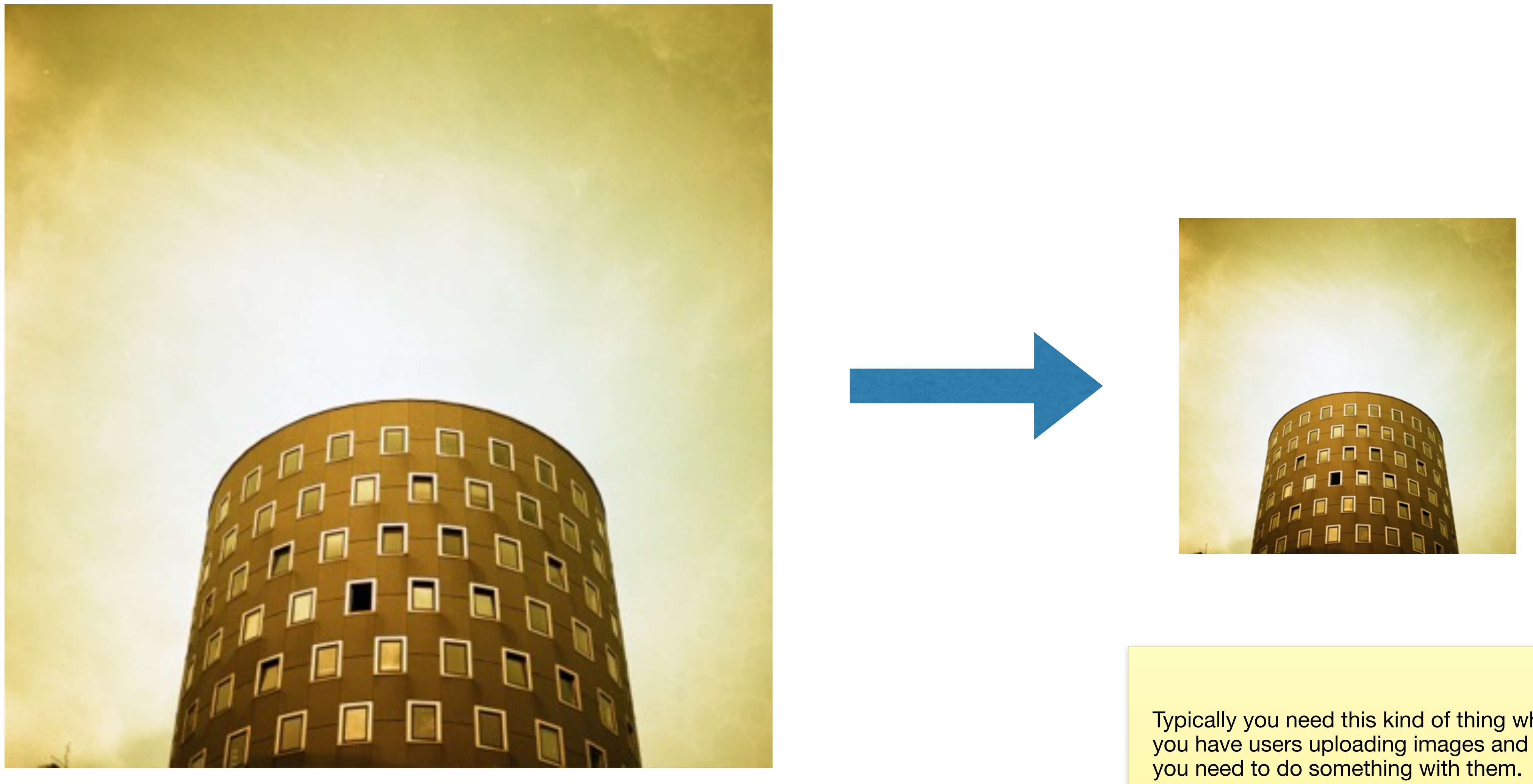
super fast image resizing with ruby

1. What?

Creating smaller or bigger versions of images (mostly jpegs) in ruby, really quickly.



Typically you need this kind of thing when you have users uploading images and then you need to do something with them. Maybe display a smaller version, maybe crop them to fit into your layout. Something like that where you can’t resize the image during development,

Traditionally accomplished by something like:
rmagick or mini_magick.
Both of these are ImageMagick based.

Ever used core image? it’s nice and fast, but Apple only and proprietary.

What we have been using recently is called libvips (or ruby-vips for the ruby wrapper). It has a lot of the advantages we find in Core Image but it is open source software.

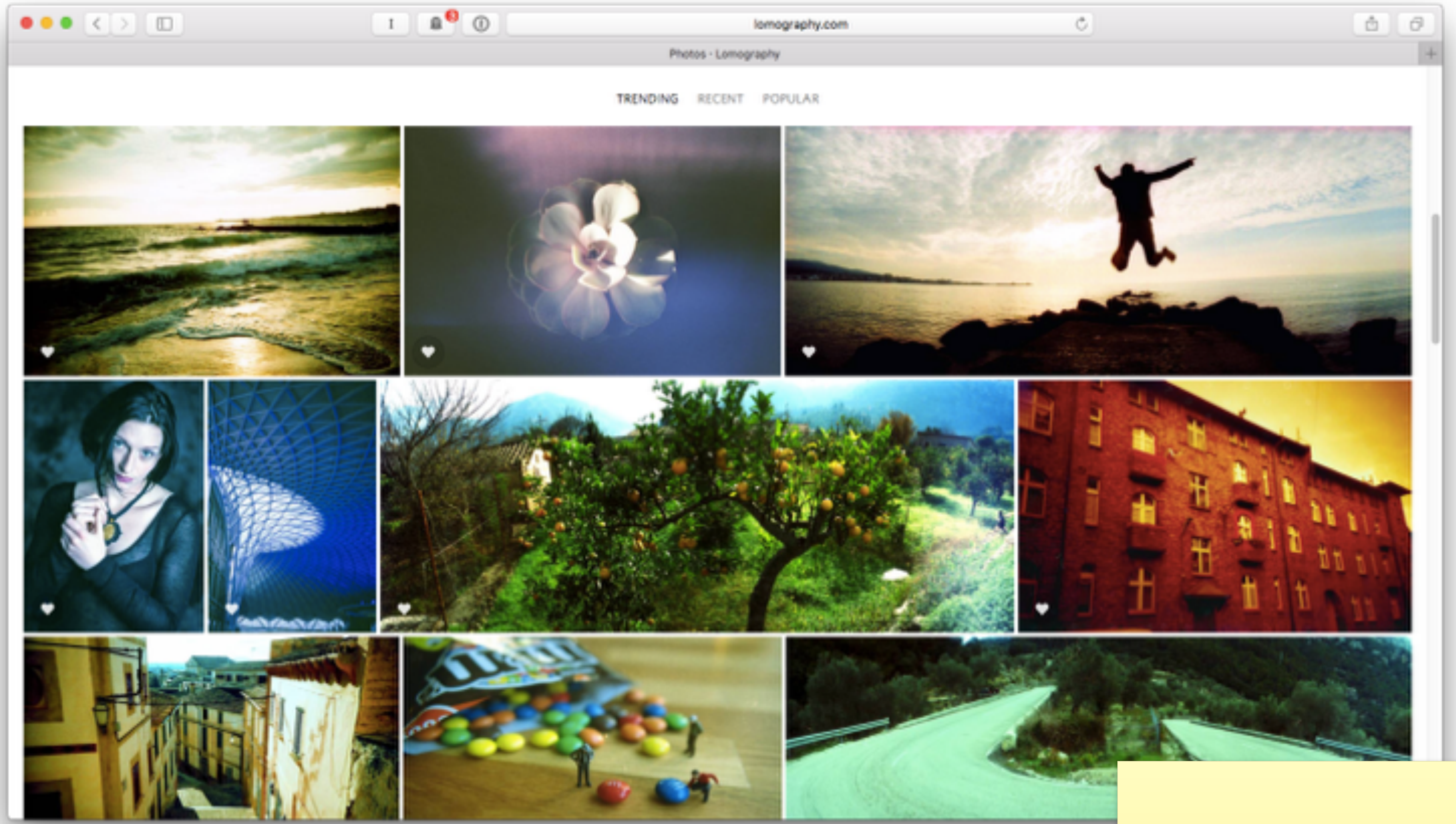
libvips

So, that's where we are going, but first I'm gonna give you some context and tell you where I'm coming from.

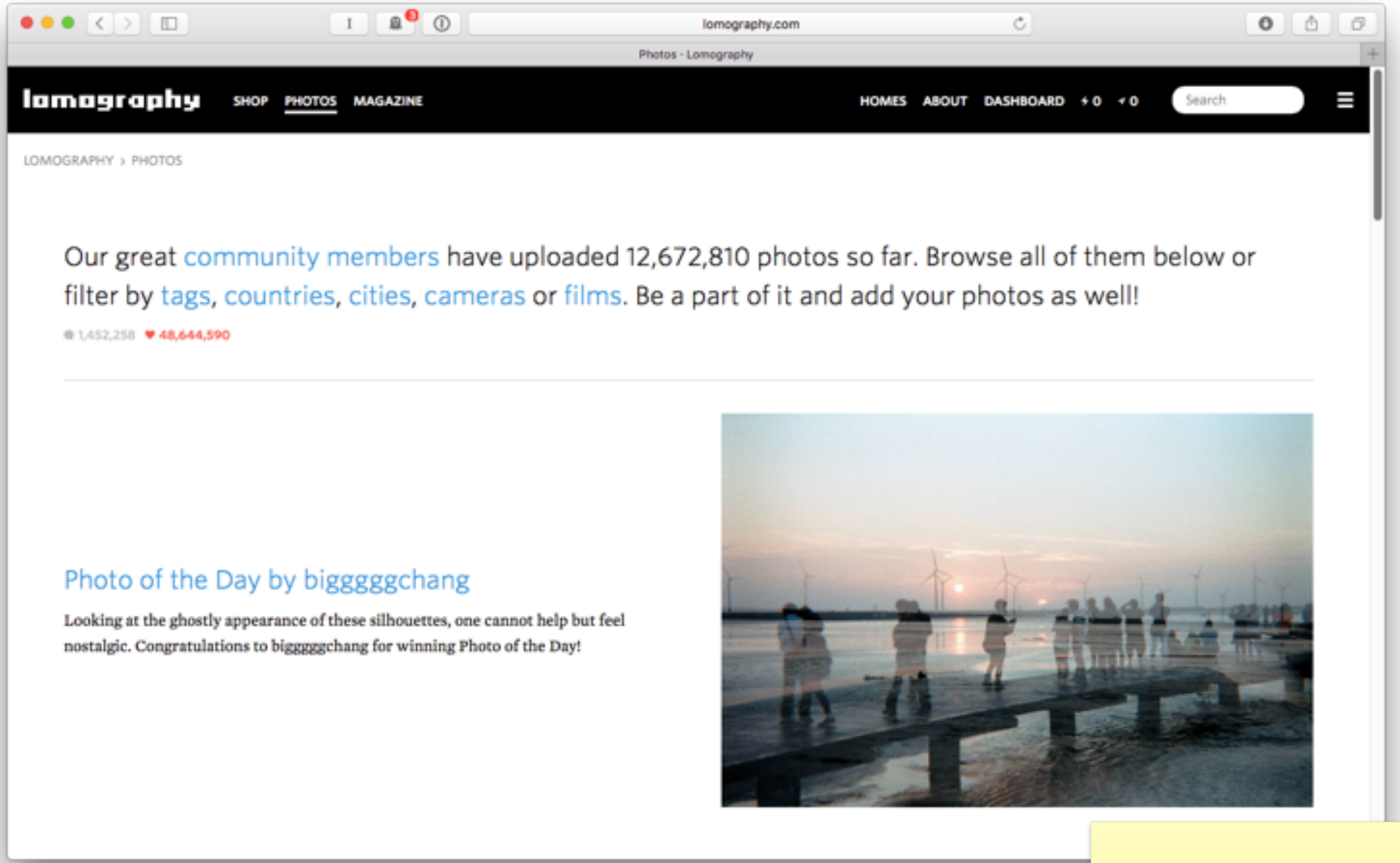
2. Why?

In my day-job I work at Lomography.
We make cameras like this one.

And a bunch of other photographic products like these lenses.



But, more relevant to what I'm talking about we also run a small photo sharing community.



lomography.com

Photos - Lomography

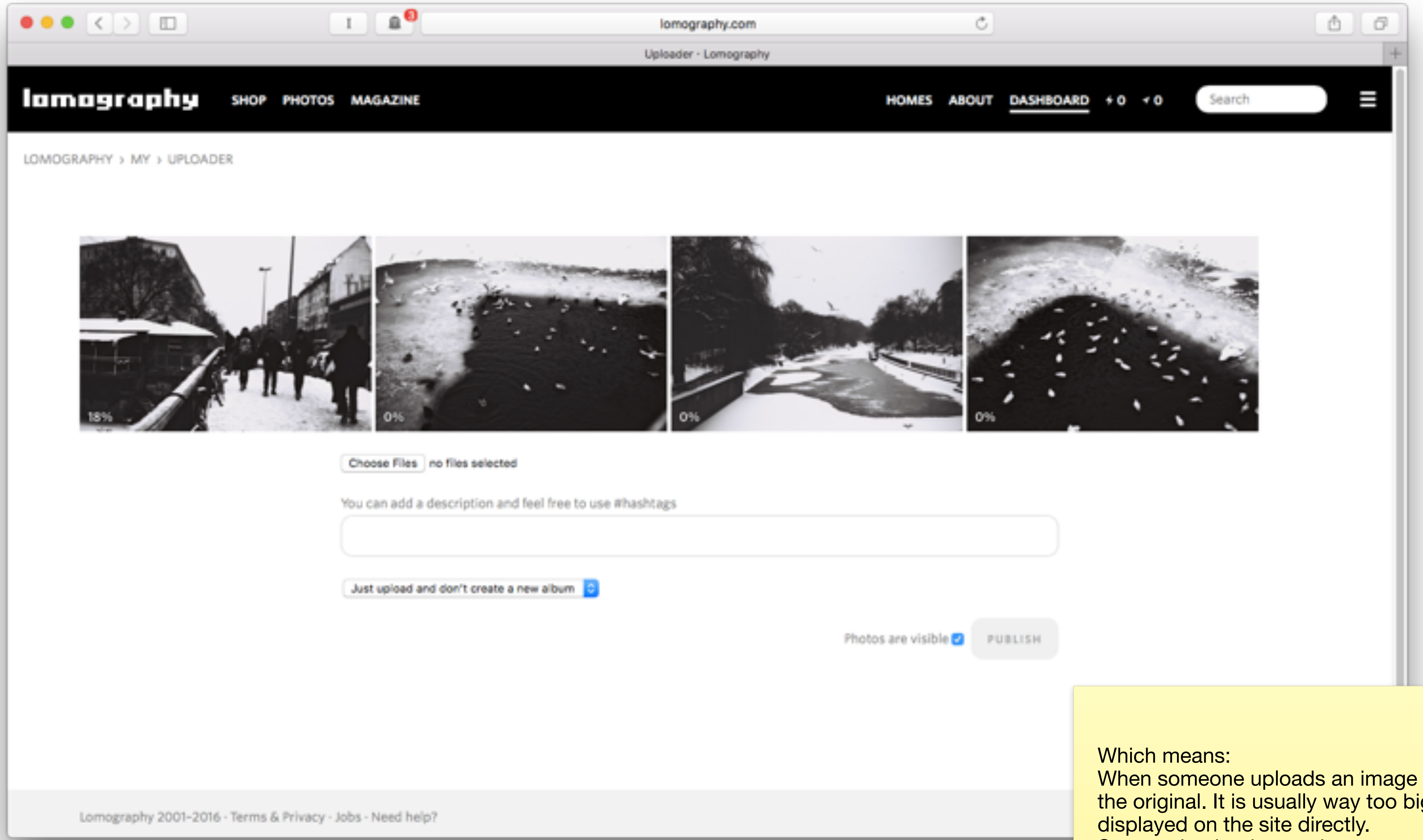
Our great community members have uploaded 12,672,810 photos so far. Browse all of them below or filter by tags, countries, cities, cameras or films. Be a part of it and add your photos as well!

1,452,258 ♥ 48,644,590

Photo of the Day by bigggggchang

Looking at the ghostly appearance of these silhouettes, one cannot help but feel nostalgic. Congratulations to bigggggchang for winning Photo of the Day!

We have around 12 million images uploaded by users.
Most of these are analog, if you are wondering that are about 400.000 rolls of film processed, scanned and uploaded.
We resize the photos on the fly.


lomography.com
Uploader - Lomography
lomography
SHOP
PHOTOS
MAGAZINE
HOMES
ABOUT
DASHBOARD
Search
LOMOGRAPHY › MY › UPLOADER
18%
0%
0%
0%
Choose Files
no files selected
You can add a description and feel free to use #hashtags
Just upload and don't create a new album
Photos are visible
PUBLISH
Lomography 2001-2016 · Terms & Privacy · Jobs · Need help?

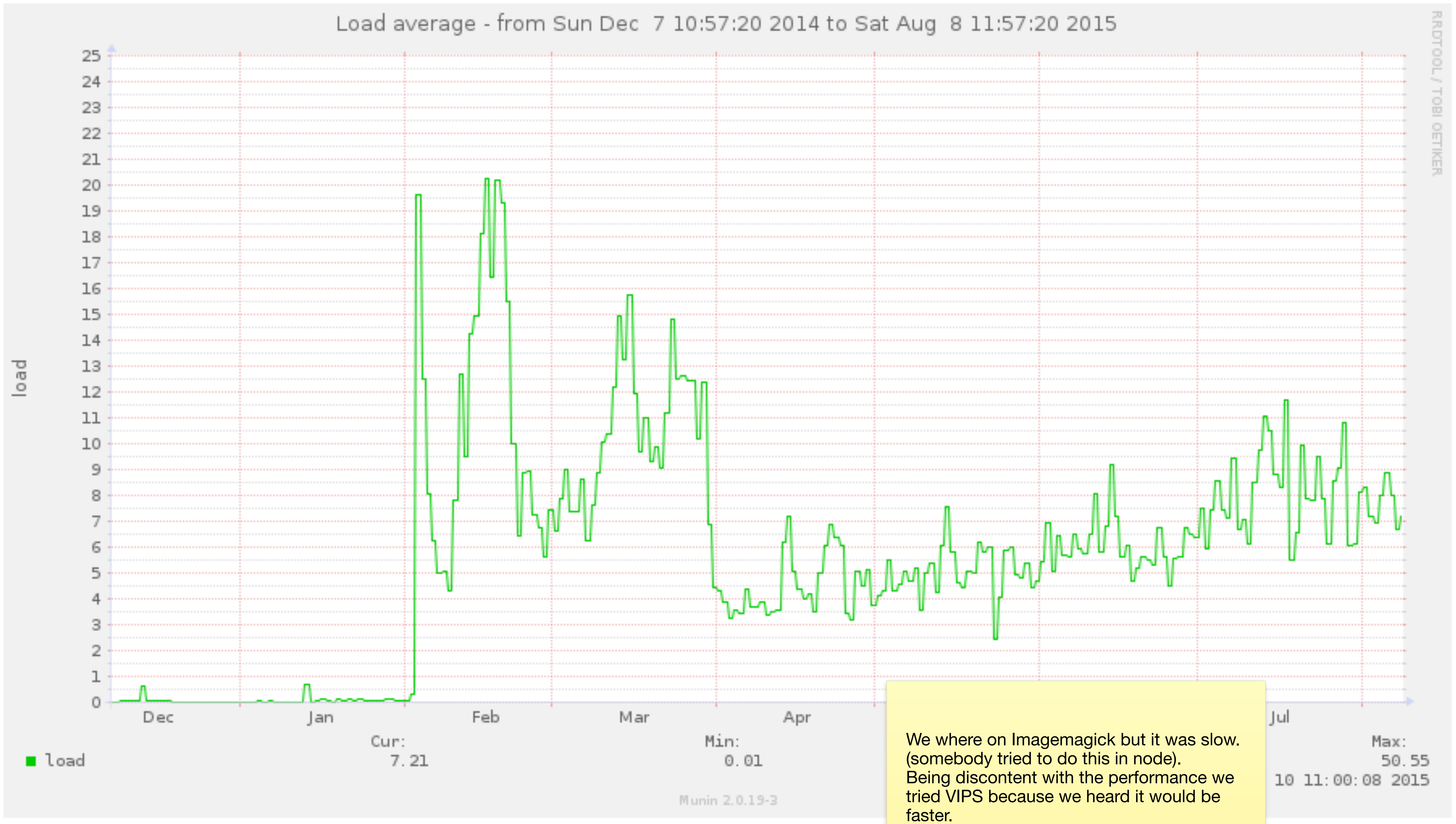
Which means:
When someone uploads an image we save the original. It is usually way too big to be displayed on the site directly.
So we take that image that we got and pass it on to a second rails app, which then returns a sha1 hash of the contents and the ratio of the image.

```
http://cdn1.lomography.com
/bb/4b8e36af4a4fe8d484af74da2f23706e914d17/576x374x2.jpg
?auth=e9904660a0f0090381c358db531d3d9204e2dbfb
```

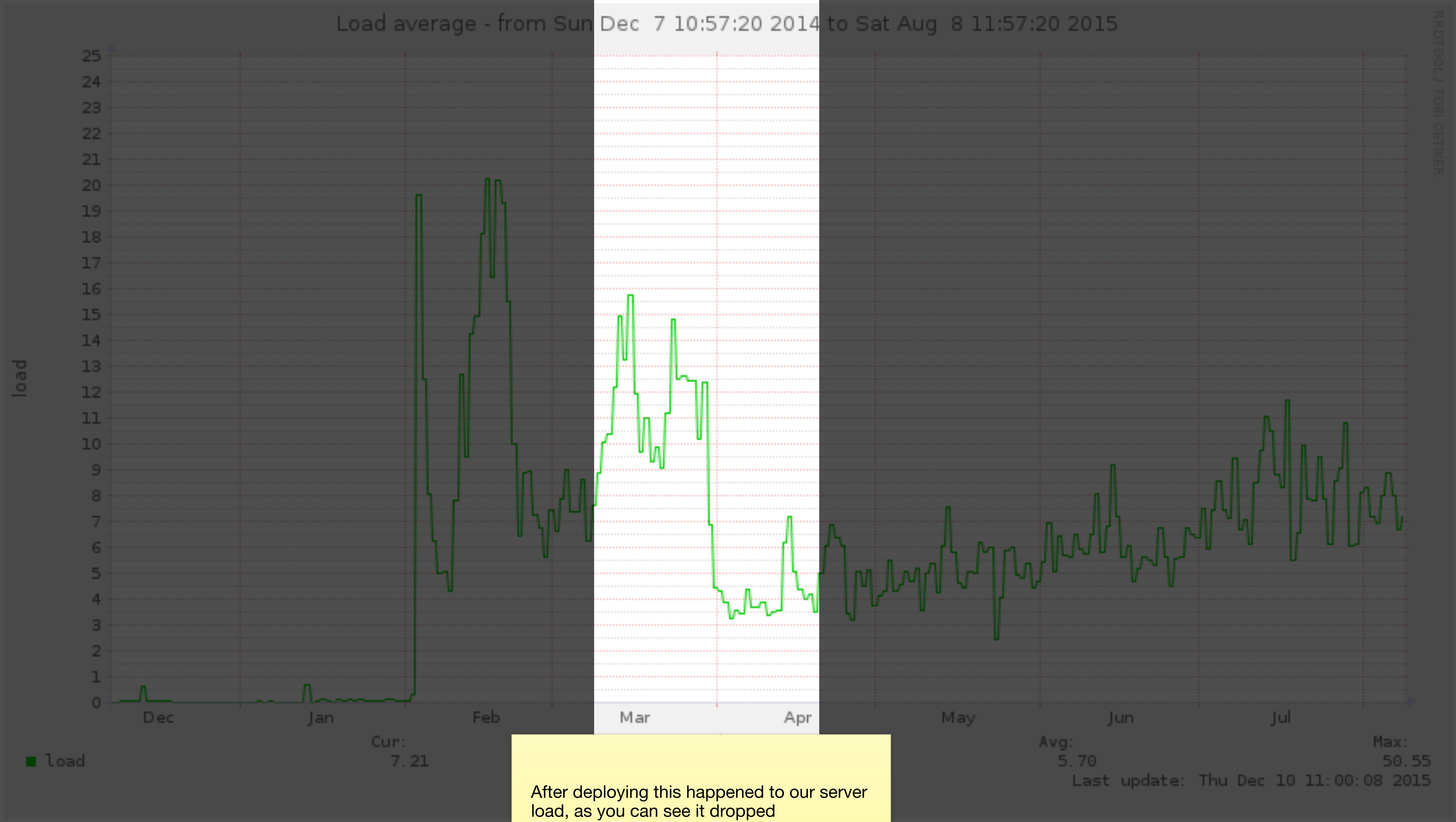
With the hash and the ratio to community app knows how to build links to images when we place them on the site. When someone views the image on the site we transparently create a resize and save that.

upside: the community decoupled form the specifics of resizing and storing photos, just knows how to build links to them.

downside: the fist time anybody sees a certain size of a photo it needs to be created on the fly.
So resizing needs to be _fast_.



We where on Imagemagick but it was slow.
(somebody tried to do this in node).
Being discontent with the performance we tried VIPS because we heard it would be faster.
What you see here is the load graph form our assets server.
We ported our assets server to use VIPS.



After deploying this happened to our server load, as you can see it dropped dramatically. We also got shorter response times (5x maybe).

3. How?

Let’s look at some code!

lazy loading

```ruby
require 'rubygems'
require 'vips'

include VIPS

image = Image.jpeg('photo.jpg')

# You can read all the header data without triggering a pixel load.
puts image.x_size

```

Once you require and include vips you can instantiate an Image object with a path.
The fist thing that is nice and different from most libraries is that VIPS does a lot of lazy loading.
You can access almost all metadata, including image dimensions, before any pixel data is loaded into memory.
This makes that kind of operation super fast.

shrink on load

```ruby
require 'rubygems'
require 'vips'

include VIPS

image = Image.jpeg('photo.jpg', shrink_factor: 2)

writer = JPEGWriter.new(image, :quality => 80)
writer.write('output.jpg')
```

Once you are actually loading, say a jpeg, into memory and you know that you will only need a smaller version you can do what is called a shrink load.
This is a way to load your jpeg form disk and instantly reduce it's dimensions by a factor of 2, 4 or 8.
And if that is all you needed you can already write the image back out to disk.

✨shrink on load✨

```ruby
require 'rubygems'
require 'vips'

include VIPS

image = Image.jpeg('photo.jpg', shrink_factor: 2)

writer = JPEGWriter.new(image, :quality => 80)
writer.write('output.jpg')

```

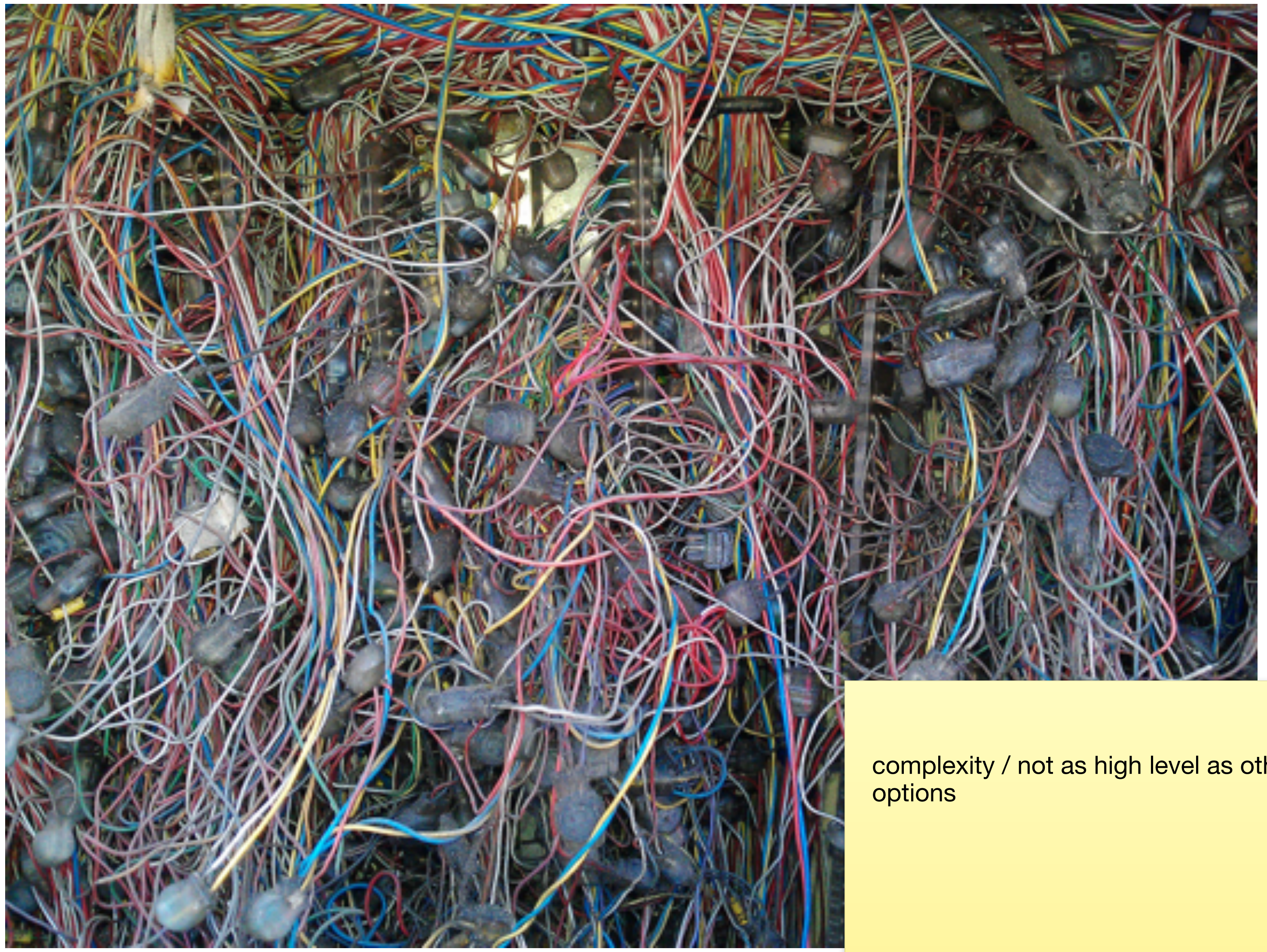
This is the thing that causes us to use VIPS! Only works well for a factor of 2, 4 or 8 so it is usually an intermediate step.

scale

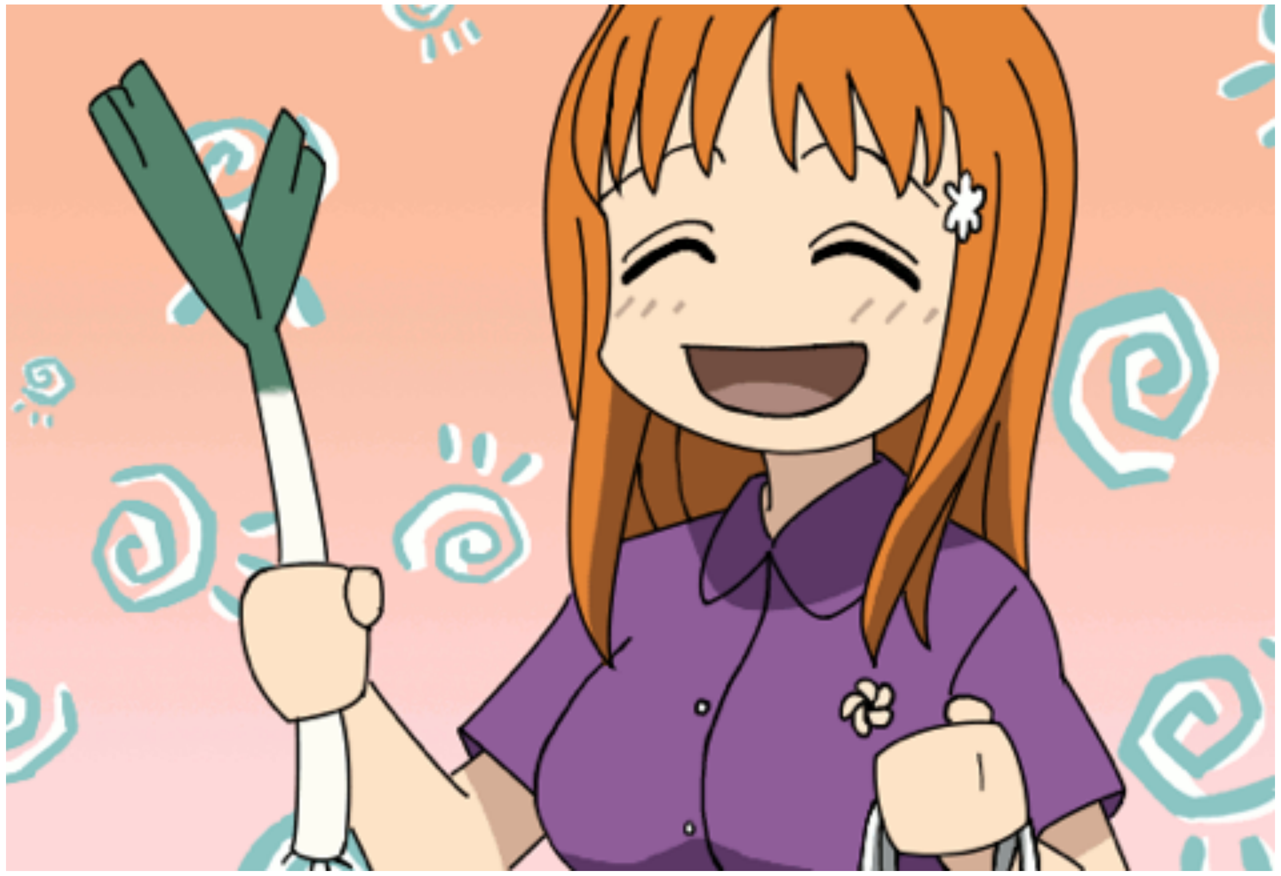
```

target_x_size = 2000
target_y_size = 1000

# load just header:
image = Image.jpeg('photo.jpg')

if (image.x_size / 2 > target_x_size) && (image.y_size / 2 > target_y_size)
  shrink = 2
else
  shrink = 1
end

# pixel load and shrink:
image = Image.jpeg('photo.jpg', shrink_factor: shrink)

# actually scale the rest of the way:
x_scale = target_x_size.to_f / image.x_size.to_f
y_scale = target_y_size.to_f / image.y_size.to_f
image = image.affinei_resize(:bicubic, x_scale, y_scale)

# write rsult to disk:
writer = JPEGWriter.new(image, :quality => 80)
writer.write('output.jpg')

```

Let's look at something more complex.
When you are actually resizing an image you probably want a specific width and height.
To achieve this as quickly as possible, you would use a load shrink to get as close as possible to the desired size and then do a 'proper' scale for the rest of the way.
This work's way faster than ImageMagick and produces comparable results.

So, that’s all nice, what are the downsides?

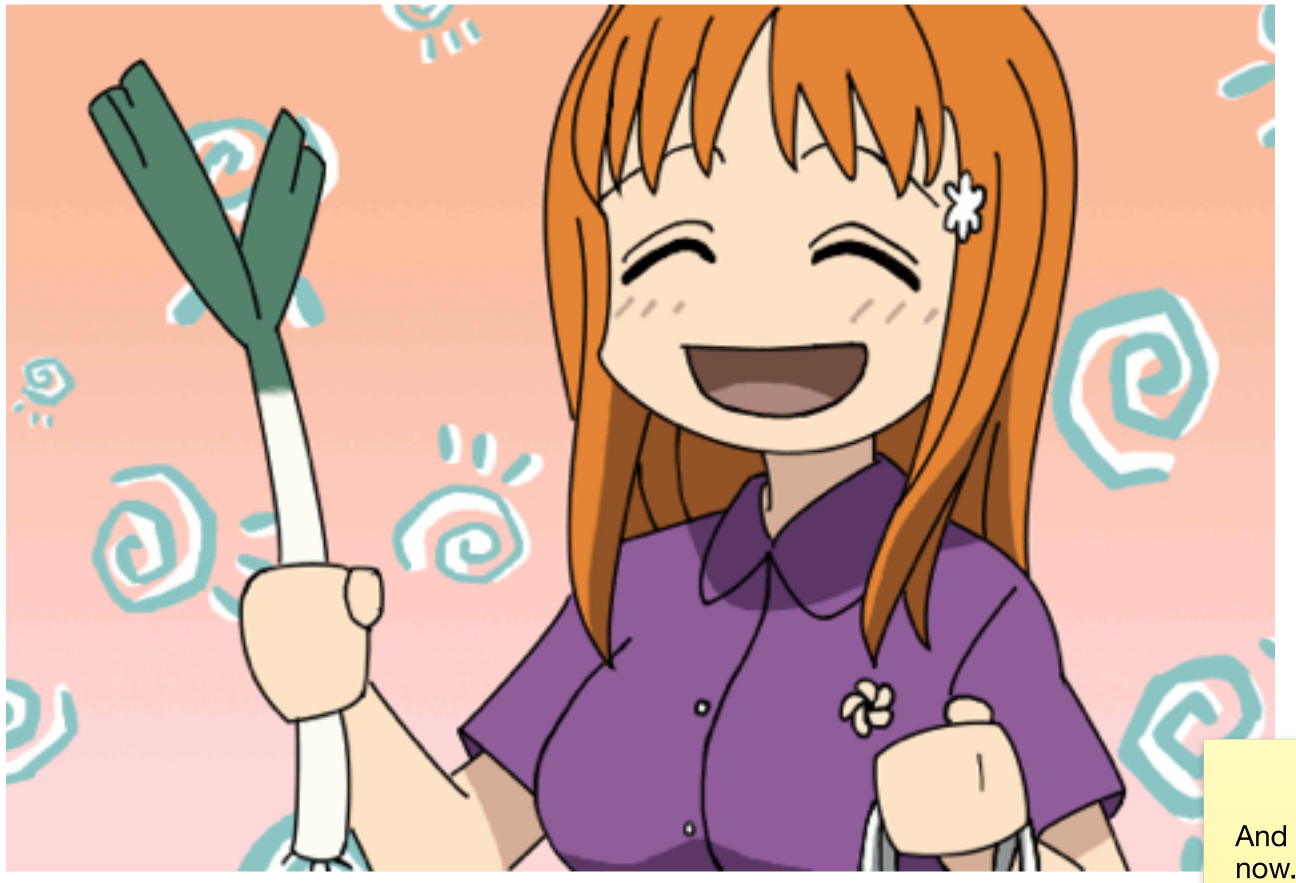


complexity / not as high level as other options



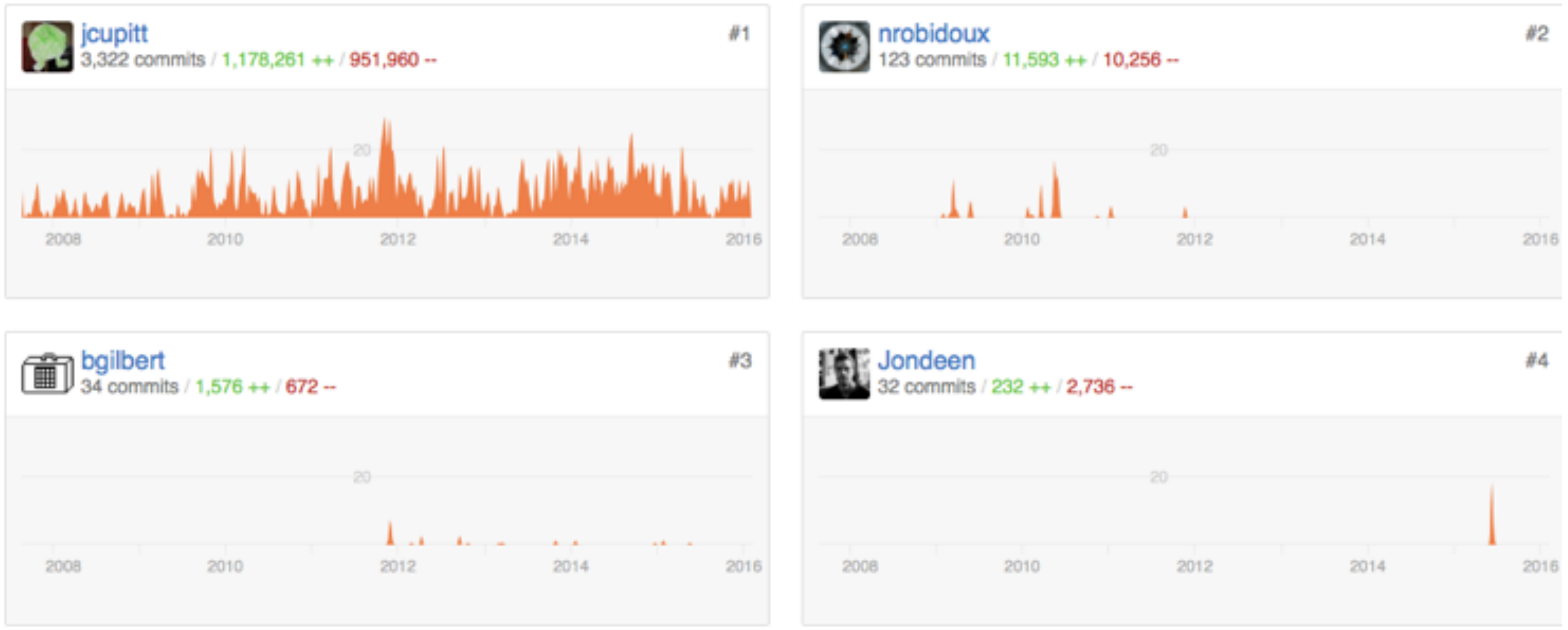
it’s leeky

We have encountered some memory leaks.



And ‚solved' them with rolling restarts for now.

passenger_max_requests = 750 :(


jcupitt #1
3,322 commits / 1,178,261 ++ / 951,960 --

nrobidoux #2
123 commits / 11,593 ++ / 10,256 --

bgilbert #3
34 commits / 1,576 ++ / 672 --

Jondeen #4
32 commits / 232 ++ / 2,736 --

There is mostly one contributor right now.

libvips

1. image resizing library and gem
2. about 5 times faster™
3. some trade-offs

https://github.com/jcupitt/libvips
https://github.com/jcupitt/ruby-vips

@michaelem

Typeface: »Vollkorn« by Friedrich Althausen, vollkorn-typeface.com