ASMC API - API Reference

Complete reference for all ASMC API endpoints, request/response formats, and usage examples.

Table of Contents

Base Configuration

Base URLs

- **Development**: `http://localhost:7055`
- **Production**: `https://api.asmcdae.in`
- **API Documentation**: `{base_url}/api-docs`

Headers

```
Content-Type: application/json
Authorization: Bearer {jwt_token}
```

Authentication

Login Endpoints

Admin Login

```
POST /auth/admin-login
```

Request Body:

```
{
    "username": "admin@asmc.com",
    "password": "password123"
}
```

Response:

```
{
    "success": true,
    "message": "Login successful",
    "result": {
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
        "user": {
            "id": "user_id",
            "username": "admin@asmc.com",
            "role": "admin"
```

```
        }
    }
}
```

Member Login

```
POST /auth/member-login
```

Request Body:

```
{
    "memberId": "00001",
    "password": "password123"
}
```

User Management

```
GET /auth/me                    # Get current user
PUT /auth/change-password       # Change password
POST /auth/send-reset-password-otp  # Send reset OTP
PUT /auth/reset-password        # Reset password
```

Members API

Member Management

```
GET /members                    # List all members (paginated)
POST /members                   # Create new member
GET /members/:id               # Get member details
PUT /members/:id               # Update member
DELETE /members/:id            # Delete member
POST /members/multiple         # Bulk member creation
GET /members/export            # Export members data
```

Create Member

```
POST /members
```

Request Body:

```
{
    "firstName": "John",
    "lastName": "Doe",
    "email": "john.doe@example.com",
    "phone": "9876543210",
    "dateOfBirth": "1990-01-01",
    "address": {
        "street": "123 Main St",
        "city": "Mumbai",
        "state": "Maharashtra",
        "pincode": "400001"
```

```
    },
    "familyMembers": [
        {
            "name": "Jane Doe",
            "relation": "spouse",
            "age": 28
        }
    ]
}
```

Response:

```
{
    "success": true,
    "message": "Member created successfully",
    "result": {
        "id": "member_id",
        "memberId": "00001",
        "firstName": "John",
        "lastName": "Doe",
        "email": "john.doe@example.com",
        "phone": "9876543210",
        "createdAt": "2024-01-01T00:00:00.000Z"
    }
}
```

Staff API

```
GET /staff                      # List all staff
POST /staff                     # Create new staff
GET /staff/:id                 # Get staff details
PUT /staff/:id                 # Update staff
DELETE /staff/:id              # Delete staff
```

Masters API

Facility Management

```
GET /masters/facilities         # List facilities
POST /masters/facilities        # Create facility
GET /masters/facilities/:id     # Get facility details
PUT /masters/facilities/:id     # Update facility
DELETE /masters/facilities/:id  # Delete facility
```

Location Management

```
GET /masters/locations          # List locations
POST /masters/locations         # Create location
GET /masters/locations/:id      # Get location details
```

```
PUT /masters/locations/:id      # Update location
DELETE /masters/locations/:id   # Delete location
```

Category Management

```
GET /masters/categories         # List categories
POST /masters/categories        # Create category
GET /masters/categories/:id     # Get category details
PUT /masters/categories/:id     # Update category
DELETE /masters/categories/:id  # Delete category
```

🏃‍♂️ Activity API

```
GET /activity                   # List activities
POST /activity                  # Create activity
GET /activity/:id               # Get activity details
PUT /activity/:id               # Update activity
DELETE /activity/:id            # Delete activity
POST /activity/enroll           # Enroll in activity
GET /activity/enrolled          # Get enrolled activities
```

💳 Payment API

Payment Processing

```
GET /payment                    # List payments
POST /payment                   # Create payment
GET /payment/:id                # Get payment details
POST /payment/verify            # Verify payment
GET /payment/history            # Payment history
POST /payment/ccavenue-response # CCAvenue callback
```

Create Payment

```
POST /payment
```

Request Body:

```
{
    "memberId": "00001",
    "paymentType": "membership",
    "amount": 5000,
    "currency": "INR",
    "paymentMethod": "online",
    "description": "Annual membership fee"
}
```

🔐 Biometric API

Machine Management

```
GET /biometric/machines         # List machines
POST /biometric/machines        # Add machine
GET /biometric/machines/:id     # Get machine details
PUT /biometric/machines/:id     # Update machine
DELETE /biometric/machines/:id  # Delete machine
```

Attendance Management

```
GET /biometric/attendance       # Get attendance
POST /biometric/attendance      # Mark attendance
GET /biometric/attendance/:id   # Get member attendance
PUT /biometric/attendance/:id   # Update attendance
```

Notifications

```
GET /biometric/notifications    # Get notifications
POST /biometric/notifications   # Create notification
PUT /biometric/notifications/:id # Update notification
DELETE /biometric/notifications/:id # Delete notification
```

Regularization

```
GET /biometric/regularization   # Get regularization requests
POST /biometric/regularization  # Submit regularization request
PUT /biometric/regularization/:id # Update regularization
```

Halls API

Hall Management

```
GET /halls                      # List halls
POST /halls                     # Create hall
GET /halls/:id                 # Get hall details
PUT /halls/:id                 # Update hall
DELETE /halls/:id              # Delete hall
```

Hall Booking

```
POST /halls/book               # Book hall
GET /halls/availability        # Check availability
GET /halls/bookings            # Get bookings
PUT /halls/bookings/:id        # Update booking
DELETE /halls/bookings/:id     # Cancel booking
```

Book Hall

```
POST /halls/book
```

Request Body:

```
{
    "hallId": "hall_id",
    "memberId": "00001",
    "bookingDate": "2024-01-15",
    "timeSlots": [
        {
            "startTime": "09:00",
            "endTime": "11:00"
        }
    ],
    "guestDetails": [
        {
            "name": "Guest Name",
            "phone": "9876543210",
            "relation": "friend"
        }
    ]
}
```

Events API

Event Management

```
GET /events                     # List events
POST /events                    # Create event
GET /events/:id                 # Get event details
PUT /events/:id                 # Update event
DELETE /events/:id              # Delete event
```

Event Booking

```
POST /events/book               # Book event
GET /events/upcoming            # Upcoming events
GET /events/bookings            # Get bookings
PUT /events/bookings/:id        # Update booking
DELETE /events/bookings/:id     # Cancel booking
```

Plans API

```
GET /plans                      # List plans
POST /plans                     # Create plan
GET /plans/:id                  # Get plan details
PUT /plans/:id                  # Update plan
DELETE /plans/:id               # Delete plan
POST /plans/recommend           # Recommend plan
GET /plans/recommended          # Get recommended plans
```

Reports API

```
GET /reports/members           # Members report
GET /reports/payments          # Payments report
GET /reports/enrollment        # Enrollment report
GET /reports/batch-wise        # Batch-wise report
GET /reports/renewal           # Renewal report
GET /reports/analytics         # Analytics report
```

Response Formats

Success Response

```
{
    "success": true,
    "message": "Operation successful",
    "result": {
        // Response data
    }
}
```

Error Response

```
{
    "success": false,
    "message": "Error description",
    "result": {
        "error": "Detailed error information",
        "code": "ERROR_CODE"
    }
}
```

Paginated Response

```
{
  "success": true,
  "message": "Data retrieved successfully",
  "result": {
    "data": [...],
    "pagination": {
      "page": 1,
      "limit": 10,
      "total": 100,
      "pages": 10
    }
  }
}
```

Error Handling

HTTP Status Codes

- `200` - Success
- `201` - Created
- `400` - Bad Request
- `401` - Unauthorized
- `403` - Forbidden
- `404` - Not Found
- `422` - Validation Error
- `500` - Internal Server Error

Common Error Codes

- `INVALID_CREDENTIALS` - Login failed
- `TOKEN_EXPIRED` - JWT token expired
- `INSUFFICIENT_PERMISSIONS` - Access denied
- `VALIDATION_ERROR` - Input validation failed
- `DUPLICATE_ENTRY` - Resource already exists
- `RESOURCE_NOT_FOUND` - Requested resource not found

Rate Limiting

Limits

- **Default**: 100 requests per 15 minutes per IP
- **Authentication**: 5 login attempts per 15 minutes per IP
- **File Upload**: 10 uploads per hour per user

Rate Limit Headers

```
X-RateLimit-Limit: 100
X-RateLimit-Remaining: 95
X-RateLimit-Reset: 1640995200
```

Rate Limit Exceeded Response

```
{
    "success": false,
    "message": "Too many requests",
    "result": {
        "error": "Rate limit exceeded",
        "retryAfter": 900
    }
}
```

Usage Examples

Complete Authentication Flow

```
// 1. Login
const loginResponse = await fetch('/auth/admin-login', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
```

```
    body: JSON.stringify({
        username: 'admin@asmc.com',
        password: 'password123',
    }),
});

const { result } = await loginResponse.json();
const token = result.token;

// 2. Use token in subsequent requests
const membersResponse = await fetch('/members', {
    headers: { Authorization: `Bearer ${token}` },
});

const members = await membersResponse.json();
```

File Upload Example

```
const formData = new FormData();
formData.append('file', fileInput.files[0]);
formData.append('memberId', '00001');

const response = await fetch('/members/upload-image', {
    method: 'POST',
    headers: { Authorization: `Bearer ${token}` },
    body: formData,
});
```

Pagination Example

```
const response = await fetch('/members?page=2&limit=20&sort=createdAt&order=desc', {
    headers: { Authorization: `Bearer ${token}` },
});

const { result } = await response.json();
const { data, pagination } = result;
```

Additional Resources

- **Interactive API Docs**: http://localhost:7055/api-docs
- **Quick Start Guide**: QUICK_START_GUIDE.md
- **Architecture Overview**: ARCHITECTURE_OVERVIEW.md
- **Installation Guide**: INSTALLATION_SETUP.md

This API reference provides comprehensive documentation for all endpoints. For interactive testing, use the Swagger UI at `/api-docs`.