**Data layer (expense/data)**

**data\_source/**

* account\_local\_ds: insert/update/archive, list accounts + balance (from view), transfer(tx pair) in txn.
* category\_local\_ds: CRUD, tree fetch (parent/children), by type (EXPENSE/INCOME).
* payee\_local\_ds: CRUD, search/normalize, top payees.
* transaction\_local\_ds: add expense/income/transfer, add splits, list by filters (date/account/category/tag/payee/type), attachments CRUD, tags link.
* exchange\_local\_ds (optional MVP-lite): upsert daily rates, get rate.

**models/**

* DTOs mirroring DB rows: AccountDto, CategoryDto, PayeeDto, TransactionDto, SplitDto, TagDto, AttachmentDto, ExchangeRateDto.
* Simple mappers to domain entities.

**repo/**

* AccountRepoImpl, CategoryRepoImpl, PayeeRepoImpl, TransactionRepoImpl, TagRepoImpl, AttachmentRepoImpl, ExchangeRepoImpl.

**Domain layer (expense/domain)**

**entities/**

* Account, Category(with optional parent), Payee, Money(amount,currency), Transaction(type, splits, tags, payee, note, dates), Tag, Attachment, ExchangeRate, BalanceSnapshot.

**repo/**

* AccountRepo: list/create/update/archive, transfer().
* CategoryRepo: listByType(), tree(), create/update/delete.
* PayeeRepo: search(), create/update/delete, stats().
* TransactionRepo: add/update/delete, addSplit/removeSplit, list(Filter), attach/detach tag & file.
* TagRepo: list/create/delete.
* ExchangeRepo (optional): rateFor(date, base, quote).

**use\_cases/**

* Accounts: GetAccounts, CreateAccount, ArchiveAccount, MakeTransfer.
* Categories: GetCategoriesByType, CreateCategory, UpdateCategory, DeleteCategory.
* Payees: SearchPayees, CreatePayee.
* Transactions: AddExpense, AddIncome, AddTransfer, EditTransaction, DeleteTransaction, GetTransactionsByFilter, AddSplitLine, RemoveSplitLine, AttachFiles, TagTransaction, UntagTransaction.
* Reports (MVP): GetMonthTotals, GetAccountBalances (via running-balance view).
* (Optional) Rates: UpsertRate, ConvertToBase.

**Presentation (expense/presentation)**

**bloc/**

* DashboardBloc: loads month totals, recent txns, account balances.
* TxnListBloc: handles filters, paging, refresh, tag/category/account filters.
* TxnEditBloc: add/edit expense/income/transfer, manage splits, attachments, tags.
* AccountsBloc: list, create/archive, quick transfer action.
* CategoriesBloc: list by type, add/edit/delete, tree maintenance.
* PayeesBloc: search + CRUD.
* TagsBloc: list + CRUD.

**pages/**

* DashboardPage (cards: income/expense/net, balances, recent).
* TransactionsPage (list + filter sheet).
* TransactionEditorPage (type switch, fields, splits, attachments, tags).
* AccountsPage (list + FAB; transfer shortcut).
* CategoriesPage (tabs: Expense/Income; tree view).
* PayeesPage (search + list).
* TagsPage (list + counts).

**widgets/**

* Reusable: MoneyField, CategoryPicker, AccountPicker, PayeeAutocomplete, TagChips, SplitEditor, AttachmentTile, FilterChipsRow, DateRangePicker, BalanceCard, PieByCategory, TrendLine.

**MVP scope (recommend)**

* Must-have: Accounts, Categories, Payees (basic), Transactions (with splits), Tags, Attachments, Dashboard (month totals + balances), Txn list with filters.
* Defer: Budgets, Recurring, Advanced reports, Exchange rates (unless you need base-currency now).

**Cross-cutting ideas**

* **Validation**: domain validators for sign rules (redundant with triggers but gives UX errors early).
* **Filters**: a single TransactionFilter value object (date range, accountIds, categoryIds, tagIds, type, payeeId).
* **Undo ops**: soft-delete in UI (actual delete in DB).
* **Seed data**: default categories/payees for new user.
* **Tests**: in-memory DB for data sources; bloc tests with fake repos.