```
require 'act_as_page_extractor/version'

require 'active_record'

require 'awesome_print'
require 'filesize'
require 'total_compressor'
require 'docsplit'
require 'pdf_utils'
require 'prawn'
require 'pdf-reader'

require 'act_as_page_extractor/modules/tools.rb'
require 'act_as_page_extractor/modules/validating.rb'
require 'act_as_page_extractor/modules/unzipping.rb'
require 'act_as_page_extractor/modules/extracting.rb'
require 'act_as_page_extractor/modules/saving.rb'

require 'act_as_page_extractor/modules/interface'

module ActAsPageExtractor

  extend ActiveSupport::Concern

  included do
    before_create { self.page_extraction_state = EXTRACTING_STATES[:new] }
    before_destroy :remove_files
  end

  # attr_reader :options

  module ClassMethods
    def act_as_page_extractor(options: {})
      define_method(:save_as_pdf){|*args| options[:save_as_pdf] }
      define_method(:extracted_filename){|*args| self.send(options[:filename].to_sym) }
      ActAsPageExtractor.define_singleton_method(:extracted_filename) {|*args| options[:filename] }
      ActAsPageExtractor.define_singleton_method(:document_class) {|*args|
options[:document_class].constantize }
      define_method(:extracted_document_id){|*args| options[:document_id] }
      define_method(:additional_fields){|*args| options[:additional_fields] }
    end
  end

  EXTRACTING_STATES = {
    new: 'new',
    extracting: 'extracting',
    extracted: 'extracted',
    'error.extraction': 'error.extraction'
  }.freeze
```

```
TMP_EXTRACTION_FILE_STORAGE = "#{Dir.pwd}/tmp/page_extraction".freeze
FILE_STORAGE = "#{Dir.pwd}/public".freeze
PDF_STORAGE = "#{FILE_STORAGE}/uploads/extracted/pdf".freeze

def initialized
  # add all need callbacks
    #on destroy remove pdf

  #Add to Readme!!
  #rails g act_as_page_extractor:migration Document category_id user_id
  # add to [Document] model:
  # has_many :extracted_pages, dependent: :destroy
  create_pdf_dir
end

def page_extract!
  initialized
  cleanup_pages
  create_tmp_dir
  begin
    copy_document
    # debug_info
    unzip_document
    if valid_document
      extract_pages
      save_to_db
    end
  ensure
    update_state
    save_pdf
    finish
  end
end

def create_pdf_dir
  if save_as_pdf
    FileUtils::mkdir_p(PDF_STORAGE) unless File.exists?(PDF_STORAGE)
  end
end

def create_tmp_dir
  @tmp_dir = "#{TMP_EXTRACTION_FILE_STORAGE}/#{SecureRandom.hex(6)}"
  FileUtils::mkdir_p(@tmp_dir) unless File.exists?(@tmp_dir)
end

def copy_document
  @origin_document_path = "#{FILE_STORAGE}#{self.send(:extracted_filename).url.to_s}"
  ap @origin_document_path
```

```
    FileUtils.cp(@origin_document_path, @tmp_dir)
    @copy_document_path = "#{@tmp_dir}/#{@origin_document_path.split("/").last}"
    @document_filename = @origin_document_path.split("/").last
  end

  def finish
    remove_tmp_dir
  end

  def remove_tmp_dir
    FileUtils.rm_rf(@tmp_dir) if @tmp_dir =~ /\/tmp\//
  end
end

# rails g model ExtractedPage page:text document_id:integer category_id:integer page_number:integer

# Rails 4 way
# 9.2.7.1 Multiple Callback Methods in One Class
# 258 page

# class ActiveRecord::Base
#   def self.acts_as_page_extractor(document_field=:filename)
#     auditor = Auditor.new(audit_log)
#     after_create auditor
#     after_update auditor
#     after_destroy auditor
#   end
# end
```