



PROJECT TIMELINE (From 12 Nov to 9 Dec 2025)

Week	Dates	Phase	Tasks & Deliverables
Week 1	Nov 12–17	Planning & Setup	<ul style="list-style-type: none">✓ Review requirements (from manuscript)✓ Define user stories & features (mobile + admin)✓ Create system architecture diagram Choose stack (Flutter + Node.js + PostgreSQL + React)✓ Set up GitHub repo & CI/CD structure✓ Prepare UI wireframes using Figma
Week 2	Nov 18–24	UI/UX Design & Database Design	<ul style="list-style-type: none">✓ Build full UI mockups in Figma for all app screens (mobile + admin)✓ Design database schema (ERD diagram) for donors, campaigns, transactions, receipts✓ Set up database on PostgreSQL✓ Set up Flutter & backend environments

Week 3	Nov 25–Dec 1	Core Development Phase I (Mobile App)	<ul style="list-style-type: none"> ✓ Develop onboarding, campaign listing, and campaign details pages ✓ Implement donation flow UI (amount selection, recurrence, payment method) ✓ Integrate M-Pesa Daraja sandbox ✓ Develop basic donor profile management ✓ Connect backend API for campaigns & donors
Week 4	Dec 2–6	Core Development Phase II (Admin Console + Integration)	<ul style="list-style-type: none"> ✓ Build React admin console (campaign management, transaction list, reports) ✓ Add authentication & roles (OAuth2.0/JWT) ✓ Connect donation verification (M-Pesa webhook) ✓ Add receipt generation (PDF) ✓ Finalize backend API routes
Week 5	Dec 7–9	Testing, Deployment & Handover	<ul style="list-style-type: none"> ✓ Conduct end-to-end testing (mobile + admin + payment) ✓ Fix UI & logic bugs ✓ Host backend (AWS/GCP/Azure) ✓ Deploy Flutter app (Play Store) ✓ TestFlight internal release) ✓ Prepare documentation & presentation for the event

PROJECT COMPONENTS

1. Mobile App (Flutter)

- Campaign listing (with progress bars)
- Campaign detail view
- Donation flow (amount, recurrence, method)
- Payment integrations (M-Pesa, Card, Bank)
- Donor receipts (PDF generation)
- Impact stories & notifications
- Donor profile management

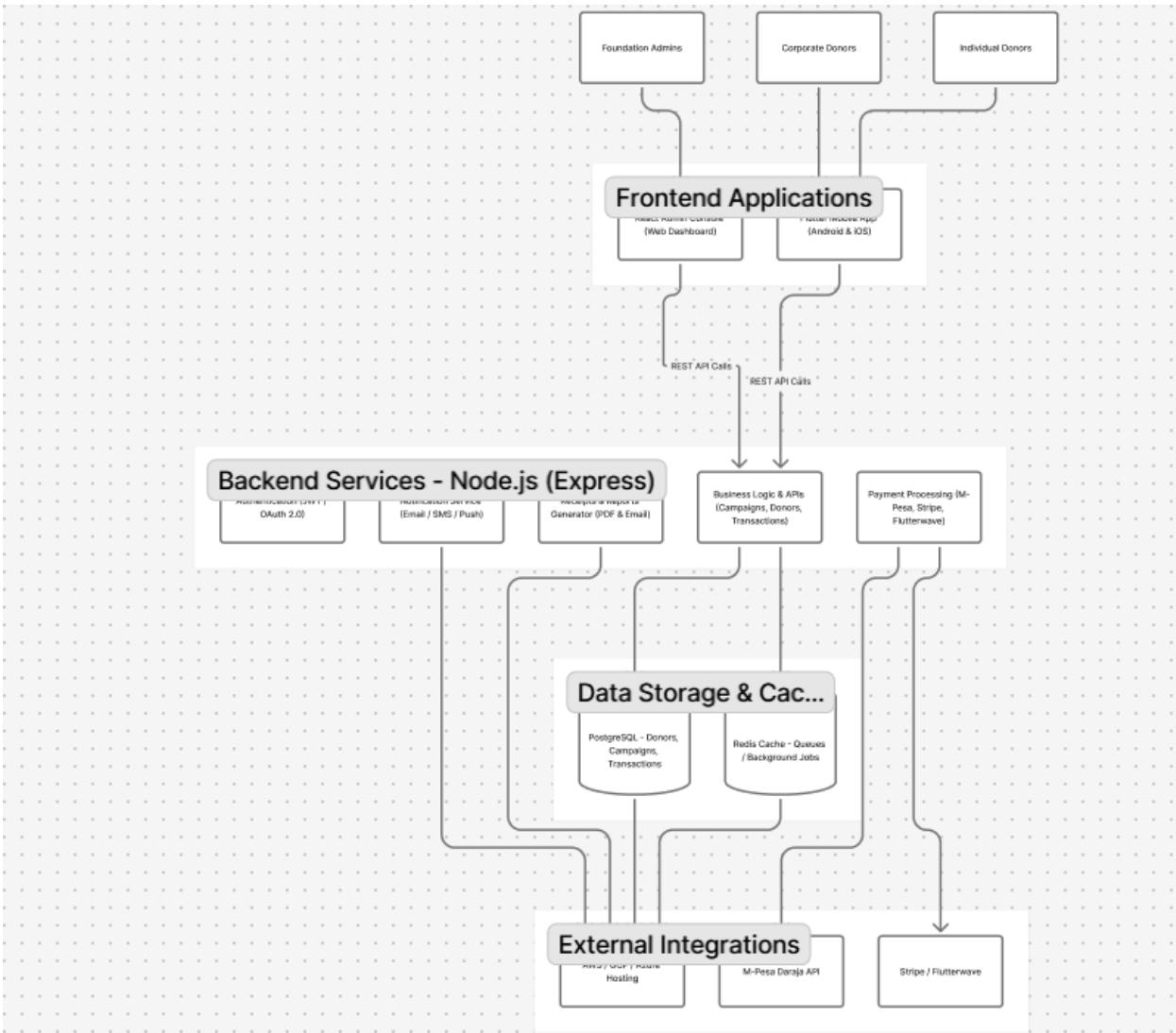
2. Admin Web Console (React + Node.js API)

- Campaign CRUD (Create, Edit, Delete)
- Transaction & Donor Management
- Reports (donations by campaign, donor type, etc.)
- Reconciliation logs
- Communication module (email/SMS notifications)

3. Backend (Node.js + PostgreSQL)

- RESTful API endpoints
- Authentication & roles (Admin, Donor)
- Payment processing via webhooks
- Data models (Campaigns, Donors, Transactions, Receipts)
- Automated receipts & notifications

SYSTEM ARCHITECTURE DIAGRAM



Top Layer — Users

Shapes: Rounded rectangles or stick-figure icons

Labels:

- **Individual Donors (Mobile Users)**
- **Corporate Donors**
- **Foundation Staff / Admins**

These connect **downward** to the mobile app or admin console.

Second Layer — Application Interfaces

Use **rectangles** labeled as:

Left: Mobile App (Flutter)

- Campaigns
- Donations
- Receipts
- Notifications
- Profile & Settings

Arrow: to Backend (Node.js API)

Right: Admin Console (React)

- Manage Campaigns
- View Donors
- Generate Reports
- Reconcile Transactions

Arrow: to Backend (Node.js API)



3. Middle Layer — Backend Services

Shape: Large rectangle labeled **Node.js / Express Backend**

Inside, list submodules:

- Authentication (JWT / OAuth 2.0)
- Payment Processing (M-Pesa API, Card)
- Business Logic
- REST API Endpoints
- Receipt Generator (PDF)
- Notification Service (Email/SMS)

Arrows: from both Frontend apps (Mobile + Admin) into this box.

4. Bottom Layer — Databases & Integrations

Use cylinders (database symbols) and boxes for APIs:

Type	Label	Notes
Database	PostgreSQL	Donors, Campaigns, Transactions, Receipts
Cache/Queue	Redis	For background jobs, confirmations
API Integration	M-Pesa Daraja API	Payment confirmations via webhook
API Integration	Stripe / Flutterwave	Card payments
Storage	AWS S3 / Cloud Storage	For receipts, images

Arrows:

- Node.js → PostgreSQL
- Node.js → Redis
- Node.js ↔ M-Pesa API (two-way)
- Node.js → S3 Storage

5. Hosting Layer

Box at the bottom labeled:

“AWS / GCP / Azure (Deployed Services)”

Include:

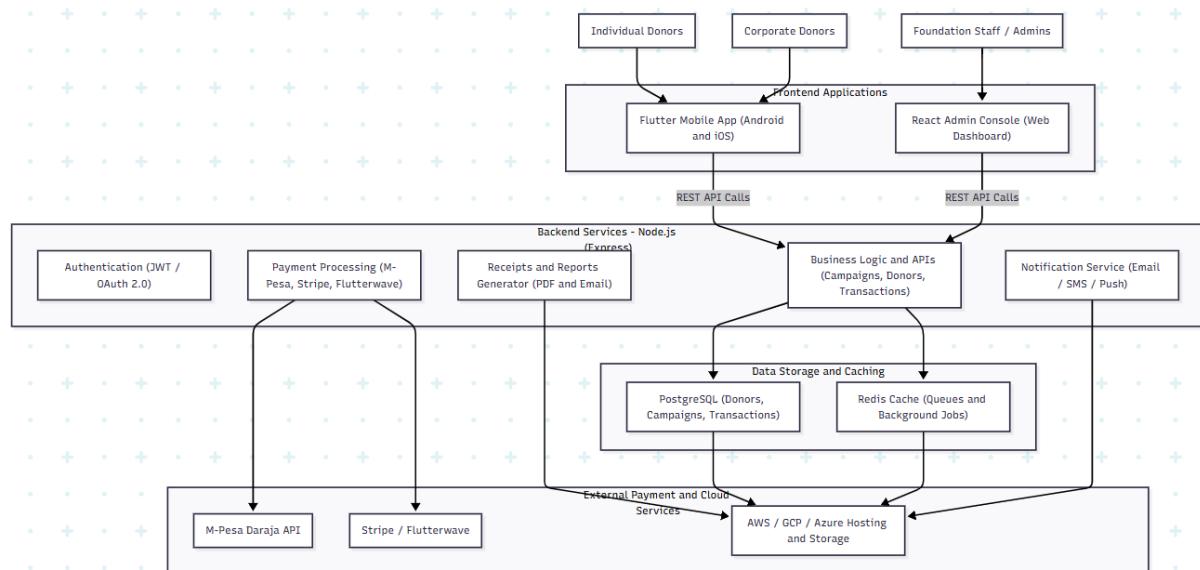
- Backend hosting
- Database instance
- File storage

- Monitoring/logging

6 Optional Extras

Add a dotted arrow for:

- **Analytics Dashboard** (tracking metrics like donors, total donations, recurring users).



DATABASE STRUCTURE (Simplified ERD)

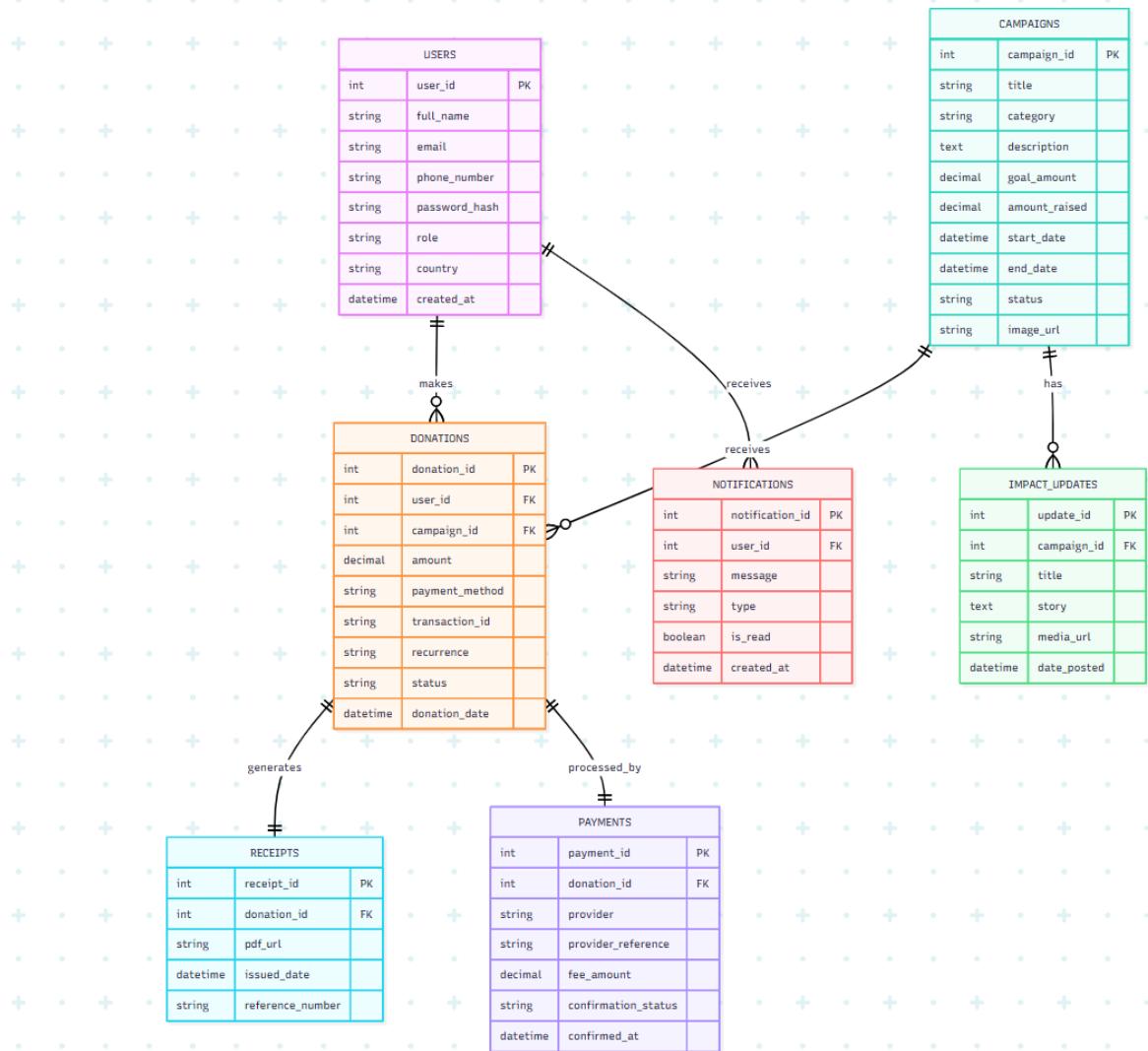
Explanation of Tables

Table	Purpose	Key Fields
-------	---------	------------

USERS	Stores donor and admin account data	user_id, role, email, phone_number
CAMPAIGNS	Defines fundraising campaigns	title, category, goal_amount, status
DONATIONS	Logs each donation event	donation_id, user_id, campaign_id, amount, payment_method
RECEIPTS	Stores generated donor receipts	receipt_id, donation_id, pdf_url, issued_date
PAYMENTS	Tracks payment confirmations from gateways	payment_id, provider, provider_reference, confirmation_status
IMPACT_UPDATES	Holds stories or milestones for campaigns	update_id, campaign_id, story, media_url
NOTIFICATIONS	Sends messages or reminders to donors	notification_id, user_id, message, is_read

Relationships Summary

- **One user** → can make **many donations**
- **Each donation** → belongs to **one campaign**
- **Each donation** → generates **one receipt**
- **Each donation** → has **one payment record** (via M-Pesa, Stripe, etc.)
- **One campaign** → can have many **impact updates**
- **One user** → can have many **notifications**



Visual Hierarchy Layout (Simplified)

busy easy

What This Diagram Shows

This beautiful visual hierarchy clearly demonstrates:

1. User Layer

- Individual Donors

- Corporate Donors
- Foundation Staff/Admins

2. Application Layer

- Flutter Mobile App (for donors)
- React Admin Console (for staff)

3. Backend Layer

- Node.js API handling authentication, payments, receipts, notifications

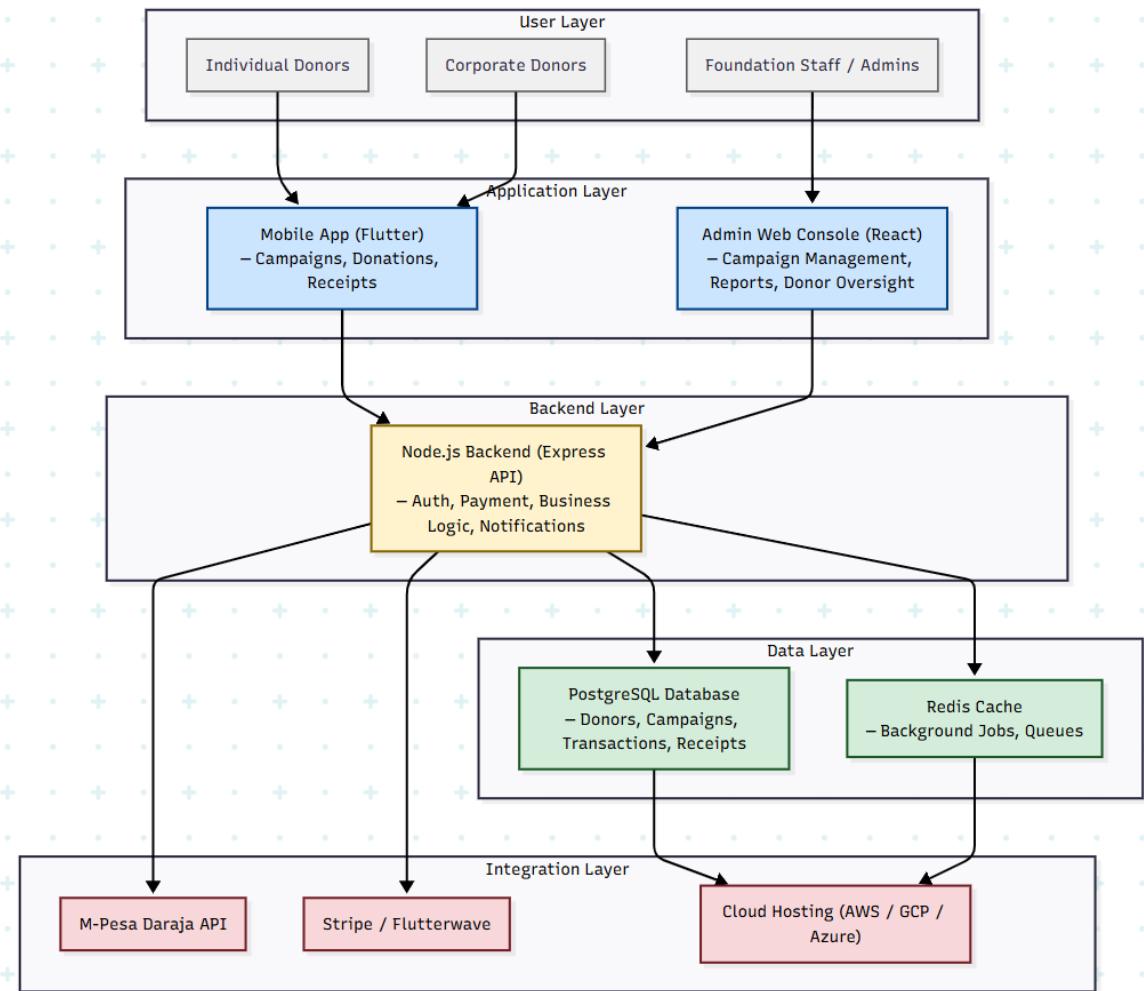
4. Data Layer

- PostgreSQL for core data
- Redis for caching and background tasks

5. Integration Layer

- M-Pesa, Stripe/Flutterwave for payments
- Cloud hosting for deployment

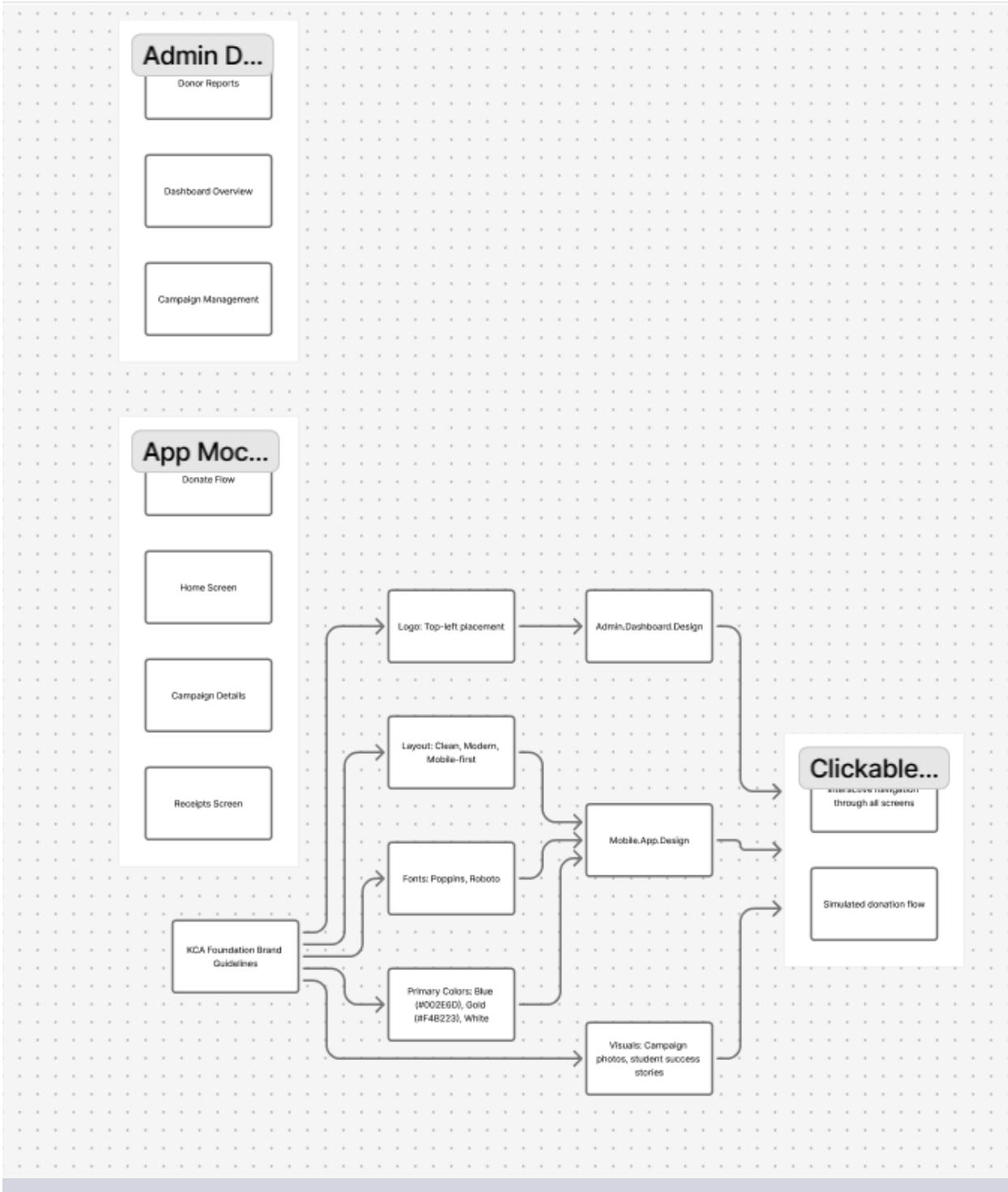


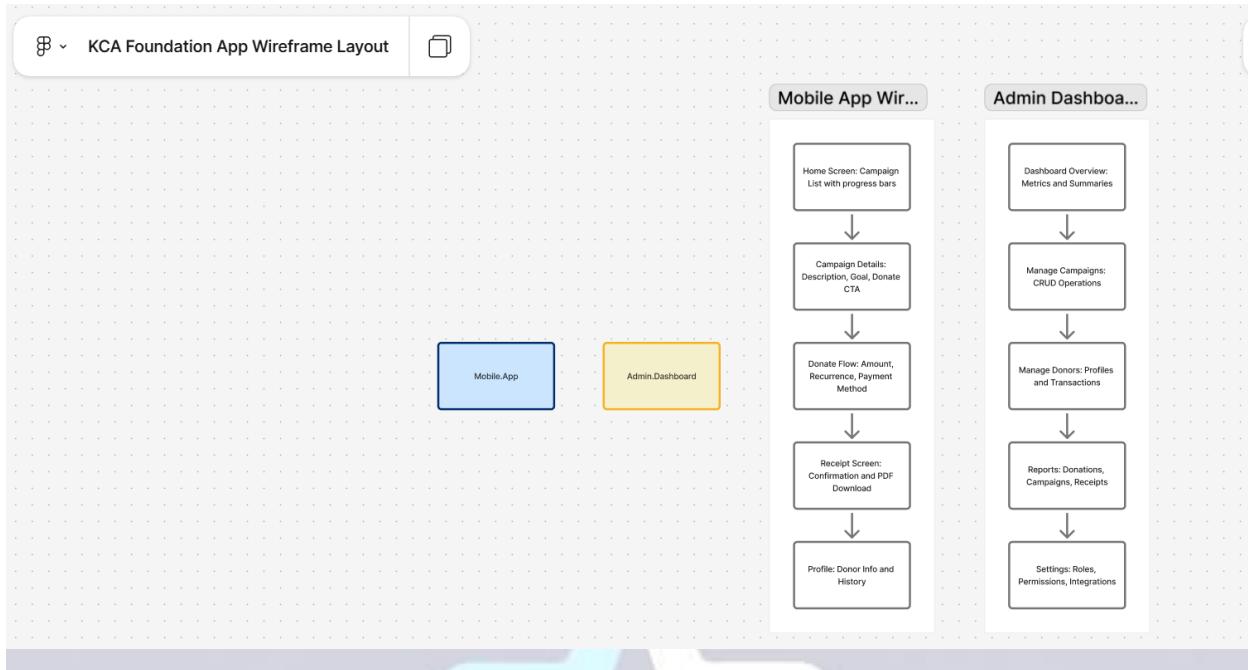


DESIGN REQUIREMENTS

Created in **Figma**:

- App Mockups (Home, Campaign Details, Donate Flow, Receipts)
- Admin Dashboard Design
- Prototype (clickable flow for presentation)





DELIVERABLE CHECKLIST

Deliverable	Status	Tool
Requirement breakdown	✓	Manuscript
System Architecture Diagram	<input type="checkbox"/>	Draw.io / Lucidchart
Database ERD	<input type="checkbox"/>	dbdiagram.io
Figma UI Designs	<input type="checkbox"/>	Figma
Flutter App (Android + iOS)	<input type="checkbox"/>	Flutter SDK
Admin Dashboard	<input type="checkbox"/>	React
Backend API	<input type="checkbox"/>	Node.js + Express
Payment Integration (M-Pesa Sandbox)	<input type="checkbox"/>	Safaricom Daraja
Deployment (Cloud + Domain)	<input type="checkbox"/>	AWS / Firebase
User Testing & Documentation	<input type="checkbox"/>	Word / PDF

Prepared by

Otieno Joshua Onyango

NovaJOO

