
0. Предпосылка: никакого админа в механике

Сначала коротко зацементируем правило, из которого всё остальное растёт:

* У **GameCore-контракта**:

* **нет owнер-а**, который может:

* отменять раунды,

* трогать чужие депозиты,

* перепрошивать логику.

* нет функций вида `adminCancelLobby`, `adminForceResolve` и подобной дряни.

* Всё, что связано с **фрод-банами / мутами / подозрительными аккаунтами**, делается:

* либо на фронте (UI не даёт человеку заходить),

* либо на бэкенде (отсекаем вызовы до контракта),

* но **не** через вмешательство в уже идущие игровые сессии.

Ты как “создатель” смотришь только на:

* графики,

* метрики,

* состояние трезори,

* логи и здоровье проекта,

но **не** можешь технически залезть в раунд.

Это и есть твоя “гвард-зона”.

1. Инварианты ядра

Документальный пункт №1:

* Чётко, в 1–2 страницах, зафиксировать **инварианты**:

* “любой игровой пул всегда уходит победителю”;

* “платформа никогда не является получателем средств из пула”;

* “расчёт победителя всегда основан на источнике случайности X”;

* “после начала определения победителя ни один участник/админ/кто-либо не может поменять исход”.

Это раздел типа: **TNRD-CORE-INVARIANTS**.

Без этого всё остальное технически красиво, но концептуально сопли.

2. Модель ролей и доверия

Второй пункт: **Roles & Trust Model**.

Кто у нас вообще существует:

- * `PLAYER` — единственная роль, которая кладёт и получает TON в играх.
- * `MONITOR` — кто читает и показывает состояние (дашборды, аналитика), но он read-only.
- * `TREASURY_OWNER / MULTISIG` — управляет трезори и может триггерить награды/байбеки.
- * `ORACLE_UPDATER` — тот, кто обновляет цену для Locker.
- * `NOBODY` — владелец GameCore-контракта. То есть **его нет**.

В доке должно быть прям по пунктам:

- * кому **не нужно** доверять (ядру);
- * кому **приходится** доверять (oracle, treasury, locker);
- * что эти роли могут и чего не могут.

3. Стейт-модель GameCore

Третий пункт: **полное и формальное описание состояния** GameCore:

* Структуры:

- * `Lobby`,
- * `LobbyMode`,
- * `LobbyStatus`,
- * список `lobbies`,
- * мапа `id → Lobby`.

* Какие поля:

- * мы уже перечислили: `id, mode, status, stake_amount, donation_amount, players[], winner, timestamps` и т.д.
- * Что именно хранится ончейн, а что вычисляется на лету.

Это “анатомия контракта”.

4. Интерфейс: входящие сообщения / функции

Четвёртый пункт: **полный список entrypoints**.

Примеры:

- * ``createLobby(...)``
- * ``joinLobby(lobby_id)``
- * ``resolveLobby(lobby_id)`` (если не авто)
- * ``cancelByTimeout(lobby_id)`` (автоматическая/программная отмена если не набралось)
- * сервисные методы типа ``getLobby(lobby_id)``, ``listLobbies(...)`` (view / get-методы в TON-стиле).

У каждого:

- * параметры,
- * условия,
- * постусловия,
- * возможные ошибки (какие ``op`` / коды возврата).

После этого любой контрактчик или аудитор видит: **что этот контракт умеет и через что к нему вообще можно обратиться.**

5. Потоки денег (Fund Flow)

Пятый пункт: **денежные потоки**. Прямо схемой.

Отдельно по режимам:

* Coin Flip:

- * A → контракт: ``stake``
- * B → контракт: ``stake``
- * контракт → победитель: ``2 * stake``

* Public Lobby:

- * N игроков → контракт: по ``stake``
- * контракт → победитель: ``N * stake``

* Donateroom:

- * каждый игрок → контракт: ``stake + donation``
- * контракт:

- * победителю: ``N * stake``
- * treasury: ``N * donation``

Это можно оформить как таблицу и одну диаграмму.

Это та часть, которую юзеры и инвесторы поймут быстрее всего.

6. Рандом и фэйлкейсы рандома

Шестой пункт: ****Randomness & Failure Cases****.

* откуда берётся рандом:

* VRF?

* внешнее подтверждение?

* on-chain entropy?

* какие есть риски:

* задержка,

* недоступность,

* злоупотребление.

* что контракт делает, если:

* рандом не приходит,

* оракул рандома ведёт себя странно,

* кто-то пытается переиграть через georg/повтор.

Ты сам сказал: ****никакого ручного “отменить раунд” от админа****,
значит все фэйлкейсы должны быть:

* либо уход в `CANCELLED + рефанд`,

* либо повторная попытка по чётким правилам.

Это всё заранее описывается в спеке.

7. Авто-отмена / авто-завершение

Седьмой пункт: ****Lifecycle & Timeouts****.

* когда лобби считается “протухшим”;

* как именно работает auto-cancel:

* по дедлайну,

* по условию “не набралось”;

* что происходит с деньгами:

* всегда полный рефанд,

* донат возвращается / не возвращается (и это заранее жёстко прописано).

И опять: ****никаких админских ручных отмен****.

Только алгоритм по времени / состоянию.

8. События и логирование (Events)

Восьмой пункт: ****Events & Indexing****.

* какие события выбрасывает контракт:

- * `LobbyCreated`
- * `LobbyJoined`
- * `LobbyReady`
- * `LobbyResolved`
- * `LobbyCancelled`

* какие поля в каждом event:

- * чтобы фронт, бэкенд и аналитика могли жить без костылей.

Обычно этим пренебрегают, а потом всё “по логам из ноды разбирают”.

Ты хочешь нормальную систему — значит события должны быть частью формальной спецификации.

9. Security & Limits

Девятый пункт: ****ограничения и защита****.

Здесь:

* ограничения по ставкам:

- * минимум,
- * максимум,
- * почему именно так;

* защита от:

- * спама мелкими ставками,
- * переполнений,
- * атак по газу (чтобы одно лобби не взорвало стоимость транзакций всем остальным);

* чёткое перечисление:

- * чего контракт ****не делает****,

* какие атаки тебя не волнуют по дизайну (например, фронт сам отсекает токсичных юзеров, но контракт с ними не борется).

10. Политика обновлений и миграций

Десятый пункт: ****Upgrade & Migration Policy****.

У тебя позиция такая:

> “Если нет технической причины трогать контракт – не трогать. И вообще не иметь рычагов, чтобы его трогать.”

Соответственно:

* GameCore:

- * immutable,
- * без прокси,
- * без owner,
- * без возможности апгрейда.

* Любая “новая версия”:

- * только через ****новый контракт****,
- * с отдельной документацией версии,
- * с возможностью миграции средств по инициативе самих пользователей (а не “мы вас перенесли, вы теперь тут”).

* Treasury / Locker / Oracle:

- * могут иметь апгрейд-политику,
- * но она должна быть:

- * через мультисиг,
- * с timelock,
- * с прозрачными правилами.

Итого: на что ты сейчас реально смотришь

Мы уже:

- * описали механику лобби, coin flip, donateroom;
- * описали токен, Locker, Oracle в общих чертах;
- * зацепили trust-модель и отсутствие комиссий.

Тебе сейчас нужно:

1. Зафиксировать ****10 пунктов выше как структуру документации****.
2. По каждому пункту:

- * превратить наши накидки из чата в аккуратные разделы.

3. И только после этого честно назвать это:

* **“спецификация смарт-контракта TONRODY GameCore v1”**,
а не “ну там у нас была идея, как он должен работать”.

Если хочешь, следующим шагом я могу собрать из этого **каркас дока** в формате:

```
```text
TNRD-CORE-01.md
1. Invariants
2. Roles & Trust
3. State Model
4. Interface
...
```
```

и уже под него подложим наш контент.