Event Fund Claims

Author: Josh S. Sakweli, Backend Lead — QBIT SPARK CO LIMITED
Version: v1.0
Base URL: `/api/v1/e-events/claims`

Overview

When an event organizer sells tickets, revenue is held in escrow — one escrow entry per buyer checkout session. A **Fund Claim** is a formal request to release **all currently** `HELD` **escrows** for an event into the organizer's wallet. Every claim goes through a `PENDING → APPROVED / REJECTED` lifecycle, and funds only move on explicit admin approval.

Claim approval is an all-or-nothing escrow operation — it releases every `HELD` escrow for that event at the moment of approval. Partial escrow release is not supported by design (see Financial Safety). What the organizer does with the wallet balance after release is handled by the separate `DisbursementService`.

Authentication: All endpoints require `Authorization: Bearer <jwt_token>`

Business Rules

Rule	Detail
One pending per event	At most ONE `PENDING` claim per event at any time
Claimable amount formula	`totalRevenue − totalRefunded − totalClaimed − totalPendingClaims`
Admin-only for active events	If event has not ended, only ADMIN can initiate a claim
Refund deadline	3 days before event start. Past this → no refunds, organizer may claim early
Escrow release	Approval releases ALL `HELD` escrows for the event atomically → organizer wallet. No partial release
No partial amounts	Escrows are atomic per buyer. "80% release" has no business meaning at the escrow layer — staging happens via `DisbursementService` after funds land in wallet
Admin claims require note	`adminNote` is mandatory on admin-initiated claims

Rule	Detail
Refund safety	Refunds and approvals use pessimistic write lock on EscrowAccount to prevent race conditions

Claim Status Lifecycle

Status	Meaning
`PENDING`	Submitted, awaiting admin review. Escrow NOT yet released
`APPROVED`	Admin approved. Escrow atomically released to organizer wallet
`REJECTED`	Admin rejected. No funds moved. Organizer may submit a new claim
`CANCELLED`	Organizer cancelled before admin action. No funds moved

Standard Response Format

Success

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Fund claim submitted successfully",
  "action_time": "2026-04-27T10:30:45",
  "data": { }
}
```

Error

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "A pending claim already exists for this event",
  "action_time": "2026-04-27T10:30:45",
  "data": "A pending claim already exists for this event"
```

```
}
```

Endpoints

1. List All Claims (Admin)

Endpoint: `GET /api/v1/e-events/claims`
Access: 🔒 `ROLE_SUPER_ADMIN` or `ROLE_STAFF_ADMIN`
Purpose: Returns all fund claims across all events. Optionally filter by status.

Query Parameters:

Parameter	Type	Required	Description	Default
`status`	`ClaimStatus` enum	No	Filter by: `PENDING`, `APPROVED`, `REJECTED`, `CANCELLED`	— (all returned)

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Fund claims retrieved",
  "data": [
    {
      "claimId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
      "claimNumber": "EFC-2026-000001",
      "eventId": "uuid",
      "eventTitle": "Dar Jazz Night",
      "eventStatus": "PUBLISHED",
      "organizerId": "uuid",
      "organizerName": "Amina Hassan",
      "status": "PENDING",
      "claimedAmount": 50000.00,
      "totalRevenueSnapshot": 80000.00,
      "totalRefundedSnapshot": 5000.00,
      "totalPreviouslyClaimedSnapshot": 0.00,
      "totalPendingAtSubmission": 0.00,
```

```
      "currency": "TZS",
      "adminInitiated": false,
      "adminId": null,
      "adminNote": null,
      "organizerNote": "Requesting first partial claim",
      "reviewedById": null,
      "reviewerName": null,
      "reviewNote": null,
      "reviewedAt": null,
      "escrowsReleasedCount": 0,
      "escrowsSkippedCount": 0,
      "actualReleasedAmount": null,
      "initiatedAt": "2026-04-20T08:00:00",
      "updatedAt": "2026-04-20T08:00:00"
    }
  ]
}
```


Response Fields:

Field	Type	Description
`claimId`	`UUID`	Unique claim identifier
`claimNumber`	`string`	Human-readable reference, e.g. `EFC-2026-000001`
`eventId`	`UUID`	The event this claim belongs to
`eventTitle`	`string`	Display name of the event
`eventStatus`	`EventStatus`	Current event status
`organizerId`	`UUID`	Organizer's account ID
`organizerName`	`string`	Organizer's display name
`status`	`ClaimStatus`	Current claim status
`claimedAmount`	`BigDecimal`	Amount the organizer requested
`totalRevenueSnapshot`	`BigDecimal`	Organizer's net revenue at time of submission
`totalRefundedSnapshot`	`BigDecimal`	Total refunds issued at time of submission
`totalPreviouslyClaimedSnapshot`	`BigDecimal`	Already released to wallet before this claim
`totalPendingAtSubmission`	`BigDecimal`	Other pending claim amounts at submission time

Field	Type	Description
`currency`	`string`	Currency code, e.g. `TZS`
`adminInitiated`	`boolean`	`true` if an admin submitted on behalf of organizer
`adminId`	`UUID`	Admin's account ID (null if organizer-initiated)
`adminNote`	`string`	Admin's reason for initiating (null if organizer-initiated)
`organizerNote`	`string`	Optional note from organizer
`reviewedById`	`UUID`	ID of admin who reviewed (null if pending)
`reviewerName`	`string`	Name of reviewing admin (null if pending)
`reviewNote`	`string`	Admin's review note (null if pending)
`reviewedAt`	`LocalDateTime`	Timestamp of review (null if pending)
`escrowsReleasedCount`	`Integer`	Number of escrow entries released on approval
`escrowsSkippedCount`	`Integer`	Escrow entries skipped (e.g. already refunded)
`actualReleasedAmount`	`BigDecimal`	Actual amount credited to wallet on approval
`initiatedAt`	`LocalDateTime`	When the claim was submitted
`updatedAt`	`LocalDateTime`	Last updated timestamp

Possible Errors:

HTTP Status	Description
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Caller does not have admin role

---

2. Get My Claims (Organizer)

Endpoint: `GET /api/v1/e-events/claims/my-claims`
Access: □□ Authenticated organizer (any role)
Purpose: Returns all fund claims submitted by the currently authenticated organizer, across all their events.

No query parameters.

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "My fund claims retrieved",
  "data": [ /* array of EventFundClaimResponse — same structure as endpoint 1 */ ]
}
```

Possible Errors:

HTTP Status	Description
`401 UNAUTHORIZED`	Invalid or expired JWT

---

3. Get Claim by ID

Endpoint: `GET /api/v1/e-events/claims/{claimId}`
Access: □□ Authenticated — admin sees any claim; organizer sees only their own
Purpose: Returns full details of a single fund claim.

Path Parameters:

Parameter	Type	Required	Description
`claimId`	`UUID`	Yes	The claim's unique identifier

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Claim retrieved",
  "data": { /* EventFundClaimResponse */ }
}
```

Possible Errors:

HTTP Status	Description
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Organizer attempting to view another organizer's claim

HTTP Status	Description
`404 NOT_FOUND`	Claim with given ID does not exist

---

4. Get Event Claims

Endpoint: `GET /api/v1/e-events/claims/event/{eventId}`
Access: □□ Authenticated — organizer (own events) or admin (any event)
Purpose: Returns all claims ever submitted for a specific event.

Path Parameters:

Parameter	Type	Required	Description
`eventId`	`UUID`	Yes	The event's unique identifier

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Event claims retrieved",
  "data": [ /* array of EventFundClaimResponse */ ]
}
```

Possible Errors:

HTTP Status	Description
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Organizer does not own this event
`404 NOT_FOUND`	Event not found

---

5. Get Claimable Amount

Endpoint: `GET /api/v1/e-events/claims/event/{eventId}/claimable-amount`
Access: □□ Authenticated — organizer (own events) or admin
Purpose: Returns the current claimable amount breakdown for an event, including eligibility status and the active pending claim if one exists.

Path Parameters:

Parameter	Type	Required	Description
`eventId`	`UUID`	Yes	The event's unique identifier

Success Response:


```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Claimable amount retrieved",
  "data": {
    "eventId": "uuid",
    "eventTitle": "Dar Jazz Night",
    "totalRevenue": 80000.00,
    "totalRefunded": 5000.00,
    "totalClaimed": 0.00,
    "totalPendingClaims": 0.00,
    "claimableAmount": 60000.00,
    "currency": "TZS",
    "eligible": true,
    "ineligibilityReason": null,
    "activePendingClaimId": null,
    "refundDeadline": "2026-05-10T00:00:00+03:00",
    "pastRefundDeadline": false
  }
}
```


> **Note on claimable amount**: `claimableAmount = totalRevenue − totalRefunded − totalClaimed − totalPendingClaims`. This is a **snapshot estimate** used to indicate eligibility — the `actualReleasedAmount` on approval is the real number, as it reflects the live `HELD` escrow balance at the moment of approval.

Response Fields:

Field	Type	Description
`totalRevenue`	`BigDecimal`	Net organizer share of all non-refunded tickets
`totalRefunded`	`BigDecimal`	Gross amount returned to buyers
`totalClaimed`	`BigDecimal`	Already released to organizer wallet

Field	Type	Description
`totalPendingClaims`	`BigDecimal`	Locked by current `PENDING` claims
`claimableAmount`	`BigDecimal`	What the organizer can claim right now
`eligible`	`boolean`	Whether a new claim can be submitted
`ineligibilityReason`	`string`	Human-readable reason if `eligible = false`
`activePendingClaimId`	`UUID`	ID of the existing pending claim, or `null`
`refundDeadline`	`ZonedDateTime`	`eventStartDate − 3 days`
`pastRefundDeadline`	`boolean`	`true` = refund window closed, full amount claimable

Possible Errors:

HTTP Status	Description
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Organizer does not own this event
`404 NOT_FOUND`	Event not found

---

6. Get Revenue Summary (Admin)

Endpoint: `GET /api/v1/e-events/claims/event/{eventId}/revenue-summary`
Access: 🔒 `ROLE_SUPER_ADMIN` or `ROLE_STAFF_ADMIN`
Purpose: Returns a full financial summary of an event including gross sales, platform fees, refunds, claimed amounts, and escrow balance. Used for admin audit and oversight.

Path Parameters:

Parameter	Type	Required	Description
`eventId`	`UUID`	Yes	The event's unique identifier

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Event revenue summary retrieved",
```

```
  "data": {
    "eventId": "uuid",
    "eventTitle": "Dar Jazz Night",
    "grossRevenue": 100000.00,
    "totalRefunded": 5000.00,
    "platformFees": 10000.00,
    "netOrganizerRevenue": 85000.00,
    "totalClaimed": 0.00,
    "totalPendingClaims": 0.00,
    "escrowBalance": 85000.00,
    "currency": "TZS"
  }
}
```

Possible Errors:

HTTP Status	Description
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Caller does not have admin role
`404 NOT_FOUND`	Event not found

---

7. Submit Claim (Organizer)

Endpoint: `POST /api/v1/e-events/claims/event/{eventId}`
Access: □□ Authenticated organizer — must own the event
Purpose: Organizer submits a fund claim for the full current claimable amount. The claim amount is computed server-side — organizer cannot override it.

Path Parameters:

Parameter	Type	Required	Description
`eventId`	`UUID`	Yes	The event to submit a claim for

Request Body (optional):

```
{
  "organizerNote": "Requesting first partial claim before event ends"
}
```

Request Body Fields:

Field	Type	Required	Description	Validation
`organizerNote`	`string`	No	Optional note from the organizer	Max 1000 characters

Guard Rules Applied Server-Side:

- Event must belong to the authenticated organizer
- Event must have ended **OR** `pastRefundDeadline = true` (otherwise only admin can claim)
- `claimableAmount` must be > 0
- No existing `PENDING` claim for this event

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Fund claim submitted successfully",
  "data": { /* EventFundClaimResponse with status: PENDING */ }
}
```

Possible Errors:

HTTP Status	Description
`400 BAD_REQUEST`	A pending claim already exists for this event
`400 BAD_REQUEST`	Claimable amount is zero — nothing to claim
`400 BAD_REQUEST`	Event has not ended and refund deadline has not passed — only admin can claim
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Authenticated user does not own this event
`404 NOT_FOUND`	Event not found

---

8. Admin-Initiate Claim

Endpoint: `POST /api/v1/e-events/claims/event/{eventId}/admin-initiate`
Access: 🔒 `ROLE_SUPER_ADMIN` or `ROLE_STAFF_ADMIN`
Purpose: Admin submits a fund claim on behalf of the organizer. Used when the event is still active (ticket sales ongoing). The claim is flagged `adminInitiated = true` with a mandatory audit note.

Path Parameters:

Parameter	Type	Required	Description
`eventId`	`UUID`	Yes	The event to claim funds for

Request Body:

```
{
  "adminNote": "Organizer requested early release. Event ending tomorrow, no pending refunds."
}
```

Request Body Fields:

Field	Type	Required	Description	Validation
`adminNote`	`string`	**Yes**	Mandatory reason for admin-initiated claim	Not blank

Guard Rules Applied Server-Side:

- `claimableAmount` must be > 0
- No existing `PENDING` claim for this event
- Claim is capped at `claimableAmount` (never full `totalRevenue`)

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Admin-initiated claim created",
  "data": {
    "claimId": "uuid",
    "claimNumber": "EFC-2026-000003",
    "status": "PENDING",
    "adminInitiated": true,
    "adminId": "uuid-of-admin",
    "adminNote": "Organizer requested early release. Event ending tomorrow, no pending
refunds.",
    "claimedAmount": 64000.00,
    "currency": "TZS"
  }
}
```

Possible Errors:

HTTP Status	Description
`400 BAD_REQUEST`	A pending claim already exists for this event
`400 BAD_REQUEST`	Claimable amount is zero — nothing to claim
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Caller does not have admin role
`404 NOT_FOUND`	Event not found
`422 UNPROCESSABLE_ENTITY`	`adminNote` is blank

---

9. Approve Claim (Admin)

Endpoint: `POST /api/v1/e-events/claims/{claimId}/approve`
Access: 🔒 `ROLE_SUPER_ADMIN` or `ROLE_STAFF_ADMIN`
Purpose: Admin approves a pending claim. This atomically acquires a pessimistic write lock on the `EscrowAccount`, fetches **all escrows currently in `HELD` status** for the event, releases every one of them, and credits the full released amount to the organizer's wallet.

> **Design note**: Approval releases ALL `HELD` escrows — not just the `claimedAmount` subset. Each escrow is atomic (one buyer's payment for one checkout session) so partial release has no valid business meaning. The `claimedAmount` on the claim is a snapshot estimate taken at submission time. The `actualReleasedAmount` stored on approval is the authoritative number — it reflects the live `HELD` balance at the moment the admin approves, and may differ if refunds were processed in between.

Path Parameters:

Parameter	Type	Required	Description
`claimId`	`UUID`	Yes	The claim to approve

Request Body (optional):

```
{
  "reviewNote": "Verified escrow balance. Approved for release."
}
```

Request Body Fields:

Field	Type	Required	Description
`reviewNote`	`string`	No	Optional note from reviewing admin

> **Critical**: Approval acquires a `PESSIMISTIC_WRITE` lock on the `EscrowAccount` row. Concurrent refund operations for the same event will block until the approval transaction completes, preventing escrow from going negative.

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Fund claim approved and funds released",
  "data": {
    "claimId": "uuid",
    "claimNumber": "EFC-2026-000001",
    "status": "APPROVED",
    "claimedAmount": 50000.00,
    "actualReleasedAmount": 48000.00,
    "escrowsReleasedCount": 12,
    "escrowsSkippedCount": 1,
    "reviewedById": "admin-uuid",
    "reviewerName": "Admin John",
    "reviewNote": "Verified escrow balance. Approved for release.",
    "reviewedAt": "2026-04-27T14:30:00"
  }
}
```

> **Note on `actualReleasedAmount`**: May be less than `claimedAmount` if one or more escrow entries were refunded between claim submission and admin approval. The `escrowsSkippedCount` indicates how many entries were skipped for this reason.

Possible Errors:

HTTP Status	Description
`400 BAD_REQUEST`	Claim is not in `PENDING` status

HTTP Status	Description
`400 BAD_REQUEST`	Escrow balance insufficient to cover claim amount
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Caller does not have admin role
`404 NOT_FOUND`	Claim not found

---

10. Reject Claim (Admin)

Endpoint: `POST /api/v1/e-events/claims/{claimId}/reject`
Access: 🔒 `ROLE_SUPER_ADMIN` or `ROLE_STAFF_ADMIN`
Purpose: Admin rejects a pending claim. No funds are moved. The organizer may submit a new claim after rejection.

Path Parameters:

Parameter	Type	Required	Description
`claimId`	`UUID`	Yes	The claim to reject

Request Body (optional):

```
{
  "reviewNote": "Pending dispute investigation. Please resubmit after resolution."
}
```

Request Body Fields:

Field	Type	Required	Description
`reviewNote`	`string`	No	Reason for rejection (shown to organizer)

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Fund claim rejected",
  "data": {
    "claimId": "uuid",
    "claimNumber": "EFC-2026-000001",
    "status": "REJECTED",
```

```
    "reviewedById": "admin-uuid",
    "reviewerName": "Admin John",
    "reviewNote": "Pending dispute investigation. Please resubmit after resolution.",
    "reviewedAt": "2026-04-27T14:35:00"
  }
}
```

Possible Errors:

HTTP Status	Description
`400 BAD_REQUEST`	Claim is not in `PENDING` status
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Caller does not have admin role
`404 NOT_FOUND`	Claim not found

---

11. Cancel Claim (Organizer)

Endpoint: `DELETE /api/v1/e-events/claims/{claimId}`
Access: □□ Authenticated organizer — must be the claim owner
Purpose: Organizer cancels their own pending claim before any admin action. This frees up the pending amount, allowing a new claim to be submitted.

Path Parameters:

Parameter	Type	Required	Description
`claimId`	`UUID`	Yes	The claim to cancel

No request body.

Success Response:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Fund claim cancelled",
  "data": null
}
```

Possible Errors:

HTTP Status	Description
`400 BAD_REQUEST`	Claim is not in `PENDING` status (already reviewed)
`401 UNAUTHORIZED`	Invalid or expired JWT
`403 FORBIDDEN`	Organizer does not own this claim
`404 NOT_FOUND`	Claim not found

Financial Safety: Refund & Claim Concurrency

This section explains how the system prevents money from being over-released when a refund and a claim approval happen simultaneously, and why partial escrow release is not supported.

Why Partial Release Is Not Supported

Each escrow is atomic — it represents one buyer's payment for one checkout session. To release "80% of claimable funds" the service would have to arbitrarily pick specific escrow entries to release and leave others `HELD`. There is no business logic that justifies which individual buyer's payment stays locked — the organizer delivered the event to all of them equally.

```
10 escrows × 950 TZS = 9,500 TZS claimable
"Release 80%" = 7,600 TZS = release 8 escrows, hold 2
→ WHY are those 2 buyers' payments still locked?
→ No valid answer exists at the escrow layer.
```

The correct model is:

```
Claim approval → ALL HELD escrows released → Organizer wallet (full amount lands)
                                                          ↓
                                                Organizer controls staged
                                                payouts via DisbursementService
```

The organizer's wallet is the right layer for staged payouts. The claim domain's job is simply: verify eligibility, get admin approval, atomically convert all `HELD` escrows into wallet balance.

The Race Condition Risk

```
Revenue = 10,000 TZS in HELD escrows
PENDING claim submitted
→ Refund of 2,000 processed concurrently
→ Claim approved → releases all HELD (now 8,000)
→ Refund deducts 2,000 from already-decremented escrow
→ Escrow goes negative □□
```

Protections in Place

Layer	Mechanism
Refund deadline	No refunds allowed within 3 days of event. Eliminates the race entirely post-deadline.
Pessimistic DB lock	Both `approveClaim()` and `processRefund()` acquire `PESSIMISTIC_WRITE` lock on the `EscrowAccount` row — serializing the two operations.
Balance check	Before any deduction, service verifies `escrow.balance >= amount`. Throws `InsufficientEscrowException` if not.
Snapshot vs actual	`claimedAmount` is a snapshot estimate at submission time. `actualReleasedAmount` is the real released amount at approval time — reflects live `HELD` balance after any intervening refunds.
`escrowsSkippedCount`	Tracks how many escrow entries were already `REFUNDED` or `DISPUTED` at approval time and therefore skipped.

Claimable Amount Formula

```
claimableAmount = totalRevenue − totalRefunded − totalClaimed − totalPendingClaims
```

This formula gives an estimate of what's claimable. The actual amount released on approval equals the live sum of all `HELD` escrow `sellerAmount` values at that moment.

Endpoint Quick Reference

#	Method	Path	Access	Description
1	`GET`	`/claims`	Admin	List all claims (filter by status)
2	`GET`	`/claims/my-claims`	Organizer	Get my own claims
3	`GET`	`/claims/{claimId}`	Organizer / Admin	Get single claim by ID

#	Method	Path	Access	Description
4	`GET`	`/claims/event/{eventId}`	Organizer / Admin	Get all claims for an event
5	`GET`	`/claims/event/{eventId}/claimable-amount`	Organizer / Admin	Get claimable amount breakdown
6	`GET`	`/claims/event/{eventId}/revenue-summary`	Admin	Full event revenue summary
7	`POST`	`/claims/event/{eventId}`	Organizer	Submit a new claim
8	`POST`	`/claims/event/{eventId}/admin-initiate`	Admin	Admin-initiate claim for active event
9	`POST`	`/claims/{claimId}/approve`	Admin	Approve claim + release escrow
10	`POST`	`/claims/{claimId}/reject`	Admin	Reject claim
11	`DELETE`	`/claims/{claimId}`	Organizer	Cancel pending claim

Standard Error Reference

HTTP Status	Scenario
`400 BAD_REQUEST`	Business rule violation (duplicate pending, zero claimable, wrong status)
`401 UNAUTHORIZED`	Missing, expired, or malformed JWT
`403 FORBIDDEN`	Insufficient role or ownership violation
`404 NOT_FOUND`	Event or claim does not exist
`422 UNPROCESSABLE_ENTITY`	Validation failure (e.g. blank `adminNote`)
`500 INTERNAL_SERVER_ERROR`	Unexpected server error

Revision #3
Created 27 April 2026 15:04:36 by Admin Qbit
Updated 27 April 2026 15:07:10 by Admin Qbit