Box Loading System – Layout for Programmers

Purpose

- Provide players with a **package-loading mini-game** tied to the shift mechanic.
- Success/failure affects **narrative progression** (meeting quota → continue; failing quota → reset/day restart).
- Ensure clarity for writers (story consistency), artists (UI & environment), and programmers (system logic).

Player Experience Flow

1. **Start of Shift:**
    - Player is assigned a **quota** (e.g., 50 packages)
    - More packages than quota spawn to allow for mistakes.

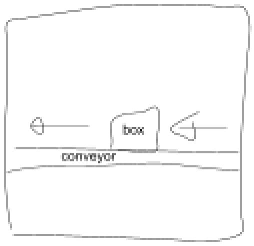
2. **Boxes Spawn:**
    - Packages move along a **conveyor belt**.
    - Each package has a **code** (symbol + number)
    - Holding a package shows floating text (ex 137)

3. **Loading Trucks:**
    - Multiple trucks are lined up along the walkway.
    - Each truck has a **symbol** above its back doors (matches first part of code)
    - Inside each truck, **shelves are labeled with numbers** (matches second part of code)
    - Player must place box in **correct truck + correct shelf**.

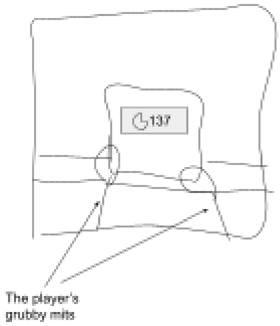
4. **Quota Verification:**
    - At end of shift, game checks whether quota was met.
    - Packages in wrong spots or left unloaded **do not count**.
    - Missed packages that reach the end of conveyor are deleted.

5. **Outcome:**
    - **Meet quota** → continue narrative.
    - **Fail quota** → game over → restart day.

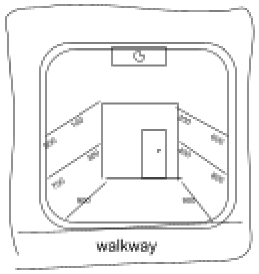
POV: You're staring at the box on the conveyor



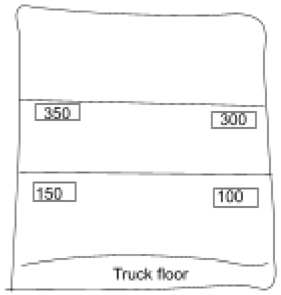
POV: You're holding the box and floating text appears to tell you where to load it



POV: Looking at truck from walkway



POV: Looking at left side of truck from inside



System Rules & Requirements

- **Quotas:**
  - Defined per shift (writers decide number).
  - Exceed slightly with extra packages (e.g., quota = 50 → ~60 packages spawned).
- **Package Logic:**
  - Each box spawns with randomized `(symbol, number)` pair.
  - Floating text shows assignment when player holds it.
  - Must match both **truck symbol** and **shelf number**.

- **Conveyor / Destruction Zone:**
  - Boxes that reach the end despawn.
  - Unloaded boxes disappear after each shift.

- **Verification:**
  - At shift end, system checks:
    - Total packages correctly loaded.
    - Compare against quota.
    - Trigger continue or game over state.
- **Note:** Random code/symbol generation must be within **editable parameters** so daily variations can be set (different # of trucks, symbols, quotas).

Special Packages (Future Feature)

- Some packages will require **extra steps** before being considered valid (ex hit with bat, cremation, special handling).
- These will use the same **code/symbol system**, but with added **unique logic checks** (verify step completed before counting towards quota).
- Next step after base system: implement these conditions once core system is stable.
- (Future) Special package conditions must be met before counting.

Programmer Tasks

1. **Box Spawning System**
   - Random code assignment `(symbol + number)`.
   - Floating text display when picked up.

2. **Truck / Shelf System**
   - Trucks positioned along walkway.
   - Symbols displayed above each truck door.
   - Shelf slots with number labels inside trucks.

3. **Loading Interaction**
   - Detect if package dropped in **correct truck + correct shelf**.
   - Count only valid placements.

4. **Quota Tracking**
   - Increment counter for valid packages.

   - At shift end, compare total vs. quota.

5. **Conveyor & Fail Conditions**
   - Conveyor deletes boxes that reach end.
   - Unloaded boxes cleared after shift.
   - If quota not met → game over → restart shift.

Dependencies

- **Writers:** Define quota numbers, story context, failure dialogue.
- **Artists:**
  - Visual design of trucks, shelves, symbols.
  - UI for quota tracking and floating text.
- **Programmers:** Implement full system logic + testing.

Testing Checklist

- Box spawns correctly with random `(symbol, number)`.
- Floating text shows correctly when box is held.
- Player can carry/drop box into correct slot.
- Wrong placement does not count.
- Quota counter updates only with correct placements.
- End-of-shift check correctly compares quota.
- Fail state resets day loop.
- Success state progresses story.