

# EventHub API Documentation

**Version:** 1.2.0

**Author:** Nathaniel Sebastian (sc232ns@leeds.ac.uk)

**Last Updated:** 5th February 2026

---

## Table of Contents

1. [Base URLs](#)
  2. [Authentication](#)
  3. [Events](#)
  4. [Attendees](#)
  5. [RSVPs](#)
  6. [Analytics & Recommendations](#)
  7. [Admin & Dataset Management](#)
  8. [Error Handling](#)
  9. [Running Locally](#)
- 

## Base URLs

Environment	URL
Production	https://comp3011-cw1-api.onrender.com
Local Development	http://127.0.0.1:8000

---

**Interactive Documentation:** - Swagger UI: /docs - ReDoc: /redoc

---

## Authentication

Most endpoints that modify data (POST, PATCH, DELETE) require a valid JWT token. Read operations (GET) are generally public.

### Auth Flow Summary

1. POST /auth/register → Create account
2. POST /auth/login → Receive JWT token
3. Include token in requests: Authorization: Bearer <token>

## Register a New User

**Endpoint:** POST /auth/register

**Auth Required:** No

**Request Body:**

```
{
  "username": "johndoe",
  "email": "john@example.com",
  "password": "mySecurePassword123"
}
```

**Success Response (201 Created):**

```
{
  "id": 1,
  "username": "johndoe",
  "email": "john@example.com",
  "created_at": "2026-02-04T12:30:00Z"
}
```

**Error Responses:** | Status | Meaning | | — — — — — | | 400 | Bad Request | Username or email already registered | | 422 | Unprocessable Entity | Validation failed (e.g., password too short) |

---

## Login

**Endpoint:** POST /auth/login

**Auth Required:** No

**Content-Type:** application/x-www-form-urlencoded

**Request Body (Form Data):**

username=johndoe  
password=mySecurePassword123

**Success Response (200 OK):**

```
{
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "token_type": "bearer"
}
```

**Error Response (401 Unauthorized):**

```
{
  "detail": "Incorrect username or password"
}
```

---

## Using the Token

Include the token in the Authorization header for all protected requests:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

**Token Expiry:** 30 minutes

---

## Events

### List Events

**Endpoint:** GET /events

**Auth Required:** No

**Query Parameters:**

Parameter	Type	Description	Example
limit	integer	Max results (default: 10, max: 100)	?limit=20
offset	integer	Skip N results (default: 0)	?offset=10
sort	string	Sort field, prefix – for desc	?sort=--start_time
q	string	Search title (partial match)	?q=tech
location	string	Filter by location	?location=Leeds
start_after	datetime	Events starting after	?start_after=2026-03-01T00:00:00
start_before	datetime	Events starting before	?start_before=2026-04-01T00:00:00
min_capacity	integer	Minimum capacity	?min_capacity=50
status	string	upcoming or past	?status=upcoming

**Example Request:**

GET /events?limit=5&sort=--start\_time&status=upcoming

**Success Response (200 OK):**

```
{
  "items": [
    {
      "id": 1,
      "title": "Tech Meetup Leeds",
      "description": "Monthly gathering for tech enthusiasts",
      "location": "Leeds Digital Hub",
```

```
    "start_time": "2026-03-15T18:00:00Z",
    "end_time": "2026-03-15T21:00:00Z",
    "capacity": 50,
    "created_at": "2026-02-01T10:00:00Z"
  }
],
"total": 42,
"limit": 5,
"offset": 0
}
```

---

## Create Event

**Endpoint:** POST /events

**Auth Required:** Yes

### Request Body:

```
{
  "title": "Freshers Welcome Social",
  "description": "Welcome event for new students at Leeds
    University.",
  "location": "Leeds Student Union, Riley Smith Hall",
  "start_time": "2026-09-25T19:00:00Z",
  "end_time": "2026-09-25T23:00:00Z",
  "capacity": 200
}
```

**Validation Rules:** | Field | Rules || — — | — — | | title | Required, 1–200 characters || description | Optional, max 1000 characters || location | Required, 1–200 characters | | start\_time | Required, ISO 8601 datetime || end\_time | Required, must be after start\_time || capacity | Required, integer ≥ 1 |

### Success Response (201 Created):

```
{
  "id": 2,
  "title": "Freshers Welcome Social",
  "description": "Welcome event for new students at Leeds
    University.",
  "location": "Leeds Student Union, Riley Smith Hall",
  "start_time": "2026-09-25T19:00:00Z",
  "end_time": "2026-09-25T23:00:00Z",
  "capacity": 200,
  "created_at": "2026-02-04T14:00:00Z"
}
```

---

## Get Event by ID

**Endpoint:** GET /events/{id}

**Auth Required:** No

**Success Response (200 OK):** Returns single event object.

**Error Response (404):** {"detail": "Event not found"}

---

## Update Event

**Endpoint:** PATCH /events/{id}

**Auth Required:** Yes

**Request Body (partial update):**

```
{
  "title": "Tech Meetup Leeds 2026",
  "capacity": 75
}
```

Only include fields you want to update.

---

## Delete Event

**Endpoint:** DELETE /events/{id}

**Auth Required:** Yes

**Success Response:** 204 No Content

---

## Get Event Statistics

**Endpoint:** GET /events/{id}/stats

**Auth Required:** No

**Success Response (200 OK):**

```
{
  "event_id": 1,
  "going": 35,
  "maybe": 10,
  "not_going": 5,
  "remaining_capacity": 15
}
```

**Formula:** remaining\_capacity = capacity - going

---

# Attendees

## Create Attendee

**Endpoint:** POST /attendees

**Auth Required:** Yes

**Request Body:**

```
{
  "name": "Alice Smith",
  "email": "alice.smith@leeds.ac.uk"
}
```

**Success Response (201 Created):**

```
{
  "id": 1,
  "name": "Alice Smith",
  "email": "alice.smith@leeds.ac.uk"
}
```

**Error:** 409 Conflict if email already registered.

---

## Get Attendee by ID

**Endpoint:** GET /attendees/{id}

**Auth Required:** No

---

## Get Events for Attendee

**Endpoint:** GET /attendees/{id}/events

**Auth Required:** No

Returns all events the attendee has RSVP'd to.

---

# RSVPs

## Create RSVP

**Endpoint:** POST /events/{event\_id}/rsvps

**Auth Required:** Yes

**Request Body:**

```
{
  "attendee_id": 1,
```

```
    "status": "going"
  }
```

**Valid status values:** going, maybe, not\_going

**Error Responses:** - 404 – Event or attendee not found - 409 – Duplicate RSVP (attendee already RSVP'd to this event)

---

## List RSVPs for Event

**Endpoint:** GET /events/{event\_id}/rsvps

**Auth Required:** No

---

## Delete RSVP

**Endpoint:** DELETE /events/{event\_id}/rsvps/{rsvp\_id}

**Auth Required:** Yes

---

# Analytics & Recommendations

These endpoints provide derived insights beyond basic CRUD operations.

## Event Seasonality

**Endpoint:** GET /analytics/events/seasonality

**Auth Required:** No

**Purpose:** Aggregate events by month to identify seasonal patterns.

**Success Response (200 OK):**

```
{
  "items": [
    {"month": "2026-01", "count": 5, "top_categories": ["General"]},
    {"month": "2026-02", "count": 12, "top_categories": ["General"]},
    {"month": "2026-03", "count": 8, "top_categories": ["General"]}
  ]
}
```

---

## Trending Events

**Endpoint:** GET /analytics/events/trending

**Auth Required:** No

**Query Parameters:**

Parameter	Type	Default	Description
— — — —   window_days	integer	30	Days to consider for “recent” RSVPs
— — — —   limit	integer	5	Max events to return

### Trending Score Formula:

$\text{score} = (\text{recent\_rsvps} \times 1.5) + (\text{total\_rsvps} \times 0.5)$

### Success Response (200 OK):

```
[
  {
    "event_id": 3,
    "title": "AI Workshop",
    "trending_score": 15.5,
    "recent_rsvps": 8
  },
  {
    "event_id": 7,
    "title": "Networking Night",
    "trending_score": 12.0,
    "recent_rsvps": 6
  }
]
```

---

## Personalised Recommendations

**Endpoint:** GET /events/recommendations

**Auth Required:** Yes

**Purpose:** Suggest upcoming events based on user's RSVP history.

**Algorithm:** 1. Find attendee record matching the authenticated user's email. 2. Identify locations the user has previously attended. 3. Recommend future events at those locations. 4. Cold start: If no history, return top upcoming events.

### Success Response (200 OK):

```
{
  "recommendations": [
    {
      "event_id": 12,
      "title": "Python Meetup",
      "score": 0.9,
      "reason": "Based on your interest in Leeds Digital Hub",
      "location": "Leeds Digital Hub",
      "start_time": "2026-03-20T18:00:00Z"
    }
  ],
  "user_id": 5
}
```

---



# Admin & Dataset Management

These endpoints manage external data integration and are protected by authentication.

## Trigger Dataset Import

**Endpoint:** POST /admin/imports/run

**Auth Required:** Yes

**Query Parameters:**

Parameter	Type	Default	Description
source_type	string	xml	xml (Leeds TEN) or csv (local file)
source_url	string		Leeds XML URL   URL or file path to import

**Example Request:**

```
curl -X POST "http://127.0.0.1:8000/admin/imports/run?source_type=xml" \
-H "Authorization: Bearer $TOKEN"
```

**Success Response (201 Created):**

```
{
  "message": "Import finished successfully",
  "source": "https://opendata.leeds.gov.uk/downloads/Licences/temp-event-notice/temp-event-notice.xml"
}
```

---

## List Import Runs

**Endpoint:** GET /admin/imports

**Auth Required:** Yes

**Query Parameters:**

Parameter	Type	Default	Description
limit	integer	10	Max results to return

**Success Response (200 OK):**

```
[
  {
    "id": 1,
    "data_source_id": 1,
    "status": "success",
    "started_at": "2026-02-05T01:00:00Z",
    "rows_inserted": 487,
    "sha256_hash": "a3f2c4e5b6d7...",
    "duration_ms": 2134
  }
]
```

---

# Get Dataset Metadata

**Endpoint:** GET /admin/dataset/meta  
**Auth Required:** Yes

**Success Response (200 OK):**

```
{
  "source_name": "Leeds Temporary Events",
  "source_url":
    "https://opendata.leeds.gov.uk/downloads/Licences/temp-event-notice/temp-event-notice.xml",
  "last_import": "2026-02-05T01:00:00Z",
  "rows_inserted": 487,
  "sha256_hash": "a3f2c4e5b6d7..."
}
```

**If No Data Imported:**

```
{
  "message": "No dataset imported yet"
}
```

---

## Error Response Format

All error responses follow a consistent format:

```
{
  "detail": "Human-readable error message"
}
```

For validation errors (422):

```
{
  "detail": [
    {
      "loc": ["body", "capacity"],
      "msg": "Input should be greater than or equal to 1",
      "type": "greater_than_equal"
    }
  ]
}
```

---

## Common Failure Modes

Scenario	HTTP Status	Response Example	Reason
Missing Token	401 Unauthorized	{"detail": "Not authenticated"}	Endpoint requires login

Scenario	HTTP Status	Response Example	Reason
<b>Invalid Token</b>	401 Unauthorized	{"detail": "Could not validate credentials"}	but no header sent. Token expired, tampered, or wrong secret.
<b>Duplicate Email</b>	400 Bad Request	{"detail": "Email already registered"}	User/Attendee registration with existing email.
<b>RSVP Conflict</b>	409 Conflict	{"detail": "Attendee already has an RSVP for this event"}	User tries to RSVP twice to same event.
<b>Empty Dataset</b>	200 OK	{"message": "No dataset imported yet"}	Analytics/Admin endpoints called before import run.

## HTTP Status Codes Summary

Code	Meaning	When Used
200	OK	Successful GET, PATCH
201	Created	Successful POST
204	No Content	Successful DELETE
400	Bad Request	Invalid data (duplicate username)
401	Unauthorized	Missing/invalid JWT
404	Not Found	Resource doesn't exist
409	Conflict	Duplicate entry
422	Unprocessable Entity	Validation failed
500	Internal Server Error	Unexpected error

## Running Locally

### Prerequisites

- Python 3.11+
- pip

### Setup

```
# Clone repository
git clone https://github.com/NathS04/comp3011-cw1-api.git
cd comp3011-cw1-api
```

```
# Create virtual environment
python -m venv venv
source venv/bin/activate # Windows: venv\Scripts\activate

# Install dependencies
pip install -r requirements.txt

# Set environment variables
export DATABASE_URL="sqlite:///./app.db"
export SECRET_KEY="your-secret-key"

# Run migrations
alembic upgrade head

# Start server
uvicorn app.main:app --reload
```

## Running Tests

```
pytest -v
```

**Result:** 31 tests passing

---

## Demo Script (cURL)

```
# 1. Health check
curl http://127.0.0.1:8000/health

# 2. Register
curl -X POST http://127.0.0.1:8000/auth/register \
  -H "Content-Type: application/json" \
  -d '{"username":"demo","email":"demo@test.com","password":"password123"}'

# 3. Login
TOKEN=$(curl -s -X POST http://127.0.0.1:8000/auth/login \
  -d "username=demo&password=password123" | jq -r '.access_token')

# 4. Create event
curl -X POST http://127.0.0.1:8000/events \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"title":"Demo Event","location":"Leeds","start_time":"2026-03-01T10:00:00","end_time":"2026-03-01T12:00:00","capacity":50}'

# 5. Get trending
curl http://127.0.0.1:8000/analytics/events/trending
```

```
# 6. Get recommendations
curl -H "Authorization: Bearer $TOKEN"
      http://127.0.0.1:8000/events/recommendations
```

---

# Changelog

Version	Date	Changes
1.1.0	2026-02-04	Added analytics and recommendations endpoints
1.0.0	2026-02-01	Initial release

---

*Document generated for COMP3011 CW1 submission*