

Gallera

CST-452 Capstone Project

Francisco Gonzalez

Milestone 6

Amr Elchouemi

11/30/2025

**Github link:** <https://github.com/dankiee0916/gallerycreator>

**Video Link:** [https://youtu.be/s0eSRpWx\\_oQ](https://youtu.be/s0eSRpWx_oQ)

1. Requirements Outcome Table

Requirement	Outcome	Justification
R1: Users can register and log in	Met	Spring Security is fully working and users can register, login, and stay authenticated across pages.
R2: Users can create a gallery	Met	GalleryController + GalleryService let logged-in users create their own galleries. Tested by creating multiple galleries.
R3: Users can edit and delete their own galleries	Met	Ownership checks prevent users from editing galleries that aren't theirs. Editing/deleting works without errors.
R4: Users can upload photos	Met	Photo uploading works, saves to /uploads/, and displays the image correctly.
R5: Users can edit and delete their own photos	Met	Photo edit page and delete functionality work and enforce ownership.
R6: Guests can view all galleries	Met	SecurityConfig allows public access to /galleries/all.

Requirement	Outcome	Justification
R7: Users can comment on photos	Met	Comment model + CommentController work. Comments show usernames and users can delete their own comments.
R8: Users can delete comments if they own them OR if they own the gallery	Met	Ownership logic implemented and tested.
R9: Preview images show correctly on galleries page	Met	Preview URLs tested and fixed.
R10: System is deployed online	Met	Fully deployed on Render with Neon PostgreSQL.

## 2. Implementation Plan

For this milestone, the main goal was to finish the important features of the Gallery Creator app and make everything stable. Most of the core work was already in progress, so this iteration focused on polishing the functionality, improving database performance, and making sure authentication, galleries, photos, and comments all worked smoothly.

Here is the updated summary of the features completed during this milestone:

User Story / Feature	Development Tasks	Estimate (hrs)	Actual (hrs)	% Complete

User registration & login with Spring Security	Set up authentication, password hashing, DB connection	5	4	100%
Gallery creation & editing	CRUD operations, link gallery to logged-in user	4	3.5	100%
Photo upload & display	Upload image files, attach to gallery, render thumbnails	5	4.5	100%
Database migration to Neon PostgreSQL	Update properties, connect to new DB	3	3	100%
Deployment to Render	Build Dockerfile, deploy working version	4	4	100%
Comments feature (new)	Add commenting system for logged-in users	4	3.5	100%
Security refinements	Add @Transactional, secure routes	2	2	100%

**Iteration Completion:** 100%

**Overall Project Completion:** About 90% (final polish and enhancements left for the end)

### 3. Functional Requirements Mapping (Traceability Matrix)

This table links each functional requirement to the actual implementation in the code and where it was tested.

<b>Requirement / User Story</b>	<b>Page / Feature</b>	<b>Code Module / File</b>	<b>Test Case ID</b>
Users can register an account	Register page	UserController.registerUser()	TC-1
Users can log in	Login page	SecurityConfig, UserDetailsServiceImpl	TC-2
Logged-in users can create galleries	Create Gallery page	GalleryController.createGallery()	TC-3
Users can edit or delete their galleries	Edit Gallery page	GalleryService.deleteGallery()	TC-4
Users can upload photos	Upload Photo page	PhotoController.addPhoto()	TC-5
Photos display under a gallery	Gallery Detail page	PhotoService.getPhotosByGallery()	TC-6

<b>Requirement / User Story</b>	<b>Page / Feature</b>	<b>Code Module / File</b>	<b>Test Case ID</b>
Users can comment on photos	Photo Detail page	CommentController.addComment()	TC-7
Restricted routes require login	All pages	SecurityConfig.securityFilterChain()	TC-8

#### 4. Test Plan and Module Test Cases

<b>Test Case ID</b>	<b>Module / Feature</b>	<b>Objective</b>	<b>Test Steps</b>	<b>Expected Result</b>	<b>Status</b>
TC-1	User Registration	Make sure users can register and save to the DB	Fill out form > submit	Redirect to login page, user added	Pass
TC-2	Login	Check that valid credentials work	Enter login info > submit	Redirect to gallery list	Pass

<b>Test Case ID</b>	<b>Module / Feature</b>	<b>Objective</b>	<b>Test Steps</b>	<b>Expected Result</b>	<b>Status</b>
TC-3	Gallery Creation	Confirm new gallery saves	Fill form > submit	Gallery appears in list	Pass
TC-4	Gallery Deletion	Make sure deletion works	Click delete	Gallery removed	Pass
TC-5	Photo Upload	Test uploading a photo	Choose file > submit	Photo shows under gallery	Pass
TC-6	Comments	Ensure comment posting works	Type comment > submit	Comment shows with username	Pass
TC-7	Security	Check access control	Try opening private page while logged out	Redirect to login	Pass

## 5. Testing Overview

During this milestone, testing was done through a mix of manual and integrated tests:

- Unit-level testing:

Tested individual controller actions by submitting forms and checking database writes.

- Integration testing:

Made sure the full flow works together (register > login > create gallery > upload photo > comment).

- System testing:

Verified everything works on the live Render deployment using the Neon database.

## 6. Source Code Listing (Summary)

Here's a short breakdown of the major components in the project and what each file is responsible for:

Component / File	Purpose / Description
GalleryController.java	Handles creating, editing, deleting, and viewing galleries.
PhotoController.java	Manages uploading photos, showing them, and deleting



Component / File	Purpose / Description
	them.
CommentController.java	Handles adding and removing comments.
UserController.java	Handles registration, login, and logout.
SecurityConfig.java	Sets up Spring Security, login routes, and password encoding.
UserDetailsServiceImpl.java	Loads user login info from the database.
GalleryService.java	Core logic for creating and managing galleries.
PhotoService.java	Logic for saving and retrieving photos.
UserService.java	Handles user creation and credential validation.
application.properties	Database settings for Neon PostgreSQL and JPA.
fragments.html	Shared UI layout for head and navbar.
login.html, register.html, gallery.html, etc.	All the main views built with Thymeleaf.

## 7. Future Enhancements

1. Add user profile pages with customization.

2. Create public shareable gallery links.
3. Add gallery search and filtering tools.
4. Add basic photo editing options (crop, rotate, effects).