MyOral.AI Public API (v1.1)

[Overview 1](#_Toc788832697)

[API Endpoints 1](#_Toc1121866494)

[User Registration 2](#_Toc595865630)

[Login 2](#_Toc1167110345)

[Update User Information 2](#_Toc658525964)

[Retrieve User Information 2](#_Toc1581366573)

[Purpose and Notes 2](#_Toc1619144617)

[Upload Teeth Photos 2](#_Toc2084264949)

[Analyze Photos 2](#_Toc1102667898)

[List Mask Image Channel Values 2](#_Toc1522172842)

[Get Analysis History 2](#_Toc907268405)

[Client Specification 2](#_Toc747434506)

[Enforcement of HTTPS on the Client Side 2](#_Toc1419549524)

[Secure Data in Transit 2](#_Toc1311303050)

[Server Specification 2](#_Toc2003637328)

[Authentication 2](#_Toc859713749)

[Access Control 2](#_Toc15843475)

[Configure the Server 2](#_Toc1819821753)

[Secure Data in Transit 2](#_Toc763612651)

[Authentication and Authorization 2](#_Toc704857939)

[Additional Best Practices 2](#_Toc654194204)

[Secure URLs 2](#_Toc2098669942)

[Appendix A 2](#_Toc341480202)

[Appendix B 2](#_Toc532133438)

[Mask and its Offset 2](#_Toc1821754951)

[List of mask values 2](#_Toc502074283)

# Overview

The API endpoints specified in this document are for the **end-user client app** and the **reviewer client app** to exchange information with our backend server. All API endpoints should use **HTTPS** instead of **HTTP** to ensure secure communication between the client and server. HTTPS encrypts the data being transmitted, protecting sensitive user information such as credentials, diagnosis details, and personal data from being intercepted by malicious actors.

# API Endpoints

|  |  |
| --- | --- |
| Base URL | https://api.myoral.ai |

## User Registration

* **Endpoint**: POST /v1/users/register
* **Description**: Creates an account for the user.
* **Headers**:
  + Content-Type: application/json
* **Request Body**:

json

{  
 "email": "string",  
 "phoneNumber": "string",  
 "password": "string",

"usertype": "enduser | reviewer"

}

* **Response**:
  + **Success**: 200 Created successfully
  + **Response Body**:

json

{  
 "userId": "string",  
 "message": "Account created successfully"  
}

* **Error**: 400 Bad Request, 409 Conflict (already registered), 500 Internal server error

## Login

* **Endpoint**: POST /v1/users/login
* **Description**: Logs in a registered user.
* **Headers**:
  + Content-Type: application/json
* **Request Body**:

json

{

"userId": "string",  
 "email": "string",  
 "phoneNumber": "string",  
 "password": "string"  
}

* **Response**:
  + **Success**: 200 OK
  + **Response Body**:

json

{  
 "userId": "string",

"usertype": "enduser | reviewer",  
 "token": "string",  
 "message": "Login successful"  
}

* **Error**: 400 Bad Request, 401 Unauthorized, 500 Internal server error

Users can use userId, email, or phoneNumber to login. Phone numbers should be in the format of +<country code>-<phone number>, without space within the string.

## Update User Information

* **Endpoint**: PUT /v1/users/{userId}
* **Description**: Updates user information such as gender and age group.
* **Headers**:
  + Authorization: Bearer <token>
  + Content-Type: application/json
* **Request Body**:

json

{  
 "surname": "string",  
 "givename": "string",  
 "sex": "string",  
 "birthYear": "number"  
}

* **Accepted Values for** sex:
  + M
  + F
* **Response**:
  + **Success**: 200 OK
  + **Response Body with sample value**:

json

{

"surname": "Chan",  
 "givename": "Tai Man",  
 "sex": "M",  
 "birthYear": "1980",  
 "message": "User information updated successfully"  
}

* **Error**: 400 Bad Request, 401 Unauthorized, 404 Not Found, 500 Internal server error

## Retrieve User Information

* **Endpoint**: GET /api/users/{userId}
* **Description**: Retrieves detailed information about a specific user.
* **Headers**:
  + Authorization: Bearer <token> (to ensure only authorized access).
* **Path Parameters**:
  + userId: The unique identifier of the user whose information is being retrieved.
* **Response**:
  + **Success**: 200 OK
  + **Response Body**:

json

{  
 "userId": "string",  
 "fullName": "string",  
 "email": "string",  
 "phoneNumber": "string",  
 "usertype": "enduser | reviewer",  
 "createdAt": "YYYY-MM-DDTHH:mm:ssZ",  
 "lastUpdatedAt": "YYYY-MM-DDTHH:mm:ssZ"  
}

* **Error**:
  + 400 Bad Request: Invalid userId format or missing required parameters.
  + 404 Not Found: User with the specified userId does not exist.

### **Purpose and Notes**

1. **Access Control**:
   1. Only authorized users (e.g., admins, reviewers, or the user themselves) should be allowed to retrieve user info.
2. **Use Case**:
   1. Allows retrieving basic user profile details like name, email, phone number, and user type.
   2. Useful for displaying user information in apps or dashboards.

## Upload Teeth Photos

* **Endpoint**: POST /v1/upload/{userId}
* **Description**: Uploads front, left, and right teeth photos.
* **Headers**:
  + Authorization: Bearer <token>
  + Content-Type: multipart/form-data
* **Form Data**:
  + front: File (image)
  + left: File (image)
  + right: File (image)
  + captureTimestamp: String (ISO 8601 format, e.g., 2025-04-16T12:30:00Z)
  + metadata: (Optional) JSON string containing additional metadata about the photos.

json

{  
 "cameraModel": "string",  
 "deviceModel": "string",  
 "appVersion": "string"  
}

**Metadata description**

* cameraModel: Describes the camera used (e.g., "Sony Alpha 7").
* deviceModel: Specifies the device model used (e.g., "iPhone 13 Pro").
* appVersion: Indicates the version of the app used to capture and upload the photos.
* **Response**:
  + **Success**: 200 OK
  + **Response Body with sample values**:

json

{  
 "photoUrls": {  
 "front": "string",  
 "left": " string",  
 "right": "string"  
 },  
 "captureTimestamp": "2025-04-16T12:30:00Z",  
 "message": "Photos uploaded successfully"  
}

photoUrls format: https://api.myoral.ai/uploads/{userId}/{captureTimestamp}/{photoType}.{jpg/png}

**Example Generated URLs**

For a user with ID user123, who took photos on 2025-04-16T12:30:00Z:

* Front photo: [https://api.myoral.ai/uploads/user123/2025-04-16T12:30:00Z/9b8a7c6d-front.png?expiry=2025-04-17T12:00:00Z&signature=abcd1234](https://api.myoral.ai/uploads/user123/2025-04-16T12:30:00Z/front.png)
* Left photo: [https://api.myoral.ai/uploads/user123/2025-04-16T12:30:00Z/s6d6f5c7-left.png?expiry=2025-04-17T12:00:00Z&signature=abcd1234](https://api.myoral.ai/uploads/user123/2025-04-16T12:30:00Z/s6d6f5c7-left.png)
* Right photo: <https://api.myoral.ai/uploads/user123/2025-04-16T12:30:00Z/a8d7f6b5-right.png?expiry=2025-04-17T12:00:00Z&signature=abcd1234>
* **Error**: 400 Bad Request, 404 Not Found, 500 Internal server error

Please refer to Appendix A for technical requirements for uploaded image files.

Discussion: do we want the image URL returned? If such is returned, the client apps can verify the image is correctly uploaded.

## Analyze Photos

* **Endpoint**: POST /v1/analysis/{userId}
* **Description**: Processes uploaded photos to analyze oral health.
* **Headers**:
  + Authorization: Bearer <token>
* **Response**:
  + **Success**: 200 OK
  + **Response Body with sample values**:
    - json
    - {  
       "results": {  
       "diagnosis: Your teeth are healthy | Early symptoms of dental issues detected. Please improve your oral hygiene | Dental illnesses detected. Please consult a dentist.",  
       "maskImageUrl": “<https://api.myoral.ai/analysis/user123/mask/9b8a7c6d-front-mask.png?expiry=2025-04-23T12:00:00Z&signature=abcd1234”>,  
       "maskOffset": {  
       "x": 50, // integer  
       "y": 30 // integer  
       }  
       },  
       "message": "Analysis completed successfully"  
      }
* **Error**: 400 Bad Request, 404 Not Found, 500 Internal server error

Please refer to Appendix B for specification of the mask image and its offset. Client app should display the annotations embedded in the mask image to the user.

## List Mask Image Channel Values

* **Endpoint**: GET /api/list-mask-values
* **Description**: Retrieves the list of values and their meanings for each channel (Red, Green, Blue) in the mask image.
* **Headers**:
  + Authorization: Bearer <token>
* **Response**:
  + **Success**: 200 OK
  + **Response Body**:

json

{  
 "channels": {  
 "Red": {  
 "description": "Encodes teeth identifiers",  
 "values": {

"0": "No teeth at this pixel",  
 "1": "Upper left central incisor (perm)",

...

}  
 },  
 "Green": {  
 "description": "Encodes teeth-related issues",  
 "values": {

"0": "No issue at this pixel",  
 "1": "Moderate caries",  
 "2": "Severe caries",

...

}  
 },  
 "Blue": {  
 "description": "Encodes gum-related issues",  
 "values": {

"0": "No issue at this pixel",  
 "1": "Moderate gum inflammation ",  
 "2": "Severe gum inflammation ",

...

}  
 }  
 },  
 "message": "Mask channel values retrieved successfully"  
}

* **Error**: 401 Unauthorized, 500 Internal server error

Description

1. **Red (R) Channel**:
   1. Represents teeth, both permanent and additional ones (like dentures or extra teeth).
   2. Supports up to 255 values to accommodate typical adult teeth and additional cases.
2. **Green (G) Channel**:
   1. Encodes teeth-specific issues, such as caries or fractures.
3. **Blue (B) Channel**:
   1. Encodes gum-specific issues, including inflammation, gingivitis, and bleeding.

This API provides a comprehensive reference for interpreting mask images for diagnosis results. The client app should render the annotation using this information to decode the mask and let users understand the analysis visually. Appendix B lists the mask values and their interpretations.

## Get Analysis History

* **Endpoint**: GET /v1/analysis/{userId}/history
* **Description**: Retrieves a user's previous analysis records.
* **Headers**:
  + Authorization: Bearer <token>
* **Response**:
  + **Success**: 200 OK
  + **Response Body**:

json

{  
 "records": [  
 {  
 "analysisId": "string",  
 "dateTime": "YYYY-MM-DDTHH:mm:ssZ",  
 "photos": {  
 "front": "string",  
 "left": "string",  
 "right": "string"  
 },  
 "results": {  
 "status": "Healthy | Early Symptoms | Dental Illness",  
 "diagnosis": "string",  
 "annotationMaskUrl": "string",

"annotationMaskOffset": "string"  
 }  
 }  
 ]  
}

* **Error**: 401 Unauthorized, 404 Not Found, 500 Internal server error

# Client Specification

We shall have two client apps: 1. end-user client app. 2. reviewer client. The end-user app is a mobile app that runs on either mobile phone or tablet. The reviewer client is used by dentists and can be a web site instead of a mobile app. The end-user app can only access the user’s own record, while the reviewer client can access all user records.

## Enforcement of HTTPS on the Client Side

* Make sure the client app only communicates with the server using HTTPS URLs.
* Block any attempts to access resources over HTTP.

## Secure Data in Transit

* Use strong TLS protocols, such as **TLS 1.2** or **TLS 1.3**, to encrypt communications.
* Disable weak ciphers to ensure the highest level of security.
* Use HSTS (HTTP Strict Transport Security) headers to enforce HTTPS on browsers or clients.

# Server Specification

## Authentication

Bearer tokens are generated by the server during the authentication process:

1. **User Login**: When a user logs in using their credentials (e.g., email and password), the server verifies that the credentials are correct.
2. **Token Generation**: If the credentials are valid, the server creates a bearer token. This token is usually a cryptographically signed string that contains information like:
   1. User identification (e.g., userId).
   2. Expiration time to limit how long the token is valid.
   3. Other metadata required for authentication.
3. **Token Delivery**: The server sends the token to the client (the app) as part of the login response.

Bearer tokens can be created using protocols like OAuth or JSON Web Tokens (JWT). These methods ensure the token is secure, tamper-proof, and usable only by the intended user.

## Access Control

The usertype field is used for access control in various endpoints:

* **End Users**: Can access APIs related to photo uploads, user’s own health analysis, and history retrieval.
* **Reviewers**: Can access APIs related to diagnosis reviews and scoring on all records.

Secure Communication

### **Configure the Server**

* Obtain an **SSL/TLS Certificate** from a trusted Certificate Authority (CA), such as Let's Encrypt, DigiCert, or GoDaddy.
* Install the certificate on your server to enable HTTPS.
  + For web servers like Apache or Nginx, update their configurations to include the SSL certificate and private key.
* Redirect all HTTP traffic to HTTPS to ensure all communications are encrypted.

### **Secure Data in Transit**

* Use strong TLS protocols, such as **TLS 1.2** or **TLS 1.3**, to encrypt communications.
* Disable weak ciphers to ensure the highest level of security.
* Use HSTS (HTTP Strict Transport Security) headers to enforce HTTPS on browsers or clients.

### **Authentication and Authorization**

* Ensure secure handling of tokens (e.g., JWT bearer tokens) over HTTPS.
* Include the Authorization: Bearer <token> header for API endpoints requiring authentication.

### **Additional Best Practices**

* Implement **CORS (Cross-Origin Resource Sharing)** policies for secure API access.
* Use rate limiting and other security measures to prevent abuse or malicious attacks.
* Regularly update and patch the server and SSL/TLS libraries to defend against vulnerabilities like Heartbleed or POODLE.

### **Secure URLs**

* **Use Expiring Signed URLs**: Generate time-limited URLs that require a secure signature to access.
  + Example: <https://api.myoral.ai/uploads/user123/9b8a7c6d-front.png?expiry=2025-04-17T12:00:00Z&signature=abcd1234>
  + The signature ensures only authorized requests can access the file, and the expiry time limits its validity.
* Use randomly generated unique identifiers for file paths (e.g., https://api.myoral.ai/uploads/{userid}/9b8a7c6d-front.png) instead of predictable names.

# Appendix A

Technical requirements for uploaded image file:

1. Format: JPG or PNG only
2. File Size: Up to 10 MB
3. Dimensions: Minimum 1000 x 1000 pixels

# Appendix B

The annotation mask is a PNG image with annotations encoded in the red, green and blue channels of the image.

## Mask and its Offset

This mask only covers the rectangular area where there is annotation. The following illustrations the mask in its offset inside the input teeth photo:

## List of mask values

Red channel (Teeth identifiers):

|  |  |
| --- | --- |
| Value | Description |
| 0 | No teeth at this pixel |
| 1 | Upper left central incisor (perm) |
| 2 | Upper left lateral incisor (perm) |
| 3 | Upper left canine (perm) |
| 4 | Upper left first premolar (perm) |
| 5 | Upper left second premolar (perm) |
| 6 | Upper left first molar (perm) |
| 7 | Upper left second molar (perm) |
| 8 | Upper left third molar (perm) |
| 9 | Upper right central incisor (perm) |
| 10 | Upper right lateral incisor (perm) |
| 11 | Upper right canine (perm) |
| 12 | Upper right first premolar (perm) |
| 13 | Upper right second premolar (perm) |
| 14 | Upper right first molar (perm) |
| 15 | Upper right second molar (perm) |
| 16 | Upper right third molar (perm) |
| 17 | Lower left central incisor (perm) |
| 18 | Lower left lateral incisor (perm) |
| 19 | Lower left canine (perm) |
| 20 | Lower left first premolar (perm) |
| 21 | Lower left second premolar (perm) |
| 22 | Lower left first molar (perm) |
| 23 | Lower left second molar (perm) |
| 24 | Lower left third molar (perm) |
| 25 | Lower right central incisor (perm) |
| 26 | Lower right lateral incisor (perm) |
| 27 | Lower right canine (perm) |
| 28 | Lower right first premolar (perm) |
| 29 | Lower right second premolar (perm) |
| 30 | Lower right first molar (perm) |
| 31 | Lower right second molar (perm) |
| 32 | Lower right third molar (perm) |
| 33 | Upper left central incisor (primary) |
| 34 | Upper left lateral incisor (primary) |
| 35 | Upper left canine (primary) |
| 36 | Upper left first molar (primary) |
| 37 | Upper left second molar (primary) |
| 38 | Lower right central incisor (primary) |
| 39 | Lower right lateral incisor (primary) |
| 40 | Lower right canine (primary) |
| 41 | Lower right first molar (primary) |
| 42 | Lower right second molar (primary) |
| 43 | Lower left central incisor (primary) |
| 44 | Lower left lateral incisor (primary) |
| 45 | Lower left canine (primary) |
| 46 | Lower left first molar (primary) |
| 47 | Lower left second molar (primary) |
| 48 | Lower left central incisor (primary) |
| 49 | Lower left lateral incisor (primary) |
| 50 | Lower left canine (primary) |
| 51 | Lower left first molar (primary) |
| 52 | Lower left second molar (primary) |
| 53 | Upper left central incisor (denture) |
| 54 | Upper left lateral incisor (denture) |
| 55 | Upper left canine (denture) |
| 56 | Upper left first premolar (denture) |
| 57 | Upper left second premolar (denture) |
| 58 | Upper left first molar (denture) |
| 59 | Upper left second molar (denture) |
| 60 | Upper left third molar (denture) |
| 61 | Upper right central incisor (denture) |
| 62 | Upper right lateral incisor (denture) |
| 63 | Upper right canine (denture) |
| 64 | Upper right first premolar (denture) |
| 65 | Upper right second premolar (denture) |
| 66 | Upper right first molar (denture) |
| 67 | Upper right second molar (denture) |
| 68 | Upper right third molar (denture) |
| 69 | Lower left central incisor (denture) |
| 70 | Lower left lateral incisor (denture) |
| 71 | Lower left canine (denture) |
| 72 | Lower left first premolar (denture) |
| 73 | Lower left second premolar (denture) |
| 74 | Lower left first molar (denture) |
| 75 | Lower left second molar (denture) |
| 76 | Lower left third molar (denture) |
| 77 | Lower right central incisor (denture) |
| 78 | Lower right lateral incisor (denture) |
| 79 | Lower right canine (denture) |
| 80 | Lower right first premolar (denture) |
| 81 | Lower right second premolar (denture) |
| 82 | Lower right first molar (denture) |
| 83 | Lower right second molar (denture) |
| 84 | Lower right third molar (denture) |

86-255: Reserved for other cases e.g. Extra tooth between xxx and yyy  
  
Green channel (Teeth-related issues):  
0: No issue at this pixel  
1: Moderate caries  
2: Severe caries

3: Fracture  
4-255: Reserved for other teeth-related issues  
  
Blue channel (Gum issues):

0: No issue at this pixel  
1: Moderate gum inflammation  
2: Severe gum inflammation  
3: Gingivitis  
4: Gum swelling  
5: Redness  
6: Receding gum  
7: Gum bleeding  
8: Gum boil  
9-255: Reserved for other gum-related issue