"photos\_attributes"=>

{"0"=>{"description"=>"Gavin"}, "1"=>{"description"=>"Gavin"}}}

"photos\_attributes"=>

{"0"=>

{"file"=>

#<ActionDispatch::Http::UploadedFile:0x83305e0

@content\_type="image/jpeg",

@headers=

"Content-Disposition: form-data; name=\"authority[photos\_attributes][0][

file]\"; filename=\"AlbumArtSmall.jpg\"\r\nContent-Type: image/jpeg\r\n",

@original\_filename="AlbumArtSmall.jpg",

@tempfile=

#<File:C:/Users/Embassy/AppData/Local/Temp/RackMultipart20180301-5036-1d

2kli9.jpg>>,

"description"=>""}}}

heroku git:remote -a gavinjell

git push heroku master --app gavinjell

heroku addons:create heroku-postgresql --app gavinjell

heroku run rake db:schema:load --app gavinjell

gem 'ckeditor', '4.1.3'

Include this inside your config/initializers/assets.rb:

Rails.application.config.assets.precompile += %w( ckeditor/\*)

gem 'paperclip'

rails generate ckeditor:install --orm=active\_record --backend=paperclip

**Mongoid + paperclip**

gem 'mongoid-paperclip', require: 'mongoid\_paperclip'

rails generate ckeditor:install --orm=mongoid --backend=paperclip

**Load generated models**

All ckeditor models will be generated in the app/models/ckeditor directory. Models are autoloaded in Rails 4. For earlier Rails versions, you need to add them to the autoload path (in application.rb):

config.autoload\_paths += %w(#{config.root}/app/models/ckeditor)

Mount the Ckeditor::Engine in your routes (config/routes.rb):

mount Ckeditor::Engine => '/ckeditor'

Include ckeditor javascripts in your app/assets/javascripts/application.js:

//= require ckeditor/init

Make sure that you put it before:

//= require\_tree .

### Form helpers

= form\_for @page do |form|

= form.cktext\_area :notes, class: 'someclass', ckeditor: { language: 'uk'}

= form.cktext\_area :content, value: 'Default value', id: 'sometext'

= cktext\_area :page, :info, cols: 40, ckeditor: { uiColor: '#AADC6E', toolbar: 'mini' }

It also works with bootstrap-form

= bootstrap\_form\_for resource do |form|

= form.cktext\_area :text, ckeditor: { language: 'uk'}

### Pundit integration

Just like CanCan, you can write this code in your config/initializers/ckeditor.rb file:

Ckeditor.setup do |config|

config.authorize\_with :pundit

end

Then, generate the policy files for model **Picture** and **AttachmentFile**

$ rails g ckeditor:pundit\_policy

By this command, you will got two files:

app/policies/ckeditor/picture\_policy.rb app/policies/ckeditor/attachment\_file\_policy.rb

By default, only the user that logged in can access the models (with actions index and create) and only the owner of the asset can **destroy** the resource.

You can customize these two policy files as you like.

#### Custom toolbars example

Adding a custom toolbar:

# in app/assets/javascripts/ckeditor/config.js

CKEDITOR.editorConfig = function (config) {

// ... other configuration ...

config.toolbar\_mini = [

["Bold", "Italic", "Underline", "Strike", "-", "Subscript", "Superscript"],

];

config.toolbar = "mini";

// ... rest of the original config.js ...

}

When overriding the default config.js file, you must set all configuration options yourself as the bundled config.js will not be loaded. To see the default configuration, run bundle open ckeditor, copy app/assets/javascripts/ckeditor/config.js into your project and customize it to your needs.

In controller

authority\_params[:photos\_attributes][:"0"][:file]

In view

ff.object.file\_contents

ff.object.content\_type

ff.object.filename

SELECT "photos".\* FROM "photos" WHERE "photos"."authority\_id" = 20 AND "photos"."id" = 16 [["authority\_id", 20]]