date Alerts:** Allow the shopkeeper to set a "Promise Date" for payment.

5. Reports & Summaries

Simple insights to help the owner see how the business is doing.

- **Daily Summary:** Total collections vs. total credit given today.
- **PDF Reports:** Generate a PDF ledger for a specific customer to share with them if they dispute the balance.
- **Monthly View:** Total outstanding (how much money is "stuck" in the market).

6. Safety & Backup

For a shopkeeper, their data is their money. If they lose their phone, they shouldn't lose their records.

- **Auto-Cloud Backup:** Sync data instantly to the cloud so it can be restored on a new device.
- **Offline Mode:** Crucial for shops with poor internet. Allow entries to be made offline and sync later.
- **App Lock:** Simple PIN or Fingerprint lock to keep the "Khata" private from others.

Suggested MVP Tech Features

To keep it **"Naa Katha" (My Story/My Account)** branded and localized:

- **Multi-language Support:** At minimum, English and your local regional language (e.g., Telugu/Hindi).
- **Voice-to-Text:** Allow the owner to speak the item names (e.g., "two liters of oil") instead of typing them.

Screens That We Need To Design

🧭 NK MVP UI PRINCIPLES

- **Calm UI:** Off-white backgrounds (`#F9FAFB`), deep charcoal text (`#1F2937`). No neon colors.
- **High Contrast:** Red (`#DC2626`) for *Gave* (Udhaar) and Green (`#16A34A`) for *Got* (Jama).
- **Touch-First:** All buttons are at least 48px in height.
- **One Question:** Each screen does exactly one thing (e.g., "Add Money" or "Search Name").

📱 SCREEN LIST & UI STRUCTURE

1. The Gateway (Auth & Setup)

- **Screen 1.1: Mobile Login**
  - *UI:* Large numeric input field + "Request OTP" button.
  - *PrimeReact Component:* `InputText` (Keyfilter for numbers) + `Button` .
- **Screen 1.2: OTP Verification**
  - *UI:* 4-box digit entry + "Resend" text link.
- **Screen 1.3: Shop Onboarding**
  - *UI:* Vertical list of 3 fields: Shop Name, Your Name, Language Selection (Dropdown).

2. The Command Center (Main Dashboard)

The "Home" screen where the shopkeeper sees the status of their business at a glance.

- **Header:** Shop Name + Profile Icon.
- **Balance Summary Card:** * Two distinct zones: **You will get** (Green) and **You will give** (Red).

- **Customer Search:** A persistent search bar with a "Add New" shortcut.
- **Customer List:** Rows showing Name, Last Transaction Date, and Balance.
  - *PrimeReact Component:* `DataTable` (Web) or `FlatList` (Mobile) with custom templates.

3. The Personal Ledger (Customer Detail)

This is the digital version of a single page in a notebook.

- **Top Bar:** Customer Name, Phone Number, Call Action, WhatsApp Action.
- **The Timeline:** A vertical feed of entries sorted by date (Newest at bottom).
  - *UI:* "Gave" entries are right-aligned cards; "Got" entries are left-aligned cards.
- **Sticky Footer Action:** Two large, equally weighted buttons:
  - `[ - GAVE ]` (Red) | `[ + GOT ]` (Green)

4. The Entry Pad (Transaction Screen)

Triggered when the user taps "Gave" or "Got."

- **Header:** "Adding Entry for [Name]"
- **Large Amount Input:** Massive font size for the amount.
- **Remarks Field:** Simple text area (e.g., "3kg Rice").
- **Date Selector:** Default "Today" chip; tap to open calendar.
- **Attachment:** "Add Photo" button (PrimeReact `FileUpload` / React Native `ImagePicker`).
- **Save Button:** Full-width bottom button.

5. The Output (Reports & Settings)

- **Reports Tab:** * "Daily Summary" (Total In/Out today).
  - "Monthly PDF Report" (Generate and Share).
- **Settings Tab:**
  - App Lock (Toggle).
  - Language Change.
  - "Backup Successful" timestamp.

Jira Epics & User Stories for Naa Khata (NK) MVP

Epic 1: Fast Onboarding (Identity)

Goal: Zero friction. The shopkeeper should be "App-ready" in 3 taps.

- **NK-1: OTP Login:** As a shop owner, I want to log in with my phone number and OTP so I don't have to manage a password.
- **NK-2: Shop Identity:** As a shop owner, I want to save my Shop Name so it appears on the reports I send to customers.
- **NK-3: Secure Access:** As a shop owner, I want to set a simple 4-digit PIN to lock the app from unauthorized customers or kids.

Epic 2: Customer Directory (The Book)

Goal: A digital version of the "Names" written on the cover of a physical book.

- **NK-4: Add Customer:** As a shop owner, I want to add a customer by Name and Phone (or via Phone Contacts) to start their ledger.
- **NK-5: Live Search:** As a shop owner, I want to search for a customer by name so I can find their ledger page instantly during a rush.
- **NK-6: Balance Overview:** As a shop owner, I want to see a list of all customers with their current "Net Balance" highlighted in color (Red for Due, Green for Settled).

Epic 3: Ledger Entries (Gave/Got)

Goal: This is the core engine. It must be as fast as writing with a pen.

- **NK-7: Add "Gave" (Credit):** As a shop owner, I want to record an amount I gave on credit with an optional note (e.g., "Soap, Milk") so I remember why they owe me.
- **NK-8: Add "Got" (Payment):** As a shop owner, I want to record a payment received from a customer to automatically reduce their outstanding balance.
- **NK-9: Transaction History:** As a shop owner, I want to see a scrollable timeline of all "Gave/Got" entries for a specific customer to resolve any disputes.
- **NK-10: Entry Correction:** As a shop owner, I want to edit or delete a wrong entry so that my accounts remain accurate.

Epic 4: Settlement & Reliability (Outcomes)

Goal: Making sure the shopkeeper gets their money back and never loses data.

- **NK-11: WhatsApp Reminder:** As a shop owner, I want to send a pre-filled WhatsApp message to a customer with their total balance and a "Pay Now" request.
- **NK-12: PDF Statement:** As a shop owner, I want to generate a simple PDF of a customer's ledger to share it as "proof" of transactions.
- **NK-13: Auto-Cloud Backup:** As a shop owner, I want my data to sync automatically to my phone number account so I don't lose my "Khata" if I lose my phone.

Execution Plan for Naa Khata (NK)

1. Module: Onboarding & Authentication

Goal: Securely identify the shopkeeper and create their digital shop profile with zero friction.

Backend Tasks (Spring Boot + JPA)

- **Database Setup:** Create `User` entity with fields: `id`, `mobileNumber`, `shopName`, `ownerName`, `pinHash`, and `isVerified`.
- **OTP Service:** Integrate a mock service (for dev) and a real SMS Gateway (for prod). Create an endpoint `POST /api/auth/send-otp`.
- **JWT Security:** Implement Spring Security with JWT. Create `POST /api/auth/verify-otp` which returns a Bearer token.
- **Profile API:** Create `PUT /api/user/profile` to update Shop Name and Category.

Frontend Tasks (NextJS / React Native)

- **Auth State:** Set up **Zustand** store (`useAuthStore`) to persist the JWT and User Profile.
- **Login UI:** Create a minimal screen with a large numeric input for the mobile number.
- **OTP View:** Create a 4-digit auto-focus input component using **Zod** for validation.
- **Shop Setup Form:** A simple **Shadcn** (Web) or **NativeWind** (Mobile) form to collect Shop Name.

Output: A registered user who is logged in and redirected to an empty Dashboard.

---

2. Module: Customer Management

Goal: Allow the shopkeeper to build their "address book" of debtors.

Backend Tasks (Spring Boot + JPA)

- **Customer Entity:** Fields: `id`, `shopkeeperId` (FK), `name`, `mobile`, `totalBalance` (BigDecimal).
- **Repository:** Create `CustomerRepository` with a custom query `findByShopkeeperIdOrderByLastTransactionDesc`.
- **Endpoints:** * `POST /api/customers` (Create)
  - `GET /api/customers` (List with search filter)

Frontend Tasks (NextJS / React Native)

- **Contact Integration:** (Mobile) Use `react-native-contacts` to fetch and filter phone contacts.
- **Search Bar:** A real-time filter using **React Query** to fetch customers as the user types.
- **Customer Card UI:** Design a clean row showing Name, Mobile, and a "Balance Summary."

Output: A searchable list of customers that the shopkeeper can tap on to open their specific ledger.

---

3. Module: The Core Ledger (Gave/Got)

Goal: The "Money Engine." Fast entry and real-time balance calculation.

Backend Tasks (Spring Boot + JPA)

- **Transaction Entity:** Fields: `id`, `customerId` (FK), `amount`, `type` (ENUM: GAVE/GOT), `description`, `imagePath`, `createdAt`.
- **Atomic Service Logic:** Create a `@Transactional` service.
  - When a "Gave" entry is added, **add** to Customer's `totalBalance`.
  - When a "Got" entry is added, **subtract** from Customer's `totalBalance`.
- **History API:** `GET /api/customers/{id}/transactions` with pagination.

Frontend Tasks (NextJS / React Native)

- **Transaction Timeline:** A "Chat-style" UI. Left side for "Got" (Green), Right side for "Gave" (Red).
- **Entry Overlay:** A specialized numeric keypad overlay.
  - **Red Theme** for "Gave" (Udhaar).
  - **Green Theme** for "Got" (Jama).
- **State Sync:** Use **React Query** `invalidateQueries` to refresh the customer balance immediately after a successful POST.

Output: The shopkeeper can add a transaction in under 5 seconds and see the balance update instantly.

---

4. Module: Reminders & PDF Reports

Goal: Helping the shopkeeper recover money and provide physical/digital proof.

Backend Tasks (Spring Boot + JPA)

- **PDF Engine:** Implement a service using **iText7** to generate a table-based ledger for a specific `customerId`.
- **Report API:** `GET /api/reports/customer/{id}/download` returning a byte stream (PDF).
- **WhatsApp Utility:** A helper class to format localized strings: *"Namaste [Customer], your balance at [Shop] is [Amount]."*

Frontend Tasks (NextJS / React Native)

- **WhatsApp Trigger:** Use `Linking.openURL('whatsapp://send?phone=...')` to trigger the local WhatsApp app with the pre-filled message.
- **Download Button:** A "Download PDF" button that triggers the browser download (Web) or opens the File Share sheet (Mobile).

Output: A professional PDF statement and a WhatsApp message sent to the customer.

---

5. Module: Cloud Sync & Offline Support

Goal: Data reliability. The shopkeeper never loses their "Katha."

Backend Tasks (Spring Boot + JPA)

- **Sync Logic:** Implement a "Last Updated" timestamp check to resolve conflicts if multiple devices are used.
- **Postgres Backup:** Set up a cron job for automated daily Postgres backups to S3 or local storage.

Frontend Tasks (NextJS / React Native)

- **React Query Persistence:** Use `createSyncStoragePersister` so the app works even when the shopkeeper is in a basement with no signal.
- **Visual Indicator:** A small "Cloud" icon in the header that turns green when data is synced.

Output: Data is safe even if the app is closed or the internet is disconnected.

Summary of Final Output (The "Naa Katha" MVP)

- **Mobile App:** For daily entries, scanning contacts, and sending WhatsApp reminders.
- **Web Dashboard:** For the shopkeeper to view detailed reports on a laptop/tablet at the end of the day.

- **Data Integrity:** A PostgreSQL 18 database managed by a high-performance Spring Boot API.

Technology Stack

API - Spring Boot - JPA

IDE : VSCode

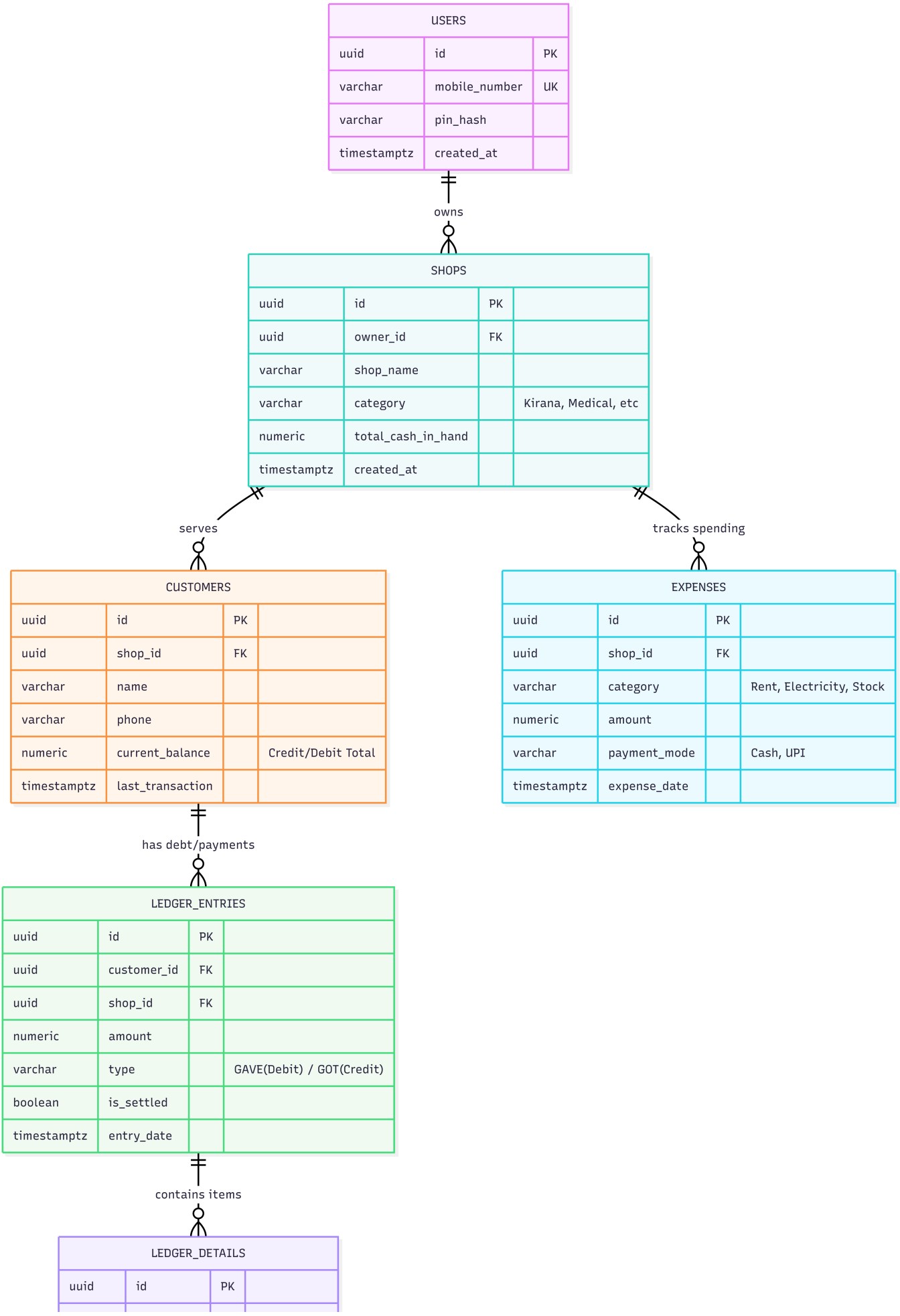
Web Application UI - NextJS — React + React Query (Tanstack) + ZOD + Zustand + Shadcn + tailwind

Mobile App - React Native with React Web Native - mobile apps both Android/iOS

WebSite — UI - Tailwind

Database : Postgress- 18

WebServer : Tomcat 11. (For APIs)


USERS
uuid
id
PK
varchar
mobile_number
UK
varchar
pin_hash
timestamptz
created_at
owns
SHOPS
uuid
id
PK
uuid
owner_id
FK
varchar
shop_name
varchar
category
Kirana, Medical, etc
numeric
total_cash_in_hand
timestamptz
created_at
serves
tracks spending
CUSTOMERS
uuid
id
PK
uuid
shop_id
FK
varchar
name
varchar
phone
numeric
current_balance
Credit/Debit Total
timestamptz
last_transaction
EXPENSES
uuid
id
PK
uuid
shop_id
FK
varchar
category
Rent, Electricity, Stock
numeric
amount
varchar
payment_mode
Cash, UPI
timestamptz
expense_date
has debt/payments
LEDGER_ENTRIES
uuid
id
PK
uuid
customer_id
FK
uuid
shop_id
FK
numeric
amount
varchar
type
GAVE(Debit) / GOT(Credit)
boolean
is_settled
timestamptz
entry_date
contains items
LEDGER_DETAILS
uuid
id
PK

uuid	entry_id	FK	
varchar	item_name		e.g. 2kg Rice
numeric	price		
int	quantity		