IDx Technologies Inc.
2300 Oakdale Blvd Coralville, IA 52241

IDx-DR API User Manual

Software Version 1
2018-08-31

ENG-DRAPIV2EU-S00140, Revision A
en

Note: The information in this manual only applies to the IDx-DR API, the marked version. It does not apply to earlier versions. Due to continuing product innovation, specifications in this manual are subject to change without notice.

Contents

1 Manual Information

1.1 Revision History

Rev.	Date	Revision Description
A	2018-08-31	Manual Creation

1.2 Purpose

This manual contains the instructions necessary to write software that interacts with the IDx-DR API.

1.3 Intended Audience

This manual is intended for a programmer who can understand English.

1.4 How to obtain a paper copy of this manual

Contact IDx at support@eyediagnosis.net to request a paper copy of the IDx-DR API manual. Please provide your organization name, full address, the product for which you are requesting a manual, and the version of the software. IDx will mail you a printed version of the manual via the postal service.

1.5 Definitions

API	Application Programming Interface

1.6 Glossary of Symbols

⚠ **Warning** - Indicates a potentially hazardous situation associated with the use or misues of the device which, if not avoided, could result in serious injury, death, or serious other adverse reaction.

ℹ **Consult Manual Before Use**

2 Warnings and Precautions

2.1 Prerequisites

In order to write software to interact with IDx-DR API safely and according to the intended purpose, the following prerequisites must be met:

- You should be a competent programmer.
- You should read and understand the IDx-DR Warnings and Indications Manual in addition to this manual.
- You must read and understand the Intended Use, Warnings, and Operating Instructions.

2.2 Warnings

The following warnings apply to all aspects of software interacting with the IDx-DR API:

- IDx-DR does not require any patient identifying information to process an exam. To help safeguard information, do not send patient information through the IDx-DR API.
- Users should regularly ensure that software interacting with IDx-DR API is operating on a computer that is free of viruses or malware, and is updated with the latest security patches.
- Programmers should take care to limit the interactions with the API to a reasonable amount. This includes properly implementing sleep and wait conditions to limit traffic to IDx-DR API to a reasonable level.

3 API Description

3.1 General Information

IDx-DR API is a REST API.

Base URL: https://idxdrapi.azurewebsites.net/api/v1

A description of the API may be found at: https://idxdrapi.azurewebsites.net/help

All errors are communicated through the API using an HTTPResponseMessage with 1) an HTTP status code in the StatusCode field, and 2) an error message, in English, in the ReasonPhrase field.

The exam result is translated to the locale of the end-user that is registered with IDx.

⚠ **The translated exam result should be displayed to the end-user as-is. This includes the result string and the result binary (PDF/JPEG). This ensures that meaning is not lost in a subsequent translation.**

There are 5 controllers that make up the IDx-DR API:

- Certificate
- Label
- Submit
- Status
- Result

The rest of this document describes how to write software to effectively use the API. Parameters are passed to the API using URL parameters and JSON in the body. Reponses from the API use JSON.

3.2 CertificateController

3.2.1 Description

HTTP GET
URL: “{BASE_URL}/certificate/{registrationCode}”

{registrationCode}: An alpha-numeric code that must be obtained from IDx as part of a registration process.

⚠ **Registration codes are single use. The returned certificate should be saved as-is for future use with API controllers.**

3.2.2 Response JSON

```
{
  "data": "string",
  "md5": "string"
}
```

“data”: base64 string containing the certificate information. **Store this as-is (as a base64 string) for use in future requests.**
“md5”: MD5 hash of the underlying byte array represented by “data”.

3.3 LabelController

3.3.1 Description

HTTP GET
URL: “{BASE_URL}/label”

The HTTP header must contain the key “Token” mapped to the base64 string obtained from the CertificateController.

3.3.2 Response JSON

```
{
  "product": "string",
  "label": {
    "data": "string",
    "md5": "string"
  },
  "manual": {
    "link": "string"
  }
}
```

“product”: A string representing the version of IDx-DR supported by the IDx-DR API.

“label.data”: The contents of the .svg label file. To view the label in a major internet browser, write the contents to a file with the .svg extension.

“label.md5”: An md5 hash of “label/data”

“manual.link”: A link to the IDx-DR manual (not this manual which is for the IDx-DR API).

3.4 SubmitController

3.4.1 Description

HTTP POST
URL: “{BASE_URL}/submit”

The SubmitController requires a JSON body containing the data for the submission.

The HTTP header must contain the key “Token” mapped to the base64 string obtained from the CertificateController.

Slow processing times may be caused by large image sizes transmitted over a slow internet connection. Decreasing the size of image files may improve processing times.

3.4.2 Submission JSON

```
{
  "submissionData": {
    "patientID": "string",
    "resubmitID": "string",
    "clientID": "string"
  },
  "image": [
    {
      "imageID": "string",
      "data": "string",
      "md5": "string"
    },
    {
      "imageID": "string",
      "data": "string",
      "md5": "string"
    },
    {
      "imageID": "string",
      "data": "string",
      "md5": "string"
    },
    {
      "imageID": "string",
      "data": "string",
      "md5": "string"
    }
  ]
}
```

“submissionData.patientID”: A custom exam identifier. This information is stored with the exam request to aid in troubleshooting a particular exam. Note: when contacting IDx Customer Service it is prefereable to provide the submissionID rather than the patientID. For more information on the submissionID see Section 3.4.3.

“submissionData.resubmitID”: When resubmitting a patient exam, this value must be the submissionID for the prior exam. If the exam indicated by the resubmitID is not found within IDx, then the exam will not process. If this is a new submission, then this value is an empty string.

“submissionData.clientID”: The name and version of the software interacting with IDx-DR API. This information is stored with the exam request to aid in troubleshooting a particular exam.

“image.imageID”: A unique identifier for the image within the exam.

“image.data”: A base64-encoded byte array for the image. The image byte array must correspond to one of the supported filetypes (JPEG, PNG, TIF, BMP).

“image.md5”: An MD5 hash of the byte array in the corresponding “image/data” field. The MD5 hash is calculated on the byte array, not on the base64 string. If IDx-DR API cannot validate the “image/data” contents using the MD5 hash, then the submission will be assumed corrupt and not processed.

3.4.3 Response JSON

```
{
  "submissionID": "string",
  "status": 0
}
```

“submissionID”: **A unique submission identifier for the exam, store this value.** This value is required to identify exams within IDx-DR API and must be submitted to request the processing status or to get the result of the exam.

“status”: An integer representing the processing status of the exam. Interpretation of the values is as follows:

Code	Description
4	Processing complete
3	Processing
2	Submitted for processing

1	Submission received
0	Internal error
-1	Error during processing
-2	License exceeds allowed submissions
-3	License not active
-4	License not active

3.5 StatusController

3.5.1 Description

HTTP GET
URL: “{BASE_URL}/status/{submissionID}”

{submissionID}: The submissionID that was returned from the SubmitController.

The HTTP header must contain the key “Token” mapped to the base64 string obtained from the CertificateController.

3.5.2 Response JSON

```
{
  "submissionID": "string",
  "status": 0
}
```

3.6 ResultController

3.6.1 Description

HTTP GET
URL: “{BASE_URL}/result/{submissionID}/{binaryFileType}”

{submissionID}: The submissionID that was returned from the SubmitController.
{binaryFileType}: Optional. One of “NONE”, “PDF”, or “JPEG” will populate the resultBinary field in the Response JSON. If “NONE” or omitted, then the resultBinary field will not be populated in the Response JSON.

The HTTP header must contain the key “Token” mapped to the base64 string obtained from the CertificateController.

3.6.2 Response JSON

```
{
  "submissionID": "string",
  "patientID": "string",
  "analysisVersion": "string",
  "analysisDateTime": "string",
  "examResult": {
    "resultBinaryFileType": "string",
    "resultString": "string",
    "resultCode": 0,
    "resultBinary": "string",
    "resultBinaryMD5": "string",
    "examQuality": {
      "examQualityCode": 0,
      "imageQuality": [
        {
          "imageID": "string",
          "imageQualityCode": 0
        },
        {
          "imageID": "string",
          "imageQualityCode": 0
        },
        {
          "imageID": "string",
          "imageQualityCode": 0
        },
        {
          "imageID": "string",
          "imageQualityCode": 0
        }
      ]
    }
  }
}
```

“submissionID”: The submissionID that was returned from the SubmitController.

“patientID”: The patientID for the exam submitted to the SubmitController.

“analysisVersion”: The IDx-DR product and version that processed the exam.

“analysisDateTime”: A string representing *analysis server date and time*.

“examResult.resultBinaryFileType”: Either “NONE”, “PDF”, or “JPEG” depending on the request submitted to the ResultController.

“examResult.resultString”: A translated string explaining the result in the language of the end-user as registered with IDx. **It is important to display this result string to the user as-is for consistent result interpretation.**

“examResult.resultCode”: An integer code representing the exam result. Possible values are:

Code	Interpretation

1	Poor Exam Quality. See individual image quality codes for low-quality images.
2	Invalid Number of Left and Right Eye Images. Require 2 left eye images and 2 right eye images.
3	Invalid Optic Disc Location. Require 1 optic disc-centered and 1 fovea-centered image per eye.
4	Duplicate Images Detected.
5	General Image Analysis Failure. Image processing failed for one of the images.
6	General Exam Analysis Failure. Processing of exam failed not specific to a given image.
7	None-or-mild DR Detected. Use resultString to present to user.
8	More-than-mild DR Detected. Use resultString to present to user.
9	Vision-threatening DR Detected. Use resultString to present to user.

“examResult.resultBinary”: A base64-encoded byte array for the result in a report form. The underlying format of the report is specified by the request to the ResultController (one of “NONE”, “PDF”, or “JPEG”).

“examResult.resultBinaryMD5”: An MD5 hash of the byte array in the “examResult.resultBinary” field.

“examResult.examQuality.examQualityCode”: An integer representing the exam quality assessment result. Possible values are listed in the table below.

Code	Interpretation
0	Sufficient Quality
1-6	The exam quality code will match the resultCode. See the resultCode definitions for interpretation.

“examResult.examQuality.imageQuality.imageID”: The unique image identifier.

“examResult.examQuality.imageQuality.imageQualityCode”: An integer indicating the result of the image quality assessment. Possible values are listed in the table below.

Code	Interpretation
0	Sufficient Quality
1	Poor Image Quality
2	Unknown Left/Right Eye. Require 2 left eye images and 2 right eye images.
3	Poor Optic Disc Location. Require 1 optic disc-centered and 1 fovea-centered image per eye.
4	Duplicate Images Detected

5	Image Analysis Failed.

4 Reference Information

Input image specifications	
Image file format	JPEG, TIF, PNG, BMP
Maximal image compression	Fundus Camera Manufacturer’s default or less compression
Imaging protocol	Per eye: 1 optic disc centered image 1 fovea centered image of the human retina
Image format	At least 1000x1000 pixels per image
Output specifications	
File format	JSON, PDF, JPG
Output values	Vision-threatening diabetic retinopathy detected Moderate diabetic retinopathy detected Negative Exam quality insufficient
Performance	
Processing time	Expected time 30s with above configuration
Retinopathy detection diagnostic performance	Clinical evaluation on a known clinical dataset surpassed the minimum standard of >80% sensitivity for vtDR on a sufficient quality image set.
Published reference system performance	Abramoff et al, Ophthalmology 2010 Abramoff et al, Diabetes Care 2008
Internal clinical evaluation	Performance testing on 8384 people with diabetes from an internal dataset with a reference standard based on grading by retinal experts