V1

Nouns & Noun Phrases → Candidate Classes / Value Objects

- Need → Need (interface / abstract)
- Basic need → BasicNeed (leaf in Composite)
- Bundle → Bundle (composite)
- Child part (bundle member) → BundlePart (value object: needName, count)
- Catalog (collection of all needs/bundles) → Catalog
- Log / Ledger / Entry → Ledger, LogEntry
- Funds (daily value) → represented as LogEntry with type FUND
- Goal (daily value) → LogEntry with type GOAL
- Fulfillment (acquire a need) → LogEntry with type NEED (name+count)
- Repository (persistence) → Repository (CSV I/O)
- Controller / App → AppController
- View / Chart / Summary → View, DTOs: DaySummary, SliceBreakdown
- Money/Date → Money (value object), LocalDate (JDK)
- Synonyms normalized: “log/ledger” → Ledger
- “funds/goal entries” → LogEntry typed
- “need/bundle” → Need with BasicNeed/Bundle

Adjectives & Adverbs → Attributes

- Need “fixed / variable / fees / total” →
    - BasicNeed: name: String, fixed: Money, variable: Money, fees: Money
    - Derived total = fixed + variable + fees
- Bundle “composed of… counts” →
    - Bundle: name: String, parts: List<BundlePart>, computed totals via recursion
- Ledger “daily” / “last known” →
    - Ledger: methods fundOn(date), goalOn(date), lastKnownBefore(date)
- Log rules “duplicates allowed” (do not merge) →
    - Ledger stores entries as-is; aggregation happens in queries
- Catalog “no forward references” →
    - Two-phase build or deferred resolution (resolve() after parsing)

Verbs → Methods (first → refined home)

- “load/save” →
    - Repository.loadNeeds()

    - Repository.loadLog()
    - Repository.appendLog()
    - Repository.saveAll()
- “add donation / fulfillment” →
    - AppController.addDonation()
    - AppController.addFulfillment()
- “compute today’s totals” →
    - AppController.computeDaySummary()
- “total/fixed/variable/fees” →
    - Need.total()
    - Need.fixed()
    - Need.variable()
    - Need.fees()
- “resolve need names” →
    - Catalog.get(name)
    - Catalog.resolveAll()

V2

Nouns & Noun Phrases → Candidate Classes / Value Objects

- **Need** → NeedComponent (interface / abstract)
- **Basic need** → BasicNeed (leaf in Composite)
    - Concrete implementation of NeedComponent
- **Bundle** → Bundle (composite)
    - Concrete implementation of NeedComponent
- **Child part** (bundle member) → BundlePart (value object: needName, count)
    - Used by Bundle to store components
- **Catalog** (collection of all needs/bundles) → NeedsRepository
    - Repository for need data
- **Log / Ledger / Entry** → LedgerRepository, LedgerEntry
    - Repository for log data
- **Abstraction layer** → AbstractLedgerEntry (interface)
    - Ensures all entries share common methods
- **Funds** (daily value) → represented as LedgerEntry with type FUND
    - Concrete type of AbstractLedgerEntry
- **Goal** (daily value) → LedgerEntry with type GOAL
    - Concrete type of AbstractLedgerEntry
- **Fulfillment** (acquire a need) → LedgerEntry with type NEED (name+count)
    - Concrete type of AbstractLedgerEntry
- **Repository** (persistence) → Repository (CSV I/O)
    - NeedsRepository, LedgerRepository
    - Isolates persistence logic (CSV I/O)
- **CSV File Handler** → CSVManager
    - Isolates low-level CSV read/write details
- **Controller / App** → AppController
    - NeedsRepository, LedgerRepository
    - Coordinates logic
- **View / Chart / Summary** → View, DTOs: DaySummary, SliceBreakdown
    - ConsoleView, DaySummary (DTO)
    - Presentation layer
- **Money/Date** → Money (value object), LocalDate (JDK)
    - Primitive types/value objects
- Synonyms normalized: “log/ledger” → Ledger
- “funds/goal entries” → LedgerEntry typed

- “need/bundle” → NeedComponent with BasicNeed/Bundle

Adjectives & Adverbs → Attributes

- Need “fixed / variable / fees / total” →
    - BasicNeed: name: String, fixed: Money, variable: Money, fees: Money
    - Derived total = fixed + variable + fees
- Bundle “composed of… counts” →
    - Bundle: name: String, parts: List<BundlePart>, computed totals via recursion
- Ledger “daily” / “last known” →
    - LedgerEntry: methods fundOn(date), goalOn(date), lastKnownBefore(date)
- Log rules “duplicates allowed” (do not merge) →
    - Ledger stores entries as-is; aggregation happens in queries
- Catalog “no forward references” →
    - Two-phase build or deferred resolution (resolve() after parsing)

Verbs → Methods (first → refined home)

- “load/save” → (NeedsRepository, LedgerRepository)
    - Repository.loadNeeds()
    - Repository.loadLog()
    - Repository.appendLog()
    - Repository.saveAll()
- “read/write CSV lines” → (CSVManager)
    - CSVManager.readAllLines(filename)
    - CSVManager.appendLine(filename, line)
    - CSVManager.writeAllLines(filename, lines)
- “add donation / fulfillment” → (LedgerController)
    - AppController.addDonation()
    - AppController.addFulfillment()
- “compute today’s totals” → (LedgerController)
    - AppController.computeDaySummary()
- “total/fixed/variable/fees” → (NeedComponent, BasicNeed, Bundle)
    - Need.getTotalCost()
    - Need.getFixedCost()
    - Need.getVariableCost()

    - Need.getFeesCost()
- “resolve need names” → (NeedsRepository)
    - Catalog.getNeed(name)
    - Catalog.resolveAll()
- “display” → (ConsoleView)
    - View.displaySummary()
    - View.displayGoal()
- “convert” → (NeedsRepository)
    - Repository.concertNeedsToNeedsObject()
    - Repository.toCSVLine()
- “get summary/need” → (NeedsRepository, LedgerRepository)
    - Repository.getSummary()
    - Repository.getNeed()