Personal Finance Manager – Project Documentation

1️⃣ Project Overview

Name: Personal Finance Manager
Type: Full-stack web application
Frontend: React + Tailwind CSS
Backend: Django Rest Framework
Database: PostgreSQL
Purpose: Enable users to manage finances including accounts, transactions, budgets, recurring bills, and analytics.

2️⃣ Features / Modules

Module	Description
User Authentication	Login, registration, Google OAuth, password reset, email verification
Accounts	CRUD accounts, soft delete / archive
Transactions	CRUD transactions, splits, attachments, soft delete
Budgets	CRUD budgets, monthly limits, rollover
Recurring Bills	CRUD recurring bills, automatic transaction generation
Dashboard	Cards, charts, trends, upcoming bills
Notifications	Real-time notifications for key events
Search	Global search for transactions

3️⃣ Priority / Roadmap

Priority	Feature	Status	Notes

High	Transactions page UI & design	❌	Needs proper styling, responsive layout
High	Transaction splits & attachments	❌	Backend exists; frontend integration / design missing
High	Form validation & clear error messages	❌	Email uniqueness, password mismatch, transaction errors need clear messages
High	Forget password production URL	❌	Currently uses localhost link; must update for deployed app
Medium	Archive / restore buttons in UI	❌	Backend supports soft delete; UI buttons missing
Medium			
Medium	Real-time notifications / toasts	⚠	Backend working; frontend partially implemented (transaction example only)

4️⃣ API Integration Status

Module	API Ready	Notes
Users	✅	Register, login, logout, profile, password reset implemented
Currencies	✅	GET only
Categories	✅	CRUD + archive/restore
Accounts	✅	CRUD + archive/restore
Transactions	✅	CRUD + search + soft delete
Transaction Splits	✅	Backend ready
Attachments	✅	Backend ready
Recurring Bills	✅	CRUD + generate transaction
Budgets	✅	CRUD

Notifications	⚠	Backend ready; frontend partially implemented
Exchange Rates	✅	Read-only
Audit Logs	✅	Read-only

6️⃣ Notes / Challenges

- **UI Styling:** Transactions, splits, attachments need better design.
- **Form Validation:** Email uniqueness, password mismatch, transaction errors need proper frontend display.
- **Forget Password:** Must update link to production; current localhost link only works locally.
- **Archive / Restore UI:** Backend supports soft delete; frontend buttons not implemented.
- **Notifications:** Backend fully functional; frontend shows only transaction toast example.

7️⃣ Future Improvements

- Complete Transactions page UI with full styling and responsive layout.
- Integrate splits and attachments with modals.
- Implement full toast notifications for all events.
- Add Archive / Restore buttons in UI.
- Form validation with clear messages for all inputs.
- Deploy forget password URL to production environment.

💡 **Summary:**

- Backend APIs mostly ready and functional.
- Frontend needs **UI improvements, form validation, notifications, archive buttons**.

- High priority items: **Transactions page UI, splits/attachments, form validation, forget password link**.