Finance Management – Project Documentation

1. Project Overview

The Finance Management Project is a lightweight yet robust API to help users manage personal and small business finances efficiently. It enables tracking of income, expenses, budgets, and financial goals in a secure and organized manner.

2. Key Features

- Create, edit, and delete transactions (income and expense)
- Categorize transactions by type (optional: income/expense categories)
- Track account balances
- Generate simple financial reports
- Set budgets and receive notifications if limits are exceeded
- User authentication and secure access
- Optional tags or notes for transactions

3. Technologies Used

- Backend: Django REST Framework
- Database: PostgreSQL
- Authentication: JWT (JSON Web Token)
- Testing: pytest (each app has its own test suite)
- Deployment (Optional): Docker containers
- Optional Tools: Django Admin for backend management

4. Target Audience

- Individuals managing personal finances
- Freelancers tracking income/expenses
- Small teams or startups managing business finances

5. Project Structure

The project structure is organized using Django apps, each with its own permissions and tests:

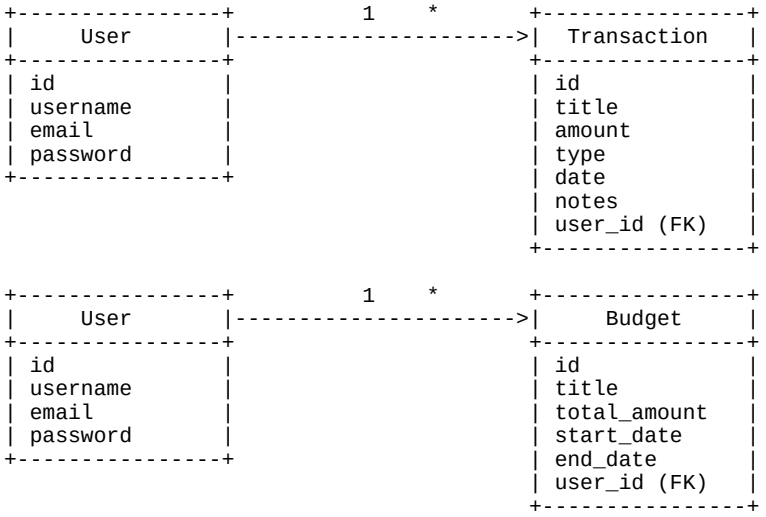
```
finance_management/
■■■ manage.py
■■■ finance_management/
■   ■■■ __init__.py
■   ■■■ settings.py
■   ■■■ urls.py
■   ■■■ asgi.py
■   ■■■ wsgi.py
■
■■■ accounts/
■   ■■■ __init__.py
■   ■■■ admin.py
■   ■■■ apps.py
■   ■■■ managers.py
■   ■■■ migrations/
■   ■   ■■■ __init__.py
■   ■■■ models.py
■   ■■■ permissions.py
■   ■■■ serializers.py
■   ■■■ tests/
```

```
■   ■   ■■■ test_accounts.py
■   ■■■ views.py
■
■■■ transactions/
■   ■■■ __init__.py
■   ■■■ admin.py
■   ■■■ apps.py
■   ■■■ migrations/
■   ■   ■■■ __init__.py
■   ■■■ models.py
■   ■■■ permissions.py
■   ■■■ serializers.py
■   ■■■ tests/
■   ■   ■■■ test_transactions.py
■   ■■■ views.py
■
■■■ budgets/
■   ■■■ __init__.py
■   ■■■ admin.py
■   ■■■ apps.py
■   ■■■ migrations/
■   ■   ■■■ __init__.py
■   ■■■ models.py
■   ■■■ permissions.py
■   ■■■ serializers.py
■   ■■■ tests/
■   ■   ■■■ test_budgets.py
■   ■■■ views.py
■
■■■ requirements.txt
■■■ pytest.ini
■■■ README.md
```

6. Database Models Overview

- User: Custom user model for authentication.
- Transaction: Fields include title, amount, type (Income/Expense), date, notes, user, created_at, updated_at.
- Budget: Fields include title, total_amount, start_date, end_date, user.

7. UML Diagram



8. API Endpoints (Example)

- POST /api/auth/register/ → Register new user

- POST /api/auth/login/ → Login and get JWT token
- GET /api/transactions/ → List transactions
- POST /api/transactions/ → Create new transaction
- PUT /api/transactions// → Update transaction
- DELETE /api/transactions// → Delete transaction
- GET /api/budgets/ → List budgets
- POST /api/budgets/ → Create new budget
- PUT /api/budgets// → Update budget
- DELETE /api/budgets// → Delete budget