# 6. Application

## 6.1 About

The application is a storage system for photos. Users can create photos, pull photos, and request information about photos through the provided api. It allows them to remotely store their photos in an easily shareable fashion.

## 6.2 API

### 6.2.1 F1 Getting Picture Information

Description: Allows querying about whether a file exists with the specified filename. Will return code 200 if the file exists. Will return code 404 if the file does not exist.

Method: GET

URI: /photo/get/exists

Request Body:

{

“filename”:”<FILE\_NAME>”

}

Response Body:

{

“code”: 200 or 404

}

Development Status: TODO

### 6.2.1 F1 Getting Picture Information Cont.

Description: Allows getting the image specified by the filename requested. Returns the image file encoded as a base 64 string. Returns 404 if the image is not found.

Method: GET

URI: /photo/get/return

Request Body:

{

“filename”:”<FILE\_NAME>”

}

Response Body Success:

{

“code”:200,

“image”:”<IMAGE\_DATA in Base 64>”

}

Response Body Failure:

{

“code”:404

}

Development Status: TODO

### 6.2.2 F2 Creating a new Picture

Description: Allows the client to upload a new image to the server. Image must be encoded as a base 64 string. Returns back 200 if the image was successfully uploaded. Returns 500 if the server had an error while processing. Returns 403 if an image already exists with that filename.

Method: POST

URI: /photo/create

Request Body:

{

“filename”:”<FILE\_NAME>”,

“image”:”<IMAGE\_DATA in Base64>”

}

Response Body:

{

“status”:200 or 500 or 403

}

Development Status: TODO

### 6.2.3 F4 Deleting a Picture

Description: Allows the client to delete a previously uploaded image. Returns 200 if the image with that name was deleted. Returns 500 for internal server error while attempting to delete. Returns 404 if the image with the filename requested was not found on the server.

Method: DELETE

URI: /photo/delete

Request Body:

{

“filename”:”<FILE\_NAME”>

}

Response Body:

{

“status”: 200 or 500 or 404

}

Development Status: TODO

### 6.2.4 F3 Editing a Picture

Description: Allows the client to replace the contents of a previously uploaded image. Returns 200 on successful upload. Returns 500 on error. Returns 404 if no file with that name was found to edit.

Method: PUT

URI: /photo/edit

Request Body:

{

“filename”:”<FILE\_NAME>”,

“image”:”<IMAGE\_DATA in Base64>”

}

Response Body:

{

“status”: 200 or 500 or 404

}

Development Status: TODO

### 6.2.5 F5 Querying about Images

Description: Returns a list of filenames which match the criteria of the search term using it as a prefix for the files on the server. If no search term is specified, then all files on the server will be returned.

Method: GET

URI: /photo/list

Request Body:

{

“search-term”:”<FILE\_SEARCH\_TERM>” <OPTIONAL>

}

Response Body:

{

“files”: [

“filename 1”,

“filename 2”,

…

],

“count”: 0 (number of files returned)

}

Development Status: TODO

### 6.2.5 F5 Deleting Matched Images

Description: Deletes all photos on the server which match the criteria. If a file extension is specified it will delete all photos with that file extension. If no extension is specified, then all photos will be removed from the server.

Method: DELETE

URI: /photo/list

Request Body:

{

“file-extension”:”<IMAGE\_EXTENSION>” <OPTIONAL>

}

Response Body:

{

“status”:200 or 500

“count”: 0 (count of the files deleted)

}

Development Status: TODO