Drop System Integration Guide

📁 File Structure

Create a new package: `dev.main.drops`

Add these 5 files:

```
dev/main/drops/
├── DropRarity.java      (Enum for drop rarities)
├── DropItem.java        (Template for droppable items)
├── DroppedItem.java     (Result of a drop)
├── DropTable.java       (Configuration of all drops)
└── DropSystem.java      (Main drop generation logic)
```

🔧 GameLogic.java Changes

1. Add Import

```java
import dev.main.drops.*;
```

2. Add Field

```java
public class GameLogic {
    private GameState state;
    private DropSystem dropSystem;  // ★ NEW
    // ... other fields
}
```

3. Initialize in Constructor

```java
public GameLogic(GameState state) {
    this.state = state;
    this.dropSystem = new DropSystem();  // ★ NEW
}
```

4. Replace handleMonsterDeath() Method

Replace your entire `handleMonsterDeath()` method with the version I provided above.

5. Add Helper Methods

Add these two new methods to GameLogic:

- `calculateDropCapacity(Entity monster)`
- `addDropsToInventory(Entity player, List<DroppedItem> drops)`

🎮 How It Works

When a Monster Dies:

1. **Generate Drops -** DropSystem rolls for items
2. **5% Lucky Drop Chance -** All items at max quantity
3. **Regular Drop -** 1 to (dropCapacity-1) random items
4. **Add to Inventory -** Items automatically added

Drop Capacity by Tier:

- **TRASH**: Max 2 items
- **NORMAL**: Max 3 items
- **ELITE**: Max 4 items
- **MINIBOSS**: Max 5 items

Available Drops:

- **Common (50%)**: Wooden Tablet, Clay, Carving Stone, Essence
- **Rare (30%)**: Scroll
- **Epic (15%)**: Verdant Shard, Rune of Return
- **Legendary (4%)**: Arcane Scroll
- **Mythic (1%)**: To be implemented

🧪 Testing

Kill a monster and you'll see:

Goblin Lv1 TRASH has died!
Goblin Lv1 TRASH dropped:
- 3x Clay [COMMON]
- 1x Carving Stone [COMMON]

Added 4 items to inventory
→ XP Reward: 25

Lucky drop example:

✨ ✨ ✨ LUCKY DROP! ✨ ✨ ✨
Elite Goblin Lv3 ELITE dropped:
- 5x Wooden Tablet [COMMON]
- 8x Clay [COMMON]
- 3x Carving Stone [COMMON]
- 4x Essence [COMMON]
- 2x Scroll [RARE]
- 3x Verdant Shard [EPIC]
- 1x Rune of Return [EPIC]
- 1x Arcane Scroll [LEGENDARY]

Added 27 items to inventory

🔄 Customization

Add New Drops

Edit `DropTable.java`:

```java
epicItems.add(new DropItem("Fire Rune", DropRarity.EPIC, 1, 2,
    ItemManager::createFireRune));
```

Adjust Drop Rates

Edit `DropRarity.java`:

```java
COMMON(0.60),    // Increase common to 60%
RARE(0.25),      // Decrease rare to 25%
```

Change Lucky Drop Chance

Edit `DropSystem.java`:

```java
private static final double LUCKY_DROP_CHANCE = 0.10; // 10% chance
```

✅ Checklist

- [ ] Create `dev.main.drops` package
- [ ] Add 5 drop system files
- [ ] Add `dropSystem` field to GameLogic
- [ ] Initialize dropSystem in constructor
- [ ] Replace handleMonsterDeath() method
- [ ] Add calculateDropCapacity() method
- [ ] Add addDropsToInventory() method
- [ ] Test by killing monsters

🎯 Features

- ✅ Uses ThreadLocalRandom for all RNG
- ✅ Regular drops: 1 to (max-1) items
- ✅ Lucky drops: ALL items at MAX quantity
- ✅ No duplicate items in single drop
- ✅ Tier-based drop capacity
- ✅ Weighted rarity system
- ✅ Automatic inventory integration
- ✅ Console feedback for drops

Note: The drop system uses `DropRarity` (separate enum) to avoid conflicts with your existing `Item.Rarity` enum.