

RUBY ON RAILS PROJECT

Getting project up & running

Firstly, make sure that **mysql2** is installed on the machine.

To run the project run following commands in the project directory:

```
bundle install
rails s
```

Creating a database

In case there are issues with the database and the project isn't loading, in `~/config/database.yml` file you want to change the *database.yml* file.

```
rake db:create
```

Basically it's recommended to use different gems for each environment, you should also create three databases, each for **development**, **testing**, and **production** environment. You can configure them in your `~/config/database.yml` file.

```
default: &default
  adapter: mysql2
  host: localhost
  username: bunbun
  password: Admin123admin!
  encoding: utf8
  pool: 5
  # timeout: 5000

development:
  <<: *default
  database: db/database_blog

# Warning: The database defined as "test" will be erased and
# re-generated from your development database when you run "rake".
# Do not set this db to the same as development or production.
test:
  <<: *default
  database: db/database_blog

production:
  <<: *default
  database: db/database_blog
```

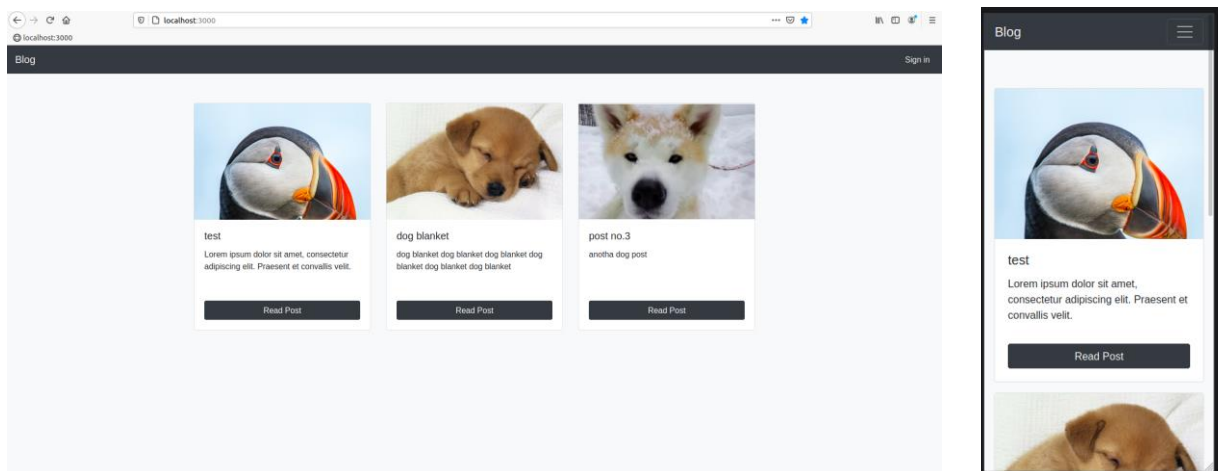
Migrating

Migrations setup the tables in the database. After that, you want to run the following command to get it up and running:

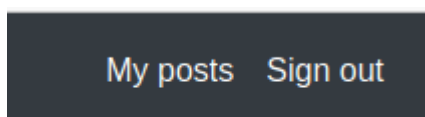
```
rake db:migrate
```

This should set our project up and it should be up and running.

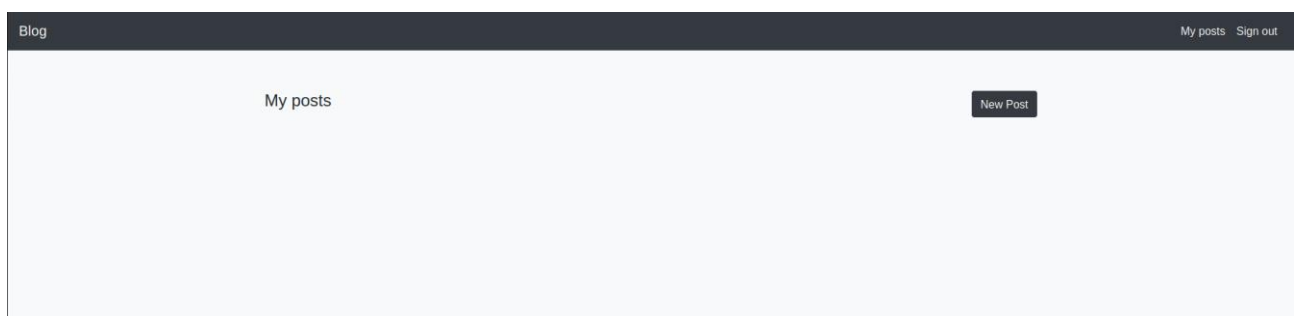
To test it, we can run *rails s* and head to <http://localhost:3000/> to see our project. For front-end styling we used bootstrap.



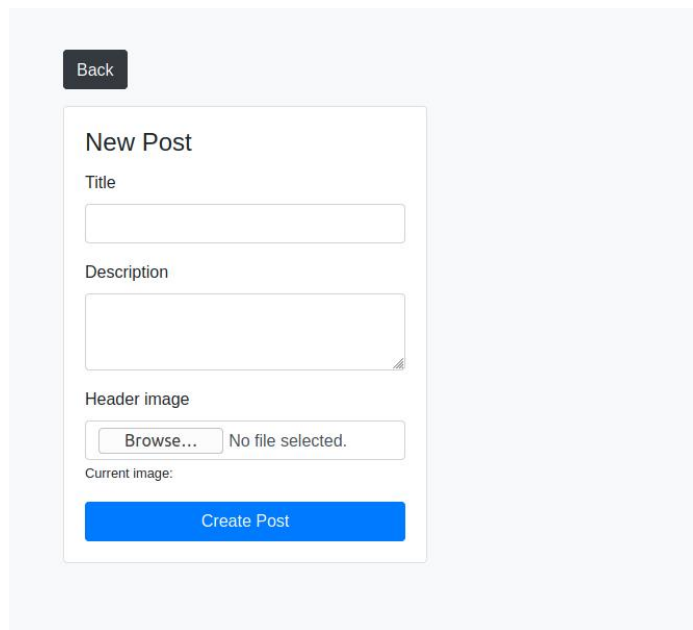
User doesn't need to be logged in to read the posts, but will need to create / log into an account if he/she wants to create a post.



Once logged in, user will see two buttons at the top right corner. One is to log out of the account and another one is to see all the posts made by a user.

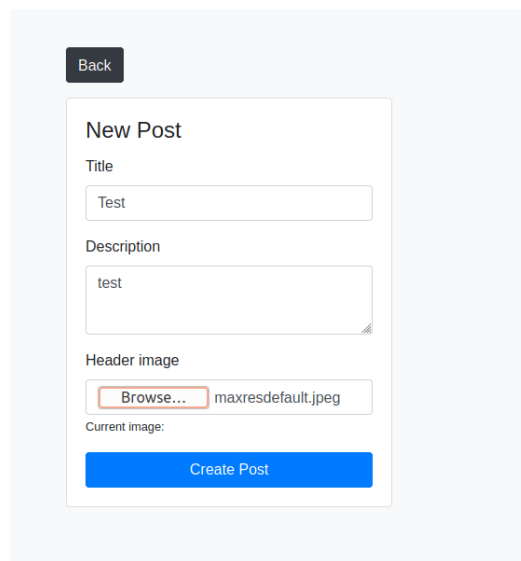


In this case we do not have any posts made with this user. So to create one we just click on „New Post“.



The screenshot shows a web interface for creating a new post. At the top left is a dark button labeled 'Back'. Below it is a white form box titled 'New Post'. Inside the form, there are three input fields: 'Title' (a single-line text box), 'Description' (a multi-line text area), and 'Header image' (a file upload area). The 'Header image' section includes a 'Browse...' button and the text 'No file selected.'. Below these fields is a label 'Current image:' followed by a large blue button labeled 'Create Post'.

This will bring the user into `/posts/new`` and a user can write down initial information about the post. User can give post a title, description and add a header image which will be displayed as a main image.



This screenshot shows the same 'New Post' form, but now it contains sample data. The 'Title' field has the text 'Test', and the 'Description' field has the text 'test'. In the 'Header image' section, the 'Browse...' button is highlighted with a red border, and the text next to it is 'maxresdefault.jpeg'. The 'Current image:' label and the blue 'Create Post' button remain at the bottom of the form.

After writing some information about the post, once we click on „*Create Post*“ it will create a post in a database and it will bring user into post edit screen. And it will now be shown in the home screen.

Back

ParagraphImage

Editing Post

Title

Test


Description

test

Header image

Browse... No ...d.

Current image:




Update Post




Delete Post




Click to add content.


Browse... No file selected.



Save Cancel

B I S 



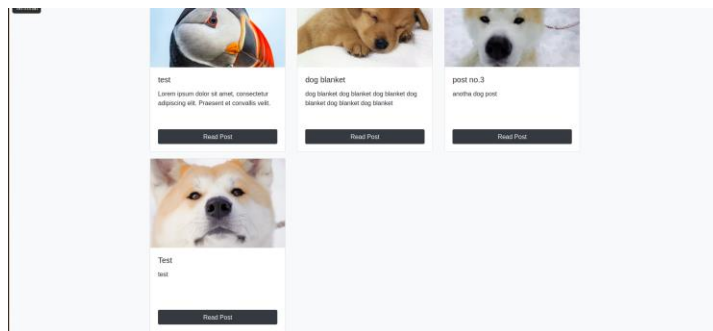
 

Save Cancel

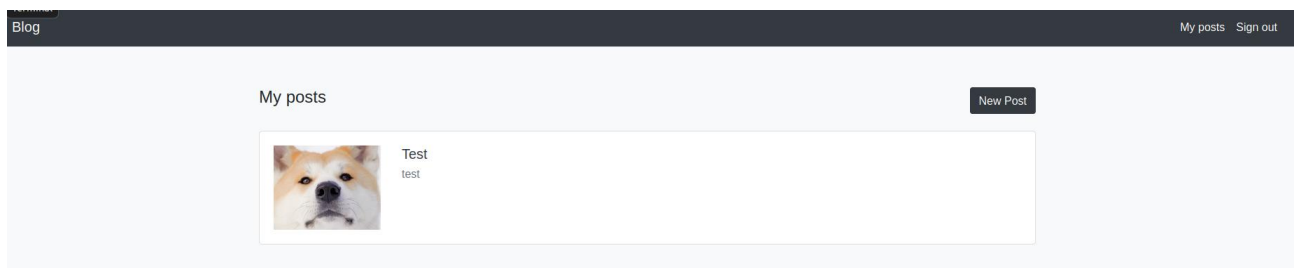
Delete

If we click on the „Update Post“ button we will save the post as it is.

This is how our home page looks like:



And since a user made a post, we can now go into `/posts/` and see all the user posts that have been made.



```
Inspecting 66 files
.....C.....

Offenses:

db/schema.rb:15:1: C: Metrics/BlockLength: Block has too many lines. [60/25]
ActiveRecord::Schema.define(version: 20_201_221_150_256) do ...
~~~~~

66 files inspected, 1 offense detected
```

When running the only offence we get is for schema. Schema is too big. But schema was generated so we did not refactor.

Posts & authors

Logged in user can have many posts assigned to them. Logged in user is an Author.

Author validation is handled by devise gem.

```
1 class Author < ApplicationRecord
2   # Include default devise modules. Others available are:
3   # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable
4   devise :database_authenticatable, :registerable,
5         :recoverable, :rememberable, :validatable
6
7   has_many :posts
8 end
```

Each user can see their own posts because, posts are linked to the authors (users) they belong to:

```
1 class Post < ApplicationRecord
2   belongs_to :author
3   has_many :elements
4
5   # published nil because they all are nil by default
6   # and there is no published handling
7   scope :published, -> do
8     where(published: nil)
9   end
10
11   has_one_attached :header_image
```

This is a html for the my posts page of logged in user.

```
1 <h4 class="d-flex justify-content-between mb-4">
2   <span>My posts</span>
3   <%= link_to 'New Post', new_post_path, class: "btn btn-dark"%>
4 </h4>
5
6 <%= @posts comes from the posts_controller.rb where
7 it gets the current author's posts %>
8 <% @posts.each do |post| %>
9   <%= link_to edit_post_path(post), class: "text-decoration-none" do %>
10     <div class="card mb-3 post-card">
11       <div class="card-body">
12         <div class="row">
13           <div class="col-md-2">
14             <% if post.header_image.present? %>
15               <div class="header-img" style="background-image: url(<%= url_for(post.header_image) %>);"></div>
16             <% end %>
17           </div>
18           <div class="col-md-10">
19             <h5 class="card-title mb-1 text-dark"><%= post.title %></h5>
20             <p class="text-secondary mb-0"><%= post.description %></p>
21           </div>
22         </div>
23       </div>
24     </div>
25   <% end %>
26 <% end %>
```

Posts are taken by the current signed in author

```
1 module Authors
2   class PostsController < AuthorsController
3     before_action :set_post, only: [:edit, :update, :destroy]
4
5     # GET /posts
6     def index
7       @posts = current_author.posts
8     end
9   end
10 end
```