# **Applying the Account/Entry Pattern to Gamified Gym**

This document outlines how to model the workout history for the Gamified Gym application using the **Account and Entry** analysis pattern, as described in Lecture 13.

### **1. The Core Objective**

As stated in the task, the system must ensure that a user's "Account" (e.g., Muscle Group Rank) is **determined by the sum of its "Entries"** (e.g., all logged exercises) rather than being a simple, editable value. This creates an auditable, non-editable history where the current state is a provable consequence of past actions.

### **2. The Analysis Pattern**

* **Account**: Holds the *current state* (the "balance"). This is the user's current rank, total volume, or personal best.
* **Entry**: Represents an *immutable historical record* of a transaction (the "amount").
* **The Constraint**: The core principle is that the Account.balance is **sum(entries.amount)**. All history is kept, and entries are never removed.

### **3. Mapping Gamified Gym to the Pattern**

| **Pattern Concept** | **Gamified Gym Implementation** | **Description (from Milestone 2 Doc)** |
| --- | --- | --- |
| **Account** | MuscleGroupProgress | "a Muscle Group Progress tracker (le bronze, silver, gold...)" (1.2.3) |
| **Account** | PersonalBest | "system... automatically updates his personal record" (2.1.1, #9) |
| **Entry** | ExerciseLog | "He logs 3 sets of 8 reps at 90 lb" (2.1.1, #6) |

### **5. Enforcing the Constraint (The Core Task)**

This model ensures the Account is determined by the sum of its Entries by defining a clear update process. **The ExerciseLog (Entry) is the source of truth.** The MuscleGroupProgress (Account) is a *cached calculation* based on those entries.

#### **Scenario: User logs a "Chest" exercise**

Here is the step-by-step process that enforces the pattern:

1. **User Input**: A user logs "Bench Press: 3 sets of 5 at 150lbs".
2. **Create Immutable Entry**: The system *does not* just update a "Chest" value. Instead, it creates a **new, non-editable ExerciseLog entry**:
   * exerciseName: "Bench Press"
   * associatedMuscleGroups: ["Chest", "Triceps"]
   * weight: 150
   * sets: 3
   * reps: 5
   * dateLogged: (current time)
   * calculatedVolume: 150 \* 3 \* 5 = **2250** (This is the "amount")
3. **Update Account (Posting)**: The system takes this Entry and "posts" it to the relevant Account(s).
   * It finds the MuscleGroupProgress for "Chest".
   * It updates its totalVolume (the "balance"):  
     Chest.totalVolume = Chest.totalVolume + 2250
   * It also finds the MuscleGroupProgress for "Triceps" and updates it:  
     Triceps.totalVolume = Triceps.totalVolume + 2250
4. **Derive State**: The system then checks if the new totalVolume for "Chest" or "Triceps" is enough to change the currentRank (e.g., if totalVolume > 50,000, set currentRank = "Gold").
5. **Update PersonalBest Account**: The system also checks the PersonalBest account for "Bench Press":
   * if (150 > BenchPressPR.maxWeight) { BenchPressPR.maxWeight = 150 }

### **6. Conclusion & Benefits**

This model directly fulfills the requirements of the task and your project documentation:

* **Rank is "Calculated"**: As required in section 2.2.3, the rank is "calculated based on the total historical training volume," not just set by the user.
* **History is Preserved**: The ExerciseLog list provides a full, auditable history of every workout. The user's "Chest: Gold" rank can be proven by summing all ExerciseLog entries associated with "Chest".
* **Immutability**: The ExerciseLog is an "entry" in the truest sense—it's a "personal and non-editable record" that, once written, is never changed.