# UNITE Blood Bank System: API Technical Reference

Version: 1.0.0

Date: December 06, 2025

Document Status: Final

## 1. Introduction

The UNITE Backend API is a RESTful service designed to manage the Bicol Medical Center Blood Bank Event Management System. It facilitates the planning, approval, scheduling, and monitoring of blood-related activities through a centralized platform.

### 1.1 Base Configuration

- **Protocol:** HTTP/1.1 (HTTPS in Production)
- **Data Format:** JSON
- **Base URL (Dev):** http://localhost:3000/api
- **Base URL (Prod):** Configurable via environment variables.

## 2. Authentication & Security

The API utilizes **JWT (JSON Web Token)** for stateless authentication.

- Header Requirement:
  All protected endpoints require the following header:
  Authorization: Bearer <your_jwt_token>
- Token Payload:
  Tokens contain the user's id, role, email, and StaffType.
- **Role-Based Access Control (RBAC):**
  - **Admin:** Full system access.
  - **Coordinator:** District-specific operations and event requests.
  - **Stakeholder:** Limited access for event requests and profile management.

## 3. Response Conventions

All API responses follow a standardized envelope structure.

**Success Response:**

```
JSON
{
  "success": true,
  "data": { ... },     // Requested resource or object
  "message": "..."     // Optional status message
}
```

**Error Response:**

```
JSON
{
```

```
  "success": false,
  "message": "Error description",
  "error": "..."       // Detailed stack trace (Development only)
}
```

**Paginated Response:**

```
JSON
{
  "success": true,
  "data": [ ... ],
  "pagination": {
    "page": 1,
    "limit": 10,
    "total": 100,
    "pages": 10
  }
}
```

**4. API Endpoint Reference**

**4.1 Authentication**

| **Method** | **Endpoint** | **Description** | **Access** |
|---|---|---|---|
| **POST** | /auth/login | Authenticates a user. Body: { email, password }. Returns JWT. | Public |
| **POST** | /auth/stakeholders/login | Specific login for external stakeholders. | Public |
| **GET** | /auth/me | Retrieves currently authenticated user details. | Private |

| | | | |
|---|---|---|---|
| **POST** | /auth/logout | Clears authentication cookies. | Public |

### 4.2 User Management

Manages the three primary user roles: Admins, Coordinators, and Stakeholders.

#### System Administrators

- GET /admin/dashboard: Retrieval of admin-specific dashboard metrics.
- GET /admin/statistics: System-wide statistics.
- POST /admin: Create a new System Admin (Admin only).

#### Coordinators

- GET /coordinators: List all coordinators.
- GET /coordinators/:id/dashboard: Retrieval of district-specific metrics.
- POST /coordinators/:id/registration-codes: Generate registration codes for stakeholders.

#### Stakeholders

- POST /stakeholders/register: Public registration endpoint.
- GET /stakeholders: List stakeholders (Admin/Coordinator access).

### 4.3 Event Management

Comprehensive management of blood drives, advocacy, and training events.

#### Calendar & Visualization

- GET /calendar/month: Aggregated events for monthly view.
- GET /calendar/upcoming: Summary of imminent events.
- GET /events/statistics/dashboard: High-level event metrics for dashboard widgets.

#### Core Event Operations

- GET /events: Retrieve all events with filtering, sorting, and pagination.
- GET /events/:eventId: Detailed view of a single event including category-specific data.
- GET /events/:eventId/completeness: Utility to check if all required event fields are populated.

### 4.4 Request Workflow

The core logic for scheduling events, enforcing the "Double-Confirmation" approval process.

#### Event Requests

- POST /requests: Submit a new event proposal (Coordinator/Stakeholder).
- POST /events/direct: Bypass workflow to create an immediate event (Admin/Coordinator).
- POST /requests/validate: Pre-check for rule violations (e.g., weekend bans, capacity limits).

#### Workflow Actions

- POST /requests/:id/admin-action: Admin approves, rejects, or reschedules.
- POST /requests/:id/coordinator-action: Coordinator accepts or rejects an assignment.
- POST /requests/:id/coordinator-confirm: Final confirmation step for the coordinator.
- POST /requests/:id/staff: Assign staff members to an approved event (Admin only).

#### Validation Utilities

- GET /requests/check-overlap: Checks if a coordinator has overlapping commitments.
- GET /requests/check-double-booking: Checks for location/date conflicts.

### 4.5 Chat System

Real-time communication endpoints supporting direct and group messaging.

- POST /chat/messages: Send a text message.
- GET /chat/messages/:conversationId: Retrieve message history.
- GET /chat/conversations: List all active conversations for the user.
- GET /chat/presence/online: Retrieve a list of currently online users.

#### Permission Rules:

- **Admins:** Chat with Coordinators.
- **Coordinators:** Chat with Admins and Stakeholders.
- **Stakeholders:** Chat with Coordinators only.

### 4.6 Inventory (Blood Bags)

- POST /bloodbags: Add new blood bags to inventory.
- GET /bloodbags: Inventory list.
- PUT /bloodbags/:id: Update status (e.g., available, reserved, expired).

#### 4.7 Utility & Location

- **Notifications:** GET /notifications (User's list), PUT /notifications/mark-all-read.
- **Geography:** GET /locations/provinces, GET /locations/districts/:id/municipalities (For dropdown population).
- **Signup Requests:** POST /signup-requests (Public signup request submission).

## 5. System Architecture & Workflows

### 5.1 Request State Machine

The system enforces a strict state machine for event requests to ensure data integrity.

**Key States:**

- pending-review: Initial submission.
- review-rescheduled: Admin/Reviewer proposed a new date.
- awaiting-confirmation: Approved by reviewer, waiting for requester to finalize.
- approved: Fully confirmed and scheduled.

### 5.2 Real-Time Architecture

The backend uses **Socket.IO** for live updates.

- **Events:** new_message, typing_start, user_online.
- **Connection:** Authenticated via JWT in the handshake.

## 6. Error Handling

The API returns standard HTTP status codes alongside the JSON error envelope.

- **200 OK:** Successful operation.
- **400 Bad Request:** Validation failure (e.g., missing fields).
- **401 Unauthorized:** Missing or invalid JWT.
- **403 Forbidden:** Valid JWT but insufficient permissions (e.g., Stakeholder accessing Admin route).
- **404 Not Found:** Resource does not exist.
- **409 Conflict:** Duplicate key error (e.g., email already exists).
- **500 Internal Server Error:** General system failure.

## 7. Deployment

- **Health Check:** GET /health returns server status and database connection state.
- **Environment Variables:** Requires MONGODB_URI, JWT_SECRET, and NODE_ENV.