```
module Jekyll
  module ListingPages
    VERSION = "0.1.7"
    class ListingPages < Generator
      # This generator is safe from arbitrary code execution.
      safe true

      # This generator should be passive with regard to its execution
      priority :lowest

      # Generate paginated pages if necessary.
      #
      # site - The Site.
      #
      # Returns nothing.
      def generate(site)
        all_pages = site.pages
        all_pages.each do |page|
          if page.data['isdup'] != true
            if page.data['list_for'] and page.data['list_limit']
              all = []
              if page.data['list_for'] == 'posts'
                all = page.site.posts.reject { |entry|
entry['hidden'] }.sort_by { |el|
                  date = el.data['date']
                  date.to_s + el.data['title']
                }.reverse
              else
                all =
page.site.collections[ page.data['list_for'] ].docs.sort_by { |el|
                  pub_date = el.data['pub_date'] == nil ?
Time.new( 1900,01,01, 00,00,00) : el.data['pub_date']
                  pub_date.to_s + el.data['title']
                }.reverse
              end
              pages = get_pages(all, page.data['list_limit'].to_i)
              (1..pages).each do |num_page|
                newpage = Page.new(site, site.source, "", page.name )
                if num_page == 1
                  add_pagedata(page,num_page, all,
page.data['list_limit'].to_i);
#                  site.pages << page # add newpage to site pages
(causes dup pg bug...)
                end
                newpage.data['page_baseurl'] = page.url
                add_pagedata(newpage,num_page, all,
page.data['list_limit'].to_i)
                newpage.data['permalink'] += num_page.to_s + '/
index.html'
                newpage.data['menuexclude'] = true
                site.pages << newpage # add newpage to site pages
                Jekyll.logger.debug "Pagination: Generating",
num_page.to_s + "... source:" + site.source + "\n"

              end
            end
          end
        end
      end

      def add_pagedata(newpage,num_page,pages,limit)

        newpage.data['isdup'] = true

        # Pagination data
        newpage.data['curpage'] = num_page
        newpage.data['total'] = pages.size
        newpage.data['pages'] = ( pages.size.to_f / limit.to_i ).ceil
        min_rec = ( ( num_page - 1 ) * limit )
        newpage.data['min_rec'] = min_rec + 1
        max_tmp = ( min_rec + limit );
        max_rec = max_tmp >= pages.size ? pages.size : max_tmp
        newpage.data['max_rec'] = max_rec
        newpage.data['list'] = pages.slice( min_rec.to_i, limit )
        newpage.data['count'] = newpage.data['list'].size
      end

      def get_pages(payload, per_page)
        (payload.size.to_f / per_page.to_i).ceil
      end
    end
  end
end
```