

BLOG PROJECT

Sin embargo, podemos corregir algunas malas prácticas en nuestro código.

PARTAMOS CON GIT COMMIT

BLOG PROJECT

Cuando trabajamos en nuestra vista debemos evitar crear las consultas en el html, para corregir esto haremos lo siguiente.

BLOG PROJECT

posts_controller.rb

```
def show
  @comments = @post.comments
end
```

Agregamos la línea remarcada en el método show del controlador de posts.

show.html.erb

```
</ul>
  <% Comment.where(post_id: @post.id).each do |comment| %>
    <li><%= comment.content %></li>
  <% end %>
<ul>
```

Reemplazamos este código

```
</ul>
  <% @comments.each do |comment| %>
    <li><%= comment.content %></li>
  <% end %>
<ul>
```

Por este

BLOG PROJECT

Otro punto importante que debemos corregir de nuestro código es el siguiente.

BLOG PROJECT

Si nos fijamos en este código, tenemos 2 malas prácticas

```
<%= form_with(model: Comment.new, local: true) do |f| %>  
  <%= hidden_field_tag :post_id, @post.id %>  
  <%= f.text_area :content %>  
  <%= f.submit :enviar %>  
<% end %>
```

BLOG PROJECT

Si nos fijamos en este código, tenemos 2 malas prácticas


Podemos entregar este dato mediante una variable desde el controller de post

```
<% form_with(model: Comment.new, local: true) do |f| %>
  <%= hidden_field_tag :post_id, @post.id %>
  <%= f.text_area :content %>
  <%= f.submit :enviar %>
<% end %>
```

BLOG PROJECT

Si nos fijamos en este código, tenemos 2 malas prácticas

Este dato se está enviando desde el formulario, y podría ser modificada fácilmente por el usuario con el inspector de elementos.

A yellow arrow points from the left towards the second line of the code block, specifically highlighting the `<%= hidden_field_tag :post_id, @post.id %>` line.

```
<%= form_with(model: Comment.new, local: true) do |f| %>  
  <%= hidden_field_tag :post_id, @post.id %>  
  <%= f.text_area :content %>  
  <%= f.submit :enviar %>  
<% end %>
```


BLOG PROJECT

Para solucionar este problema, existe el método
build

BLOG PROJECT

El método **build** permite crear un objeto a partir de otro.

```
@post = Post.find[:post_id]  
@comment = @post.comments.build
```

BLOG PROJECT

De esta forma, un objeto nuevo ya pertenece a otro, por lo que ya no es necesario pasarle el id, ya que lo hace internamente.

```
@post = Post.find[:post_id]  
@comment = @post.comments.build
```

BLOG PROJECT

En primer lugar, debemos agregar el siguiente código en el método show del controlador de posts.

```
@comment = @post.comments.build
```