

# Photo Gallery App

Nume: Mic Mihai Valentin

Grupa: 30236

Cuprins

[Photo Gallery App 1](#_Toc131068144)

[Cuprins 2](#_Toc131068145)

[Deliverable 1 3](#_Toc131068146)

[1. Specificatia proiectului 3](#_Toc131068147)

[2. Cerinte functionale 3](#_Toc131068148)

[3. Cazuri de utilizare (usecase) 4](#_Toc131068149)

[4. Cerinte non-functionale 7](#_Toc131068150)

[5. Design Constraints 8](#_Toc131068151)

[6. Glossary 8](#_Toc131068152)

# Deliverable 1

## **1. Specificatia proiectului**

Acest proiect reprezinta o aplicație web pentru o galerie foto care permite utilizatorilor să încarce, gestioneze și împărtășească propriile galerii foto. Aplicația include operații CRUD pentru fotografii și albume, precum și funcționalități adiționale, cum ar fi etichetarea, comentarea și căutarea fotografiilor. Aplicația este construită în .NET, folosind SQL Server și Entity Framework.

## **2. Cerinte functionale**

Aplicația este concepută pentru a gestiona trei tipuri de utilizatori: administrator, fotograf și client. Acești actori au atât caracteristici comune, precum nume, prenume, adresă de e-mail, parolă, cât și elemente specifice: fotograful are un portofoliu de lucrări și poate adăuga etichete fotografiilor, în timp ce clientul poate salva albume și pot adăuga comentarii la fotografiile care îi plac.

Fiecare tip de utilizator dispune de funcționalități diferite:

• **Administrator:** poate gestiona baza de date și efectua diverse interogări, precum: să adauge utilizatori de orice tip în baza de date, să șteargă utilizatori, să adauge albume, să șteargă albume sau să actualizeze informațiile existente.

• **Fotograf:** poate să-și creeze un cont, să își actualizeze detaliile contului, să încarce fotografiile în portofoliul său, să adauge etichete și să șteargă fotografiile sale. Fotograful poate, de asemenea, să vadă comentariile adăugate de clienți la fotografiile sale și să răspundă la acestea.

• **Client:** poate să-și creeze un cont, să-și actualizeze diferite date, să caute fotografi după diverse criterii, precum genul de fotografie sau etichete, să creeze albume, să adauge fotografii în albume, să adauge comentarii la fotografiile care îi plac și să șteargă albumele și comentariile sale.

Fiecare acțiune menționată mai sus este însoțită de validări și verificări pentru a se asigura că aceasta nu poate fi realizată dacă nu îndeplinește anumite criterii.

La adăugarea unui utilizator, se verifică coerența datelor: câmpurile de nume, prenume, adresă de e-mail și parolă trebuie să fie nenule, iar tipul de utilizator să fie unul dintre cele trei valide. Dacă din această perspectivă este totul în regulă, se face o verificare în baza de date pentru a ne asigura că nu există un alt utilizator cu aceeași adresă de e-mail. La finalizarea cu succes a verificărilor, utilizatorul este adăugat atât în tabelul de utilizatori, cât și în tabelele aferente tipului de utilizator.

În cazul în care o acțiune nu este validată cu succes, se vor arunca excepții specifice:

**InvalidUserException, InvalidPhotographerException, InvalidClientException,**

**InvalidAlbumException.**

La actualizarea unui utilizator, se aplică din nou aceleași cerințe menționate mai sus, astfel încât să ne asigurăm că nu există actualizări inconsistente.

La ștergerea unui utilizator, dacă acesta este de tipul fotograf sau client, se vor șterge și toate albumele și comentariile aferente lui.

Interogările care au ca scop găsirea unui utilizator sau a mai multora conform diferitor criterii vor arunca excepții specifice în cazul în care nu este niciun utilizator care să îndeplinească cerințele:

**UserNotFoundException,PhotographerNotFoundException, ClientNotFoundException, AlbumNotFoundException.**

O entitate importantă a proiectului este Albumul. Acesta este alcătuit dintr-un client, un fotograf, o listă de fotografii și o descriere. Clientul poate adăuga comentarii la fotografiile care îi plac și poate răspunde la comentariile altor clienți. Pentru a salva în baza de date un album, asupra acestuia se fac multiple verificări: clientul și fotograful să fie valizi și să fie regăsiți în tabelele aferente celor două tipuri de entități și în tabelul care conține toți utilizatorii, descrierea să nu fie nulă. Dacă aceste cerințe sunt satisfăcute, albumul este adăugat în baza de date și în listele de albume ale celor două entități.

Pentru a adăuga fotografii într-un album, fotograful trebuie să se asigure că fotografiile sunt într-un format acceptat (de exemplu, JPEG, PNG) și că dimensiunea fișierului nu depășește limita stabilită. Fotograful poate, de asemenea, să adauge etichete la fotografiile sale pentru a facilita căutarea de către clienți.

La ștergerea unui album, se vor șterge și toate fotografiile și comentariile asociate cu acel album. Dacă un client șterge un album, acesta va fi șters doar din lista sa de albume și nu va afecta portofoliul fotografului.

În cazul în care un utilizator dorește să își schimbe parola, acesta va trebui să furnizeze parola actuală și să respecte cerințele de complexitate ale parolei pentru a se asigura că noua parolă este sigură.

## **3. Cazuri de utilizare (usecase)**

Actori:

* Client
* Fotograf
* Administrator

**Use Case-uri:**

1. **Încărcare fotografie:**
   * Nivel: Subfuncție
   * Actor principal: Fotograf
   * Scenariu de succes: Fotografia este încărcată cu succes în galerie de către fotograf.
2. **Ștergere fotografie:**
   * Nivel: Subfuncție
   * Actor principal: Fotograf, Administrator
   * Scenariu de succes: Fotografia selectată este ștearsă cu succes din galerie de către fotograf sau administrator.
3. **Editare fotografie:**
   * Nivel: Subfuncție
   * Actor principal: Fotograf
   * Scenariu de succes: Fotografia selectată este editată cu succes și modificările sunt salvate de către fotograf.
4. **Vizualizare galerie:**
   * Nivel: Subfuncție
   * Actor principal: Client, Fotograf, Administrator
   * Scenariu de succes: Utilizatorul (client, fotograf sau administrator) vizualizează galeria cu fotografiile încărcate.
5. **Căutare fotografie:**
   * Nivel: Subfuncție
   * Actor principal: Client, Fotograf, Administrator
   * Scenariu de succes: Fotografiile care corespund criteriilor de căutare sunt afișate pentru utilizator (client, fotograf sau administrator).
6. **Creare album:**
   * Nivel: Subfuncție
   * Actor principal: Fotograf,Client
   * Scenariu de succes: Un nou album este creat cu succes de către fotograf.
7. **Ștergere album:**
   * Nivel: Subfuncție
   * Actor principal: Fotograf, Administrator
   * Scenariu de succes: Albumul selectat este șters cu succes, împreună cu fotografiile din el, de către fotograf sau administrator.
8. **Redenumire album:**
   * Nivel: Subfuncție
   * Actor principal: Fotograf,Client
   * Scenariu de succes: Albumul selectat este redenumit cu succes de către fotograf.
9. **Adăugare fotografie în album:**
   * Nivel: Subfuncție
   * Actor principal: Fotograf,Client
   * Scenariu de succes: Fotografia selectată este adăugată cu succes în albumul ales de către fotograf.
10. **Ștergere fotografie din album:**
    * Nivel: Subfuncție
    * Actor principal: Fotograf, Administrator,Client
    * Scenariu de succes: Fotografia selectată este ștearsă cu succes din albumul ales de către fotograf sau administrator.

O imagine care conține diagramă

Descriere generată automat

## 

## **4. Cerinte non-functionale**

1. Fiabilitate

Sistemul galeriei foto este conceput să fie fiabil în ceea ce privește gestionarea și accesarea fotografiilor. Validările și manipulările de eroare sunt implementate pentru a evita erorile critice și a asigura o experiență fluidă pentru utilizatori. În cazul în care apar probleme, timpul de redresare este redus prin structura clară și robustă a aplicației. Testarea extensivă a aplicației, inclusiv cazuri de utilizare pozitive și negative, contribuie la fiabilitatea sistemului..

2. Utilizabilitate

Interfața galeriei foto este simplă și intuitivă, permițând utilizatorilor să navigheze și să interacționeze cu aplicația cu ușurință. Funcționalitățile specifice fiecărui tip de utilizator sunt clare și ușor de înțeles, oferind o experiență plăcută pentru toți utilizatorii.

3. Portabilitate si compatibilitate

Aplicația galeriei foto este portabilă și compatibilă cu diverse dispozitive și sisteme de operare, cu anumite restricții privind versiunile software-ului necesar pentru a rula aplicația. Asigurați-vă că aveți instalate toate dependențele necesare înainte de a rula proiectul pentru a evita problemele de compatibilitate.

4. Scalabilitate

Galeria foto este construită cu o arhitectură modulară și un grad redus de cuplare între componente, ceea ce permite extinderea și adăugarea de noi funcționalități cu impact minim asupra restului sistemului. Această abordare asigură că aplicația poate fi scalată și adaptată pentru a răspunde nevoilor în creștere ale utilizatorilor și a altor cerințe emergente.

## **5. Design Constraints**

**Arhitectură:**

Proiectul galeriei foto este structurat folosind arhitectura Model-View-Controler (MVC) pentru a asigura o separare clară între logica aplicației și prezentarea acesteia. Acest lucru permite un flux logic al datelor, începând cu conexiunea la baza de date prin intermediul Entity Framework, continuând cu layer-ul de business logic și încheind cu layer-ul de prezentare, implementat cu ajutorul Razor pages.

**Limbaj de programare:**

Aplicația galeriei foto este dezvoltată folosind limbajul C# și tehnologia .NET, cu baza de date SQL Server. Acest lucru asigură o dezvoltare rapidă și scalabilă a aplicației, precum și o integrare facilă cu alte tehnologii din ecosistemul .NET.

**Librării adiționale:**

Pentru a reduce complexitatea și volumul codului, au fost folosite diverse librării adiționale, cum ar fi Entity Framework pentru gestionarea conexiunii cu baza de date SQL Server și Blazor pentru crearea de interfețe web moderne și interactive.

**Conectivitate:**

Proiectul se bazează pe Entity Framework pentru a gestiona conexiunea cu baza de date SQL Server, facilitând astfel o comunicare eficientă și o manipulare ușoară a datelor.

**Interfața utilizator:**

Aplicația galeriei foto folosește Razor pages pentru a crea o interfață web modernă și interactivă, care să ofere o experiență plăcută utilizatorilor și să permită o utilizare facilă a funcționalităților aplicației.

# A screenshot of a computer Description automatically generated with medium confidenceDeliverable 3

## Design Model

### Dynamic Behavior

### A picture containing text, diagram, screenshot, font Description automatically generated

### Class Diagram

Model de date Entități:

1. Utilizator - reprezintă utilizatorii care accesează galeria foto și creează albume. • Nume utilizator • Parolă • Rol
2. Fotografie - reprezintă fotografiile disponibile în galeria foto. • Titlu • URL • Identificator • Album - relație @ManyToOne
3. Album - reprezintă un album creat de un utilizator. • Identificator • Nume utilizator • Mesaj creare album • Fotografii - relație @OneToMany
4. BaseEntity - reprezintă entitatea de bază moștenită de toate clasele • Id • Data creării • Data actualizării • externalId

# **Testarea sistemului**

Testarea unitară este un tip de testare a software-ului în care sunt testate unități individuale sau componente ale unui software. Scopul este de a valida faptul că fiecare unitate a codului software funcționează conform așteptărilor. Testarea unitară se face în timpul dezvoltării (faza de codare) a unei aplicații de către dezvoltatori. Testele unitare izolează o secțiune de cod și verifică corectitudinea sa. O unitate poate fi o funcție individuală, o metodă, o procedură, un modul sau un obiect.

În cazul aplicației noastre, o galerie foto web, testarea unitară cu XUnit a fost esențială pentru a asigura funcționarea corectă a funcționalităților cheie. De exemplu, am testat unitar metodele de încărcare și ștergere a fotografiilor, crearea și gestionarea albumelor, precum și funcționalitățile de căutare și filtrare a fotografiilor. Aceste teste ne-au ajutat să identificăm și să corectăm problemele într-un stadiu timpuriu al dezvoltării, asigurându-ne că fiecare componentă a aplicației funcționează corect înainte de a fi integrată cu restul sistemului.

**Îmbunătățiri viitoare**

Aici sunt câteva îmbunătățiri potențiale viitoare pentru o aplicație web care are funcționalitatea unei galerii foto:

1. Implementarea unui sistem de recomandare: Un sistem de recomandare ar putea fi folosit pentru a sugera utilizatorilor albume sau fotografii pe baza istoricului lor de vizualizare. Acest lucru ar putea ajuta la creșterea implicării utilizatorilor și la promovarea mai multor fotografii.
2. Permiterea utilizatorilor să lase recenzii și evaluări: Permiterea utilizatorilor să lase recenzii și evaluări pentru fotografiile pe care le-au vizualizat poate ajuta la construirea unei comunități mai puternice și la îmbunătățirea calității generale a conținutului oferit.
3. Integrarea unui program de loialitate: Un program de loialitate ar putea încuraja utilizatorii să revină și să vizualizeze mai multe fotografii, oferind recompense sau alte beneficii pentru vizualizările lor continue.
4. Implementarea unui sistem de filtrare mai avansat: Oferirea utilizatorilor cu o varietate de opțiuni de filtrare poate face procesul de căutare a fotografiilor mai convenabil și prietenos pentru utilizator. Acest lucru ar putea include opțiuni precum filtrarea după genul de fotografie, după etichete, după fotografi, etc.
5. Implementarea unui chatbot sau a unei funcții de chat live: Un chatbot sau o funcție de chat live ar putea fi folosită pentru a oferi utilizatorilor suport instant și pentru a răspunde la orice întrebări pe care le-ar putea avea despre fotografiile sau procesul de încărcare a fotografiilor.
6. Implementarea unui sistem de urmărire a activității: Oferind utilizatorilor posibilitatea de a urmări activitatea lor pe site, cum ar fi fotografiile vizualizate sau comentate, ar putea ajuta la creșterea implicării și la construirea unei experiențe personalizate.
7. Personalizarea experienței utilizatorului: Personalizarea experienței utilizatorului pe baza istoricului lor de vizualizare ar putea ajuta la creșterea implicării și la promovarea mai multor fotografii. Acest lucru ar putea include afișarea recomandărilor personalizate de albume sau oferirea de recompense personalizate.

**Concluzie**

În concluzie, dezvoltarea unei aplicații web pentru o galerie foto, cu un backend în .NET, frontend în Razor Pages și o bază de date SQL Server, a reprezentat o experiență plină de provocări, dar în același timp extrem de valoroasă. Utilizarea acestor tehnologii și instrumente a permis crearea unei aplicații robuste și scalabile, capabilă să gestioneze operațiuni complexe și să susțină un număr mare de utilizatori.

.NET a oferit un cadru de lucru backend solid, cu o gamă largă de funcționalități, care a permis dezvoltatorilor să construiască o logică de afaceri complexă și să integreze cu diverse API-uri terțe. Razor Pages, pe de altă parte, a oferit un mediu de dezvoltare frontend eficient și intuitiv, care a permis crearea unei interfețe de utilizator interactive și responsive.

Integrarea cu baza de date SQL Server a permis stocarea și recuperarea eficientă a datelor, în timp ce implementarea securității bazate pe autentificarea generată de .NET a asigurat protecția informațiilor sensibile ale utilizatorului și a restricționat accesul doar la personalul autorizat.

**Bibliografie**

1. Microsoft Docs: ".NET documentation" - [Link](https://docs.microsoft.com/en-us/dotnet/)
2. Microsoft Docs: "ASP.NET Core documentation" - [Link](https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-5.0)
3. Microsoft Docs: "Razor Pages in ASP.NET Core" - [Link](https://docs.microsoft.com/en-us/aspnet/core/razor-pages/?view=aspnetcore-5.0&tabs=visual-studio)
4. Microsoft Docs: "Entity Framework Core" - [Link](https://docs.microsoft.com/en-us/ef/core/)
5. Microsoft Docs: "SQL Server Guide" - [Link](https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver15)
6. Microsoft Learn: "Introduction to ASP.NET Core" - [Link](https://docs.microsoft.com/en-us/learn/modules/introduction-to-aspnet-core/)
7. Microsoft Learn: "Create a web API with ASP.NET Core" - [Link](https://docs.microsoft.com/en-us/learn/modules/build-web-api-aspnet-core/)
8. Microsoft Learn: "Work with data in ASP.NET Core" - [Link](https://docs.microsoft.com/en-us/learn/modules/work-with-data-in-aspnet-core/)
9. Stack Overflow: ".NET Questions" - [Link](https://stackoverflow.com/questions/tagged/.net)
10. GitHub: ".NET on GitHub" - [Link](https://github.com/dotnet)