

# EventHub API Documentation


**Version:** 1.3.0  
**Author:** Nathaniel Sebastian ([sc232ns@leeds.ac.uk](mailto:sc232ns@leeds.ac.uk))  
**Last Updated:** 5th February 2026

## Table of Contents

- 1. [Base URLs](#)
- 2. [Authentication & Authorization](#)
- 3. [Response Headers](#)
- 4. [Error Handling](#)
- 5. [Endpoints Reference](#)
- 6. [End-to-End Example](#)
- 7. [Running Locally](#)

## Base URLs

Environment	URL
Production	<a href="https://comp3011-cw1-api.onrender.com">https://comp3011-cw1-api.onrender.com</a>
Local	<a href="http://127.0.0.1:8000">http://127.0.0.1:8000</a>

 Production uses Render free tier. First request after 15 min inactivity may take 30–60s. If 503, retry after 60s.

**Interactive Docs:** [/docs](#) (Swagger) | [/redoc](#) (ReDoc)

## Authentication & Authorization

### JWT Authentication

Property	Value
Token Type	JWT (Bearer)
Algorithm	HS256
Expiry	30 minutes
Header	Authorization: Bearer <token>

### RBAC (Role-Based Access Control)

Role	Capabilities
Anonymous	Read events, attendees, analytics
Authenticated	Create/update/delete events, RSVPs, attendees
Admin	Dataset import, import logs (/admin/*)

**Admin enforcement:** All `/admin/*` endpoints require `is_admin=True` . Non-admin users receive:

```
{"detail": "The user doesn't have enough privileges"}
```

HTTP status: `403 Forbidden`

**Rate Limiting**

Scope	Limit
Global	120 requests/minute
<code>/auth/login</code>	10 requests/minute

**Response Headers**

All responses include:

Header	Value
<code>X-Request-ID</code>	UUID v4 (for tracing)
<code>X-Content-Type-Options</code>	<code>nosniff</code>
<code>X-Frame-Options</code>	<code>DENY</code>

Cache-Control varies by response type:

Response Type	Cache-Control
GET 200 (with ETag)	<code>no-cache</code>
All other responses	<code>no-store</code>

GET 200 responses also include:

Header	Value
<code>ETag</code>	<code>"&lt;sha256-hash&gt;"</code>

**Conditional GET (ETag)**

```
# First request
curl -i http://127.0.0.1:8000/events
# Response headers include: ETag: "abc123..."

# Conditional request
curl -H "If-None-Match: \"abc123...\"" http://127.0.0.1:8000/events
# Response: 304 Not Modified (no body)
# Headers: ETag, X-Request-ID, Cache-Control: no-cache
```

## Error Handling

### HTTP Status Codes

Code	Meaning
200	OK
201	Created
204	No Content (DELETE)
304	Not Modified (ETag match)
400	Bad Request
401	Unauthorized (missing/invalid JWT)
403	Forbidden (non-admin on /admin/*)
404	Not Found
409	Conflict (duplicate)
422	Validation Error
429	Too Many Requests (rate limit)
500	Internal Server Error

### Error Response Examples

#### 401 Unauthorized:

```
{"detail": "Not authenticated"}
```

#### 403 Forbidden (Admin-only endpoints):

```
{"detail": "The user doesn't have enough privileges"}
```

#### 429 Too Many Requests:

```
{"detail": "Too Many Requests", "request_id": "550e8400-e29b-41d4-a716-446655440000"}
```

Headers include: X-Request-ID: 550e8400-e29b-41d4-a716-446655440000

#### 500 Internal Server Error (sanitized):

```
{"detail": "Internal Server Error", "request_id": "550e8400-e29b-41d4-a716-446655440000"}
```

Note: Stack traces and exception details are never exposed.

---

## Endpoints Reference

### Health Check

GET /health

Returns: `status` ("online"), `database` ("ok"/"error"), `version`, `environment`, `commit`, `timestamp`

### Authentication

POST /auth/register → Create account (JSON body)  
POST /auth/login → Get JWT token (form data: username, password)

### Events

GET /events → List (paginated, filterable, ETag)  
POST /events → Create (auth required)  
GET /events/{id} → Detail (ETag)  
PATCH /events/{id} → Update (auth required)  
DELETE /events/{id} → Delete (auth required)  
GET /events/{id}/stats → RSVP statistics  
GET /events/{id}/rsvps → List RSVPs  
POST /events/{id}/rsvps → Create RSVP (auth required)  
DELETE /events/{id}/rsvps/{rsvp\_id} → Delete RSVP (auth required)

### Attendees

POST /attendees → Create (auth required)  
GET /attendees/{id} → Detail  
GET /attendees/{id}/events → Events for attendee

### Analytics

GET /analytics/events/seasonality → Monthly distribution  
GET /analytics/events/trending → Trending by RSVP activity  
GET /events/recommendations → Personalised (auth required)

### Admin (Admin-only)

POST /admin/imports/run → Trigger import  
GET /admin/imports → List import runs  
GET /admin/dataset/meta → Dataset metadata

---

## End-to-End Example

### 1. Register

```
curl -X POST http://127.0.0.1:8000/auth/register \
-H "Content-Type: application/json" \
-d '{"username":"alice","email":"alice@example.com","password":"SecurePass123"}'
```

## 2. Login

```
TOKEN=$(curl -s -X POST http://127.0.0.1:8000/auth/login \
-d "username=alice&password=SecurePass123" | jq -r '.access_token')
```

## 3. Create Event

```
curl -X POST http://127.0.0.1:8000/events \
-H "Authorization: Bearer $TOKEN" \
-H "Content-Type: application/json" \
-d '{"title":"AI Workshop","location":"Leeds","start_time":"2026-03-15T14:00:00Z","end_time":"2026-03-15T17:00:00Z","capacity":30}'
```

## 4. Create Attendee & RSVP

```
curl -X POST http://127.0.0.1:8000/attendees \
-H "Authorization: Bearer $TOKEN" \
-H "Content-Type: application/json" \
-d '{"name":"Alice Smith","email":"alice@example.com"}'
```

```
curl -X POST http://127.0.0.1:8000/events/1/rsvps \
-H "Authorization: Bearer $TOKEN" \
-H "Content-Type: application/json" \
-d '{"attendee_id":1,"status":"going"}'
```

## 5. Analytics

```
curl http://127.0.0.1:8000/analytics/events/trending
```

## 6. Conditional GET (ETag)

```
# Get ETag
ETAG=$(curl -si http://127.0.0.1:8000/events | grep -i "^etag:" | cut -d' ' -f2 | tr -d '\r')
```

```
# Conditional request
curl -i -H "If-None-Match: $ETAG" http://127.0.0.1:8000/events
```

```
# Returns 304 Not Modified
```

---

## Running Locally

```
git clone https://github.com/NathS04/comp3011-cw1-api.git && cd comp3011-cw1-api
python3 -m venv venv && source venv/bin/activate
pip install -r requirements.txt
export DATABASE_URL="sqlite:///./app.db" SECRET_KEY="dev-secret"
alembic upgrade head
pytest -q      # Expected: 39 passed
uvicorn app.main:app --reload
```

## Changelog

Version	Date	Changes
1.3.0	2026-02-05	ETag/304, security headers on all responses, 429 request_id
1.2.0	2026-02-05	RBAC, rate limiting
1.1.0	2026-02-04	Analytics endpoints
1.0.0	2026-02-01	Initial release