Game Design Spec

- Game Design Spec
  - High concept
- Core variables
  - Persistent (carry-over)
  - Run-scoped (cleared on run reset)
- Turn structure (moment-to-moment)
  - Turn start: generate the two hidden cards
  - Player options
- Resolution rules
  - If player flips Bomb
  - If player flips Treasure
- Loot system
  - Number draw
  - Joker draw (3 jokers)
    - Joker A — Shield
    - Joker B — Scrambler
    - Joker C — Jammer
- Banking
  - When allowed
  - Bank action
- Run reset rules
  - Run reset (from duplicate bust or bomb hit)
  - Game over
- Risk / reward model (why pushing matters)

Example script (playthrough narrative)

  - Turn 1
  - Turn 2
  - Turn 3
  - Turn 4
- State Machine Diagram
- Turn Loop Diagram
- BankPoints table (ready to paste)
- Trophy thresholds

High concept

A push-your-luck guessing game. Each turn the player chooses between two face-down cars/cards. Most of the time both are treasure early in a run, but as the streak grows a bomb is more likely to appear. Treasure flips award a loot draw (numbers 1–7 or jokers). Collect unique numbers in the current run to increase a convex bank payout. Duplicates bust the run. Bombs cost lives. Banking converts run progress to permanent score. Trophy tiers unlock by total banked score.

Core variables

Persistent (carry-over)

- `totalScore` (int) — accumulated banked points.
- `trophyTier` ∈ {None, Bronze, Silver, Gold}
- `bestTotalScore` (optional)

Run-scoped (cleared on run reset)

- `lives` (int) — starts at 3 each new game.
- `streak` (int) — consecutive **Treasure flips** in current run.
- `runUniques` (set) — unique numbers collected this run (1..7).
- `runHasShield` (bool) — max 1 shield active.
- `scramblerCharges` (int) — number of "anti-duplicate rerolls" remaining.
- `jammerTurns` (int) — turns remaining where bomb appearance chance is reduced.

Turn structure (moment-to-moment)

Turn start: generate the two hidden cards

1. Compute bomb appearance probability:

- Base curve:
  `pBomb = clamp(0.20 + 0.06 * streak, 0.20, 0.56)`
- Apply Jammer (if active):
  `pBomb = max(0, pBomb - 0.15)` while `jammerTurns > 0`

1. Sample whether a bomb is present:

- With probability `pBomb`: slots = `{Bomb, Treasure}`
- Otherwise: slots = `{Treasure, Treasure}`

1. Randomly assign these to Left/Right.
2. Show shuffle animation, then wait for input.

Player options

- Flip Left
- Flip Right
- Bank (enabled only if `runUniques.size > 0`)

Resolution rules

If player flips Bomb

- If `runHasShield == true`:
  - Consume shield: `runHasShield = false`
  - **Survive** without losing a life
  - **Balance rule:** reset `streak = 0` (run continues, uniques remain)
- Else:
  - `lives -= 1`
  - Run reset (see below)

If player flips Treasure

1. Increment streak:

- `streak += 1`

1. Loot draw (always occurs)

- With probability 0.15 draw a Joker, else draw a Number.

Loot system

Number draw

- Draw number `n` uniformly from 1..7 (simple and readable).
- If `n` is already in `runUniques` → duplicate event:
  - If `runHasShield == true`:
    - Consume shield; ignore bust (duplicate does not add)
  - Else if `scramblerCharges > 0`:
    - Spend 1 charge and reroll (up to 1 reroll per charge).
    - If reroll is still duplicate and no more scramblerCharges: bust.
  - Else:
    - Run reset (duplicate bust, no life loss)
- If `n` not in `runUniques`:
  - Add: `runUniques.add(n)`

Joker draw (3 jokers)

All jokers are “loot” cards drawn after Treasure.

Joker A — Shield

- Effect: `runHasShield = true` (if already true, convert to +5 bonus points on next bank or ignore; choose one behavior. Recommended: convert to **+5 "bonusBank"** to avoid wasted draw.)

Joker B — Scrambler

- Effect: `scramblerCharges += 2` (max 4).
- Meaning: On a duplicate number draw, you may automatically reroll, spending a charge.

Joker C — Jammer

- Effect: `jammerTurns = 3`
- Meaning: For the next 3 turns, reduce bomb appearance probability by 0.15 (min 0).

Banking

When allowed

- Enabled if `runUniques.size >= 1`

Bank action

1. Compute bank payout from unique count `u = runUniques.size`:

BankPoints(u):

- u=1 → 4
- u=2 → 10
- u=3 → 18
- u=4 → 28
- u=5 → 40
- u=6 → 54
- u=7 → 70 + Full Set Bonus (+30) = 100

1. Apply any "bonusBank" (if you implement shield overflow).
2. Add to total: `totalScore += payout`
3. Update trophy tier based on `totalScore`:

- Bronze: totalScore ≥ 100
- Silver: totalScore ≥ 200
- Gold: totalScore ≥ 300

1. Run reset (safe reset): clears run state but keeps lives and totalScore.

Run reset rules

Run reset (from duplicate bust or bomb hit)

Clear:

- `streak = 0`
- `runUniques = {}`
- `runHasShield = false`
- `scramblerCharges = 0`
- `jammerTurns = 0`
  (Keep `lives` unless bomb without shield consumed it, and keep `totalScore` always.)

Game over

- When `lives == 0`: end run, show summary, allow restart.
- On restart: set `lives = 3`, `totalScore = 0`, clear run state. Optionally persist best score.

Risk / reward model (why pushing matters)

- Bomb risk per flip: `0.5 * pBomb(streak)` (rises as streak grows)
- Duplicate risk per treasure (without scrambler/shield): `u/7` (rises as uniques grow)
- Bank payout is convex: marginal gains from u=3→4→5 are large, so “bank at 1” is slow.

Example script (playthrough narrative)

Assume new game: `lives=3`, `totalScore=0`, run empty.

Turn 1

- streak=0 ⇒ pBomb=0.20 ⇒ no bomb this time ⇒ {T, T}
- Player flips Left ⇒ Treasure
- Loot: Number 3 (new) ⇒ runUniques={3}, streak=1
- Bank now would give 4 points, but player pushes.

Turn 2

- streak=1 ⇒ pBomb=0.26 ⇒ bomb not present ⇒ {T, T}
- Flip Right ⇒ Treasure
- Loot: Joker Scrambler ⇒ scramblerCharges=2
- runUniques still {3}, streak=2

Turn 3

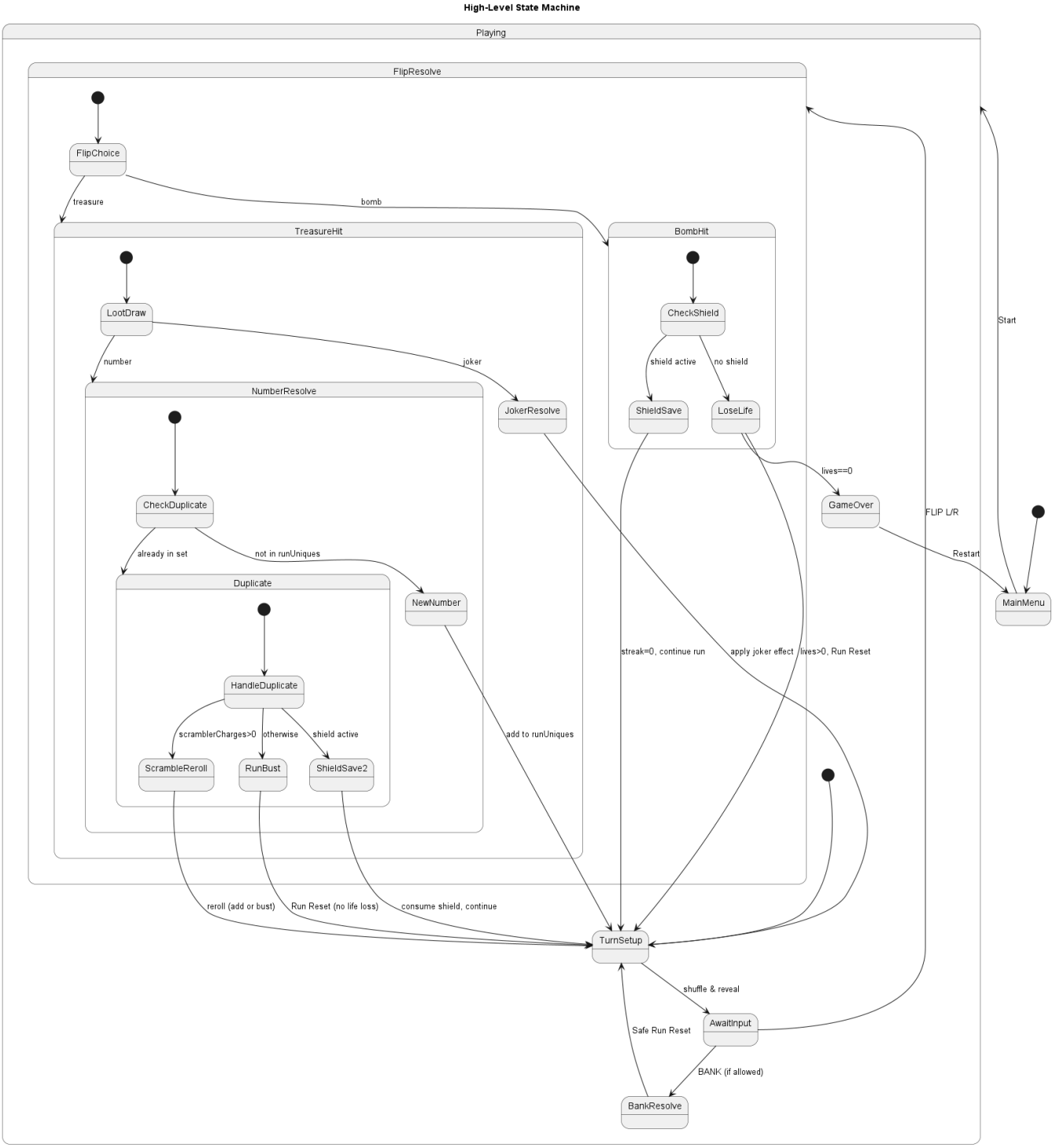
- streak=2 ⇒ pBomb=0.32 ⇒ bomb present ⇒ {B, T}
- Flip Left ⇒ Treasure (lucky)
- Loot: Number 3 (duplicate) ⇒ scrambler triggers (charge 1) reroll ⇒ Number 6 (new)
- runUniques={3,6}, streak=3, scramblerCharges=1

Turn 4

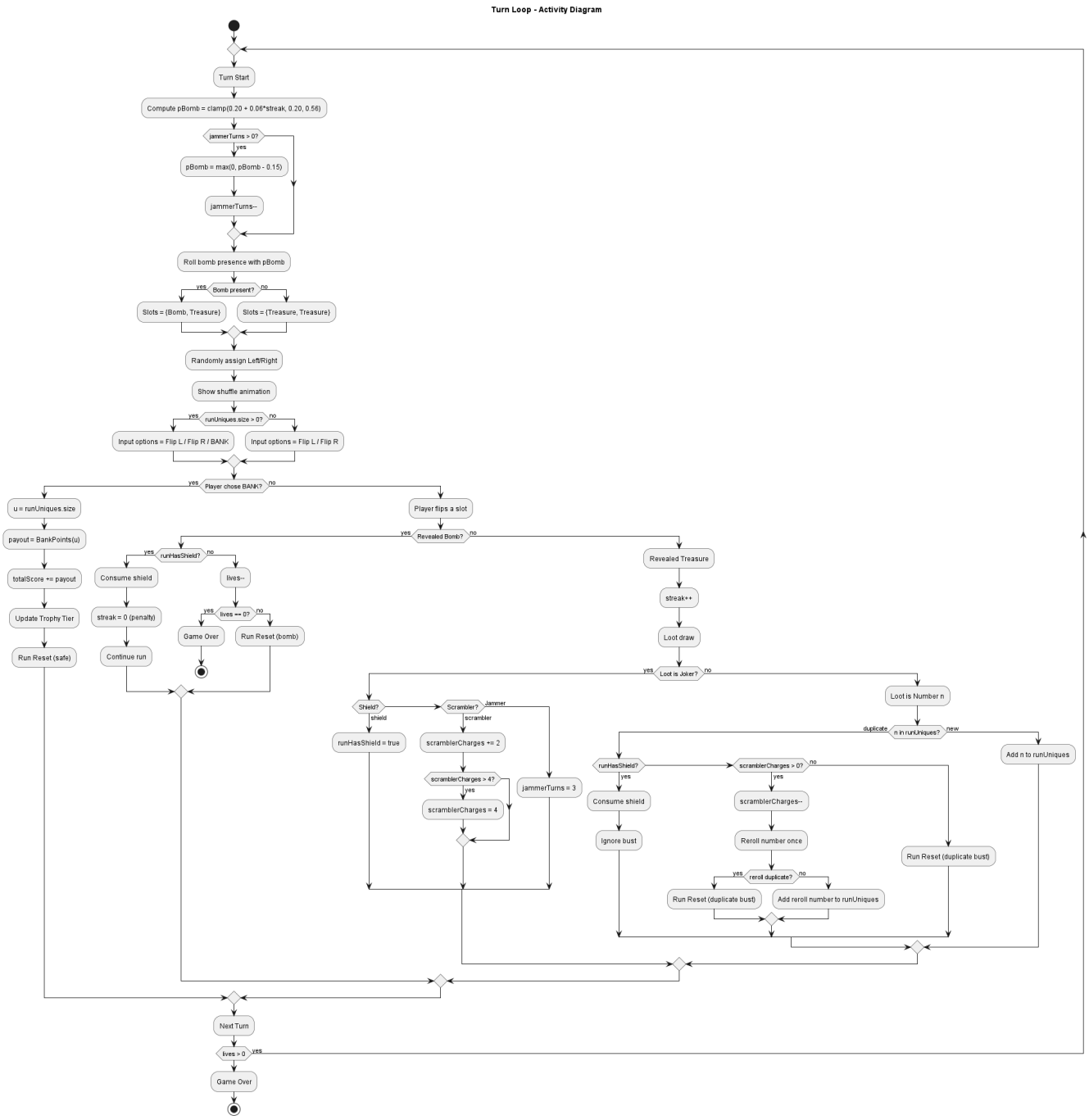
- streak=3 ⇒ pBomb=0.38 ⇒ bomb present ⇒ {B,T}
- Player chooses BANK instead of flipping.
- u=2 ⇒ payout 10 ⇒ totalScore=10
- Run resets: streak=0, runUniques cleared, scrambler cleared.

Repeat runs; a skilled player aims to bank u=4–5 often. Gold at 300 total typically takes ~7–10 decent banks or a couple big u=6/u=7 banks.

State Machine Diagram



Turn Loop Diagram



BankPoints table (ready to paste)

- 1 → 4
- 2 → 10
- 3 → 18
- 4 → 28
- 5 → 40
- 6 → 54
- 7 → 100

Trophy thresholds

- Bronze: 100
- Silver: 200
- Gold: 300