Nouns and Verbs

Complete this activity as a Team and submit the file to your Team repository under /doc directory. [check for other instructions on dropbox]

List **all** of the nouns and verbs that you find in the description of this system that have any association with its features, operation, or data.

Nouns	Verbs

• **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
	• "load/save" → (NeedsRepository, LedgerRepository)
	  ○ Repository.loadNeeds()
	  ○ Repository.loadLog()
	  ○ Repository.appendLog()
	  ○ Repository.saveAll()
	• "read/write CSV lines" → (CSVManager)
	  ○ CSVManager.readAllLines(file name)
	  ○ CSVManager.appendLine(filename, line)
	  ○ CSVManager.writeAllLines(file name, 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()

- 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
  - 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()

• "funds/goal entries" → LedgerEntry typed • "need/bundle" → NeedComponent with BasicNeed/Bundle	
1. Need / BasicNeed / NeedComponent 2. Bundle / Child part (BundlePart) 3. Cost / total cost / fixed cost / variable cost / fees 4. Catalog / Collection / NeedsRepository 5. Log / Ledger / LedgerRepository 6. Entry / LedgerEntry / AbstractLedgerEntry 7. Funds / Donations 8. Goal / Daily goal 9. Quantity / Count 10. Date / todaysDate 11. Summary / DaySummary 12. CSV / CSVManager / File 13. System / App / User / Doner 14. View / ConsoleView / UI 15. Controller / NeedsController / LedgerController 16. Model / Logic / Data persistence	1. Monitor / improve 2. Track / tracking 3. Set / goal-setting 4. Provide 5. Combine / form 6. Define 7. Enter / register / record / records fulfillment 8. Address 9. Load / save / append / write 10. Read / resolve / finishedAlert 11. Startup / exit / requests 12. findGoal / CalculateDonations / subtractDonationsFromGoal 13. Display / generate 14. Convert (to object) / add (to array) 15. Compute / calculate 16. Oversees / manages / returns / resolve

Now create one table for each noun that you think corresponds to a class in your software design. Put the "Class noun" in the header row for the table. Look at the other nouns and copy any that are relevant for this class to the "Attribute nouns" box. Some of these nouns may themselves be "Class nouns". Others will not have that level of prominence. Copy any verbs that could indicate responsibilities for this class to the "Behavior verbs" box. Extend the boxes as needed. Keep doing this until you think you have identified all of the "class nouns" in the system. Are there others that somehow have been missed that you need for a good design? Add more tables as needed.

Class noun: NeedComponent (Abstract)

Attribute nouns	Behavior verbs
● Name ● Identifier ● Interface ● Cost ● Total cost ● Fixed cost ● Variable cost ● Fee cost	● Provide (abstract methods for costs) ● Compute (cost details) ● Allow (derived classes to calculate) ● Return (cost components)

Class noun: BasicNeed	
Attribute nouns	**Behavior verbs**
● Name ● Dollar values ● Total cost ● Fixed cost ● Variable cost ● Fees ● CSV line ● Cost components	● Return (total individual cost) ● Provide (cost breakdown) ● Serialize (to/from CSV)

Class noun: Bundle	
Attribute nouns	**Behavior verbs**
● Name, list of (NeedCopmonent, count) pairs (BundlePart) ● Sub-needs ● Components ● Computed totals ● CSV line ● Quantity	● Combine ● Form ● Add (sub-needs) ● Compute (total costs, recursive operation) ● Sum ● Define

Class noun: AbstractLedgerEntry (Interface)	
Attribute nouns	**Behavior verbs**
● Date ● Type (FUND, GOAL, NEED) ● Value (Amount or Quantity) ● CSV line ● Log ● Entry	● toCSVLine

Class noun: LedgerEntry (Concrete)

Attribute nouns	**Behavior verbs**
● Date ● Type (FUND, GOAL, NEED) ● Value (Amount or Quantity) ● CSV line ● Associated name (for fulfillment) ● Log ● Entry	● Implement (AbstractLedgerEntry) ● Record (data) ● Store (data) ● toCSVLine ● Get/return (data)

Class noun: NeedsRepository	
Attribute nouns	**Behavior verbs**
● NeedComponent objects (BasicNeed/Bundle) ● Needs array ● Bundles array ● CSV file ● Needs data ● CSVManager	● Sttartup ● Get (needs, bundles) ● Returns (needs), convertNeedsToNeedsObject ● convertBundlesToBundlesObject ● addNeedsToNeedsArray ● addBundlesToBundlesArray ● finishedAlert ● Load ● Save ● Resolve (need names) ● Read

Class noun: LedgerRepostiory	
Attribute nouns	**Behavior verbs**
● Log ● Ledger ● Entry ● AbstractLedgerEntry (collection) ● Summary ● Daily totals ● CSV file ● CSVManager	● Save ● getSumary() ● Get (entry, goal) ● findGoal(todaysDate) ● CalculateDonations(todaysDate) ● Return Goal/Donations ● addEntry ● loadLog ● appendLog

Class noun: CSVManager	
Attribute nouns	**Behavior verbs**
● Filename ● CSV line ● Needs data ● Bundles data ● File	● readAllLines(filename ● appendLine(filename, line) ● writeAllLines(filename, lines) ● Load ● Save

Class noun: LedgerController

Attribute nouns	**Behavior verbs**
● LedgerRepository ● NeedsRepository ● Donation (x, y) ● need_count ● todaysDate	● Records fulfillment ● Save ● get need ● getSummary() ● getDailyGoal() ● findGoal ● CalculateDonations ● subtractDonationsFromGoal ● registerDonations ● addDonations ● subtractFromDailyGoal ● DonationCompleted ● compute

Class noun: ConsoleView (View)	
Attribute nouns	**Behavior verbs**
● NeedsController ● LedgerController ● Summary ● Goal ● Exit button ● Thank you message ● UI ● User	● Requests ● Records fulfillment ● Exit button ● getDailyGoal() ● displaySummary() ● displayGoal() ● startup() ● finishedAlert() ● enterDonations ● registerDonations ● thankYouAlert ● display