Building a Rails Engine from Scratch

Lessons from Active Storage Dashboard and more

RubyConf Thailand - Feb, 1st 2026
Bangkok

Giovanni Panasiti

https://panasiti.me

giovanni@montedelgallo.com

@giovapanasiti

It’s 2026… Why a talk about engines?

Eight Lessons

Namespace Isolation	Scope everything
Engine Config	mattr_accessor with smart defaults
The Auth Problem	Share auth with the host app
Multi-DB Queries	SQLite, MySQL, PostgreSQL, Oracle - all of them
Version Compatibility	Rails 5.2 through 8.0 in one codebase
Zero-Dependencies	No npm. No bundler conflicts.
Service Classes	Rake tasks that actually fix things
Filter Classes	Testable, composable query objects

Namespace Isolation

Scope everything to your namespace

```ruby
# lib/active_storage_dashboard/engine.rb

class Engine < ::Rails::Engine
  isolate_namespace ActiveStorageDashboard

  ...

end
```

- ✓ **Model Table Name Prefixing**
- ✓ **Route Namespace Encapsulation**
- ✓ **Helper Method Scoping**
- ✓ **Controller Inheritance Chain**
- ✓ **Asset Pipeline Namespacing**

Engine Configuration

mattr_accessor with smart defaults

```ruby
module ActiveStorageDashboard
  mattr_accessor :base_controller_class,
    default: "::ApplicationController"

  def self.base_controller
    base_controller_class.constantize
  end
end
```

- ✓ Store class names as strings, not constants
- ✓ Use .constantize at runtime, not load time
- ✓ Provide sensible defaults that just work

Engine Configuration

mattr_accessor with smart defaults

```ruby
module EmailClient
  class Configuration
    attr_accessor :admin_authentication_method,
                  :admin_user_class,
                  :items_per_page,
                  :enable_downloads,
                  :storage_service

    def initialize
      @admin_authentication_method = :authenticate_admin!
      # ...
    end
  end

  class << self
    def configuration
      @configuration ||= Configuration.new
    end

    def configure
      yield(configuration)
    end
  end
end
```

```ruby
EmailClient.configure do |config|
  config.admin_authentication_method = :authenticate_user!
  config.admin_user_class = "AdminUser"
  config.items_per_page = 50
  config.enable_downloads = false
end
```

The Authentication Problem

Share authentication with the host application

```ruby
# config/routes.rb
authenticate :user, -> (user) { user.admin? } do
  mount ActiveStorageDashboard::Engine, at: "/active-storage-dashboard"
end

# or with devise:
constraints lambda { |req| req.session[:user_id].present? || (req.env['warden'] &&
req.env['warden'].user(:user)) } do
  mount ActiveStorageDashboard::Engine, at: "/active-storage-dashboard"
end

```

The Authentication Problem

Share authentication with the host application

```ruby
# app/controllers/active_storage_dashboard/application_controller.rb
class ApplicationController < ActiveStorageDashboard.base_controller_class.constantize
  protect_from_forgery with: :exception

  ...

 end
```

```ruby
# config/initializer/active_storage_dashboard.rb
Rails.application.configure do
  config.active_storage_dashboard.base_controller_class = "AdminController"
end
```

Multi-DB Queries

support SQLite, MySQL, PostgreSQL and Oralce

```ruby
def json_extract(column, path)
  adapter = ActiveRecord::Base.connection.adapter_name.downcase

  case adapter
  when /sqlite/   then "json_extract(#{column}, '$.#{path}')"
  when /mysql/    then "JSON_UNQUOTE(JSON_EXTRACT(#{column}, '$.#{path}'))"
  when /postgres/ then "#{column}->>'#{path}'"
  when /oracle/   then "JSON_VALUE(#{column}, '$.#{path}')"
  end
end

# Usage
User.select("id, name, #{json_extract('metadata', 'locale')} AS user_locale")
    .where("#{json_extract('metadata', 'locale')} = ?", 'it')
```

Version Compatibility

Rails 5.2 through 8.0 in one codebase

```ruby
if blob.respond_to?(:variant_records)
  # Rails 7+ API
  blob.variant_records
else
  # Rails 6 fallback
  []
end
```

```ruby
if defined?(ActiveStorage::VariantRecord)
  # Include variant stats
  include_variants: true
end
```

Zero-Dependencies

No npm. No bundler conflicts. No gems.

```
source 'https://rubygems.org'
gemspec

gem 'rails'
```

Service Classes

Rake tasks that actually fix things

```ruby
class OrphanCleanupService
  def call(dry_run: true)
    orphans = find_orphaned_blobs
    return orphans if dry_run
    orphans.find_each(&:purge_later)
  end
end
```

Testable

Unit test business logic in isolation

Reusable

Call from rake, controller, or console

Safe

dry_run default prevents accidents

Filter Classes

Testable, composable query objects

```ruby
class BlobFilter
  def initialize(scope, params
    @scope, @params = scope, p
  end

  def apply
    @scope
      .then { filter_by_conter
      .then { filter_by_filena
  end
end
```

```ruby
class Admin::BlobsController < ApplicationController
  def index
    @blobs = BlobFilter.new(
      ActiveStorage::Blob.all,
      filter_params
    ).apply.page(params[:page])
  end

  private

  def filter_params
    params.permit(:content_type, :filename)
  end
end
```

Design for the host app, not yours.

Thank you!

github.com/giovapanasiti/active_storage_dashboard

https://panasiti.me

giovanni@montedelgallo.com

@giovapanasiti