Bizflow SME Nigeria - Updated Roadmap for Manus AI

Date: June 30, 2025, 11:59 AM WAT

Purpose: This plan updates the Bizflow MVP for Nigerian SMEs, addressing Lovable gaps, aligning with the provided analysis, and incorporating new withdrawal details, guiding Manus AI to implement changes, push to GitHub, and deploy on Lovable.

Project Overview

Bizflow is an affordable, locally relevant business management tool for Nigerian SMEs, targeting 40 million SMEs with a goal of 400,000–2 million users. It competes with QuickBooks, Wave, and DUMO. The MVP will validate demand with a Free Plan, upsell Silver, and tease Gold with a referral system. No offline/mobile yet, but placeholders exist for future AI (voice, AdSense).

Updated Requirements

- **Mobile-Responsiveness**: Work on all devices, small screens (⩾320px) with fluid typography (rem/em) and media queries.
- **Deployment**: Use Lovable platform (no Vercel or .env).
- **Backend**: Supabase with PostgreSQL for users, invoices, expenses, clients, salespeople, referrals.
- **Payments**: Paystack free tier for Silver subscriptions/invoice payments.
- **Local Relevance**: Naira formatting (e.g., ₦1,200), Pidgin English placeholders (e.g., "Choose plan wey de okay for you business").
- **Security**: JWT and RLS, plan NDPR compliance later.
- **Real Data**: Test live, no mock data.
- **Clean Code**: Avoid tight coupling, use separation of concerns.
- **Performance**: Lazy loading (React.lazy/Suspense), code splitting.

Development Steps

1. Project Setup

- **Environment Setup**:
- Pull Lovable repo (e.g., `git clone https://github.com/<username>/bizflow-sme-nigeria.git`).
- **Database Setup (Supabase on Lovable)**:
- Tables (update existing):
  - `users`: id, email (nullable), phone (unique), role ('Owner', 'Salesperson'), subscription_tier ('Free', 'Silver', 'Gold'), referral_code (unique, nullable), active (boolean, default true).
  - `invoices`: id, user_id, client_id (nullable), amount, status ('Pending', 'Paid', 'Overdue'), due_date.
  - `expenses`: id, user_id, category, amount, date.
  - `clients`: id, user_id, name, phone (nullable), email (nullable).

    - `salespeople`: id, user_id, name, email, active (boolean, default true).
    - `referrals`: id, referrer_id, referred_id, reward_amount (nullable), status ('Pending', 'Claimed'), withdrawal_amount (nullable), withdrawal_status ('Pending', 'Processed').
- RLS:
    - `salespeople`: Owners only insert/update/delete.
    - `invoices`, `expenses`: Salespeople insert/update, owners delete.
- **Lovable Config**: Use existing Supabase credentials.

2. Frontend Development

- **Responsive Design**:
- Use Tailwind CSS with fluid typography (e.g., `text-2xl md:text-3xl`) and media queries (e.g., `@media (max-width: 768px)`).
- **Pages and Components**:
- **Landing Page (/)**:
    - Hero: "Your Business, Simplified," "Get Started Free" button to /register.
    - Header: "Choose plan wey de okay for you business."
- **Register Page (/register)**:
    - Form: phone (required), email (optional), password, submit button, link to /login.
- **Login Page (/login)**:
    - Form: phone/email, password, submit button, forgot password link (OTP reset).
- **Dashboard (/dashboard)**:
    - Header: "Bizflow Dashboard," nav (Home, Profile, Logout).
    - Grid:
    - Welcome card with AdSense placeholder (300x250px, "Ads by Google – Coming Soon").
    - Quick Actions: "Create Invoice Quick Quick," "Check Expenses," "Manage Clients," "Get Referral Link."

  - Subscription Status: Show tier (Free/Silver), "Upgrade Plan" button to /pricing, voice placeholder ("Voice Commands – Coming Soon" for Silver).
  - Analytics: Revenue, Expenses, Pending Invoices (Recharts with lazy loading).
  - Referral Earnings: Display total earnings, withdrawal option.
  - Gold Tease: "Unlock Gold Premium for ₦6,000/month – Coming Soon."
- **Pricing Page (/pricing)**:
  - Plans:
  - Free: 5 invoices, 10 expenses, basic dashboard, referral access.
  - Silver: ₦1,400/week, ₦4,500/month, ₦50,000/year, unlimited features, client/salesperson management, Paystack.
  - Gold Tease: "Unlock Gold Premium for ₦6,000/month – Coming Soon."
- **Client Management (/clients)**:
  - List clients with edit/delete buttons (Silver only), show history.
- **Team Management (/team)**:
  - Add salespeople (name, email), toggle activation, remove option (Silver only).
- **Referral Management (/referrals)**:
  - Display referral link, earnings history, withdrawal request form (minimum ₦3,000, 15% fee).
- **Navbar**:
  - Hide "Login"/"Register" when authenticated, show "Logout" (call `signOut`, navigate to /login).
- **Styling**:
- Tailwind: `bg-gradient-to-br from-green-50 to-blue-50` (cards), `bg-gradient-to-r from-green-600 to-blue-500` (buttons), Naira text with `Intl.NumberFormat`.

3. Backend Development

- **Authentication (Supabase)**:

- Register/login with phone/email, JWT security.
- Forgot password with phone OTP (fix email delivery).
- Roles: 'Owner' (creates salespeople), 'Salesperson' (create invoices/expenses, no delete).
- **API Endpoints (Supabase Functions)**:
- `POST /users`: Register, assign 'Owner' role.
- `GET /users/{id}`: Profile.
- `PUT /users/{id}`: Update role.
- `POST /invoices`: Create (salespeople/owners).
- `GET /invoices?user_id={id}`: List.
- `PUT /invoices/{id}`: Edit (salespeople/owners).
- `DELETE /invoices/{id}`: Only owners.
- `POST /expenses`: Create (salespeople/owners).
- `GET /expenses?user_id={id}`: List.
- `PUT /expenses/{id}`: Edit (salespeople/owners).
- `DELETE /expenses/{id}`: Only owners.
- `POST /clients`: Create.
- `GET /clients?user_id={id}`: List.
- `PUT /clients/{id}`: Edit.
- `DELETE /clients/{id}`: Delete.
- `POST /salespeople`: Create (owners only, Silver).
- `PUT /salespeople/{id}`: Toggle active (owners only).
- `POST /referrals`: Create link (no limit).
- `GET /referrals?referrer_id={id}`: Status.
- `PUT /referrals/{id}`: Update reward (₦500 for monthly), process withdrawal (minimum ₦3,000, 15% fee).
- Secure with JWT and RLS.
- **Supabase Functions**:

- Deploy `reset-password` function for OTP delivery.
- Deploy `referral-reward` function for reward calculation.
- Deploy `process-withdrawal` function to handle withdrawal requests with fee logic.

4. Core Business Management Tools

- **Invoicing**:
- Create/view/edit/list with manual + dropdown client, ₦ amount, due date, notes.
- Free: 5/month, basic templates.
- Silver: Unlimited, customizable (logo, colors).
- Salespeople create, owners delete.
- Status tracking (Pending, Paid, Overdue).
- **Expense Tracking**:
- Record/view with category, amount, date, notes.
- Free: 10/month, basic categories.
- Silver: Unlimited, advanced categories, summaries.
- Salespeople create, owners delete.
- Reporting (period/category).
- **Client Management (Silver)**:
- Store name, phone, email, history.
- Edit/delete buttons, salespeople assign invoices.
- **Dashboard**:
- Revenue, expenses, recent data (Recharts with lazy loading).
- Free: Basic metrics.
- Silver: Trends, top clients.

5. Payment Processing

- **Types**: Cash, bank transfer, credit (with name).
- **Paystack (Silver)**:

- Subscriptions: ₦1,400/week, ₦4,500/month, ₦50,000/year.
- Invoice payments.
- Use Paystack JS for initialization/verification.

6. Referral System

- **Mechanics**: Generate links via dashboard, track sign-ups/upgrades.
- **Rewards**: ₦500 for monthly Silver, none for weekly or yearly.
- **Abuse Prevention**: Phone/IP verification, no limit.
- **UI**: Link, earnings history, withdrawal request form.
- **Withdrawal**: Minimum ₦3,000, 15% system fee deducted, process via `process-withdrawal` function.

7. Subscription Plans

- **Free**: 5 invoices, 10 expenses, basic dashboard, referral access.
- **Silver**: ₦1,400/week, ₦4,500/month, ₦50,000/year, unlimited features, client/salesperson management, Paystack.
- **Gold Tease**: "Unlock Gold Premium for ₦6,000/month – Coming Soon."

8. User Experience

- **Interface**: Intuitive, mobile-friendly (Tailwind).
- **Landing**: Sign-up instructions, Free highlights, Silver value, referral promotion, Naira formatting.
- **Accessibility**: Pidgin placeholders (e.g., "Manage clients sharp sharp"), plan for multi-language support.
- **Single-Owner**: One-click workflows.
- **Salespeople**: Team features.

9. Enhancements (Future)

- **Advanced Reporting**: Custom filters, visual charts, export to PDF/CSV.
- **Payment Gateway**: Integrate Paystack for direct invoice payments.

- **Inventory Management**: Track stock levels, product details, sales.
- **Notifications**: Overdue invoices, upcoming payments, events.
- **Granular Permissions**: Roles (e.g., accountant, manager).
- **Offline Capabilities**: Data entry for unreliable Internet.
- **Mobile Apps**: Native iOS/Android versions.
- **Testing**: Unit, integration, end-to-end tests.
- **Error Handling**: Specific messages, user guidance.
- **Performance**: Monitor and optimize data-intensive operations.

10. Security and Compliance

- **JWT and RLS** for auth/permissions.
- **Future NDPR**: Encryption, audit logs.

11. Performance Optimization

- **Lazy Loading**: Use React.lazy and Suspense for dashboard analytics (Recharts).
- **Code Splitting**: Split chunks for pages (e.g., `/dashboard`, `/pricing`).
- **Image Optimization**: Use next/image or lazy-loaded images.

12. Deployment

- **Lovable**:
- Push to GitHub, Lovable handles deployment.
- Test responsiveness (⩾320px).
- **Supabase**:
- Deploy functions with `supabase functions deploy`.

13. Testing and Validation

- Test auth, invoicing, expenses, clients, payments, referrals, salesperson roles, withdrawal functionality live.
- Validate on screens ⩾320px.

- Gather feedback from 10–20 SMEs.

14. GitHub Push

- Commit changes (e.g., "Implement MVP with withdrawal functionality, client/salesperson fixes").
- Push to `main` with token: `git push https://<token>@github.com/<username>/bizflow-sme-nigeria.git main`.

15. Future Roadmap

- **Framework**: Adopt Django for AI integrations.
- **Offline/Mobile**: Future iterations.

Instructions for Manus AI

- Pull repo: `git clone https://github.com/<username>/bizflow-sme-nigeria.git`.
- Follow steps 1–14, using real data.
- Fix Lovable gaps (manual client, client edit/delete, navbar, reset email, salesperson).
- Update pricing (₦1,400/week, ₦4,500/month, ₦50,000/year), no referral limit, ₦500 referrer for monthly.
- Add withdrawal functionality (minimum ₦3,000, 15% fee).
- Push to GitHub with token.
- Deploy Supabase functions (provide credentials).
- Ensure clean code (separation of concerns, no tight coupling).

16. Comparison with Existing Application Functionalities

This section compares the proposed roadmap with the currently implemented functionalities of the BizFlow SME Nigeria application. The goal is to identify areas where the application aligns with the roadmap and areas that require further development or adjustment.

16.1. Project Setup and Backend

Roadmap Points: * **Environment Setup**: Pull Lovable repo. * **Database Setup (Supabase on Lovable)**: Tables (`users`, `invoices`, `expenses`, `clients`, `salespeople`, `referrals`) and RLS policies. * **Lovable Config**: Use existing Supabase credentials.

Current Implementation: * The application is indeed hosted on a GitHub repository, which is compatible with Lovable deployment. The `git clone` operation was successful, confirming the environment setup. * Supabase is actively used as the backend, as evidenced by the `supabase/client.ts` file, which contains the Supabase URL and public key. This aligns with the roadmap's backend choice. * The presence of `invoices`, `expenses`, and `clients` tables is confirmed by their respective React components (`Invoices.tsx`, `Expenses.tsx`, `Clients.tsx`) and the data fetching logic using `supabase.from("...").select("*")`. The `users` table is implicitly used for authentication via `useAuth`. * The `salespeople` table is also present and managed in `TeamManagement.tsx`. The `referrals` table is managed in `Referrals.tsx`. * **RLS Policies**: The roadmap specifies RLS for `salespeople` (owners only insert/update/delete) and for `invoices`, `expenses` (salespeople insert/update, owners delete). While the frontend code attempts to enforce these roles (e.g., only owners can delete invoices), the actual RLS policies are configured directly within the Supabase project. I do not have direct access to view or modify these RLS policies from this environment. It is crucial to ensure that these RLS policies are correctly set up in your Supabase project to enforce the specified access controls and prevent unauthorized data manipulation. Without proper RLS, a salesperson could potentially delete invoices or expenses directly via the Supabase API, bypassing frontend restrictions.

Alignment: High, with a critical note on verifying Supabase RLS configurations externally.

16.2. Frontend Development

Roadmap Points: * **Responsive Design**: Tailwind CSS with fluid typography and media queries. * **Pages and Components**: Landing Page, Register, Login, Dashboard, Pricing, Client Management, Team Management, Referral Management, Navbar. * **Styling**: Specific Tailwind gradients and Naira formatting.

Current Implementation: * **Responsive Design**: The application extensively uses Tailwind CSS classes, including responsive prefixes (e.g., `md:grid-cols-3`), indicating an intention for mobile responsiveness. Fluid typography and media queries are inherent to Tailwind's design philosophy, so it's likely implemented. Visual inspection on different screen sizes would confirm full responsiveness. * **Pages and Components**: All listed pages (`Landing.tsx`, `Register.tsx`, `Login.tsx`, `Dashboard.tsx`, `Pricing.tsx`, `Clients.tsx`, `TeamManagement.tsx`, `Referrals.tsx`) and the `Navbar.tsx` component are present in the codebase. Their basic structures and navigation align with the roadmap. * **Landing Page**: The `Index.tsx` file serves as the landing page, containing elements like a hero section and calls to action that align with the roadmap's description. * **Register/Login**: Forms for phone/email, password, and links are present. OTP reset for forgot password is mentioned in the roadmap but not explicitly implemented in the frontend code I reviewed; it would typically involve a Supabase function or a separate API call. * **Dashboard**: The `Dashboard.tsx` provides an overview with stats, quick actions, subscription status, and analytics (using Recharts). The

AdSense placeholder and voice command placeholder are present. The Gold Tease is also there. * **Pricing Page**: The `Pricing.tsx` file displays the Free, Weekly, Monthly, and Yearly plans with their respective features and prices. The Gold Tease is also present. * **Client Management**: The `Clients.tsx` page allows users to add, edit, and delete clients. The functionality is present and aligns with the roadmap. * **Team Management**: The `TeamManagement.tsx` page allows owners to add, activate/deactivate, and remove salespeople. This aligns with the roadmap. * **Referral Management**: The `Referrals.tsx` page displays the referral link, earnings history, and a withdrawal request form. The minimum withdrawal amount and fee are also implemented. * **Navbar**: The `Navbar.tsx` component correctly hides/shows login/register/logout links based on authentication status. * **Styling**: The application uses the specified Tailwind gradients and the `Intl.NumberFormat` for Naira formatting is likely used where currency is displayed, though I would need to verify this in each component.

Alignment: High.

16.3. Backend Development (Supabase)

Roadmap Points: * **Authentication**: Phone/email login, JWT, roles (Owner, Salesperson). * **API Endpoints (Supabase Functions)**: Endpoints for users, invoices, expenses, clients, salespeople, referrals. * **Supabase Functions**: `reset-password`, `referral-reward`, `process-withdrawal`.

Current Implementation: * **Authentication**: The application uses Supabase for authentication, which inherently uses JWTs. The concept of 'Owner' and 'Salesperson' roles is present in the frontend logic, but as mentioned, the enforcement relies on RLS policies in Supabase. The phone/email login is implemented. * **API Endpoints**: The application doesn't use Supabase Functions for all API endpoints as suggested in the roadmap. Instead, it directly interacts with the Supabase database from the client-side using the `supabase-js` library. This is a valid approach for many operations, but for more complex or sensitive logic (like processing withdrawals or handling referral rewards), Supabase Functions are recommended to avoid exposing business logic to the client-side. The roadmap's suggestion to use Supabase Functions for all endpoints is a good practice for a more secure and scalable architecture. * **Supabase Functions**: The roadmap mentions three specific Supabase Functions: `reset-password`, `referral-reward`, and `process-withdrawal`. I cannot verify if these functions are deployed in your Supabase project. The frontend code for password reset is not explicit, and the referral reward and withdrawal logic is handled on the client-side in `Referrals.tsx`. This is a significant deviation from the roadmap and a potential security risk. For example, a malicious user could try to manipulate the withdrawal request on the client-side before it's sent to the database. Moving this logic to a Supabase Function is highly recommended.

Alignment: Medium. The core backend functionality is there, but the implementation deviates from the recommended use of Supabase Functions for all API endpoints and for sensitive operations like password reset, referral rewards, and withdrawals.

16.4. Core Business Management Tools

Roadmap Points: * **Invoicing**: Create, view, edit, list invoices with client association and status tracking. * **Expense Tracking**: Record, view expenses with categorization and reporting. * **Client Management**: Store client information and history. * **Dashboard**: Display key metrics and analytics.

Current Implementation: * **Invoicing**: The `Invoices.tsx` and `InvoiceForm.tsx` components provide all the specified functionalities. Users can create, view, edit, and list invoices. Invoices are associated with clients, and their status is tracked. * **Expense Tracking**: The application has an expense tracking module, but I haven't reviewed the specific file for it. However, based on the dashboard, it seems to be implemented. * **Client Management**: The `Clients.tsx` component allows users to manage their clients as specified in the roadmap. * **Dashboard**: The `Dashboard.tsx` component displays key metrics like total invoices, pending invoices, total expenses, and total payments. It also includes a chart for financial overview.

Alignment: High.

16.5. Payment Processing

Roadmap Points: * **Types**: Cash, bank transfer, credit. * **Paystack (Silver)**: Subscriptions and invoice payments.

Current Implementation: * The roadmap mentions Paystack for payment processing, but I don't see any direct integration with Paystack in the frontend code I've reviewed. The `handleUpgrade` function in `Pricing.tsx` and `Dashboard.tsx` is a simulation. This is a major gap between the roadmap and the current implementation. To enable actual payments, you will need to integrate the Paystack API.

Alignment: Low. The payment processing functionality is not implemented.

16.6. Referral System

Roadmap Points: * **Mechanics**: Generate links, track sign-ups/upgrades. * **Rewards**: ₦500 for monthly Silver. * **Abuse Prevention**: Phone/IP verification. * **UI**: Link, earnings history, withdrawal form. * **Withdrawal**: Minimum ₦3,000, 15% fee, processed via Supabase Function.

Current Implementation: * The `Referrals.tsx` component provides the UI for the referral system, including the referral link, earnings history, and a withdrawal form. The withdrawal logic is implemented on the client-side, which is not recommended. The reward mechanism is also likely handled on the client-side. The abuse prevention mechanisms (phone/IP verification) are not implemented.

Alignment: Medium. The UI is there, but the implementation of the core logic (reward calculation, withdrawal processing, abuse prevention) is either missing or insecure.

16.7. Subscription Plans

Roadmap Points: * **Free**: 5 invoices, 10 expenses, basic dashboard, referral access. * **Silver**: ₦1,400/week, ₦4,500/month, ₦50,000/year, unlimited features, client/salesperson management, Paystack. * **Gold Tease**: "Unlock Gold Premium for ₦6,000/month – Coming Soon."

Current Implementation: * The `Pricing.tsx` page accurately reflects the subscription plans and their features as described in the roadmap.

Alignment: High.

16.8. User Experience

Roadmap Points: * **Interface**: Intuitive, mobile-friendly. * **Landing**: Sign-up instructions, highlights, referral promotion, Naira formatting. * **Accessibility**: Pidgin placeholders. * **Single-Owner/Salespeople**: Workflows for different roles.

Current Implementation: * The application has a clean and intuitive interface. The use of Tailwind CSS suggests it's mobile-friendly. The landing page contains the necessary elements. The Pidgin placeholders are a nice touch for local relevance. The workflows for single-owner and salespeople are implemented in the frontend.

Alignment: High.

16.9. Enhancements (Future)

The roadmap lists several future enhancements like advanced reporting, payment gateway integration, inventory management, notifications, granular permissions, offline capabilities, mobile apps, testing, error handling, and performance optimization. These are not yet implemented, which is expected as they are part of the future roadmap.

Alignment: N/A (Future work).

16.10. Security and Compliance

Roadmap Points: * **JWT and RLS**. * **Future NDPR**.

Current Implementation: * JWT is handled by Supabase. RLS is a critical component that needs to be verified in the Supabase project. NDPR compliance is a future goal.

Alignment: Medium. The foundation for security is there, but the critical RLS policies need to be verified.

16.11. Performance Optimization

Roadmap Points: * **Lazy Loading**: React.lazy and Suspense. * **Code Splitting**. * **Image Optimization**.

Current Implementation: * The roadmap mentions lazy loading for dashboard analytics (Recharts), but I don't see it implemented in `Dashboard.tsx`. Code splitting is a feature of Vite, so it's likely being used. Image optimization is not explicitly implemented.

Alignment: Low. The performance optimization techniques mentioned in the roadmap are not fully implemented.

16.12. Deployment

Roadmap Points: * **Lovable**: Push to GitHub, Lovable handles deployment. * **Supabase**: Deploy functions.

Current Implementation: * The application is set up for deployment on Lovable. The Supabase functions are not yet implemented, so they cannot be deployed.

Alignment: Medium.

16.13. Testing and Validation

Roadmap Points: * Test all features live. * Validate on screens ⩾320px. * Gather feedback from SMEs.

Current Implementation: * This is a process that needs to be carried out after the development is complete.

Alignment: N/A (Process).

17. Supabase Tables and RLS Policies Guidance

Based on the analysis, here is my guidance on Supabase tables and RLS policies to avoid potential conflicts and ensure proper functionality, especially for offline, payment, and referral features.

17.1. Supabase Tables

The table structure defined in the roadmap is a good starting point. Here are some suggestions for improvement:

- **`users` table**: The `email` column should be unique if you want to allow users to log in with either phone or email. The `role` column is good, but you might want to create a separate `roles` table and use a foreign key in the `users` table if you plan to have more roles in the future. This would be more scalable.
- **`invoices` table**: The `client_id` should be a foreign key to the `clients` table. You might also want to add a `created_by` column to track who created the invoice (owner or salesperson).
- **`expenses` table**: Similar to the `invoices` table, a `created_by` column would be useful.
- **`clients` table**: The `user_id` should be a foreign key to the `users` table.
- **`salespeople` table**: The `user_id` should be a foreign key to the `users` table.
- **`referrals` table**: The `referrer_id` and `referred_id` should be foreign keys to the `users` table.
- **Offline Functionality**: To support offline functionality, you would need to use a local database on the client-side (e.g., IndexedDB) and a synchronization mechanism to sync data with Supabase when the user is online. This is a complex feature that requires careful planning. You would need to decide which data needs to be available offline and how to handle conflicts when the same data is modified both offline and online.
- **Payment Structure**: For payment processing with Paystack, you would need to store transaction details. You could create a `transactions` table with columns

like `id`, `user_id`, `invoice_id` (nullable), `subscription_plan` (nullable), `amount`, `status`, `reference`, `gateway`, `created_at`. This would help you track all payments, whether for subscriptions or invoices.

17.2. RLS Policies

As I mentioned earlier, I cannot view or modify your RLS policies. However, based on the roadmap, here are the policies you should have in place:

- **`users` table**: Users should only be able to view and edit their own data.
- **`invoices`, `expenses`, `clients` tables**: Users should only be able to view, create, edit, and delete their own data. Salespeople should be able to create and edit invoices and expenses, but not delete them. Owners should have full control.
- **`salespeople` table**: Only owners should be able to create, edit, and delete salespeople.
- **`referrals` table**: Users should only be able to view their own referral data.

To implement these policies, you would use the `auth.uid()` function in your RLS rules to get the ID of the currently logged-in user.

17.3. What I Need to Proceed

To help you further with the implementation, especially with the backend logic and Supabase integration, I would need the following:

1. **Confirmation of RLS Policies**: Please review your RLS policies in your Supabase project and confirm that they are set up as described above. If you need help with writing the RLS rules, I can provide you with the SQL code.
2. **Paystack API Keys**: To integrate Paystack for payments, I would need your Paystack public and secret keys. Please provide them securely.
3. **Supabase Service Role Key**: To implement the Supabase Functions for sensitive operations like password reset, referral rewards, and withdrawals, I would need your Supabase service role key. This key has admin privileges, so please provide it securely.

Once I have this information, I can proceed with implementing the remaining features and fixing the identified gaps.