**API Structure Documentation**

**Base URL**

All endpoints will use the base URL:

https://api.example.com/v1

**Endpoints**

**1. User Management**

**Create User**

* **Endpoint**: POST /users
* **Description**: Allows admins to create a new user account.
* **Request Format**:

{

"Name": "string",

"password": "string",

"email": "string",

"Role": "string" // "student", "ssg\_officer", "event\_organizer", "admin"

}

* **Response Format**:

{

"UserID": "string",

"Name": "string",

"Role": "string",

"created\_at": "string"

}

**Edit User**

* **Endpoint**: PATCH /users/{userId}
* **Description**: Allows admins to edit user account details.
* **Request Format**:

{

"Name": "string",

"Email": "string",

"Role": "string"

}

* **Response Format**:

{

"UserID": "string",

"Name": "string",

"Role": "string",

"updated\_at": "string"

}

**Delete User**

* **Endpoint**: DELETE /users/{userId}
* **Description**: Allows admins to delete a user account.
* **Response Format**:

{

"message": "User deleted successfully."

}

**List Users**

* **Endpoint**: GET /users
* **Description**: Fetches a list of all users.
* **Response Format**:

[

{

"UserID": "string",

"Name": "string",

"Email": "string",

"Role": "string"

}

]

**2. Event Management**

**Create Event**

* **Endpoint**: POST /events
* **Description**: Allows event organizers to create a new event.
* **Request Format**:

{

"Event Name": "string",

"description": "string",

"Date and Time": "string",

"Location": "string"

}

* **Response Format**:

{

"EventID": "string",

"Event Name": "string",

"description": "string",

"Date and Time": "string",

"Location": "string",

"created\_at": "string"

}

**Edit Event**

* **Endpoint**: PATCH /events/{eventId}
* **Description**: Allows event organizers to edit an event.
* **Request Format**:

{

"Event Name": "string",

"description": "string",

"Date and Time": "string",

"Location": "string"

}

* **Response Format**:

{

"EventID": "string",

"Event Name": "string",

"description": "string",

"Date and Time": "string",

"Location": "string",

"updated\_at": "string"

}

**Delete Event**

* **Endpoint**: DELETE /events/{eventId}
* **Description**: Allows event organizers to delete an event.
* **Response Format**:

{

"message": "Event deleted successfully."

}

**List Events**

* **Endpoint**: GET /events
* **Description**: Fetches a list of all events.
* **Response Format**:

[

{

"EventID": "string",

"Event Name": "string",

"description": "string",

"Date and Time": "string",

"Location": "string"

}

]

**3. Attendance Tracking**

**Mark Attendance**

* **Endpoint**: POST /attendance
* **Description**: SSG officers verify and mark attendance via face recognition.
* **Request Format**:

{

"StudentID": "string",

"EventID": "string",

"Status": "string" // "present" or "absent"

}

**Response Format**:

{

"AttendanceID": "string",

"StudentID": "string",

"EventID": "string",

"Status": "string",

"timestamp": "string"

}

**4. Attendance Reports**

**Generate Attendance Report**

* **Endpoint**: GET /reports/attendance?eventId={eventId}
* **Description**: Admins can generate event attendance reports.
* **Response Format**:

{

"EventID": "string",

"Event Name": "string",

"Attendance": [

{

"StudentID": "string",

"Name": "string",

"status": "string" // "present" or "absent"

} ] }

**View Attendance History**

* **Endpoint**: GET /attendance/history?studentId={studentId}
* **Description**: Students can view their attendance history.
* **Response Format**:

[

{

"EventID": "string",

"Event Name": "string",

"Status": "string",

"timestamp": "string"

}

]

**5. Security & Authentication**

**User Login**

* **Endpoint**: POST /auth/login
* **Description**: Allows users to log in with credentials or third-party authentication.
* **Request Format**:

{

"email": "string",

"password": "string" // Optional if logging in via Google/Facebook

}

* **Response Format**:

{

"token": "string",

"user": {

"id": "string",

"username": "string",

"role": "string"

}

}

**Google/Facebook Login**

* **Endpoint**: POST /auth/social
* **Description**: Allows users to log in via Google or Facebook.
* **Request Format**:

{

"provider": "string", // "google" or "Facebook"

"token": "string" // Token from Google/Facebook authentication

}

* **Response Format**:

{

"token": "string",

"user": {

"id": "string",

"username": "string",

"role": "string"

}

}

**Secure Communication**

* **Headers**:
  + Authorization: Bearer <token> (Required for all secured endpoints)

**Error Response Format** For all endpoints, errors will follow this format:

{

"error": "string",

"message": "string"

}