-- Работа для программистов в Германии --
www.DerTraktor.com
E-mail: cv@dertraktor.com
Skype: dertraktor.com

Uploader Bot

The goal of task is to test your php skills. Please write your “best code” to get the best contract.

1. *Introduction*

The task is to write a command line script - Bot. It resizes given images and saves them to remote cloud storage.

The workflow should be divided into the following independent steps:

- Schedule list of images to be processed.
- Resize scheduled images.
- Upload resized image to cloud storage.
  Each step works with its own queue, e.g. `scheduler` adds filename to `resize` queue, `resizer` takes filename from `resize`queue, does resize and put filename to `upload` queue and so on.

After all there are the following queues:

- `resize` - files ready to be resized.
- `upload` - files ready to be uploaded.
- `done` - completed files.
- `failed` - failed files.
  If any step fails it should move file to `failed` queue. For example, if image could not be uploaded right now (e.g. due to network problems) corresponding filename should be moved to `failed` queue and it should be possible to retry later.

2. *Requirements*

- Code should be uploaded to public github or bitbucket repository.
- Code format should conform PSR-2.
- There should be `composer.json` file with list of external dependencies and autoloading rules.
- There should be one config file in any format (`json` , `yaml` , `ini` , `php`) with path to temporary folder and credentials of remote file storage.
- Bot should work under Linux (Ubuntu, Debian, Centos) system.
- `README.md` should contain information about how to install required software, do initial provisioning, edit config file and run bot.
- Feel free to choose any storage for queues, e.g. RabbitMQ, beanstalkd or any relational database. It should be easy to install environment on test machine.

- `Vagrantfile` or `Dockerfile` will be a great advantage. Otherwise `README.md` should contain information about how to install required software.
- Tests are not required.

3. *CLI Script*

Bot should be implemented as a PHP command line script named `bot`. Running bot without arguments should output full list of supported commands:

```
$ bot
Uploader Bot
Usage:
  command [arguments]
Available commands:
  schedule  Add filenames to resize queue
  resize    Resize next images from the queue
  status    Output current status in format %queue%:%number_of_images%
  upload    Upload next images to remote storage
```

Description of commands are listed below in this section.

1. Scheduler

Accepts a path to the directory with images and schedule them for resize, i.e. adds to `resize` queue.

```
$ bot schedule ./images
```

Directory `images` contains only images in different formats:

```
$ ls images
first.png     second.jpg     third.png     5.jpg
```

2. Resizer

Takes next `count` of images from `resize` queue and resizes them to `640x640` pixels in `jpg` format. If image is not a square shape resizer should make it square by means of adding a white background. If there is an error URL should be moved to `failed`queue.

```
$ bot resize [-n <count>]
```

If parameter -n is omitted resize should work on all images from resize queue.

Resized images should be stored in directory called `images_resized`. If resize goes well original image should be removed from`images` directory.

3. Uploader

Uploads next `count` of images from `upload` queue to one of the remote storages. Type of cloud storage and corresponding credentials should be set in config file. There can be only one remote storage at the moment. Bot should support one storage from the list:

- Dropbox
- Google Drive
- Amazon S3

After image is uploaded move its filename to `done` queue. In case of any error move filename to `failed` queue.

```
$ bot upload [-n <count>]
```

If parameter -n is omitted upload should work on all images from the queue.

4. Monitoring

Outputs all queues with a count of URLs in each of them.

```
$ bot status
Images Processor Bot
Queue      Count
resize     0
upload     12
done       42
failed     4
```

5. Rescheduler

Moves all URLs from `failed` queue back to `resize` queue.

```
$ bot retry [-n <count>]
```