RENO COLLECTIVE

FULL SPEED

N V

AHEAD

ACADEMY

Looking forward

Pinterest

Twitter

Slack Bot

Assessments

Goals

Adding more models

Generate model for adding comments

```
$ bin/rails generate model Comment
commenter:string body:text article:references
```

Generated model for Comment

```
class Comment < ActiveRecord::Base
  belongs_to :article
end
```

Our generated migration

```ruby
class CreateComments < ActiveRecord::Migration
  def change
    create_table :comments do |t|
      t.string :commenter
      t.text :body
      t.references :article, index: true,
        foreign_key: true

      t.timestamps null: false
    end
  end
end
```

Run the db migration

```
$ bin/rake db:migrate
```

```
==  CreateComments: migrating
=====================================
-- create_table(:comments)
   -> 0.0115s
==  CreateComments: migrated (0.0119s)
=====================================
```

has_many

```
class Article < ActiveRecord::Base
  has_many :comments

  # other code from previous steps
end
```

Add routes for Comments

```
$ rake routes
```

Create a controller

```
$ bin/rails generate controller Comments
```

Form for adding a comment

```
<h2>Add a comment:</h2>
<%= form_for([@article, @article.comments.build])
do |f| %>
  <p>
    <%= f.label :commenter %><br>
    <%= f.text_field :commenter %>
  </p>
  <p>
    <%= f.label :body %><br>
    <%= f.text_area :body %>
  </p>
  <p>
    <%= f.submit %>
  </p>
<% end %>
```

Allow a user to create a comment

```
class CommentsController < ApplicationController
  def create
    @article = Article.find(params[:article_id])
    @comment = @article.comments.create(comment_params)
    redirect_to article_path(@article)
  end

  private
    def comment_params
      params.require(:comment).permit(:commenter, :body)
    end
end
```

Display our comments

```
<h2>Comments</h2>
<% @article.comments.each do |comment| %>
  <p>
    <strong>Commenter:</strong>
    <%= comment.commenter %>
  </p>

  <p>
    <strong>Comment:</strong>
    <%= comment.body %>
  </p>
<% end %>
```

Display our comments

```
<h2>Comments</h2>
<% @article.comments.each do |comment| %>
  <p>
    <strong>Commenter:</strong>
    <%= comment.commenter %>
  </p>

  <p>
    <strong>Comment:</strong>
    <%= comment.body %>
  </p>
<% end %>
```

Adding Bootstrap

```
bower init
```

Create a .bowerrc file in your project root

```
atom .bowerrc
```

```
{
  "directory":"vendor/assets/bower_components"
}
```

Install Bootstrap with Bower

```
bower install bootstrap
```

Asset Pipeline

/app/assets/stylesheets/application.css

```
/*
 *= require bootstrap/dist/css/bootstrap
 *= require_tree .
 *= require_self
*/
```

/app/assets/javascripts/application.js

```
//= require jquery
//= require jquery_ujs
//= require bootstrap/dist/js/bootstrap
//= require turbolinks
//= require_tree .
```

Run your app and check to see if Bootstrap is being included

DIY Project

DIY Project

- Make a new Rails app
- Come up with a concept that has two related models
- Create the models, controllers and views necessary to create, edit, delete and view those items
- Add Bootstrap