UIT2602 WEB PROGRAMMING

Exercise 6 - Implement functionality to allow users to upload files

Name: S. Selcia
Reg. No.: 3122215002098
Section: IT B

UPLOAD FILES

1. **Aim:**

   This documentation outlines the implementation of file upload functionality in a Ruby on Rails application. The goal of this functionality is to allow users to upload files, such as images, documents, etc., and store them in the application.

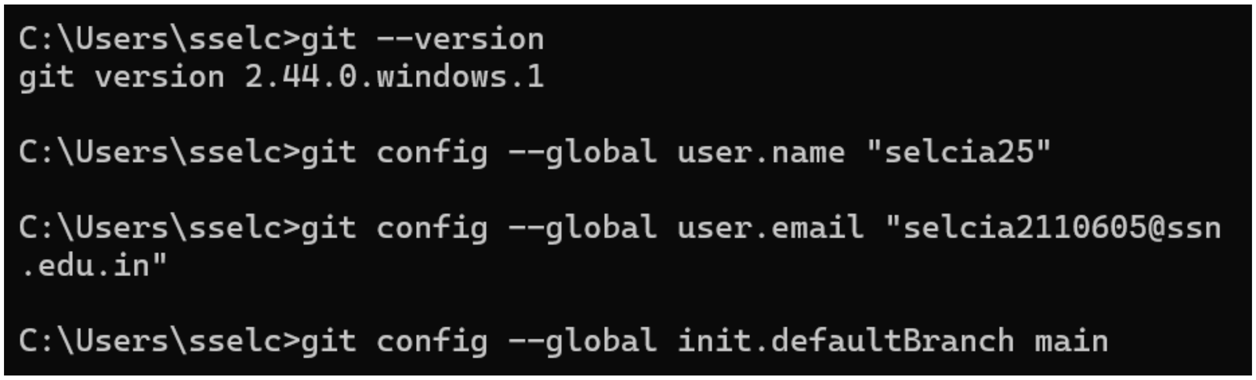
2. **Required Web Tools and Methodology:**

   - Ruby on Rails: The web framework will be used to develop the application.
   - SQLite for database
   - HTML, CSS, Javascript for frontend
   - Git for version control
   - Web Browser: To test the functionality of the website.

3. **Implementation Procedure:**

BEFORE SETTING UP RAILS - MAKE SURE YOU INSTALL GIT

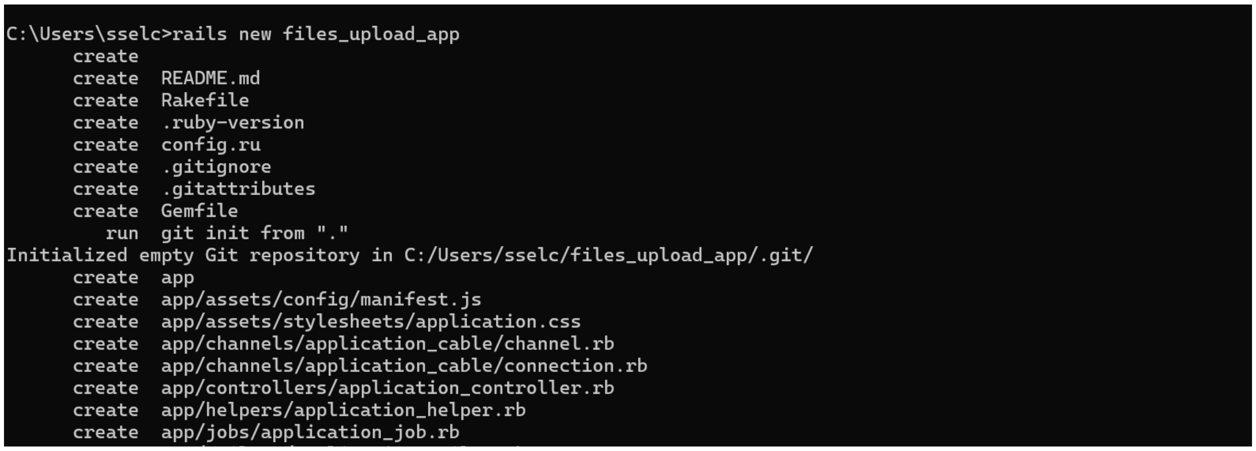
❖ Download Git Version for Windows from the official website and follow the commands given below.



```
C:\Users\sselc>git --version
git version 2.44.0.windows.1

C:\Users\sselc>git config --global user.name "selcia25"

C:\Users\sselc>git config --global user.email "selcia2110605@ssn
.edu.in"

C:\Users\sselc>git config --global init.defaultBranch main
```

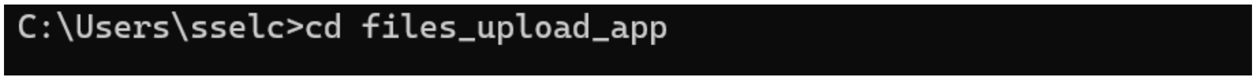
SETTING UP THE RAILS APPLICATION:

❖ The `*rails new*` command is used to create a new Ruby on Rails application



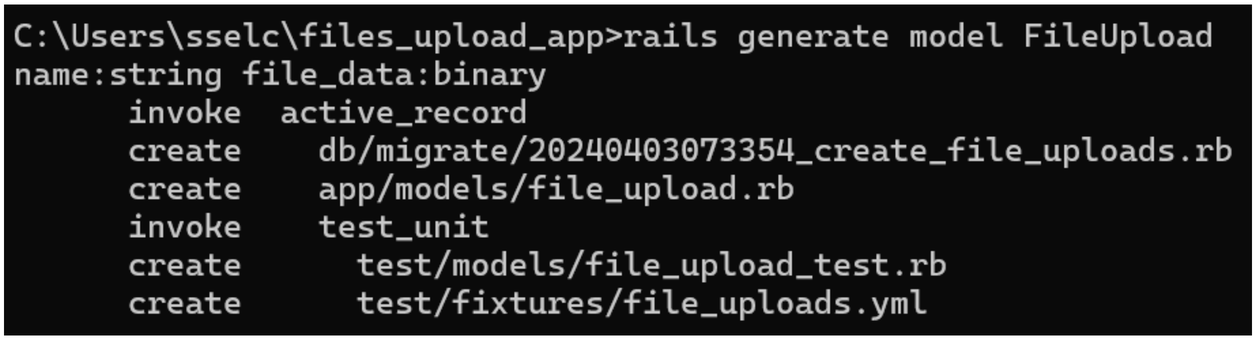
```
C:\Users\sselc>rails new files_upload_app
      create
      create  README.md
      create  Rakefile
      create  .ruby-version
      create  config.ru
      create  .gitignore
      create  .gitattributes
      create  Gemfile
         run  git init from "."
Initialized empty Git repository in C:/Users/sselc/files_upload_app/.git/
      create  app
      create  app/assets/config/manifest.js
      create  app/assets/stylesheets/application.css
      create  app/channels/application_cable/channel.rb
      create  app/channels/application_cable/connection.rb
      create  app/controllers/application_controller.rb
      create  app/helpers/application_helper.rb
      create  app/jobs/application_job.rb
```

❖ Change into the application directory



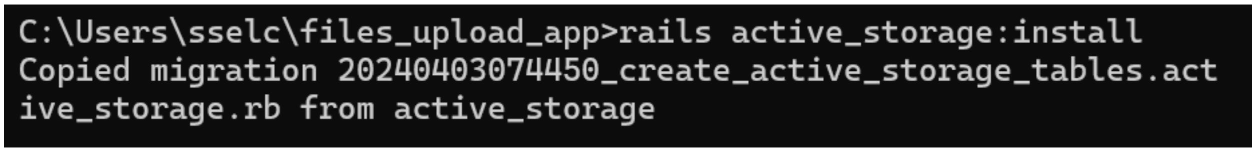
```
C:\Users\sselc>cd files_upload_app
```

❖ The `*rails generate model*` command generates a new model called FileUpload with attributes name (string) and file_data (binary) using the following command



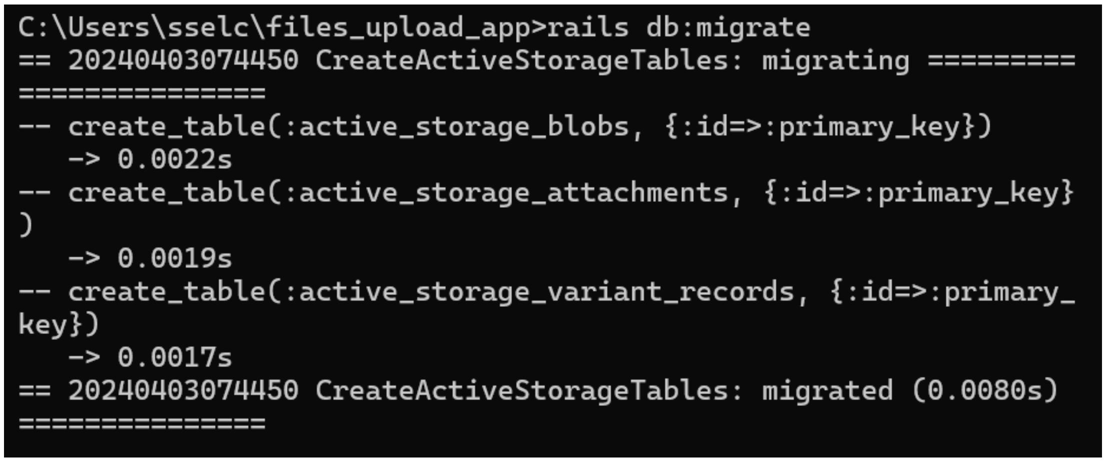
```
C:\Users\sselc\files_upload_app>rails generate model FileUpload
name:string file_data:binary
      invoke  active_record
      create    db/migrate/20240403073354_create_file_uploads.rb
      create    app/models/file_upload.rb
      invoke    test_unit
      create      test/models/file_upload_test.rb
      create      test/fixtures/file_uploads.yml
```

❖ Install and configure active storage if it is not already done



```
C:\Users\sselc\files_upload_app>rails active_storage:install
Copied migration 20240403074450_create_active_storage_tables.act
ive_storage.rb from active_storage
```

❖ The `*rails db:migrate*` runs pending database migrations to update the database schema.



```
C:\Users\sselc\files_upload_app>rails db:migrate
== 20240403074450 CreateActiveStorageTables: migrating =========
===============
-- create_table(:active_storage_blobs, {:id=>:primary_key})
   -> 0.0022s
-- create_table(:active_storage_attachments, {:id=>:primary_key}
)
   -> 0.0019s
-- create_table(:active_storage_variant_records, {:id=>:primary_
key})
   -> 0.0017s
== 20240403074450 CreateActiveStorageTables: migrated (0.0080s)
==============
```

SETTING UP THE FILE UPLOAD FORM

- Go to `app\views` and create a new folder named file_uploads - create new file `app\views\new.html.erb` and `app\views\show.html.erb`

app\views\file_uploads\new.html.erb

```
<%= form_with(model: @file_upload, url: file_uploads_path, multipart: true) do |form| %>
    <div class="field">
      <%= form.label :name %>
      <%= form.text_field :name %>
    </div>
    <div class="field">
      <%= form.label :file_data, 'Upload File' %>
      <%= form.file_field :file_data %>
    </div>
    <div class="actions">
      <%= form.submit %>
    </div>
  <% end %>
```

app\views\file_uploads\show.html.erb

```
<h1>File Details</h1>
<p>This is the file that you uploaded!</p>
<p><strong>Name:</strong> <%= @file_upload.name %></p>
<%= image_tag @file_upload.file_data if @file_upload.file_data.attached? %>
```

IMPLEMENT THE FILE UPLOAD CONTROLLER

- Go to `app\controllers` and create a new file `app\controllers\file_uploads_controller.rb`

app\controllers\file_uploads_controller.rb

```
class FileUploadsController < ApplicationController
    def new
      @file_upload = FileUpload.new
    end
    def create
      @file_upload = FileUpload.new(file_upload_params)
      if @file_upload.save
        redirect_to @file_upload, notice: 'File was successfully uploaded.'
      else
        render :new
      end
    end
    def show
      @file_upload = FileUpload.find(params[:id])
    end
    private
    def file_upload_params
      params.require(:file_upload).permit(:name, :file_data)
    end
  end
```

IMPLEMENT MODEL THE FILE UPLOAD CONTROLLER

- ❖ Go to **`app\models`** and create a new file **`app\models\file_upload.rb`**

<u>**`app\models\file_upload.rb`**</u>

```
class FileUpload < ApplicationRecord
    has_one_attached :file_data
end
```

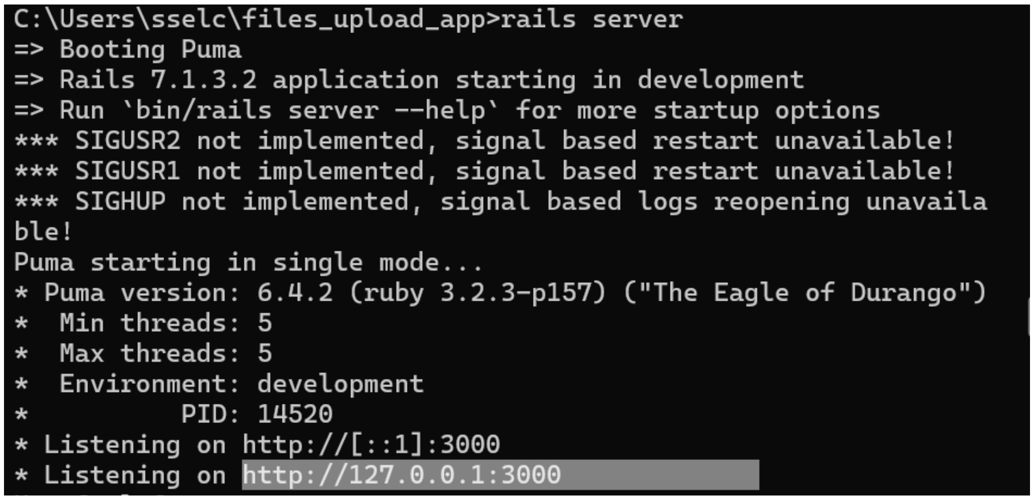
- ❖ Go to **`config`** and modify file **`config\routes.rb`**

<u>**`config\routes.rb`**</u>

```
Rails.application.routes.draw do
get "up" => "rails/health#show", as: :rails_health_check
  resources :file_uploads, only: [:new, :create, :show]
end
```

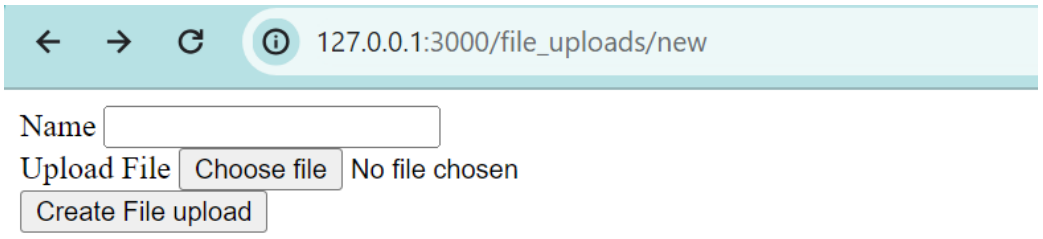
RUNNING THE APPLICATION

- ❖ The `*rails server*` starts the Rails development server, allowing you to run your Rails application locally. It serves your application so that you can access it through a web browser at `*http://127.0.0.1:3000*`.



```
C:\Users\sselc\files_upload_app>rails server
=> Booting Puma
=> Rails 7.1.3.2 application starting in development
=> Run `bin/rails server --help` for more startup options
*** SIGUSR2 not implemented, signal based restart unavailable!
*** SIGUSR1 not implemented, signal based restart unavailable!
*** SIGHUP not implemented, signal based logs reopening unavaila
ble!
Puma starting in single mode...
* Puma version: 6.4.2 (ruby 3.2.3-p157) ("The Eagle of Durango")
*  Min threads: 5
*  Max threads: 5
*  Environment: development
*          PID: 14520
* Listening on http://[::1]:3000
* Listening on http://127.0.0.1:3000
```

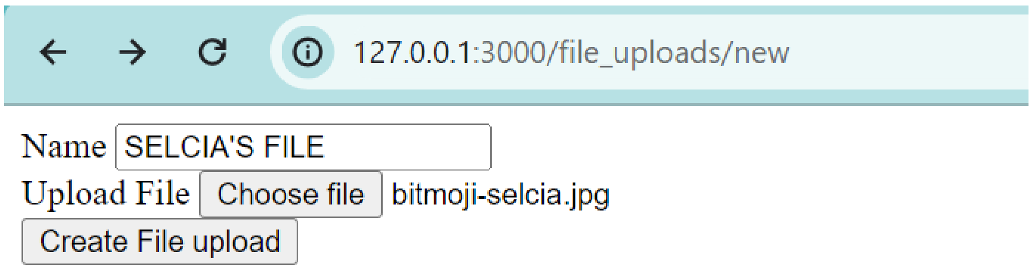
- Once we enter the browser with *http://127.0.0.1:3000/file_uploads/new*, we get this


127.0.0.1:3000/file_uploads/new

Name

Upload File No file chosen

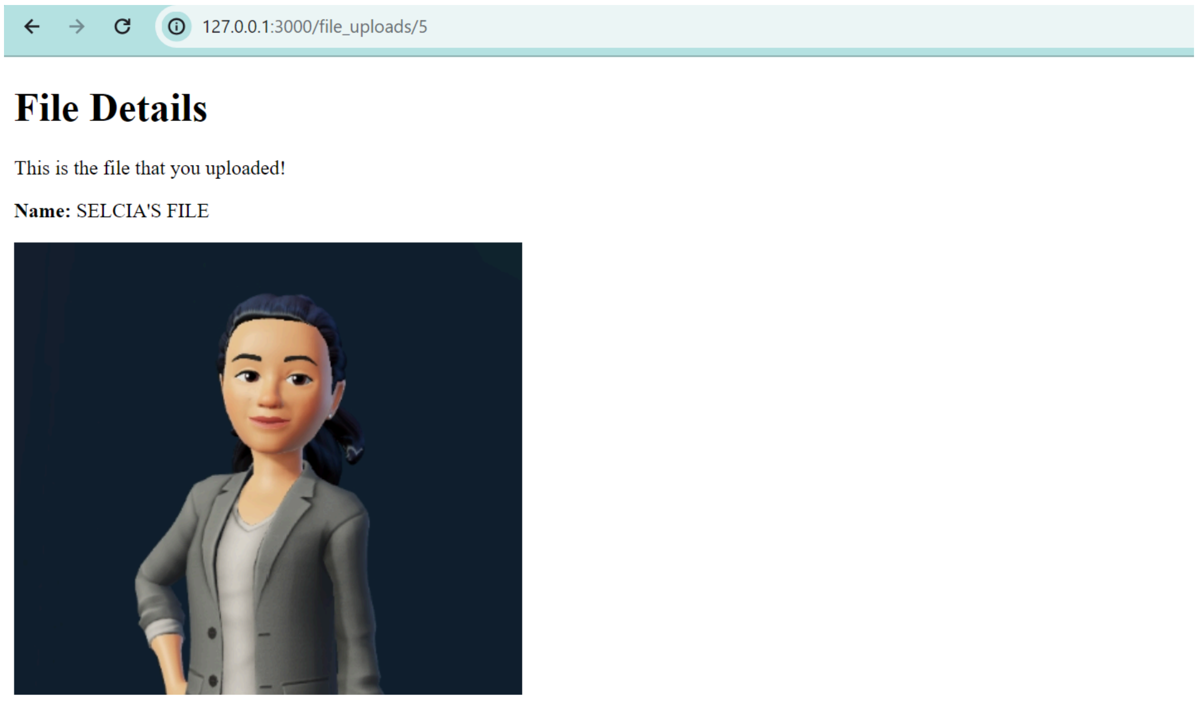
- Allows users to UPLOAD FILES


127.0.0.1:3000/file_uploads/new

Name SELCIA'S FILE

Upload File bitmoji-selcia.jpg

- OUTPUT:


127.0.0.1:3000/file_uploads/5

File Details

This is the file that you uploaded!

Name: SELCIA'S FILE

Conclusion:

The file upload functionality has been successfully implemented in the Rails application. Users can now upload files, which are stored in the database and can be displayed using the show action of the FileUploadsController.