
Moves-GAE Documentation

Release 0.0.1

Will Tekulve

Sep 09, 2017

Contents:

1	Images	1
1.1	List of Endpoints	1
1.2	API Reference	1
2	Videos	3
2.1	List of Endpoints	3
2.2	API Reference	3
3	Storage	5
3.1	API Reference	5
4	Download	7
4.1	List of Endpoints	7
4.2	API Reference	7
5	Indices and tables	9
	Python Module Index	11

List of Endpoints

- /upload/image: Handled by *image_upload()*.
- /upload/image/overlay: Handled by *image_upload_with_overlay()*.

API Reference

`images.delete_image()`

Deletes an image from Cloud Storage. Requires “post_id” url parameter.

Returns

- 204 - Image successfully deleted.
- 400 - Missing post_id url parameter or no such image found.

`images.image_upload()`

Endpoint for just image upload.

Expects form data fields “post_id”, “ext” and “content-type” which are used to save the output name. Expects image to be uploaded as multipart/file-upload and accessible under the name “image”.

Returns

- 204 - No content.
- 400 - Missing data field.
- 5XX - Request too large.

`images.image_upload_with_overlay()`

Endpoint for image + overlay upload. Has an additional form data field “overlay” that is required.

Returns

- 204 - No content.
- 400 - Missing data field.
- 5XX - Request too large.

`images.setup_routing(app: flask.app.Flask)`

Basic routing function for flask.

Parameters `app` (`flask.Flask`) – Your flask application object.

List of Endpoints

- /upload/video: Handled by `video_upload()`.
- /upload/video/overlay: Handled by `video_upload_with_overlay()`.

API Reference

`videos.delete_video()`

Deletes a video from Cloud Storage. Requires “post_id” url parameter.

Returns

- 204 - Video successfully deleted.
- 400 - Missing post_id url parameter or no such image found.

`videos.setup_routing(app: flask.app.Flask)`

Basic routing function for flask.

Parameters `app` (`flask.Flask`) – Your flask application object.

`videos.video_upload()`

This requires “post_id”, “ext” and “content-type” as form data keys along with “video” file key (also form data technically).

Returns

- 204 - No content.
- 400 - Missing data field.
- 5XX - Request too large.

`videos.video_upload_with_overlay()`

Video upload with an overlay image. This endpoint will schedule transcoding of the video (e.g. it won't happen in this request call).

Requires an extra “overlay” form data field that must be a PNG.

Returns

- 204 - No content.
- 400 - Missing data field.
- 5XX - Request too large.

Utilities for interacting with Google Cloud Storage.

API Reference

`storage.delete_image(filename: str) → bool`

Deletes an image with the given filename (without extension).

Parameters `filename` –

Returns `True` if deleted, `False` if not found.

`storage.delete_video(filename: str) → bool`

Deletes a video with the given filename (without extension).

Parameters `filename` –

Returns `True` if deleted, `False` if not found.

`storage.generate_image_path(post_id: str, ext: str) → pathlib.Path`

Generates the full path to be passed to `upload_data()`.

Parameters

- **post_id** – Output filename, usually gathered from an upload request.
- **ext** – Output extension, usually gathered from an upload request.

Returns Full path to be passed to `upload_data()`.

`storage.generate_transcoding_path() → pathlib.Path`

Generates a directory path that can be used for publishing transcode events.

Returns

`storage.generate_video_path(output_name: str, ext: str) → pathlib.Path`

Generates the full path to be passed to `upload_data()`.

Parameters

- **output_name** – Output filename, usually gathered from an upload request.
- **ext** – Output extension, usually gathered from an upload request.

Returns Full path to be passed to `upload_data()`.

`storage.get_image_url(filename: str) → typing.Union[str, NoneType]`

Gets an image's public url based on the filename without extension.

Parameters filename –

Returns Image's public URL.

`storage.get_public_url(filename: str) → typing.Union[str, NoneType]`

Get's a files public url based on the filename. Could be a video or image.

Parameters filename –

Returns Public URL

`storage.get_video_url(filename: str) → typing.Union[str, NoneType]`

Gets a video's public url based on the filename without extension.

Parameters filename –

Returns Video's public URL

`storage.upload_data(data: _io.BytesIO, content_type: str, full_path: pathlib.Path) → str`

Uploads binary data to the given path on Google Cloud Storage.

Parameters

- **data** – Binary data (for now just photos/videos).
- **content_type** – Correct content type for the data we're uploading.
- **full_path** – Full path including bucket.

Raises ValueError – If `full_path` begins with a leading slash.

Returns Public URL to access file.

List of Endpoints

- /download/image: Handled by `download_image()`.
- /download/video: Handled by `download_video()`.

API Reference

`download.download_image()`

This function expects a `post_id` URL parameter. Post ID corresponds to the filename w/o extension in Storage. This endpoint should redirect to the file's location in Storage.

Returns Redirects to the file's location in Storage.

`download.download_video()`

This function expects a `post_id` URL parameter. Post ID corresponds to the filename w/o extension in Storage. This endpoint should redirect to the file's location in Storage.

Returns Redirects to the file's location in Storage.

`download.setup_routing(app: flask.app.Flask)`

Basic routing function for flask.

Parameters `app` (`flask.Flask`) – Your flask application object.

CHAPTER 5

Indices and tables

- `genindex`
- `modindex`
- `search`

d

download, 7

i

images, 1

s

storage, 5

v

videos, 3

D

`delete_image()` (in module `images`), 1
`delete_image()` (in module `storage`), 5
`delete_video()` (in module `storage`), 5
`delete_video()` (in module `videos`), 3
`download` (module), 7
`download_image()` (in module `download`), 7
`download_video()` (in module `download`), 7

G

`generate_image_path()` (in module `storage`), 5
`generate_transcoding_path()` (in module `storage`), 5
`generate_video_path()` (in module `storage`), 5
`get_image_url()` (in module `storage`), 6
`get_public_url()` (in module `storage`), 6
`get_video_url()` (in module `storage`), 6

I

`image_upload()` (in module `images`), 1
`image_upload_with_overlay()` (in module `images`), 1
`images` (module), 1

S

`setup_routing()` (in module `download`), 7
`setup_routing()` (in module `images`), 2
`setup_routing()` (in module `videos`), 3
`storage` (module), 5

U

`upload_data()` (in module `storage`), 6

V

`video_upload()` (in module `videos`), 3
`video_upload_with_overlay()` (in module `videos`), 3
`videos` (module), 3