User Documentation for

Image Transformation API Service

Table of Content

[1. Introduction 3](#_Toc30489512)

[1.1 Content and Purpose 3](#_Toc30489513)

[1.2 Environment 3](#_Toc30489514)

[1.3 References 3](#_Toc30489515)

[2. API Usage 4](#_Toc30489512)

[2.1 Resizing Image 4](#_Toc30489513)

[2.2 Converting Images between formats 5](#_Toc30489514)

[2.3 Cropping Image 5](#_Toc30489515)

[2.4 Rotating Image 6](#_Toc30489517)

[2.5 GrayScale/ GreyScale Filter on Image 6](#_Toc30489518)

[2.6 Posterization Filter on Image 7](#_Toc30489519)

1. Introduction
   1. **Content and Purpose**

* Purpose of writing the Image Transformation service is to enable manipulation of remotely stored images of users in perspective of resizing, rotating, cropping, applying filters etc.
* Purpose of this document is to guide user on how to use this image service

* 1. **Environment**
* Server Environment - node js (8 or later)
* Requests handled by express.js - standard webserver framework for node js
* Source Code Editor – Microsoft Visual Studio Code
* Sharp – High performance image processing library
* I have used “**axios**” module to handle processing of remote url images as sharp do not have the functionality to handle remote image url
  1. **References**
* <https://sharp.pixelplumbing.com/>
* <https://expressjs.com/>

1. API Usage

**Note:**

**1. API route has domain as** [**http://localhost:8080**](http://localhost:8080)

**2. Remote Image url is common query parameter for all features**

**3. Common Error Message displayed in case of entering wrong or incomplete query parameters:**

**{error: “Invalid or Incomplete Query Parameters”}**

**4. Error message while entering wrong or invalid image url:**

**{error: “Invalid or Incomplete Remote image url”}**

2.1 **Resizing Image**

|  |  |
| --- | --- |
| Feature description | Resizing Image based on resize width and height |
| API route | http://localhost:8080/magic-image/resize-image |
| Method | GET |
| Request PARAMETERS (Query Params) | Required\*:  url=[string], w=[integer], h=[integer] |
| Success Response | -Res.ContentType as image/image-format and  -Resized Image |
| Error Response | -Error JSON with either inbuilt sharp library error message **OR**  -with custom error message (Read point 3 & 4 in **Notes** section) |
| Sample Request | <http://localhost:8080/magic-image/resize-image?url=https://cdn.pixabay.com/photo/2019/11/08/11/56/cat-4611189_960_720.jpg&w=300&h=200> |
| Notes | Click the URL in Sample Request for demo (Assuming your server is up and running on http://localhost:8080) |

* 1. **Converting Image Between Formats (jpg -> png , png->jpg etc.)**

|  |  |
| --- | --- |
| Feature description | Converting image from one format to another |
| API route | http://localhost:8080/magic-image/change-format |
| Method | GET |
| Request PARAMETERS (Query Params) | Required\*:  url=[string], format=[string] |
| Success Response | -Res.ContentType as image/image-format and  -Converted Image (Download/Save image (or by any other way) to check new format) |
| Error Response | -Error JSON with either inbuilt sharp library error message OR  -with custom error message (Read point 3 & 4 in **Notes** section) |
| Sample Request | <http://localhost:8080/magic-image/change-format?url=https://cdn.pixabay.com/photo/2019/11/08/11/56/cat-4611189_960_720.jpg&format=png> |
| Notes | -Click the URL in Sample Request for demo (Assuming your server is up and running on <http://localhost:8080>)  -values supported by **format** are (png,jpg/jpeg,webp,tiff,raw) |

* 1. **Cropping Image**

|  |  |
| --- | --- |
| Feature description | Cropping image based on Left Offset, Top Offset, width and height of cropped section. |
| API route | http://localhost:8080/magic-image/crop-image |
| Method | GET |
| Request PARAMETERS (Query Params) | Required\*:  url=[string], left=[integer], top=[integer], width=[integer], height=[integer] |
| Success Response | -Res.ContentType as image/image-format and  -Cropped Image |
| Error Response | -Error JSON with either inbuilt sharp library error message OR  -with custom error message (Read point 3 & 4 in **Notes** section) |
| Sample Request | <http://localhost:8080/magic-image/crop-image?url=https://cdn.pixabay.com/photo/2019/11/08/11/56/cat-4611189_960_720.jpg&left=70&top=100&width=400&height=300> |
| Notes | -Click the URL in Sample Request for demo (Assuming your server is up and running on <http://localhost:8080>) |

* 1. **Rotating Image**

|  |  |
| --- | --- |
| Feature description | Rotating image based on angle of rotation |
| API route | http://localhost:8080/magic-image/rotate-image |
| Method | GET |
| Request PARAMETERS (Query Params) | Required\*:  url=[string], rotate=[integer] |
| Success Response | -Res.ContentType as image/image-format and  -Rotated Image |
| Error Response | -Error JSON with either inbuilt sharp library error message OR  -with custom error message (Read point 3 & 4 in **Notes** section) |
| Sample Request | <http://localhost:8080/magic-image/rotate-image?url=https://cdn.pixabay.com/photo/2019/11/08/11/56/cat-4611189_960_720.jpg&rotate=100> |
| Notes | -Click the URL in Sample Request for demo (Assuming your server is up and running on <http://localhost:8080>) |

* 1. **GrayScale / GreyScale Filter on Image**

|  |  |
| --- | --- |
| Feature description | Applying grayscale/ greyscale filter on image |
| API route | http://localhost:8080/magic-image/apply-grayscale |
| Method | GET |
| Request PARAMETERS (Query Params) | Required\*:  url=[string], grayscale or greyscale |
| Success Response | -Res.ContentType as image/image-format and  -Image with Gray/Grey Scale filter |
| Error Response | -Error JSON with either inbuilt sharp library error message OR  -with custom error message (Read point 3 & 4 in **Notes** section) |
| Sample Request | <http://localhost:8080/magic-image/apply-grayscale?url=https://cdn.pixabay.com/photo/2019/11/08/11/56/cat-4611189_960_720.jpg&grayscale> |
| Notes | -Click the URL in Sample Request for demo (Assuming your server is up and running on <http://localhost:8080>)  -you can use **greyscale** or **grayscale** as param, its just functions with same implementation in sharp for different spelling |

* 1. **Posterization of input image**

|  |  |
| --- | --- |
| Feature description | Changing Color/Colour space of image |
| API route | http://localhost:8080/magic-image/posterize-image |
| Method | GET |
| Request PARAMETERS (Query Params) | Required\*:  url=[string], colorspace or colourspace=[string] |
| Success Response | -Res.ContentType as image/image-format and  -Image with posterized filter |
| Error Response | -Error JSON with either inbuilt sharp library error message OR  -with custom error message (Read point 3 & 4 in **Notes** section) |
| Sample Request | <http://localhost:8080/magic-image/posterize-image?url=https://cdn.pixabay.com/photo/2019/11/08/11/56/cat-4611189_960_720.jpg&colorspace=cmyk> |
| Notes | -Click the URL in Sample Request for demo (Assuming your server is up and running on <http://localhost:8080>)  **-possible values for color/colourspace are (srgb, b-w, lab, cmyk, rgb)**  **---If you pass any other value other than possible values, then it will not apply any filter and return original image as it is---**  -you can use **colorspace** or **colourspace** as param, it’s just functions with same implementation in sharp for different spelling |