Backend Query

CREATE TABLE users (

    id UUID DEFAULT gen\_random\_uuid() PRIMARY KEY,

    username VARCHAR(50) UNIQUE NOT NULL,

    password\_hash TEXT NOT NULL,

    balance DECIMAL(15,2) DEFAULT 0.00,

    created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

    updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Transactions table (updated)

CREATE TABLE transactions (

    id UUID DEFAULT gen\_random\_uuid() PRIMARY KEY,

    user\_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,

    kind VARCHAR(10) NOT NULL CHECK (kind IN ('credit', 'debit')),

    amt DECIMAL(15,2) NOT NULL,

    updated\_bal DECIMAL(15,2) NOT NULL,

    to\_user VARCHAR(50), -- username of recipient for payments

    product\_id UUID, -- reference to product for purchases

    description TEXT,

    created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Products table

CREATE TABLE products (

    id UUID DEFAULT gen\_random\_uuid() PRIMARY KEY,

    name VARCHAR(200) NOT NULL,

    price DECIMAL(15,2) NOT NULL,

    description TEXT,

    created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

    updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Indexes for better performance

CREATE INDEX idx\_users\_username ON users(username);

CREATE INDEX idx\_transactions\_user\_id ON transactions(user\_id);

CREATE INDEX idx\_transactions\_created\_at ON transactions(created\_at);

CREATE INDEX idx\_products\_name ON products(name);

-- Row Level Security (RLS) policies

ALTER TABLE users ENABLE ROW LEVEL SECURITY;

ALTER TABLE transactions ENABLE ROW LEVEL SECURITY;

ALTER TABLE products ENABLE ROW LEVEL SECURITY;

-- Users can only see their own data

CREATE POLICY "Users can view own profile" ON users

    FOR SELECT USING (auth.uid() = id);

CREATE POLICY "Users can update own profile" ON users

    FOR UPDATE USING (auth.uid() = id);

-- Transactions: users can only see their own

CREATE POLICY "Users can view own transactions" ON transactions

    FOR SELECT USING (auth.uid() = user\_id);

CREATE POLICY "Users can insert own transactions" ON transactions

    FOR INSERT WITH CHECK (auth.uid() = user\_id);

-- Products: everyone can read, authenticated users can create

CREATE POLICY "Anyone can view products" ON products

    FOR SELECT TO authenticated, anon USING (true);

CREATE POLICY "Authenticated users can create products" ON products

    FOR INSERT TO authenticated WITH CHECK (true);

-- Function to update updated\_at timestamp

CREATE OR REPLACE FUNCTION update\_updated\_at\_column()

RETURNS TRIGGER AS $$

BEGIN

    NEW.updated\_at = NOW();

    RETURN NEW;

END;

$$ language 'plpgsql';

-- Triggers to automatically update updated\_at

CREATE TRIGGER update\_users\_updated\_at BEFORE UPDATE ON users

    FOR EACH ROW EXECUTE FUNCTION update\_updated\_at\_column();

CREATE TRIGGER update\_products\_updated\_at BEFORE UPDATE ON products

    FOR EACH ROW EXECUTE FUNCTION update\_updated\_at\_column();