

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Кафедра інформаційних систем

КУРСОВА РОБОТА

на тему:

Розробка десктопної соціальної мережі на базі обміну фотографіями

Студента 3 курсу, групи ПМі-34
напряму підготовки: Інформатика
спеціалізація: Інформатика
Кізло Т. М.
Керівник: доц. Бернакевич І. Є.
Національна шкала _____
Кількість балів: _____ Оцінка: ECTS _____
Члени комісії

Зміст

Вступ	3
Мета	7
1 Постановка задачі	9
1.1 Фізична постановка задачі	9
1.2 Програмна постановка задачі	12
2 Проектування архітектури ти інтерфейсу	16
2.1 Модель даних	16
2.2 Розбиття на проекти	17
2.2.1 Проект для ручних тестів	18
2.2.2 Базовий проект	18
2.2.3 Проект з моделлю	19
2.2.4 Проект доступу до бази даних	19
2.2.5 Проект з бізнес логікою	21
2.2.6 Проект із тестами	22
3 Розробка програми	23
3.1 Побудова частини користувача	23
3.2 Побудова частини адміністратора	29
3.3 Загальне про аплікацію	36
4 Апробація	37
Висновок	39
Список використаних джерел	41

Вступ

Соціальна мережа – інтернет-співтовариство користувачів, об'єднаних за будь-якою ознакою на базі одного сайту, який і називається в цьому випадку соціальною мережею.

На сьогоднішній день соціальні мережі становлять важливу частину життя будь-якої людини. В них зареєстрований практично кожен. Причому дані ресурси зуміли підкорити користувачів будь-якого віку, від малого до великого. Різноманітних соцмереж стає все більше. Вони відрізняються функціоналом чи способом спілкування. Але мета їх однакова — об'єднати людей із однаковими чи не дуже інтересами.

Іншими словами, соціальна мережа у всесвітній павутині будується на тих же принципах, що і в реальному світі, але відрізняється від реальних людських спільнот тим, що у функціонуванні мережі не грає ролі географічна віддаленість її учасників один від одного. Головним чинником об'єднання користувачів у соціальну мережу є яка-небудь їх спільність – фінансове становище, стать, приналежність до тієї чи іншої раси, національності, віросповідання, професії і так далі.

Серед соціальних мереж найбільшими фаворитами вважають наступні:

1. **Lenkedln.** Аудиторія налічує 106 млн. Користувачів. Сюди входять в основному представники бізнес-структур. Тут підшукують нову роботу, укладають угоди, діляться новинами, інформацією.
2. **Badoo.** Це своєрідне місце для знайомства. У ньому 230 млн. користувачів. Портал має аналогічний функціонал, що і в соціальних мережах.

3. **Tumblr.** Улюблене місце блогерів. Вони ведуть свої блоги, коментують і читають чужі. Ресурс був створений в 2007 році, налічує 115 млн. користувачів.
4. **Instagram.** З'явився в 2010 році. Досить швидко зайняв лідируючі позиції. Користувачам сподобалася можливість ділитися відео, фотознімками.
5. **Pinterest.** Тут можна обмінюватися ідеями на різні теми, рецептами, проектами.
6. **Flickr.** Аудиторія включає 112 млн. користувачів. На даному проєкті виходить пересилати велику кількість фотографій.
7. **MySpace.** Була утворена в 2003 році, включає 50 млн. акаунтів. Використовується для: обміну фото, відео, повідомленнями. Тут знайомляться, ведуть блоги.
8. **Scribd.** Вважається текстовою копією youtube. Використовується для обміну книгами.
9. **Askfm.** Тут легко отримати на поставлені запитання відповіді. Зареєстровано 38 млн. акаунтів. Хто-небудь з них точно підкаже правильне рішення.
10. **Snapchat.** Тут можна не боятися бути таким, як є. Тут не передбачені коменти і лайки, тому немає тиску. Все справжнє, не накручене.

Найцікавішим є те, що за останні роки набирають великої популярності соціальні мережі, що свідомо вирішили відмовитись від стандартних засобів комунікації (розмова, листування тощо) та перейти до більш екстравагантних, таких як обмін картинками, відео, аудіо тощо.

Так, наприклад, **Instagram** — соціальна мережа, що базується на обміні фотографіями, дозволяє користувачам робити фотографії, застосовувати до них

фільтри, а також поширювати їх через свій сервіс і низку інших соціальних мереж.

Подібний функціонал має **Flickr** — веб-сайт для розміщення фотографій та відеоматеріалів, їх перегляду, обговорення, оцінки та архівування. Flickr популярний завдяки зручній та простій системі завантаження та пошуку фотографій. Дозволяє спілкуватися та створювати тематичні групи, соціальні мережі.

Власник фотографій має можливість:

1. обмежити доступ до своїх фото;
2. встановити умови використання фото;
3. супроводжувати свої фото коментарями;
4. дозволяти коментувати свої фото;
5. ставити до кожного фото мітки (теги), які дозволяють знаходити фото в архівах Flickr за темою, місцем або предметом присутніми на фото;
6. позначати певні фрагменти фото з коментарем;
7. прив'язати фото до місця зйомки або місцезнаходження об'єкта на географічній карті з великою точністю;
8. розмістити кожне своє фото в 10 тематичних групах (для платних рахунків максимальна кількість груп — 60).

Або **Pinterest** — соціальний фото-сервіс. Його основний функціонал полягає в завантаженні фото на певну тематику, та знаходження подібних за допомогою алгоритмів програми. Після того, як зображення завантажені на Pinterest, вони називаються пінами, а колекції, до яких вони належать, — дошки. Місія сайту звучить, як «об'єднати весь світ за допомогою речей, які їм цікаві».

Таким чином ці соціальні мережі націлені в основному не на встановлення зв'язку між близько знайомими людьми, а на розширення кола друзів, завдяки пошуку та аналізу спільних інтересів.

Саме на основі цих трьох соціальних мереж (Instagram, Flickr, Pinterest) і виникла ідея *створення своєї соціальної мережі* обміну фотографіями, яка б об'єднувала переваги цих мереж:

1. **Instagram** — можливість обміну фото, власний акаунт
2. **Flickr** — розміщення фотографій та відеоматеріалів, їх перегляду, обговорення, оцінки та архівування
3. **Pinterest** — автоматичний пошук фото цікавих фото на спільну тематику, оптимізаційні способи збереження мінімальної фізичної кількості фото, унормована політика завантаження та фільтрації фото

Та не буде містити їх недоліків:

1. **Instagram** — рекламні акаунти, погані алгоритми пошуку фото
2. **Flickr** — невірні алгоритми розставлення тегів, обмежена пам'ять для збереження фото
3. **Pinterest** — специфікації на обмежену групу аудиторії, спам та реклама у деяких пінах

Мета

Метою даного завдання є створення інформаційної системи для зберігання (у великій кількості), пересилання фото, сортування фото по категоріях та тегах, перегляд та оцінка фотоматеріалів їх архівування. Також будуть реалізовані алгоритми пошуку людей та фото.

Для пошуку людей має бути один спрощений функціонал, який вимагатиме від користувача лише введення тексту, а сам пошук буде наскрізний — тобто такий, що ігноруватиме помилки в тексті користувача та видаватиме результати по кращому співпадінні.

Пошук фото буде реалізований кількома варіантами:

1. ручний пошук через теги чи категорії
2. автоматичний пошук за допомогою нейронної мережі фото на спільну тематику із фото завантаженими користувачем

Також для користувача буде передбачена можливість зміни власних даних аккаунта чи видалення аккаунта повністю. Якщо користувач підтвердить видалення йому все одно буде надано певний період часу для скасування цієї дії. По завершенні цього терміну із системи будуть видалені всі дані користувача, враховуючи його коментарі, його оцінки до фото, самі фотознімки користувача, альбоми тощо.

Додавання фото буде зроблене за допомогою вікна перетягування. Таким чином користувач з легкістю може обрати декілька фотографій та перетягнути їх на інтерфейс програми. Обрані ним фото одразу ж будуть відображатись.

Для кращого розуміння потреб користувачів, для кожного користувача буде надана можливість надсилати повідомлення адміністратору і отримувати відповідь. Щоб уникнути спаму, адміністратор такої системи зможе блокувати користувачів, як на продуманий період часу, так і на необмежений.

Окрім функцій користувача система повинна також надавати можливості для адміністратора. Оскільки адміністратором може бути любий користувач, якому буде надана така можливість, то програма буде мати два різних інтерфейси і можливість переходу між ними.

Для адміністратора будуть передбачені функції модерування системи. А саме перегляд повідомлень користувача. Можливість відповідати на повідомлення, чи можливо обмежувати надтоїдливих користувачів у цій функції.

Також адміністратор такої інформаційної системи може переглядати інформацію про користувачів і їх публічні дані, підписників, фото, у зручному форматі.

Адміністратору також будуть передбачені можливості перегляду, зміни, створення та видалення, тегів, категорій, альбомів, тем повідомлень, повідомлень користувача тощо.

Оскільки планується довготривала розробка та підтримка даної системи, також будуть написані спеціальні класи, які допоможуть у вирішенні цієї проблеми, а саме логгер, загальний інтерфейс програмування застосунків тощо. Уся система також буде покрита юніт-тестами.

1 Постановка задачі

1.1 Фізична постановка задачі

Для користувачів такої інформаційної системи будуть надані наступні можливості:

1. створення персонального профілю. Користувач вводить лише нікнейм, який перевіряється на валідність та унікальність, а також пароллю (в базі даних зберігається лише хеш);
2. пошук та перегляд профілів інших користувачів;
3. підписатись на сторінку користувача аби отримувати сповіщення про виставлені ним фото;
4. бачити список підписників та на кого підписаний користувач;
5. мати можливість редагувати список своїх підписників на сторінці свого профілю, а також мати змогу змінювати *свою* підписку на сторінках інших користувачів. Якщо користувач не закрив вікно із підписниками він може відмінити дію видалення;
6. додавання багатьох фото в одному вікні та їх перегляд;
7. можливість видалення та відновлення фото, якщо користувач знаходиться в тому самому вікні;
8. писати листи адміністратору з метою повідомлення адміністратора про посилки в системі, пропозиції по покращенню, ставити запитання і отримати відповіді на неочевидність програмного інтерфейсу чи рішення розробників при проектуванні системи, або просто залишати відгук про програми. Листування буде приватний лише між адміністратором та користувачем, який пише повідомлення;
9. зміна персональних даних (імені, фото, пароллю, пошти тощо);

- 10.видалення всіх даних та при потребі їх відновлення протягом певного періоду часу;
- 11.перегляд фото в розширеному форматі із їхніми даними (кількістю позитивних та негативних оцінок, коментарями і так далі);
- 12.при наведенні на фото користувача буде бачити коротку інформацію, кількість коментарів, підпис до фото;
- 13.додавання власних оцінок до фото як негативних так і позитивних, їх відміна чи зміна оцінки;
- 14.додавання та видалення коментарів під своїм фото, або видалення своїх коментарів під чужим фото. Якщо користувач не закрив вікно із коментарями він може відмінити дію видалення;
- 15.коментар користувача, буде містити не лише фото, ім'я та текст користувача, а також автоматично згенеровану дату коментаря;
- 16.можливість оцінки коментарів інших користувачів. Оцінка може бути позитивна чи негативна. Користувач також може змінити свою оцінку із однієї на іншу чи забрати її взагалі;
- 17.керуванням рівнем доступу до профілі. Тобто, щоб профіль міг бути приватним чи публічним. Таким чином приватні репозиторії не можна знайти через пошук чи переглядати/коментувати фотографії користувача;
- 18.сповіщення про нові фотографії додані користувачем на якого відбувається підписка;
- 19.групування фотографій по альбомах;
- 20.можливість додати/ видалити фотографію з альбому, але не з профіля в цілому;
- 21.*чорний список* для користувачів в який вони зможуть додавати лише вибраних осіб. Таким чином людина яка знаходиться в чорному списку не матиме доступу до сторінки користувача, його фотографій тощо;
- 22.користувач також зможе змінити кольорову тему усієї програми для себе чи для окремих компонентів інтерфейсу користувача;

- 23.при завантаженні фото, користувач може редагувати його перед публікацією. Редагування фото включає в себе спеціальні інструменти малювання (кольорову кісточку, пензлик, тощо), а також зміну кольорової схеми зображення: на світлішу чи темнішу, змінити наповненість окремого компоненту кольору: червоного, зеленого чи синього; вибір наперед виготовлених кольорових тем: сепія, чорно-біла тема, тощо;
- 24.користувач також зможе повертати фото, щоб встановити їх правильну позицію. Також буде додана нейронна мережа яка буде аналізувати фотографії користувача і у вигляді сповіщень відправлятиме користувачу пропозиції щодо можливого повороту фотографії. Рішення чи приймати пропозицію чи ні залежить від користувача. Крім того він завжди зможе відключити аналіз нейронною мережею;
- 25.у програмі буде локалізація різними мовами;
- 26.також в програму можна буде увійти як звичайний користувач, знаходити публічні профілі і переглядати їх дані;

Частина адміністратора передбачає наступне:

1. усі дані відображені в таблицях можна фільтрувати та сортувати нажавши на необхідну колонку;
2. перегляд/редагування/видалення тем повідомлень, які користувач пише адміністратору;
3. перегляд/видалення повідомлень користувача;
4. персональний чат із користувачем;
5. перегляд/видалення/редагування даних користувача (імені, паролю, фотографії профілю, чи є користувач адміном, чи є користувач заблокованим, фотографій користувача, приватності його аккаунта, редагування його чорного списку);
6. перегляд/видалення фото;
7. перегляд/редагування/видалення коментарів до фото;

8. видалення/зміна імені альбому. Можливість переміщати фотографії між альбомами;
9. зміна алгоритму хешування пароля (HAVAL, MD2, MD4, MD5, N-Hash, RIPEMD-160, SHA, Tiger, власна реалізація);
10. додавання редагування кольорових тем програми, які будуть доступні на вибір кожному користувачеві;
11. активація/деактивація кольорових тем для фото (сепії, чорно білий режим тощо);
12. налаштування нейронної мережі (кількість проміжних шарів, вибір функції активації (Identity, Relu, Sigmoid, власна реалізація), рівень навчання), яка буде повертати фотографії;

1.2 Програмна постановка задачі

Як можна побачити дана програма передбачає велику кількість можливостей як для звичайного користувача, так і для адміністратора. Тепер коли ми визначились, які вимоги накладаються на нашу програму, залишилось вирішити за допомогою якої мови програмування буде побудована ця інформаційна система. Також варто вибрати технології розробки та платформу на якій буде розгортатись програма.

Серед численних мов програмування, я вибрав C#. C# — об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET.

Розроблена Андерсом Гейлсбергом, Скотом Вілтамутом та Пітером Гольде під егідою Microsoft Research (при фірмі Microsoft).

Серед переваг мови C# можна визначити наступні:

1. Безпечний ООП підхід, звичайне наслідування, множинне наслідування реалізується за допомогою інтерфейсів;
 2. Не потрібно використовувати вказівники;
 3. Автоматичне управління пам'яттю за допомогою збирання сміття.
- Враховуючи це, ключове слово delete в C# не зарезервоване;

4. Формальні синтаксичні конструкції для класів, інтерфейсів, структур, переліку і делегатів;
5. Перевантаження операцій для спеціальних типів без зайвої складності;
6. Підтримка узагальненого програмування;
7. Підтримка анонімних методів;
8. Підтримка суворо типізованих запитів LINQ;
9. Безпека типів;
10. Підтримка всіх можливостей середовища та сучасних технологій;

Проект буде створений на платформі .NET Framework — це програмна платформа для побудови програм на базі сімейств операційних систем Windows, а також інших систем.

Таким чином дана програма зможе розгортатись лише на Windows, тим не менш, це незначний недолік, оскільки ця операційна система займає 80% ринку.

Тепер варто визначити яким чином буде розгортатись програма: у якості веб-сервіса, аплікації чи мобільний додаток.

Зараз доволі популярні веб-додатки, оскільки їх набагато легше розробляти. Вам не треба мати справи із різними версіями операційної системи, реєстром, СОМ, тощо. Тим не менш, варто піклуватись про різні браузері, які постійно змінюють один одного, а отже підтримка програми ніколи не припиняється.

Але однією із важливих причин, являється те що користувач завжди має бути підключений до мережі. Я хочу цього уникнути і надати можливість для користувача роботи з аплікацією в зручний для нього момент. Таким чином синхронізація даних і сервером буде відбуватись лише в момент з'єднання, в інший час користувач буде обмежений в деякій функціональності, але, що важливо — не в усій.

Також десктопна програма має вразі кращу продуктивність, оскільки вона працює із компонентами операційної системи через зручний адаптер.

Продуктивність також збільшується із можливістю використання багатопотокового чи асинхронного програмування, що не завжди доступно у веб-аплікаціях. Її не потрібно перевіряти в усіх браузерах, а інтерфейс буде незмінний незалежно від версії операційної системи.

Отож, я зупинив свій вибір на розробці десктопної програми.

Мова програмування C# надає великі можливості та незліченну кількість фреймворків для розробки десктопних програм. Я буду використовувати Windows Presentation Foundation. Це одна із провідних технологій розробки програмного забезпечення, яка отримала свою популярність, завдяки сильному графічному інтерфейсу який розробляється за допомогою Extensible Application Markup Language. Хоча мова XAML — це технологія, що може бути застосована до багатьох різних предметних областей, її головне призначення — конструювання інтерфейсів користувачів WPF. Інакше кажучи, документи XAML визначають розташування панелей, кнопок та інших елементів керування, що становлять вікна в застосунку WPF.

Оскільки програма оперує великою кількістю даних, то варто їх зберігати в Базі даних. Серед різноманітних СУБД, я використовую SQL Server. Вона являється однією із найбільш популярних систем керування базами даних. Дана СУБД підходить для різноманітних проектів: від невеликих додатків, до великих, багато навантажених проектів.

Його основні особливості:

1. Продуктивність, SQL Server працює дуже швидко;
2. Надійність і безпека. SQL Server надає можливість шифрувати дані;
3. Простота. Із даною СУБД легко працювати і вести адміністрування;
4. Для організації баз даних MS SQL Server використовує реляційну модель.

Ця модель була розроблена ще в 1970 році Едгаром Коддом, а на сьогоднішній день являється стандартом для організації баз даних;

Для спрощення роботи з базою даних використаю модуль Entity Framework, який містить об'єктно-орієнтований підхід для доступу до бази даних. Використання цього модуля допоможе мапити усі дані з бази даних на колекції C# та пришвидшуватиме роботу програми.

Цей фреймворк пропонує зручну роботу із класами сутностями, які описують модель таблиці, у вигляді DTO-класів. Робота із базою даних, виглядає подібно до роботи із звичайною колекцією мови C#.

2. Проектування архітектури та інтерфейсу

Даний проект містить велику кількість операцій, які варто продумати наперед, перед тим як розробляти. Спробуємо як найкраще розбити систему на частини, їх взаємодію одна з одною, те як між ними передається інформація, як ці частини розвиваються поодиночі і як все вищеописане найкраще записати використовуючи формальну чи неформальну нотацію.

Розглянемо такі аспекти розробки, як:

1. Створення ООП моделі предметної області;
2. Розбиття програми на окремі проекти, розробка яких буде вестись незалежно один від одного;
3. Розробка Unit-тестів;
4. Архітектура програми під Windows Presentation Foundation з урахуванням eXtensible Application Markup Language ;
5. Розробка інтерфейсу користувача з розділенням моделі і представлення (MVVM);
6. Розробка UserControl для побудови інтерфейсу користувача;

2.1 Модель даних

Для вирішення поставленої задачі, сформуємо базу даних. Основною сутністю буде користувач (таблиця User) він може виставляти фотографії (таблиця Photo) та писати коментарі (таблиця Comment). Оскільки є можливість оцінювати, як фотографії, так і коментарі, додамо проміжні таблиці PhotoLike і CommentLike відповідно. Для реалізації моделі підписників нам знадобиться таблиця Followers. Можливість користувача писати повідомлення адміністратору на обрану тему відображають таблиці Messages та Subject.

ER-діаграма бази даних зображена на *рисунок 2.1*:

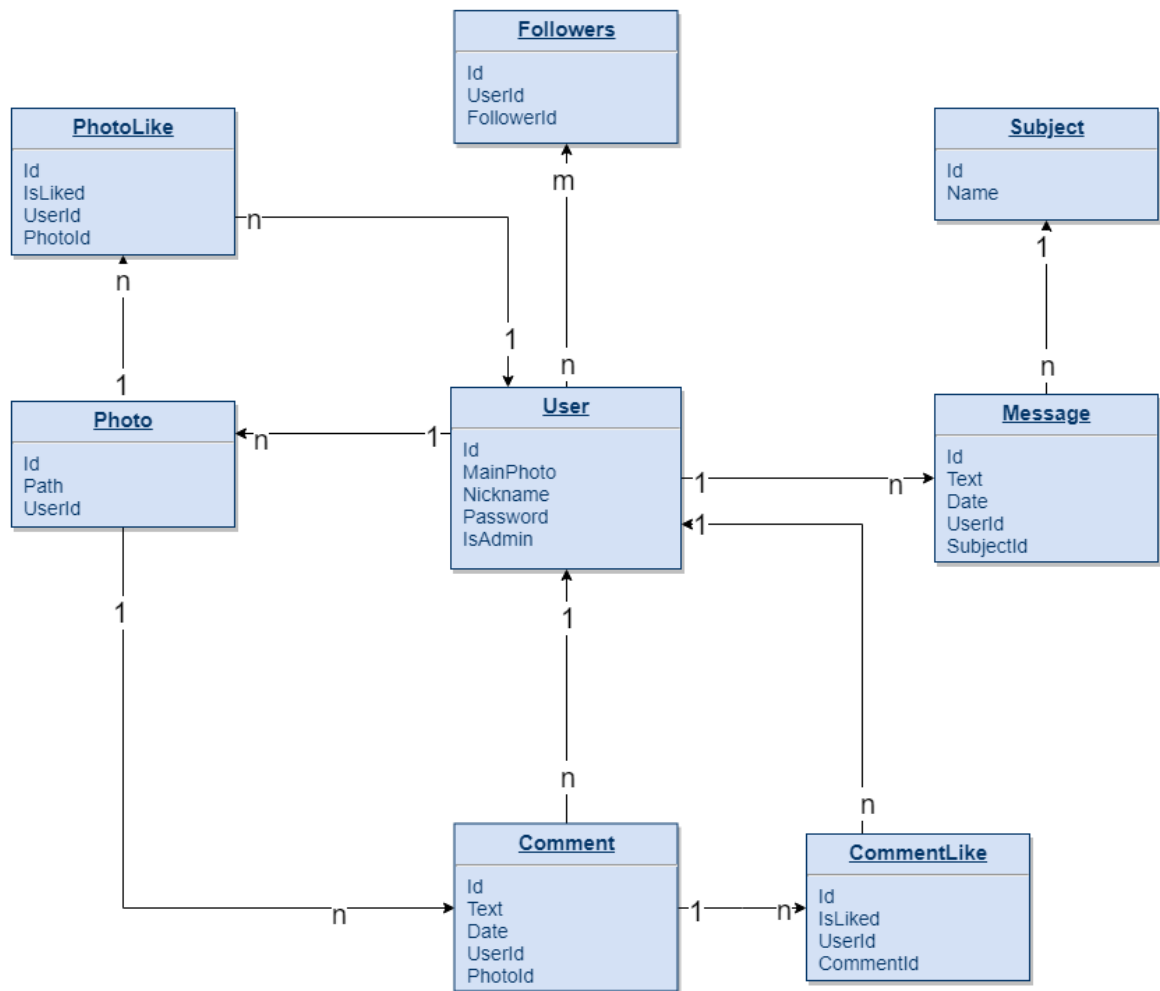


Рисунок 2.1 - ER-діаграма бази даних

Дана модель не є фінальною, але вона становить гарний початок. Також, щоб не втратити даних, але мати можливість змінити структуру бази даних, ми використовуємо механізм міграцій.

2.2 Розбиття на проекти

Варто зазначити, що я не буду розробляти проект, а потім по завершенню переходити до наступного. Проект може змінитися, можуть змінюватись вимоги, чи технічне завдання, змінитись графічний інтерфейс чи ще якийсь компонент. Я скористаюсь методикою неперервної інтеграції. Суть її в тому, що ми завжди матимемо робочу версію програми — Minimal Valuable Product — MVP (не патерн Model View Presenter). Таким чином ми завжди можемо переглянути наш

додаток, показати його майбутньому користувачу, виявити недоліки в моделі та інтерфейсі на ранніх стадіях розробки.

Цикл розробки при неперервній інтеграції виглядає так:

1. розробка моделі;
2. розробка інтерфейсу;
3. тестування;
4. рефакторинг;

Таким чином, розробка полягає в постійному внесені невеликих змін в модель та інтерфейс. Це все потрібно, оскільки набагато краще мати легку і примітивну робочу програму, аніж складну і не робочу. Якщо в кінці розробки виявиться, що ми взагалі невірно зрозуміли програмну область, то переробка невеликої програми буде значно простішою, ніж великої. Інтерфейс не обов'язково буде розроблятися швидше моделі, але якщо так станеться, то частина його можливостей буде замінена заглушками.

2.2.1 Проект для ручних тестів

Для тестування алгоритмів практичними засобами чудово підходить консоль. Тому я створю проект, який буде містити код, виклику модулів. Код у ньому буде постійно змінюватись, в залежності від того який алгоритм буде проходити дебаг. Таким чином я зможу прискорити розробку алгоритмів, та перевіряти їх результат в ручну.

2.2.2 Базовий проект

Для зберігання всіх налаштувань (спеціальних констант, конфігурацій, які впливають на роботу програми), використаємо спеціальні статичні класи.

Решта налаштувань(стрічки з'єднання, розташування вікна тощо), які містять динамічні налаштування я зберігатиму у спеціальному XML файлі. Також цей проект міститиме загальні базові класи та інтерфейси, які можуть використовуватись іншими проектами, наприклад, поліморфна фабрика, логгер.

Оскільки також програма передбачає локалізацію, усі повідомлення, які будуть відображатись користувачу збережемо у спеціальних файлах. При запуску програми, залежно від обраної локалізації, дані будуть зчитувати з потрібного файла.

2.2.3 Проект з моделлю

Перше з чого варто розпочати розробку — продумання моделі предметної області (яку також називають доменною моделлю, об'єктною моделлю, ООП моделлю тощо).

Нашою метою є створення класів, які максимально точно відображають предмети зовнішнього світу, із усіма їхніми властивостями, залежностями, та функціоналом.

У даному випадку, предметною областю є менеджер фотографій. більша частина технічного завдання, передбачає маніпуляції з даними, а в моделі можна віднести лише алгоритми пов'язані із фотографіями, різними фільтрами, тощо.

2.2.4 Проект доступу до бази даних

Наша База даних, представляє собою класичну реляційну модель, а класи бізнес-логіки оперують об'єктами. Об'єктна модель — це не те саме, що реляційна модель для представлення одних і тих самих даних. Саме для цього варто винести моделі пов'язані із даними в окремий проект.

Також цей проект буде мати алгоритми для роботи з даними (фільтрація, сортування, порівняння). Напишемо спеціальні класи, сутності, які будуть відображати модель нашої таблиці. Також створимо загальний інтерфейс і абстрактний клас для сутностей з декількома шаблонними методами, **рисунок 2.2:**

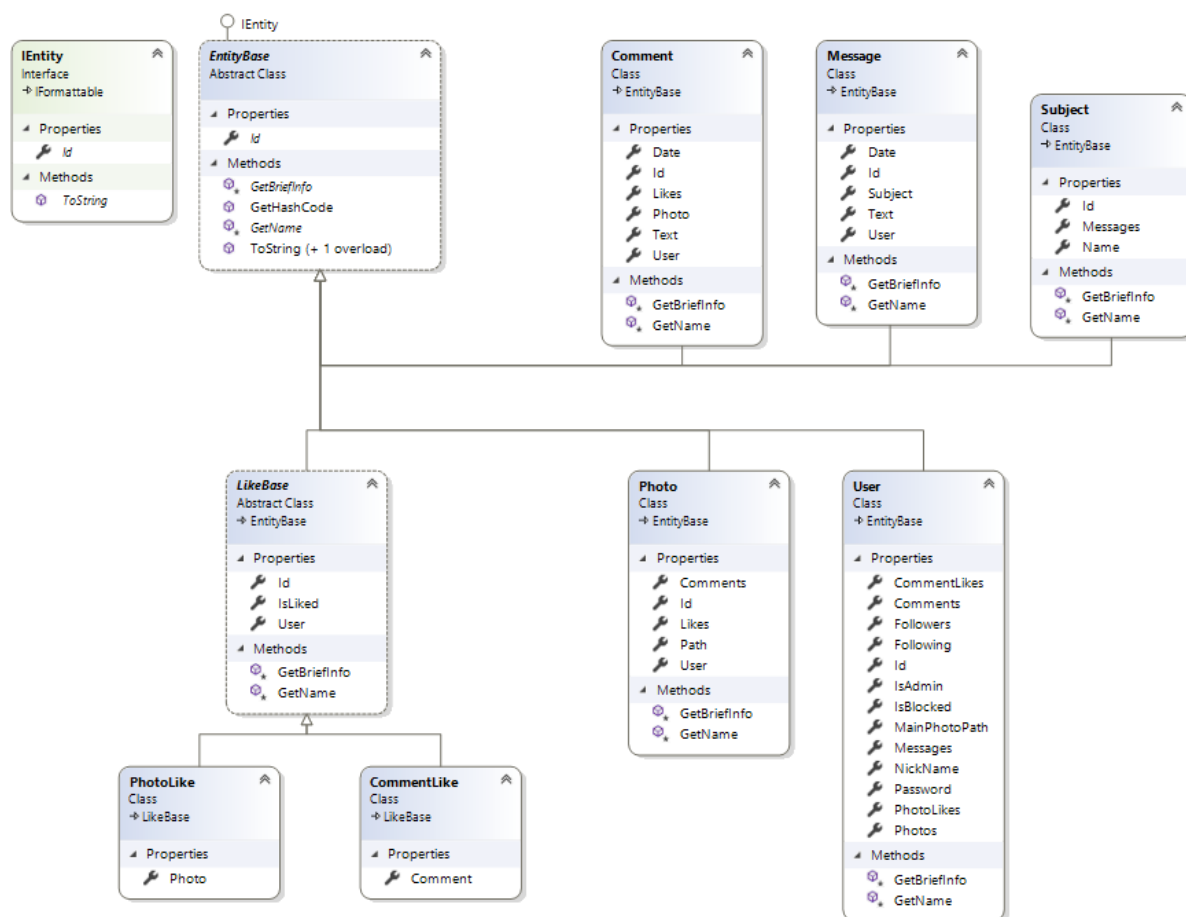


Рисунок 2.2 - Модель діаграми сутностей.

Основні CRUD операції винесемо в класи репозиторії, із використанням відповідного патерну. Напишемо узагальнений репозиторій Generic Repository, та інтерфейс для нього. Для кожної сутності теж створимо свої репозиторії та інтерфейси. Таким чином ми дотримуємось усіх принципів SOLID, та зможемо досягти Inversion of Control, тобто працювати з абстракціями, а не реалізаціями. Щоб отримати репозиторії в будь-якій точці програми створимо спеціальний клас UnitOfWork та інтерфейс для нього. Цей клас реалізовуватиме декілька патернів: спочатку патерн одинак, щоб мати можливість отримати єдиний екземпляр цього класу в будь-якій точці програми, а також шаблон лінивої загрузки, тобто репозиторії будуть створюватись лише в момент безпосереднього їх виклику. Це допоможе не навантажувати додаток зайвими операціями, **рисунок 2.3**:

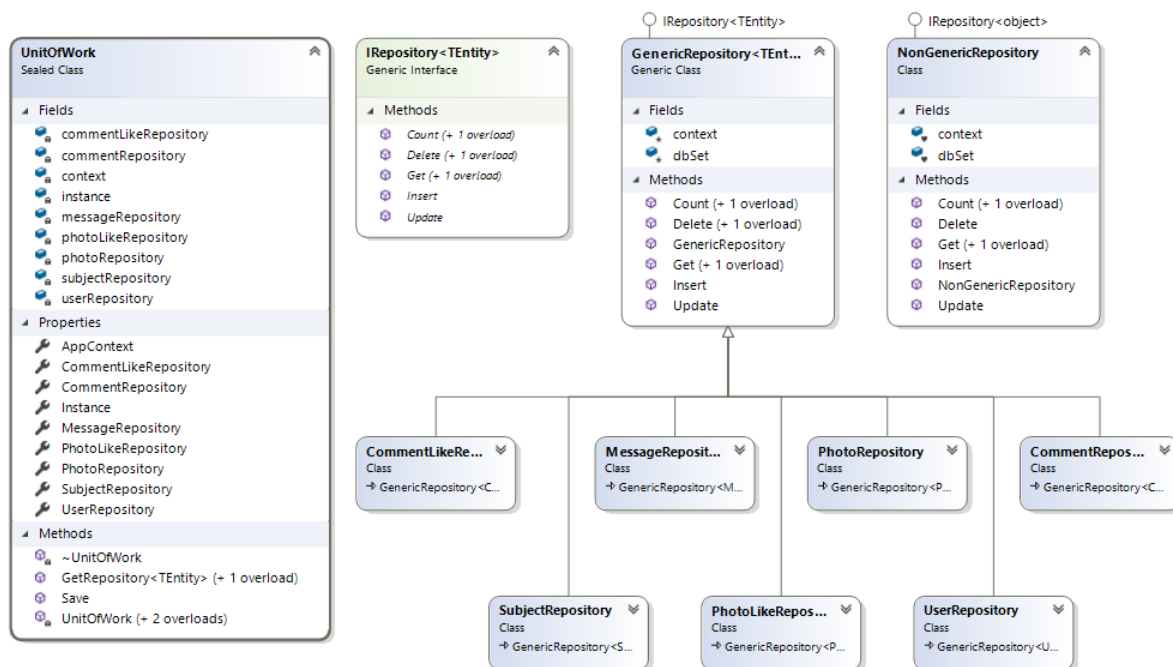


Рисунок 2.3 - Модель діаграми класів репозиторіїв.

2.2.5 Проект з бізнес логікою

Однієї із основних вимог, які ставляться до нашої програми, це її підтримка та супровід. Аби досягти легкої зміни компонентів використаємо патерн міст. Це дозволить нам відділити абстракції від її конкретної реалізації. Також, як згадувалось раніше, ми використовуємо WPF у якості графічного інтерфейсе. Гарною практикою вважається побудова програм на цій платформі із використанням архітектурного патерну MVVM (Model-View-ViewModel). Це дозволить розділити програму на три функціональні частини: графічний інтерфейс, бізнес логіку та саму модель зберігання даних і реалізації алгоритмів.

Саму бізнес-логіку зачасту поміщають в команди, а їх контекст виконання в ViewModel класи. Напишемо кілька основних команд, які нам знадобляться в майбутньому, а саме CommandBase — абстрактний клас команд, RelayCommand — узагальнена команда, яка виконує делегат, а також MultipleCommand — клас, який виконує декілька команд та припиняє їх виконання, якщо одна із команд не виконалась вірно, наприклад, дві різні команди валідація моделі, збереження даних в БД (патерн ланцюжок відповідальності). Самі команди прив'яжемо до

графічного інтерфейсу шляхом байндингу. Модель команд можна побачити на *рисунку 2.4*:

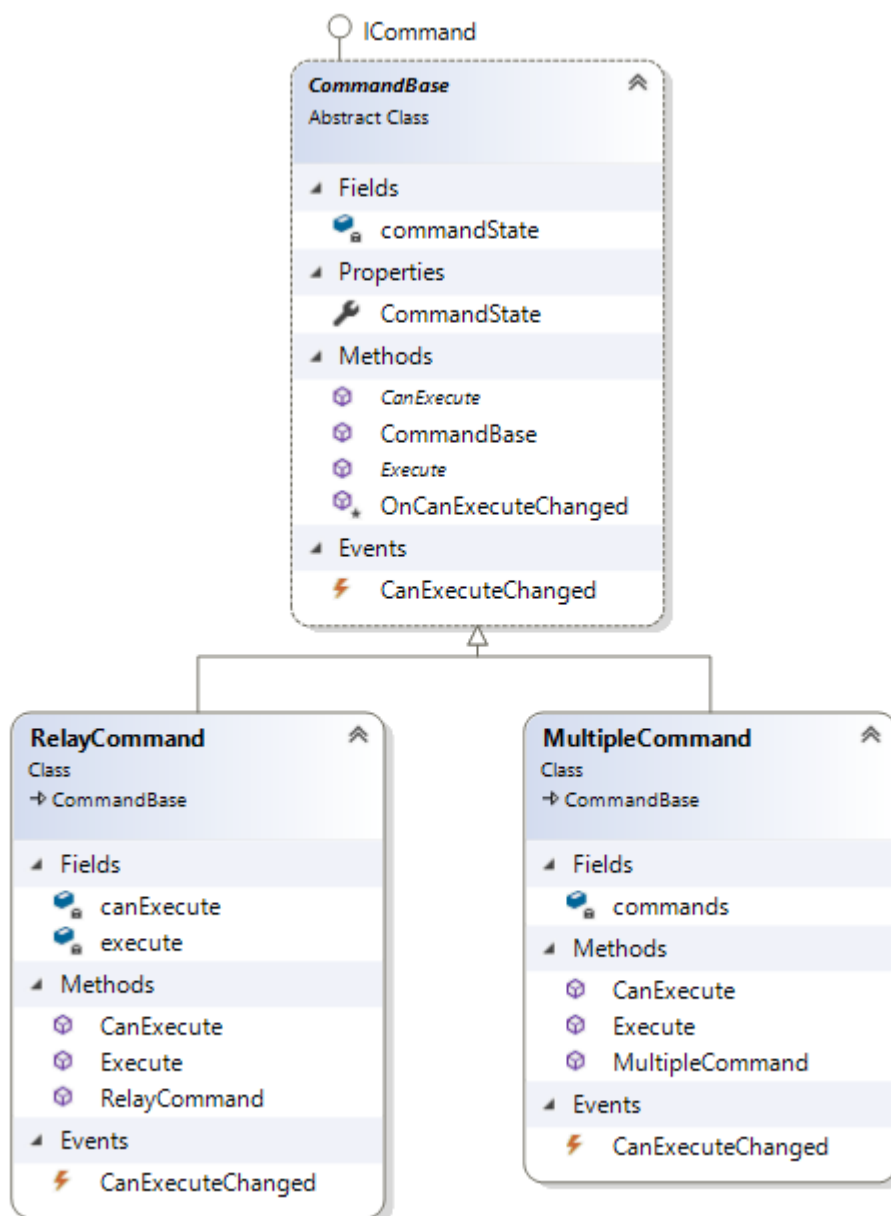


Рисунок 2.4 - Модель команд

2.2.6 Проекти із тестами

Також для підтримки проекту нам знадобляться Unit-тести. Так для кожного проекту створимо, ще проект, який міститиме тести. Таким чином, якщо десь в проекті будуть зміни, в алгоритмах чи його структурі, ми завжди можемо перевірити працездатність системи, без зайвої потреби запускати додаток і перевіряти все вручну.

3 Розробка програми

3.1 Побудова частини користувача

Для входу в систему користувачеві необхідно ввести його дані, це його Нікнейм та пароль. Цих даних достатньо, для реєстрації і для логування, *рисунок 3.1*:

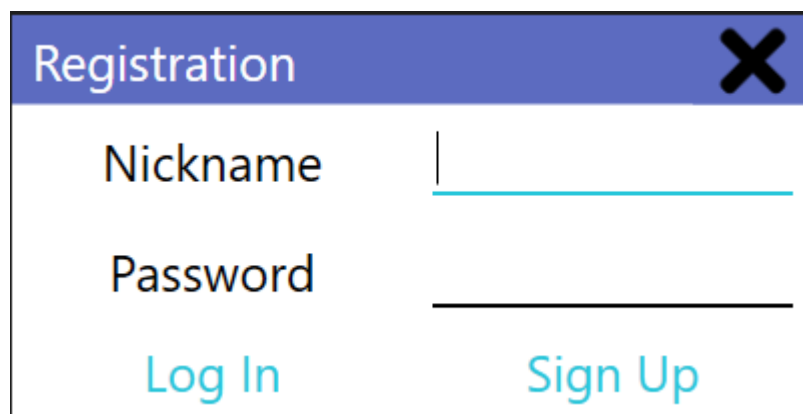
A registration dialog box with a blue header bar containing the word "Registration" and a close button (X). The main area is white and contains two input fields: "Nickname" and "Password". Below the "Nickname" field is a light blue "Log In" button, and below the "Password" field is a light blue "Sign Up" button.

Рисунок 3.1 - Вхід в систему

Після входу в систему, чи створення акаунта, користувач може побачити свій профіль із дефолтним зображенням в якості фотографії профіля, *рисунок 3.2*:



Рисунок 3.2 - Профіль користувача

Користувач може додати декілька фотографій. Це робить за допомогою перетягування їх на спеціальну панель. Таким чином можна з легкістю додати декілька фотографій, застосувати до них фільтри (при нажиманні правої клавіші миші, потрібно вибрати спеціальний пункт з контекстного меню), видалити вибрані фото, тощо, *рисунок 3.3*. Завантажені фотографії зберігатимуться на сервері, а в базі даних, міститиметься їхня унікальна назва. Для генерації унікальної назви зображення, був написаний спеціальний алгоритм. При видаленні фотографії, чи акаунта користувача усі, його фотографії теж будуть видалені із сервера, та з бази даних.

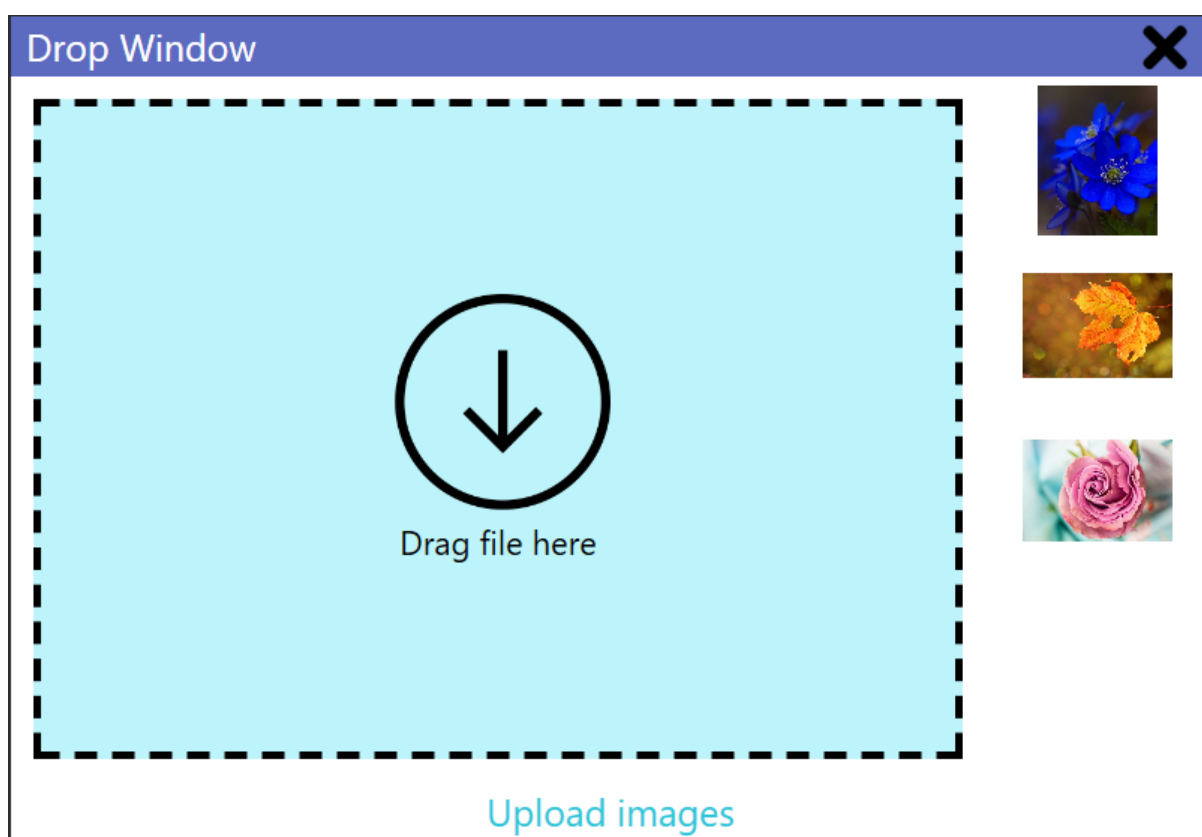


Рисунок 3.3 - Вікно для завантаження фотографій

Однієї із найважчих проблем при завантаженні фото, було те що WPF хешує фотографії за допомогою Direct, але завдяки спеціальним налаштуванням її вдалось вирішити.

Також, як зазначалось вище, користувачеві, дозволено спілкуватись із адміністрацією програми. Тобто він може задати свої запитання, чи запропонувати способи покращення програми, **рисунок 3.4**:

Ask Question

Any questions?

If you have any questions or you want to improve our service you can write a little message to us.

Subject

Message

Error

Suggestion

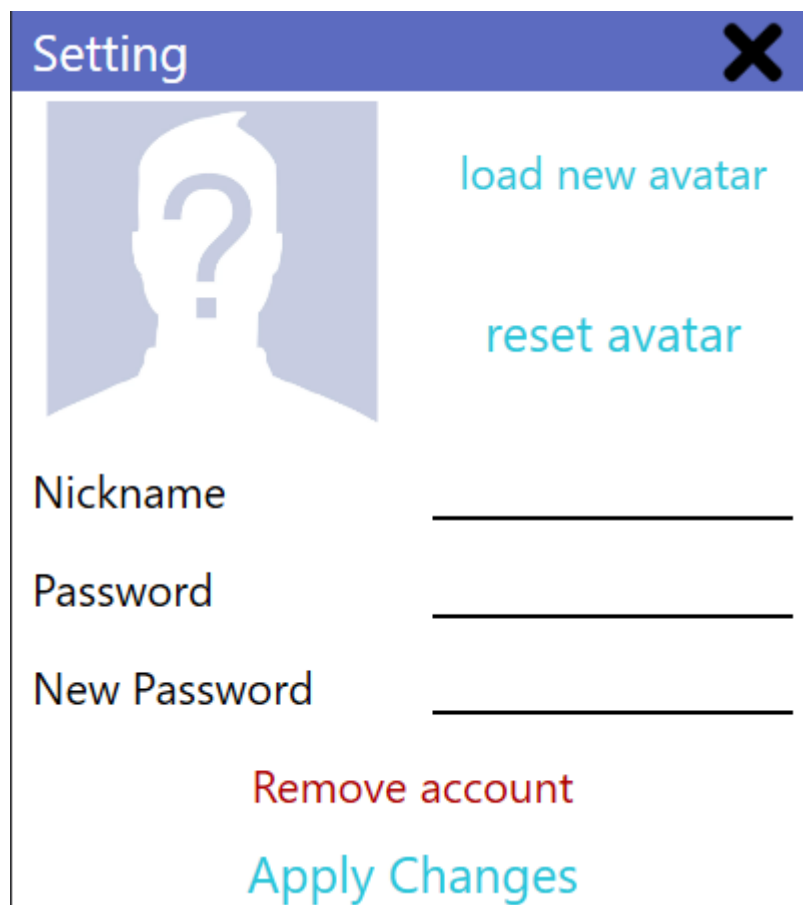
Question

Respond

Ask

Рисунок 3.4 - Вікно спілкування користувача з адміністрацією

Однієї із важливих можливостей є зміна персональних даних. Для користувача є можливість змінити свій нікнейм, пароль, фотографію профілю, тощо в зручний для нього час. Але при цьому, варто пам'ятати, що для того щоб змінити дані варто спочатку вказати пароль, **рисунок 3.5**:



The image shows a 'Setting' window with a blue header bar containing the title 'Setting' and a close button (X). The main content area has a light blue background. On the left, there is a placeholder for an avatar, represented by a white silhouette of a person's head and shoulders with a large question mark inside. To the right of the avatar placeholder are two buttons: 'load new avatar' and 'reset avatar', both in a light blue color. Below the avatar section, there are three text input fields with labels: 'Nickname', 'Password', and 'New Password'. At the bottom of the window, there are two buttons: 'Remove account' in red and 'Apply Changes' in light blue.

Рисунок 3.5 - Налаштування профілю користувача

Оскільки однієї із найважливіших причин існування соціальних мереж, є можливістю взаємодії з іншими користувачами, тому в нашій системі також доступний пошук інших користувачів, **рисунок 3.6**. При цьому варто зазначити, що для пошуку не потрібно повністю вказувати ім'я користувача, так як банально ви можете його не знати, чи вказати невірно. У системі реалізований наскрізний пошук, тому ви побачити усі співпадиння, із введеними вами даними:

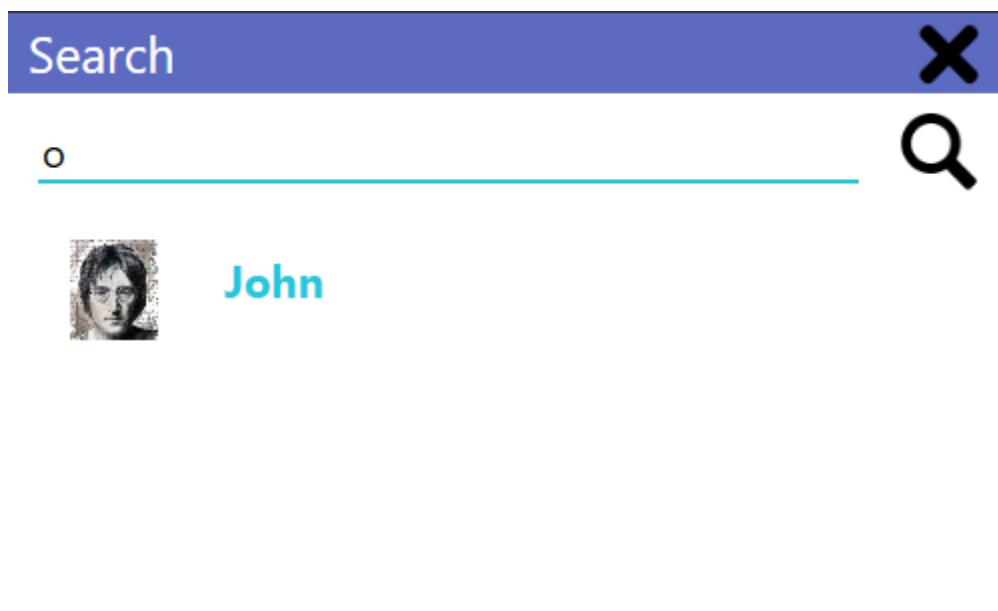
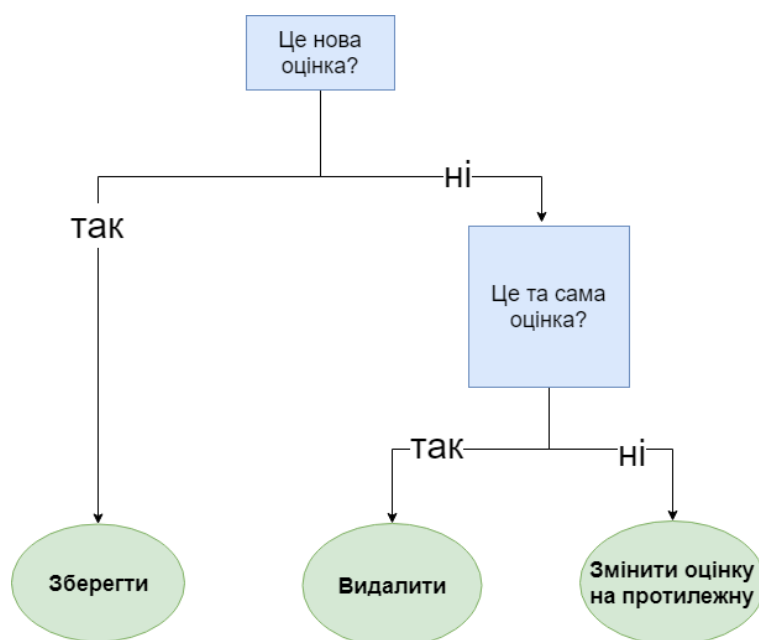


Рисунок 3.6 - Зображення реалізації наскрізного пошуку

На сторінці іншого користувача можна підписатись на нього (чи відписатись при потребі), глянути його підписників та тих на кого підписаний він сам. Також можна переглянути фотографії детальніше, поставити свою оцінку (як негативну так і позитивну), додати/видалити коментар, **рисунок 3.8**. Коментарі інших користувачів також можна оцінювати. Окрім введенного тексту в інтерфейсі також відображається дата коментаря в американському форматі, ким був написаний коментар, та фотографія автора. Схема роботи алгоритму оцінювання подана на **рисунок 3.7**:

Рисунок 3.7 - Схема роботи алгоритму оцінювання



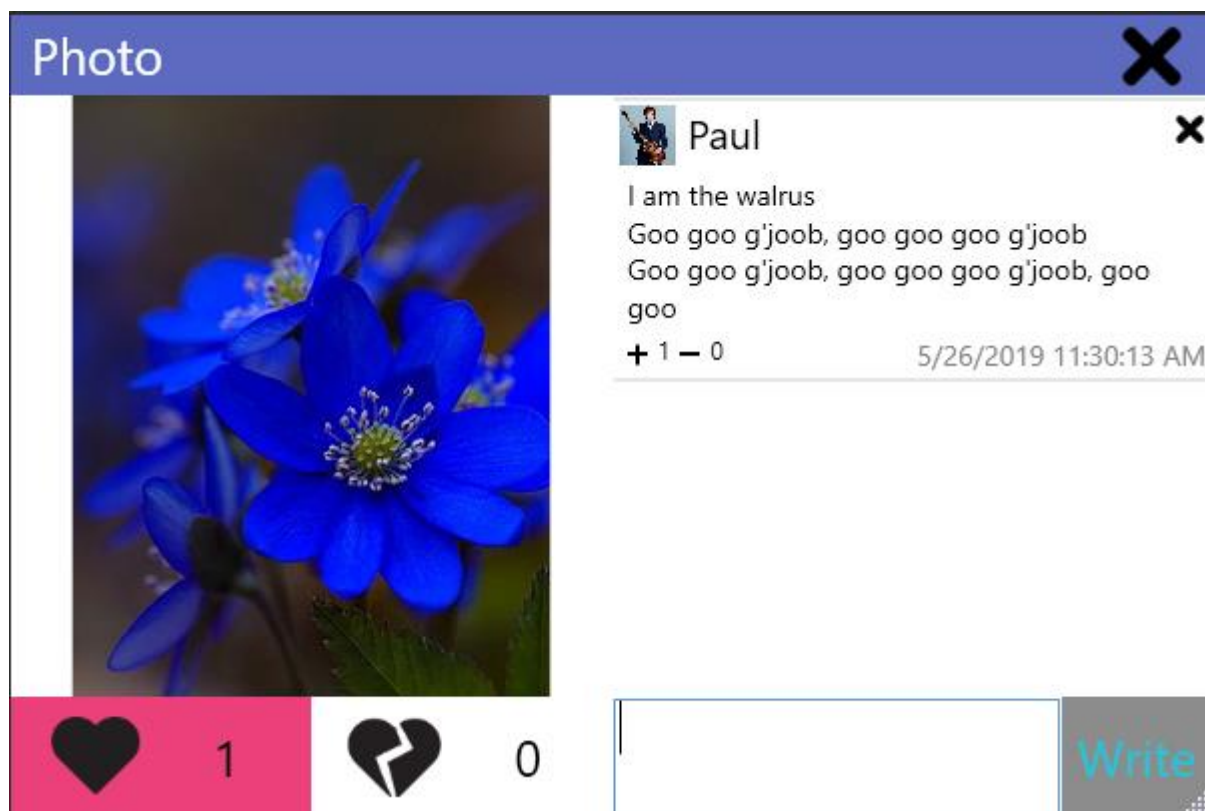


Рисунок 3.8 - Перегляд фотографії користувача

Список підписників відображається в окремому вікні, **рисунок 3.9**. Користувач завжди може відписатись від людини, чи видалити людину із своїх підписників. Також, якщо користувач знаходиться на сторінці іншого користувача, то він може відписати самого себе, або видалити себе із списку, тих на кого підписаний, себе. Варто зауважити, що ці алгоритми є узагальненими і відбуваються в одному вікні. Перевірка на чийй сторінці знаходиться користувач (на своїй чи на сторінці іншого користувача) відбувається методами XAML.

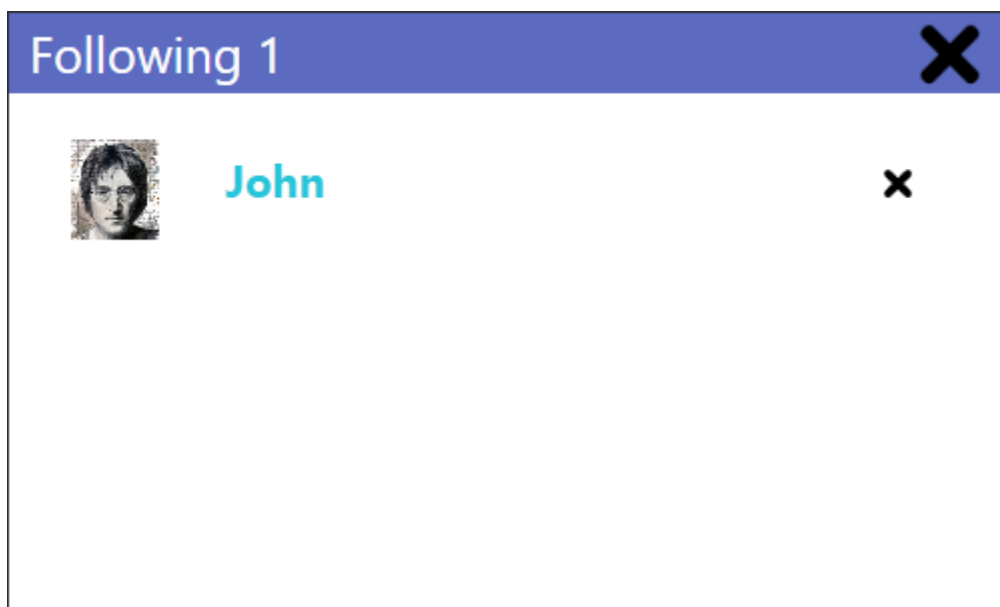


Рисунок 3.9 - Огляд вікна із підписниками.

Також користувач може змінити зовнішній вигляд програми. Йому на вибір доступна одна із 4 кольорових тем: синій колір (стандартний), рожевий, жовтий та зелений. Теми можна переглянути на **рисунку 3.10**:



Рисунок 3.10 - Приклади кольорових тем

3.2 Побудова частини адміністратора

Як і зазначалось раніше, окрім частини користувача розробляється також частини адміністратора. За замовчуванням система містить одного адміністратора, який не може бути видалений із системи. Адміністратором можна призначити

довільного користувача. Вхід у систему у ролі адміністратора, відбувається через вікно логування, **рисунок 3.1**. Якщо користувач являється адміністратором, то при вході в систему він матиме вибір між можливим способом авторизації, у ролі адміністратора чи звичайного користувача, **рисунок 3.11**.

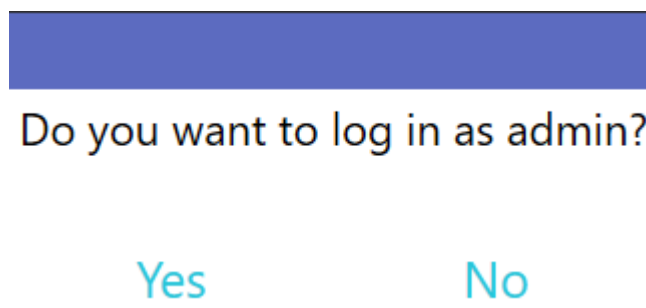


Рисунок 3.11 - Вибір способу входу в систему, як адміністратор чи користувач.

Для адміністратора доступні розширені можливості в редагування чи перегляді таких даних, як повідомлення користувача, теми повідомлень, самі користувачі, їх фотографії та коментарі.

Основна панель програмного інтерфейсі змінюється в залежності від вибраного пункту меню. Це досягнуто завдяки створенню власних компонентів інтерфейсу, а їх зміна відбувається завдяки реалізації патерну поліморфна фабрика. Таким чином, це покращить продуктивність програми, зменшить кількість використаної пам'яті, оскільки не потрібно створювати нове вікно, а лише одну робочу панель, а також, що найважливіше, це покращить сам процес використання програми.

При перегляді повідомлень, що їх пише користувач, адміністратор має можливість відфільтрувати повідомлення по їх темі чи даті написання, а також можна сортувати записи за іменем користувача, темі повідомлення, тексту, чи даті, **рисунок 3.12**. Також можна видалити повідомлення чи перейти в його розширений перегляд, **рисунок 3.13**.

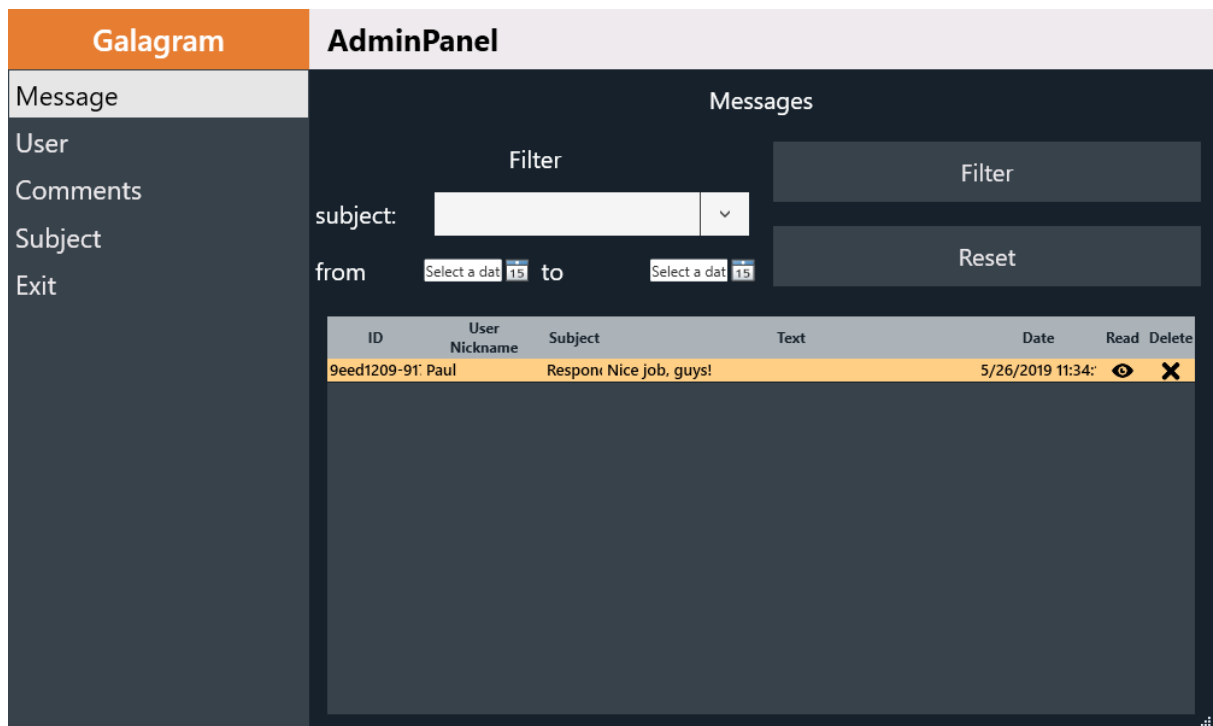


Рисунок 3.12 - Зображення інтерфейсу користувача, із відображенням всіх повідомлень.

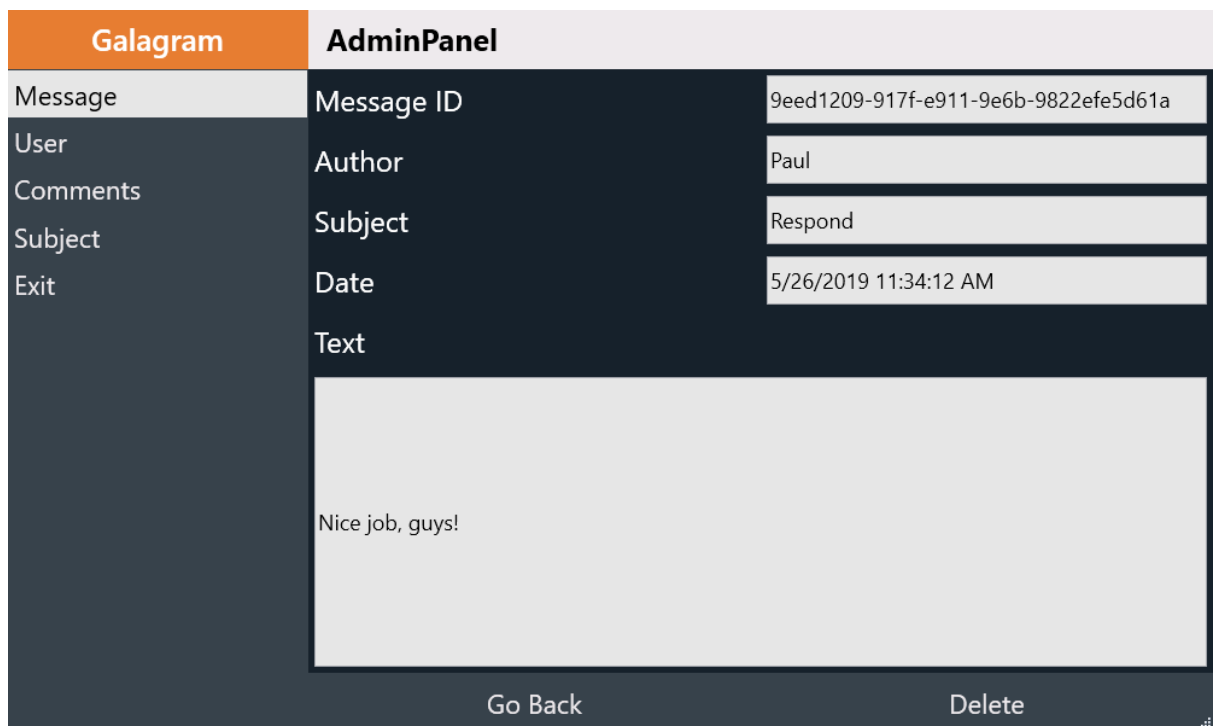


Рисунок 3.13 - Детальний перегляд повідомлень

При перегляді усіх даних, варто відмітити, що їх кількість може досягати великих об'ємів, які можуть вплинути на продуктивність програми. Щоб цього уникнути використовується механізм віртуалізації. Тобто в систему завантажуються лише та кількість даних, які відображається. Таким чином незалежно від об'єму даних, аплікація не буде зазнавати проблем із їх відображенням та завантаженням.

При управлінні користувачами адміністратор може фільтрувати дані по імені (використовуються регулярні вирази для пошуку співпадінь), та за тим чи користувач заблокований. Сортувати дані можна по імені, в лексикографічному порядку та по кількості фото, підписників, тих на кого підписаний користувач. Адміністратор може редагувати та переглядати дані чи видалити користувача із системи, *рисунок 3.14* :

The screenshot displays the 'AdminPanel' interface for the 'Galagram' application. On the left is a sidebar menu with options: Message, User, Comments, Subject, and Exit. The main area shows the details of a selected user, 'John'. The fields include: User ID (f66e16af-8c7f-e911-9e6b-9822efe5d61a), Nickname (John), Avatar (a small profile picture with a 'reset' button), Password (1111), Is Admin (checkbox), and Is Blocked (checkbox). Below these are statistics: Followers (1) and Following (0). A section for 'Photos' shows 3 photos, with the first one being a pink rose. At the bottom, there are 'Go Back' and 'Save Changes' buttons.

Рисунок 3.14 - Редагування даних користувача.

У вікна вибраного користувача адміністратор може переглянути список фотографій та коротку інформацію про кожний фотознімок (кількість позитивних

При перегляді коментарів адміністратор може відфільтрувати дані по тексту, чи слові яке зустрічається у вмісті коментаря, або ж по користувачеві, який написав коментар. Також доступна фільтрація, по даті від/до. Сортувати можна по користувачам, та тексту в лексикографічному порядку і даті. Коментарі можна переглядати, видаляти та редагувати, **рисунок 3.16**:


Galagram	AdminPanel	
Message	Comment ID	f296cd7a-907f-e911-9e6b-9822efe5d61a
User	Author	Paul
Comments	Photo	
Subject	Date	5/26/2019 12:00:00 AM 15
Exit	Text	I am the walrus Goo goo g'joob, goo goo goo g'joob Goo goo g'joob, goo goo goo g'joob, goo goo
	Like	1
	Dislike	0
	<div>Paul</div> <div></div>	
	<div>Go Back</div> <div>Save Changes</div>	

Рисунок 3.16 - Перегляд інформацію про коментар

Адміністратору також надається можливість редагувати список тем повідомлень користувача, **рисунок 3.17**. У цій панелі можна сортувати теми по їх назвав, а для редагування та створення виділяється окреме вікно, **рисунок 3.18**.

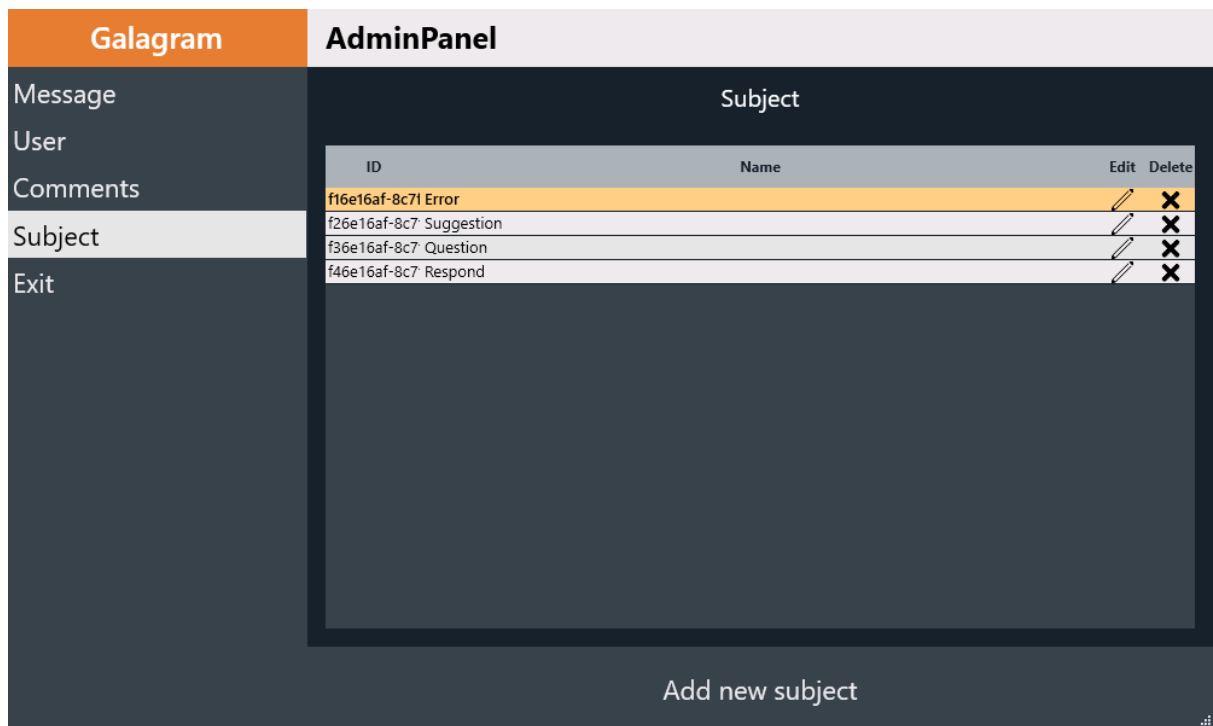


Рисунок 3.17 - Вікно із списком тем.

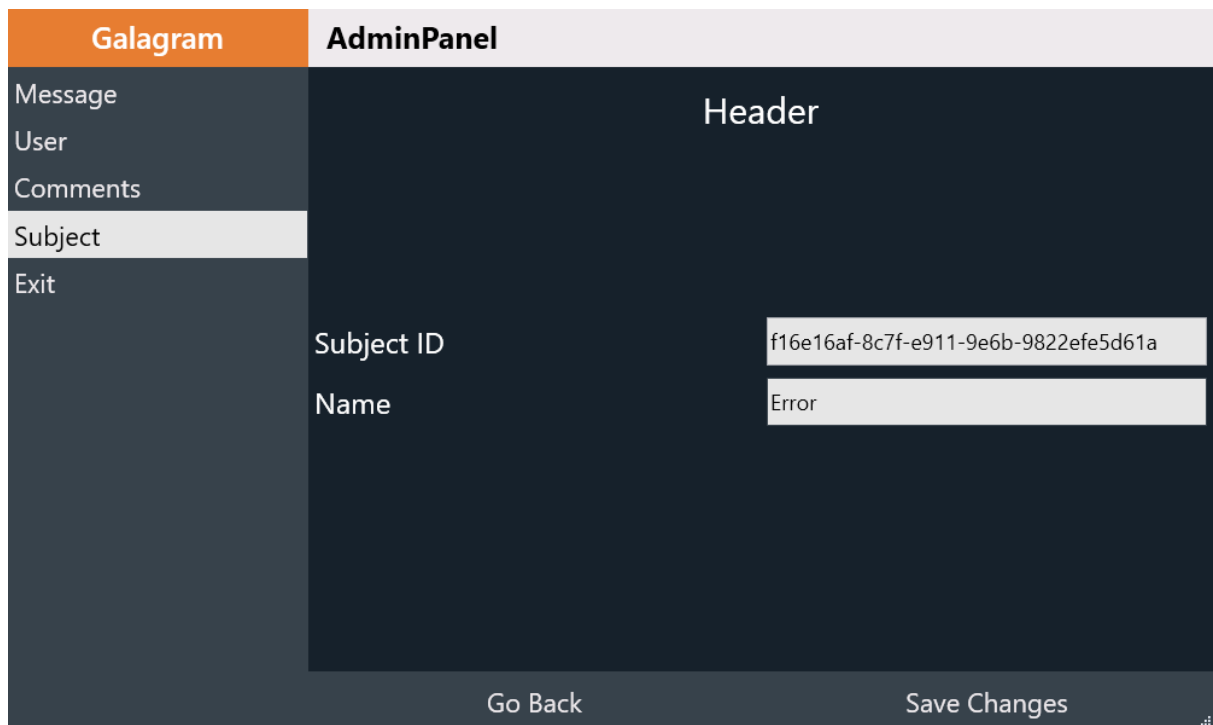


Рисунок 3.18 - Редагування/створення теми.

3.3 Загальне про аплікацію

Для встановлення програми користувачеві варто скачати спеціальний **інсталлер**, файл із розширенням `msi`, та пройти невеликий етап встановлення, вибрати шлях на встановлення, а також прочитати ліцензію.

Усі **конфігурації** користувача (розмір вікна програми, розміщення вікна, кольорова тема вікна тощо) зберігаються в системному реєстрі і при деінсталяції видаляються.

Саму аплікацію можна **запустити лише один раз**. При повторному запуску, спочатку відбувається перевірка на наявність такого процесу і при його присутності запуск нового додатку обривається, передавши перед цим керування на вже існуючий процес.

Також в програмі присутня плавна **анімації** і **локалізації** різними мовами. При зміні локалізації чи кольорової теми, аплікацію варто перезапустити, оскільки саме під час запуску програми відбувається завантаження файлів із мовою та кольорами.

4 Апробація

При розробці, використовувались модульні тести, з готовим фреймворком для написання автоматичних тестів MsTest. Усі зв'язки були замінені Moq-об'єктами.

Для тестування Баз Даних використались різні режими тестів:

1. невелика кількість даних (до 100 записів на таблицю)
2. середній розмір (до 1000 записів на таблицю)
3. великий розмір (до 10 000 записів на таблицю)
4. екстремальний розмір (понад 1 000 000 записів на таблицю)

Переключення між режимами являє собою патерн стратегію.

Це робиться не лише для перевірки працездатності Баз Даних, а також для її перевірки її ефективності.

Також для перевірки роботи усієї систему використовувались інтеграційні тести з різними підходами:

1. знизу до верху
2. зверху до низу

Для тестів були введені деякі конвенції:

Усі тест-класи мають назву відповідного класу із закінченням Test. Ця сама конвенція вжита і до методів. Тести будуть написані за технологією AAA.

Приклад вигляду тестів можна побачити на *рисунку 4.1*:

```

[TestMethod]
public void TestingBlockName()
{
    // BEHAVIOR
    // the beginning balance should be reduced by debit amount
    // TAKE
    // passed regular parameter
    // debit amount is less than beginnig balance
    // RETURN
    // a positive value after money has been dabit

    // Arrange
    double beginningBalance = 11.99;
    double debitAmount = 4.55;
    double expected = 7.44;
    BankAccount account = new BankAccount("Mr. Bryan Walton", beginningBalance);

    // Act
    account.Debit(debitAmount);
    double actual = account.Balance;

    // Assert
    Assert.AreEqual(expected, actual, 0.001, "Account not debited correctly");
}

```

Рисунок 4.1 - Структура тесту.

Для того щоб не довелося перевіряти кожний елемент програми вручну, був написаний спеціальний клас Логгер, який зберігає відомості про роботу системи у спеціальний файл.

Він являє собою реалізацію патерна одинак, та може здійснювати логування у декількох режима важливості, по одному чи в їх комбінації. Також відбувається ротація лог-файлів, тобто видалення файлу якщо він є достатньо великим. За потреби логгер можна динамічно вимкнути чи включити. Саме логування, щоб не погіршити роботи програми відбувається в двох режимах: синхронному та асинхронному.

Висновок

Отже, я написав соціальну мережу для обміну та збереження фотознімків. Система реалізує схему підписників. Також надає можливість додавати фотографії та виставляти їм оцінку чи коментувати. Присутній зручний пошук та адміністрування. Адміністратор має можливість зручного перегляду та зміни усіх даних. Взаємодіяти із користувачами, читати їх повідомлення та побажання.

Інформаційна система являє собою десктопну програму із об'єктно-орієнтованим доступом до бази даних. Для написання системи мені знадобилися різноманітні інструменти на подоби Windows Presentation Foundation, а також Entity Framework.

Для виконання даної задачі можна було скористатись і іншими технологіями. Наприклад першим ділом визначитись із середовищем розгортання систему:

1. веб-сервіс
2. десктоп
3. мобайл
4. тощо

Графічним інтерфейс міг бути побудований із використанням наступних технологій:

1. html, css, javascript
2. Windows Forms
3. Windows Presentation Foundation
4. тощо

Частина бізнес-логіки могла бути створена за допомогою різних платформ:

1. Active Server Pages
2. Hypertext Preprocessor

3. Node.js
4. тощо

Рівень доступу до бази даних міг бути побудований з використанням різноманітних підходів:

1. ActiveX Data Objects
2. Entity Framework
3. Entity Framework Core
4. тощо

В перспективі розвитку проекту є додавання публічних та приватних профілів. Система альбомів, створення користувачем власної кольорової теми, тощо. Також велика кількість завдань не була виконано як результат браку часу, але їх виконання може відбутись у наступних версіях програми.

Завдяки тому, що велика частина розробки програми зосереджена на архітектурній компонентів, використання патерну міст та MVVM, написання понад 350 unit-тестів та спеціального логгер, а також дотримання принципів архітектурної розробки (SOLID, KISS, DRY) додавання нового функціоналу не являтиме собою великої проблеми.

Список використаних джерел

1. Andrew Troelsen Pro C# 7: With .NET and .NET Core, Minneapolis, Minnesota, USA, 2017. – 1372 с.
2. Christian Nagel Professional C#7 and .NET Core 2.0 7th Edition, Indianapolis, Indiana, USA, 2017. — 1948 с
3. <https://metanit.com/sharp/wpf/>
4. <https://metanit.com/sharp/entityframework/>
5. <http://helpyourselfhere.blogspot.com/2015/02/wpf-mvvm-relaycommand-implementation.html>
6. <https://stackoverflow.com/questions/933613/how-do-i-use-assert-to-verify-that-an-exception-has-been-thrown>
7. <https://www.c-sharpcorner.com/UploadFile/dacca2/fundamental-of-unit-testing-don%E2%80%99t-test-your-private-method/>
8. <https://www.c-sharpcorner.com/UploadFile/dacca2/fundamental-of-unit-testing-understand-mock-object-in-unit/>
9. <https://stackoverflow.com/questions/31499265/followers-schema-in-entity-framework>
10. <https://docs.microsoft.com/en-us/dotnet/framework/wpf/app-development/how-to-automatically-size-a-window-to-fit-its-content>
11. <https://stackoverflow.com/questions/32084943/async-lock-not-allowed>
12. <https://stackoverflow.com/questions/17037424/undo-dbcontext-add>
13. <https://stackoverflow.com/questions/15220411/entity-framework-delete-all-rows-in-table>

14. <https://social.msdn.microsoft.com/Forums/en-US/62226de0-fa91-420d-8244-854004d5b2ea/how-to-get-all-table-names-of-db-using-ef?forum=adodotnetentityframework>
15. <https://stackoverflow.com/questions/5170429/deleting-database-from-c-sharp>
16. <https://stackoverflow.com/questions/2240421/using-binding-for-the-value-property-of-datatrigger-condition>
17. <https://stackoverflow.com/questions/20040555/process-cannot-access-the-image-file-because-it-is-being-used-by-another-process>
18. http://www.thejoyofcode.com/WPF_Image_element_locks_my_local_file.aspx
19. <https://stackoverflow.com/questions/6205472/mvvm-passing-eventargs-as-command-parameter>
20. <https://stackoverflow.com/questions/42341592/wpf-rectangle-border-with-corner-from-connecting-line-of-two-dashes>
21. <https://stackoverflow.com/questions/253849/cannot-truncate-table-because-it-is-being-referenced-by-a-foreign-key-constraint>
22. <https://social.msdn.microsoft.com/Forums/vstudio/en-US/42b3db75-e61e-4f59-bf2b-c96a40cfb4e4/how-can-i-give-another-process-focus-from-c?forum=csharpgeneral>
23. <https://stackoverflow.com/questions/27449298/setforegroundwindow-doesnt-work-with-minimized-process>
24. <https://stackoverflow.com/questions/19147/what-is-the-correct-way-to-create-a-single-instance-wpf-application?page=1&tab=votes#tab-top>
25. <https://stackoverflow.com/questions/43500/is-there-a-built-in-method-to-compare-collections>