WombTo18 Foundation

Complete Technical Architecture & Development Plan

Prepared for: WombTo18 Foundation (Section 8 Company)

Prepared by: Rohith Yerramsetti

Date: 12 February 2026

Version: 1.0

---

1. Executive Summary

WombTo18 Foundation needs a digital platform that goes beyond a brochure website and functions as a **fund management and donor engagement system** aligned with **Indian Section 8 and 80G/12A** requirements.The proposed solution delivers:

- A modern website with core sections: **Home (Hero), About Us, Services, Blog, Press, Impact Reports, Donation**.
- A **trusted donation experience** with Razorpay integration, donor privacy controls, and automated tax certificates.
- A **transparency model** where every rupee received and utilized is visible per program.
- **Automation** for:
- Instant 80G/12A certificate generation.
- Weekly progress reports to donors until each program is closed.
- A secure, scalable architecture with separate environments for **development, testing/staging, and production**.

This document summarizes the **architecture, data model, automation workflows, tech stack, environments, and implementation roadmap** in a way that is understandable to non-technical leadership while still strong enough for a technical interview.

---

2. Goals & Key Requirements

2.1 Functional Goals

- **Public Website**
- Home page with a strong **Hero** section.

- About Us, Services, Blog, Press, Impact Reports.
- Clear calls-to-action and donation buttons connected to specific programs.
- **Donation System**
- Capture donor **name, email, mobile number**.
- Donor chooses whether their name is shown publicly or masked.
- Integrated with **Razorpay** for Indian payments (UPI, cards, net banking).
- **Immediate 80G and 12A certificate** after successful payment.
- **Transparency & Reporting**
- Public **transparency dashboard** per program:
- Funds received.
- Funds utilized.
- Remaining balance.
- **Donor wall** displaying:
- Real name if consented.
- System-generated donor ID if not.
- Weekly progress reports emailed to donors **every 7 days** until the program is closed.

2.2 Non-Functional Goals

- **Security & Compliance**
- Section 8, 80G, 12A-aligned record-keeping.
- Secure handling of personal data and payments.
- **Scalability**
- Able to grow from hundreds to thousands of donors and multiple concurrent programs.
- **Maintainability**
- Clean separation between website, business logic, and data.
- Easy to extend with future features (CSR portal, recurring donations, mobile app).

---

3. High-Level System Architecture

The platform is organized into three main layers:

3.1 Presentation Layer (What Users See)

- **Public Website**
- Home (Hero), About Us, Services, Blog, Press, Impact Reports.
- Program pages with details, stories, and “Donate Now” buttons.
- SEO-friendly and fully responsive (desktop and mobile).
- **Donor Experience**
- Donation form (name, email, mobile, program, amount, consent choice).
- Thank-you and confirmation pages.
- Donor wall and transparency views.
- **Admin Panel**
- Secure login for admins, staff, and auditors.
- Manage programs, donors, donations, fund utilizations.
- View transparency dashboards and export reports.

3.2 Application Layer (Business Logic & Workflows)

- **APIs** to:
- Power the website (program data, donor wall, transparency stats).
- Drive admin features (program CRUD, utilizations, reporting).
- **Business Logic** for:
- Validating and recording donations.
- Syncing with Razorpay and verifying payment authenticity.
- Creating **immutable ledger entries** for every credit and debit.
- Generating certificates and emailing donors.
- Running **weekly reporting** jobs.
- **Background Workers**
- Handle heavy or time-consuming tasks:
- PDF generation (certificates and weekly reports).

- Sending emails.
- Weekly cron jobs for donor updates.

3.3 Data & Integration Layer

- **Primary Database**
- Stores donors, donations, programs, fund ledger entries, certificates, utilizations, donor reports, and donor wall entries.
- Designed for strong consistency and auditability.
- **File Storage**
- Cloud storage for:
- Certificates (PDF).
- Weekly reports (PDF).
- Impact reports and utilization proofs (images, documents).
- **External Integrations**
- **Razorpay** for payments.
- **Email service** (SES/SendGrid-type).
- Optional SMS/WhatsApp integration for critical alerts.

---

4. Conceptual Data Model (Non-Technical View)

The core entities and relationships:

- **Users**
- Admins, staff, and auditors who log into the admin panel.
- Used for access control and audit trails (who did what, when).
- **Donors**
- Name, email, mobile, optional PAN.
- **Public consent flag** indicating if their real name can be shown.
- **Masked donor ID** (e.g., “WTB-DN-2024-0001”) for privacy when consent is not given.
- Total donated amount and history of donations.

- **Programs**
- Each initiative (e.g., “Early Childhood Nutrition 2025”).
- Contains target amount, raised amount, utilized amount, description, dates, and status (draft, active, completed, archived).
- **Donations**
- Each payment from a donor to a specific program.
- Links to donor and program.
- Stores amount, status (pending, completed, failed, refunded), Razorpay references, and transaction date.
- **Fund Ledger (Transparency Core)**
- **Journal of all financial moves**:
- Credits = donations.
- Debits = utilizations.
- Tracks running balance per program.
- **Immutable**: entries are never edited or deleted—only new entries added.
- **Certificates**
- Each certificate linked to a single donation and donor.
- Stores certificate number (e.g., “80G/2024-25/0001”), financial year, issue date, and file link.
- **Utilizations**
- Records how funds are spent:
- Amount, category, description, date, and approvals.
- Linked to program and optionally to uploaded receipts/proofs.
- **Donor Reports**
- Weekly or final progress reports.
- Include which week, funds utilized so far, milestones, and file link.
- **Donor Wall Entries**
- Public recognition entries.

- For each relevant donation: display name (real or masked), amount, optional message, visibility flag.

This structure supports clear answers to questions like:

- “How much has Program A received, how much has been used, and what’s left?”
- “Show every in-and-out movement for this program in order.”
- “Show a donor’s full history and certificates.”

---

5. Key Automation Workflows

5.1 Donation & Payment Flow

1. Donor selects a program and amount, fills name, email, mobile, and consent choice.
2. System creates a **pending donation record** and initiates a Razorpay order.
3. Donor completes payment through Razorpay.
4. On success:

- Payment is **verified** (signature and webhook checks).
- Donation is marked as **completed**.
- A **fund ledger credit entry** is created for the program.
- The program’s raised amount is updated.

5. Then, automatically:

- Certificate generation job is queued.
- Donor wall entry is created using:
- Real name (if consented), or
- Masked donor ID (if not).
- Thank-you email and certificate email are scheduled.

5.2 Automatic 80G/12A Certificate Generation

For each successful donation:

1. Pull donor, donation, program, and legal registration data.
2. Generate a **formatted certificate number** (e.g., “80G/2024-25/0005”).

3. Create a **professional PDF certificate** including:

- WombTo18 branding and legal lines.
- Donor name, amount, date, program.
- Certificate number, financial year.
- Optional verification URL or QR code.

4. Save the PDF in secure storage and link it in the system.
5. Email the donor with:

- Thank-you message.
- Certificate attachment or link.

5.3 Weekly Donor Reports (Every 7 Days)

1. A scheduled job runs weekly (e.g., every Sunday 10:00 AM).
2. For each completed donation in a still-active program:

- Calculate how many weeks have passed since the donation.
- Check if a report for this week number already exists.

3. If not, compile:

- Utilizations and milestones from the last 7 days for that program.
- Funds used so far and current balance.

4. Generate a **concise report**:

- As a PDF or formatted email.
- Personalized with donor name, program, and week number.

5. Upload/save the report and email it to the donor.
6. Once a program is marked **completed**, send a final summary report (optional) and stop further weekly emails.

5.4 Donor Privacy & Donor Wall Logic

- During donation, donor opts:
- “Show my name publicly” or
- “Keep me anonymous (show ID only)”.

- If **consent = true**:
- Donor wall displays real name.
- If **consent = false**:
- Donor wall displays masked donor ID (e.g., “WTB-DN-2024-0007”).
- If consent changes later:
- Admin updates donor record.
- All donor wall entries for that donor update automatically.

---

6. Technology Stack (High-Level)

6.1 Frontend (Website & Donor UI)

- **Framework**: Modern JavaScript framework like **Next.js + React**.
- Benefits: Fast, SEO-friendly, supports server-side rendering for better performance and Google ranking.
- **Styling**: Utility-based CSS framework (e.g., Tailwind CSS).
- Benefits: Consistent design system, quick layout building, great mobile support.
- **UI Components**: Reusable component library.
- Benefits: Faster development and consistent user experience.

6.2 Backend (APIs & Business Logic)

- **Language & Framework**: Type-safe JavaScript backend (e.g., **Node.js with TypeScript**, Express/NestJS).
- Benefits: Shared language between frontend and backend, strong ecosystem, good for async I/O.
- **APIs**: RESTful endpoints for:
- Public site (programs, transparency, donor wall).
- Admin panel (management and reporting).
- **Background Workers & Queues**:
- To handle:
- Certificate PDF generation.

- Weekly reports.
- Email sending.
- Ledger updates.
- Using a queue system with retries to handle temporary failures.

6.3 Data & Storage

- **Primary Database**: Relational DB like **PostgreSQL**.
- Benefits: ACID compliance, ideal for financial data, strong query capabilities.
- **Cache / Queue**: In-memory store like **Redis**.
- Benefits: Faster reads for certain data, reliable task queues for background jobs.
- **File Storage**: Cloud object storage (e.g., **AWS S3** or equivalent).
- For certificates, weekly reports, impact documents, images.

6.4 Integrations & DevOps

- **Payment Gateway**: **Razorpay** for Indian context (UPI, cards, net banking).
- **Email Service**: Transactional email provider (e.g., AWS SES, SendGrid).
- **CDN & Security**: Cloudflare-type service for SSL, caching, and DDoS protection.
- **Monitoring & Logging**: Error tracking (e.g., Sentry) and log aggregation (e.g., Logtail/DataDog).

---

7. Environment Strategy & Deployment

7.1 Environments

- **Development Environment**
- Used by developers for building features.
- Local/test databases and Razorpay in **test mode**.
- Fast feedback, hot-reload, and detailed logging.
- **Testing / Staging Environment**
- Mirrors production as closely as possible.
- Separate staging database and storage.

- Razorpay sandbox keys.
- Used for QA testing, stakeholder demos, and **User Acceptance Testing**.
- **Production Environment**
- Live environment serving real donors at wombto18.org.
- Razorpay live keys, production database, full security hardening.
- Strict access controls and monitoring.

7.2 Deployment & CI/CD (Explained Simply)

- Developers push changes to a central code repository (e.g., GitHub).
- Automated pipeline:
- Runs tests and basic security checks.
- Builds the application and, if successful, deploys to **staging**.
- After manual sign-off on staging:
- The same version is deployed to **production**.
- If something breaks:
- Deployment can quickly roll back to the previous stable version.

---

8. Testing, Launch, and Post-Launch

8.1 Testing Phase

- **Functional Testing**
- Full donation flow (success and failure cases).
- Certificate generation and email delivery.
- Donor wall behavior (with and without consent).
- Weekly report scheduling and sending.
- All key admin actions (program creation, utilization logging, transparency views).
- **Security Testing**
- Confirm HTTPS everywhere.
- Test login security, role-based access, and payment webhook validation.

- Ensure fund ledger behaves as write-only, preserving auditability.
- **Performance Testing**
- Simulate traffic spikes (e.g., post-press article or campaign).
- Verify system responsiveness and stability under load.
- **User Testing**
- Non-technical staff and sample donors test:
- Donation experience.
- Reading impact reports.
- Using the admin dashboard.

8.2 Production Go-Live

- **Pre-Launch**
- Verify legal texts (80G/12A lines, privacy policy, terms of service).
- Configure domain, SSL, email domains, Razorpay live keys.
- Seed real programs and images.
- **Launch Day**
- Deploy latest approved version to production.
- Run **smoke tests**:
- Make a small real donation.
- Check certificate email.
- Verify donor wall and transparency dashboard.
- **First 1–2 Weeks**
- Closely monitor:
- Payment success rates.
- Error logs.
- Email delivery metrics.
- Quickly address any issues or UX pain points.
- **Ongoing**

- Regular backups and security updates.
- Monthly review of:
- Donation funnel metrics.
- Transparency content freshness.
- Donor engagement metrics.

---

9. Security, Compliance, and Transparency

- **Security**
- Enforced SSL (HTTPS) for all users.
- Secure admin logins with strong passwords and possible 2FA.
- Limited access to production data for authorized staff only.
- **Compliance**
- Structured storage of donations and certificates for at least **7 years**.
- Clear data retention and privacy policies.
- Audit logs for critical actions (e.g., utilization approvals, data exports).
- **Transparency**
- Fund ledger ensures **every rupee is traceable**.
- Public dashboards show:
- Funds received per program.
- Funds utilized.
- Remaining balance.
- Donor wall respects privacy while still recognizing support.

---

10. Implementation Roadmap (Weeks 1–10)

- **Phase 1 (Weeks 1–3): Foundation**
- Set up codebase, database schema, and basic admin.
- Build core website pages (Home, About, Services, Blog, Press, Impact Reports – static).

- Integrate Razorpay in test mode and store donations.
- **Phase 2 (Weeks 4–6): Core Features & Transparency**
- Implement fund ledger, certificate generation pipeline, and email sending.
- Implement donor wall with consent handling.
- Add program management and utilization recording in admin.
- Create basic transparency dashboards.
- **Phase 3 (Weeks 7–8): Automation & Optimization**
- Build weekly report job and templates.
- Optimize performance (caching, indexing).
- Add monitoring and alerting.
- **Phase 4 (Weeks 9–10): Polish & Launch**
- UI/UX refinements, mobile optimization, accessibility.
- Full testing in staging, security checks, and documentation.
- Production deployment and early monitoring.

---

11. Success Metrics & KPIs

- **Technical/Platform Health**
- Uptime: **99.5%**+.
- Average page load time: under **2 seconds** for most users.
- Payment success rate: above **95%**.
- **Donor Engagement**
- Donation conversion rate from visitors.
- Average donation amount.
- Repeat donor rate and donor retention.
- Weekly report email open and click-through rates.
- **Trust & Transparency**
- Number of donors accessing transparency pages.

- Positive feedback from donors and auditors on clarity of reports.
- Reduced manual effort for staff in preparing reports and certificates.

---

12. Conclusion

The proposed architecture gives WombTo18 Foundation a **production-grade, donor-centric, and transparency-first platform** that:

- Clearly shows **where donor money goes**, per program and over time.
- **Automates compliance** with immediate 80G/12A certificates and structured records.
- Keeps donors engaged through **weekly impact updates**.
- Respects donor **privacy choices** while recognizing support through the donor wall.
- Is built on a **scalable, modern tech stack** with safe deployment and testing practices.

This approach allows WombTo18 to launch a strong **Version 1** in about **10 weeks**,
then iteratively add advanced features (CSR reporting, recurring donations, mobile app, AI-based insights) as the foundation grows.