SBET V2 / GRADER OPS

Grader Operator Handbook

Running a grader on SBET V2 — stake, infrastructure, 6-tier AI escalation, cost ceilings, slashing scenarios, VRF panel rotation, and incident runbook.

Audience: potential and active grader operators

Prerequisites: $25,000 SBET + HSM / hardware signer + Python/Go runtime

Time to onboard: 1–2 weeks for infrastructure + shadow-grading

At a glance

Item	Value
Minimum stake	$25,000 SBET
Panel size (N)	13
Quorum (M)	9 of 13
Deregister cooldown	7 days
Max slash per match	Up to 100% of grader's stake
Auto-claim fee share (stakers)	20% of 2% SBET / 3.5% other fees
Slash share (stakers)	20% of slashed bonds
Break-even AI cost / match	~$0.05 avg, $0.40 full escalation

This handbook is a draft. Final operator terms, including fee-rebate schedules and grader-rotation cadence, will be finalized at V2 mainnet deployment (target Q3 2026).

1. What is a Grader?

Graders are the decentralized oracle for SBET V2. You stake $25,000 SBET, run infrastructure that grades match outcomes, and earn a share of auto-claim fees and slash distributions. Grade honestly and you collect rewards indefinitely. Grade dishonestly and you lose your stake.

Each match registered on-chain is assigned a VRF-selected sub-panel of 13 graders drawn randomly from the active pool. You are **not** on every match. Your grader software listens for `MatchRegistered` events, checks whether you're on the panel via a Merkle proof, waits for the match to end, runs a 6-tier AI grading pipeline, and submits your result on-chain.

1.1 Role and responsibilities

- **Sign your own analysis.** Each grader submits independently. You are responsible for the correctness of your grade.
- **Meet quorum.** 9 of 13 graders must submit grades before the aggregator can call `proposeOutcome`. Dropouts below 5 delay finalization.
- **Report confidence honestly.** confidenceBps feeds the dynamic challenge-window tier. Artificial inflation of confidence is slashable if the modal outcome is overturned.
- **Abstain when uncertain.** You are never punished for not submitting a grade. You are punished for submitting a wrong grade.
- **Monitor your panel assignments.** Missing a match where you were drawn damages reputation (not formally slashable today, but may become so).

1.2 What graders don't do

Graders do not resolve disputes (the 5-of-9 arbiter does). Graders do not trigger pauses (guardians do). Graders do not hold user stakes (Treasury does). Graders do not finalize matches themselves — they submit grades; an off-chain aggregator aggregates and calls `proposeOutcome`; anyone can call `finalize` after the challenge window elapses.

2. Economics

The grader business is simple: you stake capital, you provide uptime and AI compute, and you capture a share of protocol fees. This section walks through the stake, the revenue, the cost, and the break-even matrix.

2.1 Stake

Minimum stake is $25,000 SBET. This is the hard floor: fall below it after a slash and your status becomes `Slashed`. You cannot submit grades while slashed; top up to the floor to reactivate.

```
await sbet.approve(graderRegistryAddress, parseEther('25000'));
await graderRegistry.register(parseEther('25000'));
// GraderStatus → Active, GraderRegistered event fires.
```

2.2 Revenue streams

Stream	Amount	Path
Slash share (via stakers pool)	20% of slashed bonds	Treasury.distributeSlash → SBETStakerRewards
Auto-claim fee share	20% of 2% SBET / 3.5% other	Treasury.autoClaim → notifyReward → stakers
Community bounty (optional)	10% of slashed bonds	Independent monitoring; qualify via reports

Revenue is captured via *staking your SBET rewards* into `SBETStakerRewards`. The grader stake itself (in `GraderRegistryV2`) does not earn yield — it is security collateral. Operators typically re-stake earned fees into stakers for the yield leg.

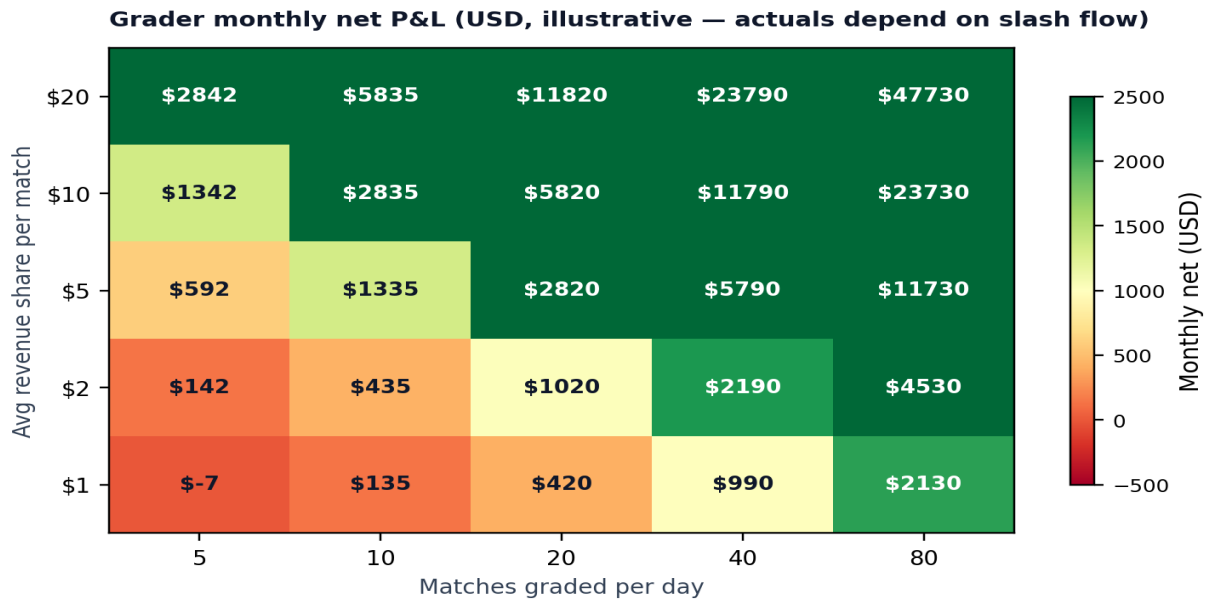
2.3 Costs

Three cost centers: locked stake (opportunity cost), infrastructure, and AI inference.

Category	Est. monthly	Notes
AI inference	$30–$50	6-tier escalation, ~5% of matches reach tier 4+
Infrastructure (VPS + RPC)	$100–$200	4 vCPU / 8GB / dedicated L2 RPC + fallback
Monitoring / alerting	$20–$50	Uptime + event subscription + dashboard
Opportunity cost on stake	— (TBD)	$25k SBET locked; opportunity cost depends on alt yield

2.4 Break-even matrix

Net monthly P&L as a function of matches graded per day × avg revenue share per match (illustrative — actuals depend on slash flow and fee mix):



At 20 matches/day × $5 avg revenue share = +$2,850/mo net. At 40/day × $10 = +$11,850/mo. Break-even for a single operator running 10 matches/day sits around $1 avg share per match.

3. Infrastructure

Your grader runs a continuous loop: listen for matches, check panel membership, wait for match end, grade via the 6-tier pipeline, submit, monitor proposals. It is **I/O bound**, not compute bound; modest hardware + reliable AI API access is enough.

3.1 Minimum requirements

Component	Recommendation
CPU / RAM	4 vCPU / 8 GB (I/O bound)
Storage	50 GB SSD (match evidence cache)
Network	99.9% uptime, <200 ms latency to L2 RPC
RPC	Dedicated L2 RPC (Base/Arbitrum) + fallback RPC
Signer	Hardware-backed signer (HSM / Fireblocks / Turnkey)
AI provider access	Claude + GPT API keys, optional Sportradar/ESPN API
Language runtime	Python 3.11+ (reference); Go public release planned
Database (optional)	Postgres for audit log of submitted grades

3.2 Software stack

```
# Grader daemon — pseudocode
while True:
for match in subscribe('MatchRegistered'):
if not verify_panel_membership(match.id, me, proof):
continue
await match_end_time(match)
result = run_6_tier_escalation(match)
if result.confidence_bps >= MIN_CONFIDENCE:
await grader_registry.submit_grade(
match.id, result.outcome, result.confidence_bps
)
else:
log_abstention(match, result)
reconcile_missed_matches()
```

3.3 Key management

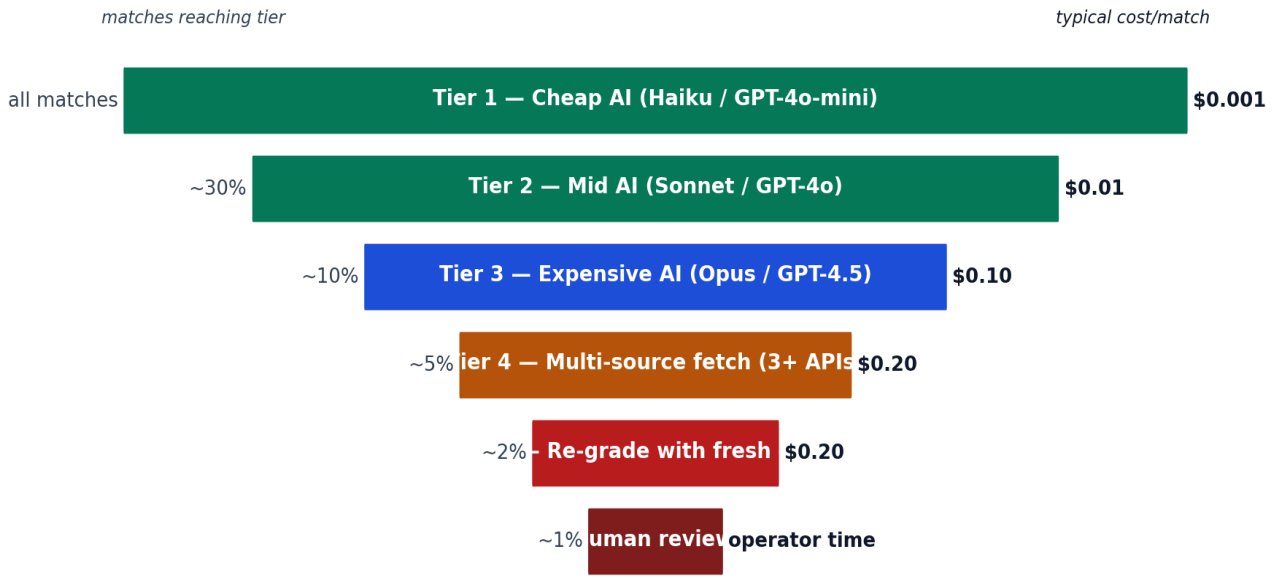
The grader key signs `submitGrade` and `increaseStake` transactions. It does **not** hold custody of user funds, but it can trigger slashing of your $25k stake if misused. Operational rules:

- Use an HSM or hardware wallet reachable via signer proxy (Fireblocks, Turnkey, Gnosis Safe module).

- Keep the key on a dedicated machine with no inbound network access.
- Separate the *grader key* (signs grades) from the *stake key* (holds SBET, calls register/increaseStake/deregister). Different roles, different blast radii.
- Rotate the key via governance if compromise is suspected (flagged as open question — per-grader key rotation not yet wired).

4. 6-Tier AI Escalation Pipeline

Grading is not a single AI call. It is a **progressive escalation pipeline** that climbs in cost and confidence until one tier yields a result you're willing to stake $25k on.



Grader submits highest-confidence tier result. Majority of matches settle at tier 1–2.

4.1 Tier-by-tier

#	Tier	Cost	When to use
1	Cheap AI (Haiku / GPT-4o-mini)	$0.001	First pass on public match metadata
2	Mid AI (Sonnet / GPT-4o)	$0.01	If tier 1 confidence < 80%
3	Expensive AI (Opus / GPT-4.5)	$0.10	If tier 2 confidence < 90%
4	Multi-source fetch (3+ independent APIs)	$0.20	If tier 3 still ambiguous — cross-check with ESPN / league / Sportradar
5	Re-grade with fresh context	$0.20	If sources disagree — re-run tier 3 with updated prompt
6	Human review / abstain	operator time	Still uncertain → abstain; optionally open dispute if someone else proposes badly

4.2 Confidence thresholds that matter downstream

The confidenceBps you submit feeds directly into `ChallengeWindowLib.computeTier`. Three thresholds:

- **confidenceBps ≥ 9500** → eligible for 90-second Fast tier (requires unanimity + source agreement ≥ 2).
- **confidenceBps < 6000** → forces 4-hour Contentious tier regardless of other signals.
- **6000 < confidenceBps < 9500** → QuorumClean or OneDissent tier based on dissents.

Slashable risk. Submitting artificially high confidence (e.g. 9800) when you are actually uncertain is dangerous. If the aggregator lumps your vote in, the match finalizes Fast-tier, and a dispute later overturns it, you are a signer of the losing proposal — *slashable*.

4.3 Source agreement

The `sourceAgreement` parameter (0..3) passed to `proposeOutcome` comes from tier 4 of your pipeline. Values: 0 = no external sources, 1 = one, 2 = two agree (Fast tier threshold), 3 = three independent sources agree. Submit high sourceAgreement **only** when you independently verified via distinct data feeds (e.g. ESPN API + official league API + Sportradar).

4.4 Cost ceilings

```
max_spend_per_match = expected_revenue_per_match × safety_factor

# Example at 20 matches/day, $5 avg share:
# expected_revenue = $5 → max_spend = $5 × 0.5 = $2.50 (50% safety)
# Upper bound = tier 6 cap of ~$0.80 per match
# Leaves buffer for infra + opportunity cost
```

4.5 When to abstain

- You can't reach quorum confidence (≥ 60%) even after tier 6
- The match has conflicting data feeds and you don't have independent verification
- The match is paused at any tier (`pauseTierOf(matchId) != None`)
- Your per-match spend cap is hit before reaching confidence threshold

Abstaining does not slash you. Submitting a wrong grade does.

5. Slashing Scenarios

5.1 What gets you slashed

- **Signing a proposal the arbiter overrules.** If the 5-of-9 arbiter rules your proposed outcome wrong, every grader who signed it is a slash candidate via DisputeManager.arbiterResolve → GraderRegistryV2.slashWithChallenger.
- **Submitting a grade outside your panel.** Currently prevented off-chain by the aggregator; future versions may enforce on-chain via Merkle proof in submitGrade.
- **Manipulating confidence to mint false Fast-tier finalizations.** Caught post-hoc by the dispute/challenger system.

5.2 What does NOT get you slashed

- Abstaining from a match (not submitting a grade)
- Submitting a grade that ends up in a successful dispute **as a dissenter** (dissenters are not slashed; only signers of the losing proposal are)
- Deregistering after a grade you feel unsure about (7-day cooldown exists precisely for this window)

5.3 Slash distribution (you are the victim)

Recipient	Share	Amount (if slashed $25k)
Challenger	50%	$12,500
Treasury retained	20%	$5,000
SBET stakers	20%	$5,000
Community bounty	10%	$2,500
Total	**100%**	**$25,000**

No burn. Your stake flows to your counterparties. The challenger who caught your cheating captures 50% — this is the economic reward that makes honest dispute-raising positive-EV.

5.4 How to recover after a slash

If your stake falls below the floor, status → `Slashed` and you cannot submit grades. To reactivate:

```
await sbet.approve(graderRegistryAddress, parseEther('25000'));
await graderRegistry.increaseStake(parseEther('25000'));
// status automatically transitions back to Active
```

5.5 Disputing a slash against you

Today, slashing flows directly from `DisputeManager.arbiterResolve`. You cannot appeal the arbiter's ruling. **Your defense path is pre-emptive:**

- If you see a proposed outcome you believe is wrong, **dissent** — submit your correct grade. Dissenters are not slashed.
- If you signed in error and the challenge window is still open, **open a dispute yourself** with the corrected outcome. Accept the bond cost to avoid slashing.
- Off-chain: maintain a detailed audit log of every grade (model outputs, sources consulted, timestamps). This is your evidence if the arbiter review is ever re-opened via governance.

6. VRF Panel Rotation

For each match, **13 graders are randomly drawn** from the active pool using Chainlink VRF v2.5. Quorum is 9. You are not on every match.

6.1 How to know you're on a panel

When a match is registered, `MatchRegistered` is emitted with a `panelRoot` (keccak256 Merkle root of sorted panel addresses). The aggregator publishes the full panel list off-chain. Your grader checks:

```
const proof = /* Merkle proof from off-chain index */;
const onPanel = await sbetCore.verifyPanelMembership(
matchId, graderAddress, proof
);
if (!onPanel) return; // skip this match
```

6.2 Drop-out tolerance

Panel 13, quorum 9 → the panel tolerates **4 dropouts**. If you're briefly offline, the remaining graders can still reach quorum. If 5+ panel members go offline, the match cannot be proposed and may eventually be voided.

6.3 VRF fulfillment delays

If Chainlink VRF does not fulfill within 24h, `seedFromBlockhashFallback(matchId)` is permissionless. Your panel membership is still determined by the sealed `panelRoot`, but the draw used `blockhash(matchCreationBlock)` instead of VRF. Still grade these matches; flag them for extra scrutiny in your monitoring.

7. Onboarding Checklist

Before going live on mainnet, confirm every item below:

- SBET in wallet ≥ $25k + gas buffer (ETH or network token)
- HSM / hardware signer configured and tested on testnet
- Dedicated L2 RPC (Base / Arbitrum) + fallback RPC provisioned
- Event subscription to MatchRegistered, VRFSeedFulfilled, OutcomeProposed, MatchFinalized, GraderSlashed
- 6-tier AI pipeline tested on ≥ 50 historical matches; accuracy ≥ 95%
- Cost budget set per match with tier-6 escalation cap
- Monitoring dashboard for your own gradesSubmitted, slashedTotal, uptime
- Alerting on pause events (MatchLocked, CategoryPauseActivated, GlobalPauseActivated)
- Runbook for abstain-and-investigate when confidence < 60%
- Shadow-grading mode completed (submit grades to logs without on-chain transactions for 1 week)

- Audit log pipeline writing every grade submission with model outputs + sources
- Key recovery procedure documented (HSM backup, stake key separation)

8. Incident Runbook

Common incidents and the steps to take:

8.1 Grader offline

Symptom: your daemon crashed or lost RPC connection. **Panel impact:** 4 dropouts tolerated; you are 1 of 13. **Actions:** (1) restart daemon / fail over to backup RPC; (2) reconcile missed matches — check which you were drawn onto; (3) if > 4 hours offline, audit your reputation impact; (4) if > 24h, consider requesting deregister temporarily.

8.2 AI cost spike

Symptom: daily spend exceeds budget. **Causes:** many matches hitting tier 4+; provider price change; prompt drift. **Actions:** (1) check per-match spend distribution — is any single match blowing through tier 6?; (2) if yes, audit and abstain; (3) throttle tier 4 to single-provider verification; (4) if provider price changed, update spend cap calculations in config.

8.3 You are a dissenter on a finalized outcome

Symptom: you graded outcome A; the panel proposed B; B was finalized. **Risk:** if B is correct, you are fine (dissenters not slashed). If B is wrong and gets disputed later, your audit log may be called on. **Actions:** (1) preserve your evidence / model outputs for this match; (2) if you are confident B is wrong and the challenge window is open, *open a dispute yourself*; (3) if the window elapsed, contact the category guardian for a category pause review.

8.4 Dispute raised against a proposal you signed

Symptom: `DisputeOpened` fires on a match you signed. **Risk:** if the arbiter rules against your proposal, you are in the slash-candidate pool. **Actions:** (1) immediately assemble your evidence for this match; (2) publish your reasoning to the grader discord channel (other signers may corroborate); (3) track the dispute state — each round adds 24h; (4) if round 3 triggers `AwaitingArbiter`, prepare to submit evidence to the 5-of-9 off-chain.

8.5 Suspected key compromise

Symptom: unusual signing activity; unexplained `submitGrade` calls. **Actions:** (1) immediately call `requestDeregister` to freeze further grade submissions; (2) rotate HSM; (3) contact governance multisig for emergency pause consideration; (4) document timeline for post-incident review.

Emergency contacts. Grader discord channel (invite-only, per active grader). Contact a CATEGORY_GUARDIAN_ROLE holder for a 3-of-5 category pause, or a GLOBAL_GUARDIAN_ROLE holder for a 5-of-9 global pause if you observe protocol-level compromise.