МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 Программное обеспечение информационных технологий (программирование интернет приложений)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема Программное средство «Социальная сеть для любителей кошек»

Исполнитель

студент (ка) 2 курса группы 6 Селицкий Николай Евгеньевич

(Ф.И.О.)

Руководитель работы ст. преподаватель Сухорукова И. Г.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Мущук А. Н.

(подпись)

Минск 2024

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой ПИ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В.Смелов

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированные технологии программирования и стандарты проектирования"

|  |  |
| --- | --- |
| Специальность: 1-40 01 01 Программное обеспечение информационных технологий  Студент: Селицкий Николай Евгеньевич | Группа:\_\_6\_\_ |
| **Тема: Социальная сеть для любителей кошек** | |

**1. Срок сдачи студентом законченной работы**: "20 мая 2024 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПC поддерживает:

* **Функции администратора сервиса:**
* Поддерживать работу c базой данных.
* Предупреждать и отправлять в бан нарушителей.
* **Функции клиента:**
* Создание и удаление аккаунта.
* Поиск и добавление друзей.
* Связь с администратором.
* Создание и/или присоединение к группам.
* Общение с друзьями и членами групп.
* Заполнение информацию о своем питомце.

**2.2.** При выполнении курсового проекта необходимо использовать принципы проектирования ООП. Приложение разрабатывается под ОС Windows и представляет собой настольное приложение (desktop). Отображение, бизнес логика должны быть максимально независимы друг от друга для возможности расширения. Диаграммы вариантов использования, классов реализации задачи, взаимодействия разработать на основе UML. Язык разработки проекта – C#. Управление программой должно быть интуитивно понятным и удобным. При разработке использовать несколько наиболее подходящих шаблонов проектирования ПО.

**3. Содержание расчетно-пояснительной записки**

* Введение
* Постановка задачи и обзор литературы
* Проектирование архитектуры проекта
* Разработка функциональной модели и модели данных ПС
* Тестирование
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненной курсовой работы:**

* + Теоретическая часть курсового проекта должны быть представлены в формате docx. Оформление записки должно быть согласно выданным правилам.
  + Листинги программы представляются в приложении.
  + Пояснительную записку, листинги, проект (инсталляцию проекта) необходимо загрузить на диск, указанный преподавателем.

#### Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 12.03.2024 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 22.03.2024 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2024 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 5.04.2024 |  |
| 5 | Кодирование программного средства | 15.04.2024 |  |
| 6 | Тестирования и отладка программного средства | 01.05.2024 |  |
| 7 | Оформление пояснительной записки | 11.05.2024 |  |
| 9 | Защита проекта | 20.05.2024 |  |

**5. Дата выдачи задания** \_\_\_\_29.02.2024\_\_\_\_\_\_

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Сухорукова И.Г.*

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Селицкий Н.Е.*

(дата и подпись студента)

Оглавление

[Введение 5](#_Toc167081337)

[1. Аналитический обзор прототипов 6](#_Toc167081338)

[1.1 Анализ прототипов 6](#_Toc167081339)

[1.1.1 Социальная сеть для всех “Вк” 6](#_Toc167081340)

[1.1.2 Социальная сеть Instogram 7](#_Toc167081341)

[1.2 Постановка задачи 8](#_Toc167081342)

[1.3 Вывод к разделу 9](#_Toc167081343)

[2. Анализ требований к программному средству и разработка функциональных требований 10](#_Toc167081344)

[2.1 Спецификация функциональных требований к программному средству 10](#_Toc167081345)

[2.2 Вывод к разделу 11](#_Toc167081346)

[3. Проектирование программного средства 12](#_Toc167081347)

[3.1 Архитектура системы 12](#_Toc167081348)

[3.2 Взаимоотношения между классами 13](#_Toc167081349)

[3.3 Модель базы данных 13](#_Toc167081350)

[3.4 Проектирование последовательности взаимодействия 15](#_Toc167081351)

[3.5 Вывод к разделу 16](#_Toc167081352)

[4. Реализация программного средства 17](#_Toc167081353)

[4.1 Реализация паттернов репозиторий и Unit of Work 17](#_Toc167081354)

[4.2 Реализация хеширования паролей 22](#_Toc167081355)

[4.3 Вывод к разделу 23](#_Toc167081356)

[5 Тестирование, проверка работоспособности и анализ полученных результатов 24](#_Toc167081357)

[5.1 Тестирование авторизации и регистрации 24](#_Toc167081358)

[5.2 Тестирование изменения личной информации 26](#_Toc167081359)

[5.3 Тестирование списка друзей 27](#_Toc167081360)

[5.4 Тестирование списка групп 28](#_Toc167081361)

[5.5 Тестирование чата 29](#_Toc167081362)

[6 Руководство по установке и использованию 30](#_Toc167081363)

[**Заключение** 35](#_Toc167081364)

[**Список использованных источников** 36](#_Toc167081365)

Введение

Люди часто забывают, что с кошками надо дружить. А ведь кошки хорошие друзья и общаться с их хозяевами должно быть очень весело. Но как же это сделать?

В наш век информационных технологий этот вопрос смешон – конечно же социальная сеть по интересам.

Цель данного курсового проекта - разработать приложение для общения и распространения информации своим друзьям и всем любителям котиков.

Основные задачи, которые должен выполнять создаваемый проект включают в себя:

* регистрация новых клиентов и хранение информации о них;
* нахождение и добавление друзей;
* общение в группах и лично.

Для реализации поставленных задач будут использоваться язык программирования C# и технология WPF (Windows Presentation Foundation), разработанная Microsoft для создания удобных пользовательских интерфейсов и отображения графических элементов в Windows-приложениях. В качестве СУБД для хранения данных будет использоваться Microsoft SQL Server, который является одним из наиболее популярных продуктов СУБД на рынке, используемый в крупных корпоративных системах, а также для построения приложений. Для улучшения дизайна и работы с интерфейсом будет использоваться библиотека Matirial Disign. Доступ к базам данных производиться с использованием EntityFramworkCore от Microsoft, который является удобным и популярным способом взаимодействия с базами данных в WPF.

1. Аналитический обзор прототипов

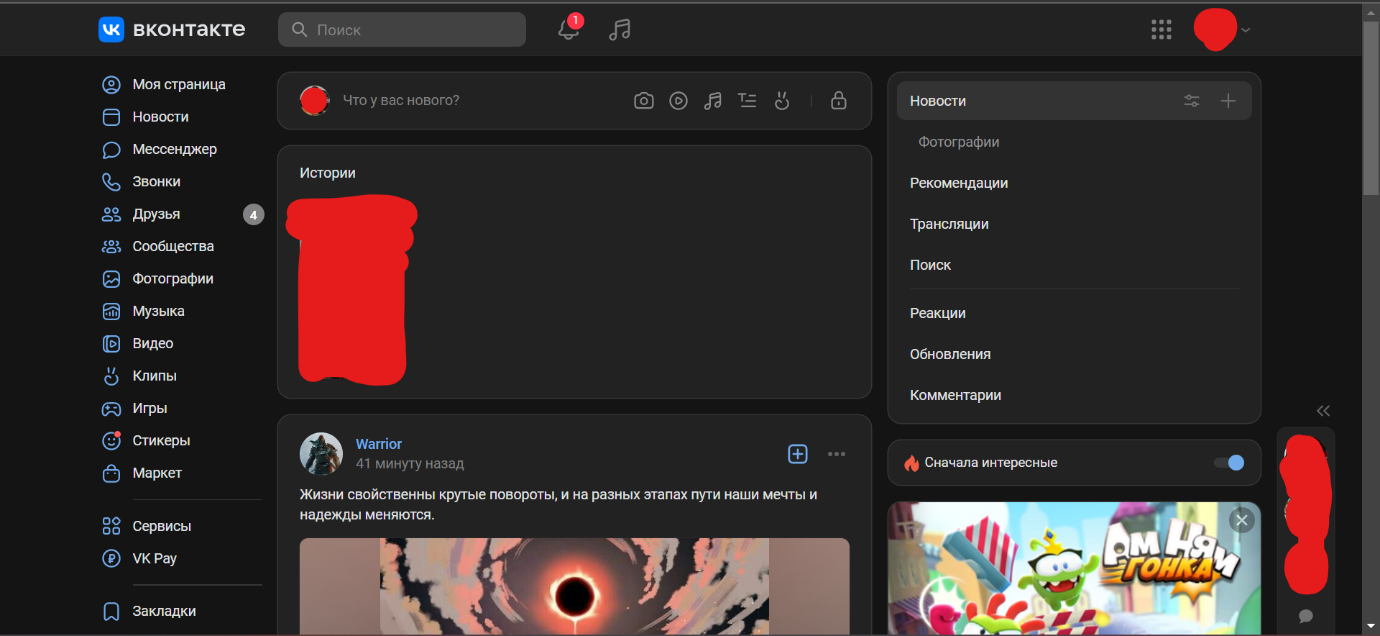
* 1. Анализ прототипов

Курсовой проект представляет из себя социальную сеть для людей имеющих у себя дома котов. Идея относительно свежая поэтому пока что имеет мало аналогов. В этом разделе будут приведены интернет-ресурсы. Приведенные примеры будут вторичными конкурентами так как первичных на данный момент нет.

* + 1. Социальная сеть для всех “Вк”

Это одна из самых популярных социальных сетей на данным момент. Она так же помогает людям общаться. В приложении есть возможность создание аккаунта, создавать группы, общаться с друзьями, добавлять и удалять людей, размещать информацию о себе и многое другое.

<https://vk.com/feed>;



Изображение №1.1 “приложение Вк”

**Преимуществами ресурса:**

* поиск представлен в виде области с лупой в верхней части, экрана которая заметна и удобна;
* навигация очень удобная и расположена слева сбоку;
* регистрация носильная при входе что хорошо для социальной сети так как не дает возможности работать как инкогнито;
* из цветов только 2 цвета основной черный который является фоновым и, например синий (цвет хорошо подходящий под черный) которым выделяются иконки и имена пользователей (рассмотрим темную тему). Оно разбито по блокам что удобно, углы сглажены что делает сайт более презентабельным.

**Недостатки:**

* насильная регистрация как плюс, так и минус так как у нас не выйдет в первый раз зайти в приложение быстро;
* cложность для новичков – пусть навигация сделана удобно, но многие необязательные функции находятся на главной панели что может запутать нового пользователя.
  + 1. Социальная сеть Instogram

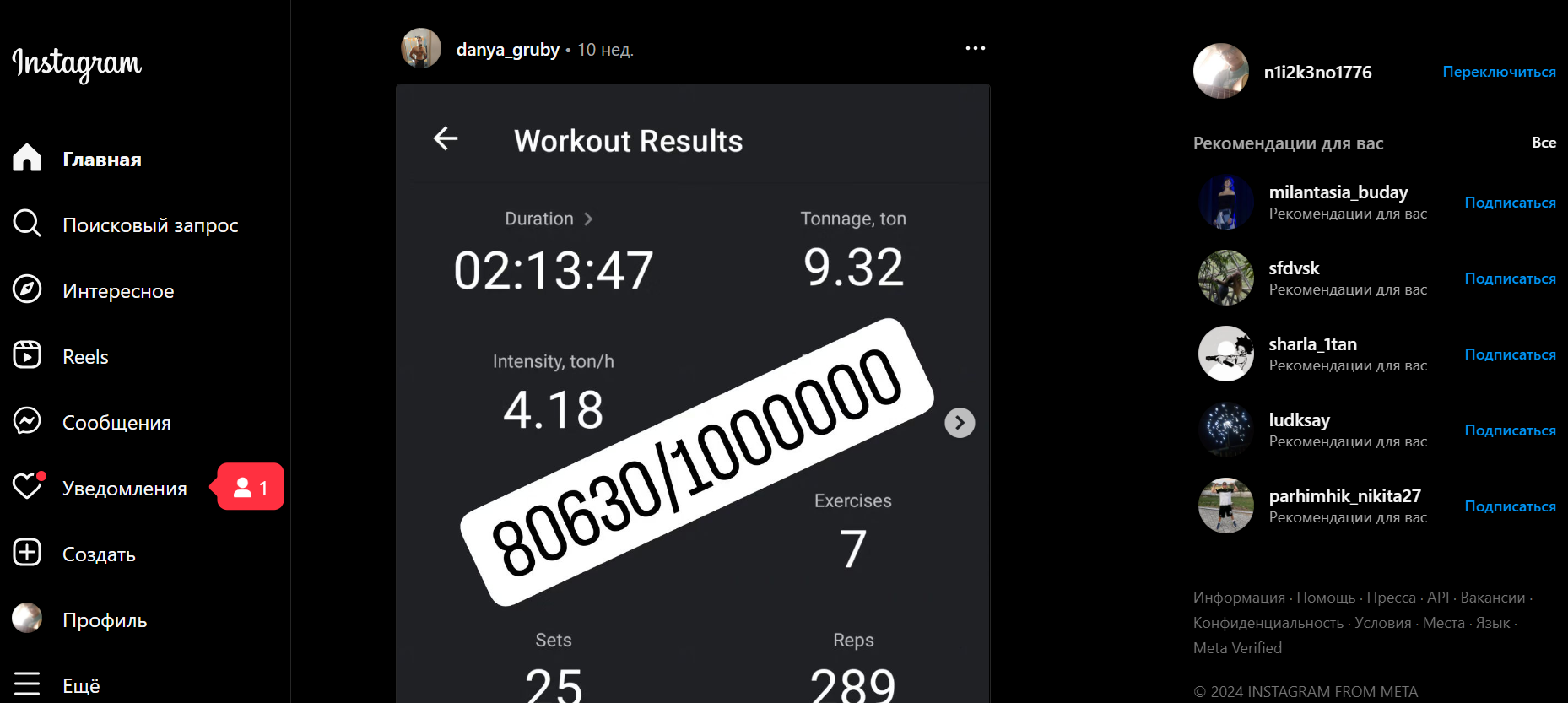
Социальная сеть, о которой идет речь, является одной из самых популярных и востребованных на сегодняшний день. В ней особое место занимают группы, которые играют важную роль в жизни пользователей. Данная платформа предоставляет множество возможностей для общения и взаимодействия, позволяя людям создавать группы по интересам, находить новых друзей и поддерживать связь с уже знакомыми людьми.

Одной из ключевых функций этого приложения является создание аккаунта, что позволяет пользователю начать свой путь в этом цифровом сообществе. Создав аккаунт, пользователи могут вступать в уже существующие группы или создавать свои собственные, где они могут делиться информацией, обсуждать общие интересы и участвовать в различных активностях.

Кроме того, приложение предоставляет обширные возможности для общения с друзьями. Пользователи могут отправлять друг другу сообщения, делиться фотографиями и видео. Интерфейс приложения интуитивно понятен и удобен, что делает общение еще более приятным и простым.

Таким образом, это приложение предоставляет все необходимые инструменты для комфортного и разнообразного общения, объединяя людей и помогая им находить единомышленников.

<https://www.instagram.com/>

****

Изображение №1.2 “приложение instagram”

**Преимуществами ресурса:**

* бесплатные сообщения и звонки: instagram позволяет пользователям отправлять текстовые сообщения, совершать голосовые и видеозвонки бесплатно через интернет, что делает его доступным и экономичным средством связи;
* широкий функционал: Приложение предлагает множество функций, включая обмен мультимедийными файлами, стикеры, групповые чаты, а также возможность отправки документов и GIF-анимаций. Это делает общение разнообразным и интересным;
* кроссплатформенность: instagram доступен на различных платформах, включая iOS, Android, Windows и Mac, что позволяет пользователям легко общаться независимо от используемого устройства.

**Недостатки:**

* использование ресурсов: Приложение может потреблять значительное количество оперативной памяти и батареи, особенно при активном использовании видеозвонков и мультимедийных функций;
* интерфейс: Некоторые пользователи могут находить интерфейс instagram. менее интуитивно понятным по сравнению с конкурентами, что может требовать времени на привыкание и освоение всех функций.
  1. Постановка задачи

После анализа аналогичного решения, можно выделить ряд основных задач, которые должно решать программное средство.

Основные задачи:

* регистрация и удаление аккаунта;
* возможность добавлять и удалять друзей;
* возможность создавать, удалять, вступать и покидать группы;
* возможность отправки сообщений в группы и в личку;
* возможность изменение информации;
* создание удобной навигации;
* спокойный неперегруженный дизайн.

При разработке программного обеспечения крайне важно принимать во внимание все требования и пожелания пользователей. Это необходимо для того, чтобы конечный продукт соответствовал их ожиданиям и решал те задачи, которые перед ним стоят. Понимание потребностей пользователей позволяет разработчикам создавать более востребованные и полезные программные средства.

Кроме того, следует использовать современные технологии и методы программирования. Это гарантирует, что разрабатываемое программное обеспечение будет соответствовать актуальным стандартам индустрии и сможет эффективно функционировать на различных платформах и устройствах. Применение передовых технологий также помогает повысить производительность и надежность программного средства.

Не менее важным аспектом является безопасность использования программного средства. Защита данных пользователей и предотвращение несанкционированного доступа к информации должны быть приоритетными задачами при разработке. Современные методы шифрования, аутентификации и другие технологии безопасности позволяют защитить данные и обеспечить надежность работы программного обеспечения.

1.3 Вывод к разделу

В конечном итоге, когда будут выполнены все поставленные задачи, должно получиться удобное и функциональное приложение, предназначенное для обмена информацией и общения с другими любителями кошек. Это приложение будет поддерживать несколько важных функций, которые значительно упростят взаимодействие между пользователями.

Прежде всего, приложение будет предоставлять возможность хранения информации о каждом пользователе. Это включает в себя создание профиля, где можно указать личные данные и загрузить фотографии.

Кроме того, приложение предложит функцию заведения друзей. Пользователи смогут находить и добавлять в друзья других любителей кошек, с которыми у них общие интересы. Это позволит создавать личные связи и расширять круг общения, находя единомышленников и людей, с которыми можно обмениваться опытом.

Одной из ключевых функций также станет возможность создания групп. Пользователи смогут объединяться в тематические группы по интересам.

Таким образом, конечная цель создания этого приложения заключается в том, чтобы предложить любителям кошек удобную платформу для общения, обмена опытом и поддержания связей с другими кото любами.

2. Анализ требований к программному средству и разработка функциональных требований

Анализ требований – это процесс понимания, документирования, оценки и управления требованиями к системе. Целью анализа требований является создание четкого и понятного описания того, что должна делать система, каким образом и какие требования необходимы для того, чтобы система была полезной и эффективной для пользователей и бизнеса.

Анализ требований включает в себя такие шаги, как сбор и документирование требований, анализ требований на противоречивость, неполноту и загрузку, разработка модели требований, их оценка и проверка, управление требованиями в течение жизненного цикла проекта. Он является критическим и необходимым этапом в разработке любой системы, поскольку он устанавливает основу для дальнейшей работы разработчиков и гарантирует, что система будет соответствовать требованиям пользователей и бизнеса и будет успешно принята ими.

2.1 Спецификация функциональных требований к программному средству

Для выявления функциональных требований были определены следующие роли и группы пользователей:

* администратор;
* пользователь.

Программное средство должно предоставлять следующие функциональные возможности:

Возможности для пользователя:

* создать свой аккаунт;
* изменение профиля аккаунта(обновление информации);
* удаление аккаунта;
* отправить запрос на дружбу(поиск по id или никнейму);
* отвечать на запросы;
* отправлять сообщения;
* удалить сообщение;
* просмотр аккаунта друзей;
* удалить друга;
* подать заявку на вступление в группу;
* покинуть группу(в которой вы состояли);
* создавать группу;
* изменение информации о группе;
* удаление группы;
* бан человека из группы.

Возможности для Администратора:

* бан аккаунта за несоблюдение правил;
* просмотр всех аккаунтов и групп;
* отправка письма с предупреждением о нарушении;
* принудительное удаление группы;
* работа с БД.

В приложении А представлена диаграмма использования программного обеспечения

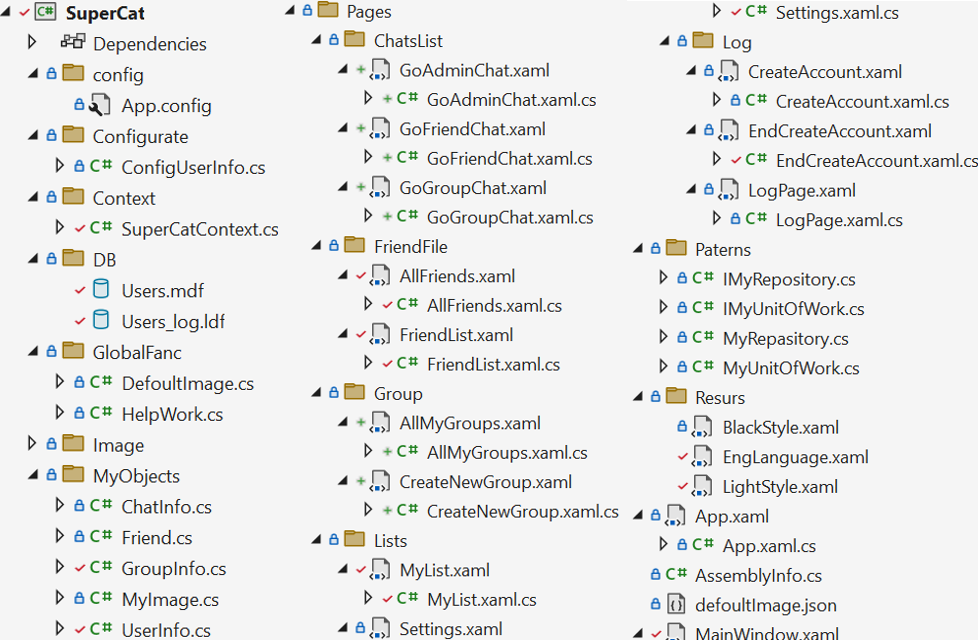
2.2 Вывод к разделу

В данной главе были определены роли и группы пользователей, а также представлена диаграмма использования программного обеспечения разными группами пользователей. В целом, спецификация функциональных требований к программному средству позволяет определить функциональность и ожидания пользователей, а также задает рамки для разработки и тестирования программного продукта.

1. Проектирование программного средства

3.1 Архитектура системы

Структура социальной сети представлена на изображении №3



Изображение №3.1-“Структура проекта”

Все оснавные папки храняться в “Page” .Описание структуры основных папок проекта представлено в таблице 3.1.

Таблица 3.1 - Описание структуры основных папок проекта

|  |  |
| --- | --- |
| Название | Содержание |
| Context | Содержит строки контектса для работы с БД |
| config | Содержит конфигурационную инфорацию для работы с БД |
| DB | Содержит таблицы базы данных |
| GlobalFanc | Содержит статические методы для работы приложения |
| Image | Содержит иконки которые используються в приложении |
| MyObjects | Содержит классы-модели которые повторяют все поля для каждой таблицы БД |
| Pages | Содержит все файлыдля работы со страницами |
| ChatsList | Содержит файлы для работы с часами |
| FriendFile | Содержит файлы для работы с друзьями |
| Group | Содержит файлы для работы с группами |
| Lists | Содержит файлы для работы с личной инфорацией |
| Log | Содержит файлы для регистарции или авторизации |
| Paterns | Содержит патерны репозиторий и Unit of work |
| Resurs | Содержит файлы ресурсов языка и стиля |

Эта таблица помогает понять структуру и устройство моего приложения

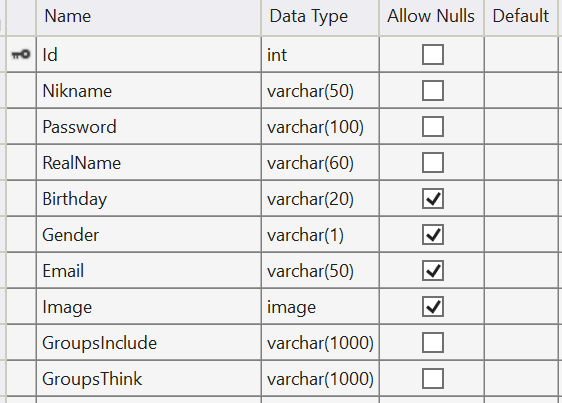
* 1. Взаимоотношения между классами

Структура классов описана в виде диаграммы классов Диаграмма классов представлена в Приложении Б.

* 1. Модель базы данных

В программе была создана база данных Users.mdf, и 5 таблиц которые хранят в себе всю информацию необходимую для корректной работы приложения.

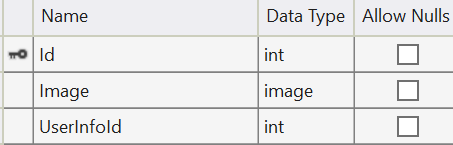
На изображении №3.2 представлена таблица UsersInfo



Изображение №3.2 – “Таблица UsersInfo”

Описание полей: Id – обязательный уникальный индекс каждого пользователя, Nickname – обязательный уникальный никнейм каждого пользователя, Password – обязательная хешированная строка дающая доступ к аккаунтам, Realname – обязательное неуникальное имя, Birthday – необязательная строка хранящая день рождения пользователя, Gender – необязательный символ который может быть только ‘m’, ‘w’ или ‘null’, Email – необязательная строка хранящая корректно записанный адрес электронной почты, Image – обязательный набор битов описывающий картинку которая является главной иконкой пользователя, GroupsInclude – обязательная строка хранящая id всех групп куда мы вступили, GroupsThink - обязательная строка хранящая id всех групп куда мы подали заявку.

На изображении №3.3 представлена таблица MyImages.

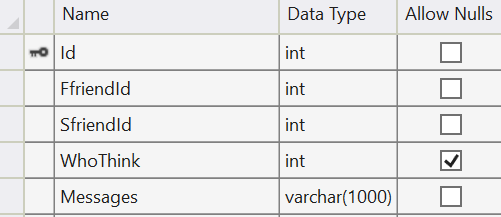
****

Изображение №3.3 – “таблица MyImages”

Описание полей: Id - обязательный уникальный индекс каждой картинки,

Image – обязательная битовая последовательность описывающая хранимую картинку, UserInfoId – обязательный индекс указывающий на владельца изображения.

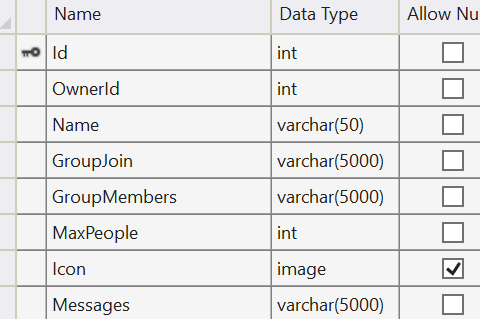
На изображении №3.4 представлена таблица Friends.



Изображение №3.4 – “таблица Friends”

Описание полей: Id - обязательный уникальный индекс каждой дружбы, FfriendId – обязательный индикатор пользователя который предложил дружбу, SfriendId - обязательный индикатор пользователя который получил запрос на дружбу, WhoThink – необязательное поле показывающее согласен ли второй пользователь на дружбу если null значит запрос одобрен, если не нал значит обдумывается, если дружба отклонена то связь удаляется первоначально для каждого пользователя есть один друг admin которого нельзя удалить, он предоставляет механизм обратной связи. Messages – строка хранящая все номера сообщений в группе

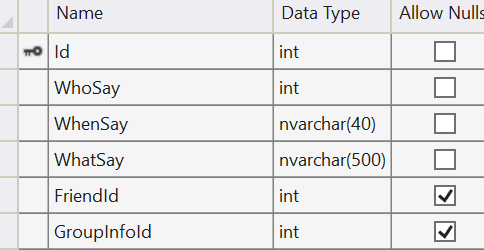
На изображении №3.5 представлена таблица GroupsInfo.



Изображении №3.5 – “таблица GroupsInfo”

Описание полей: Id - обязательный уникальный индекс каждой группы, OwneriD – обязательный индекс пользователя, создавшего эту группу(при удалении аккаунта группа также удаляеться), Name – обязательное название группы (названия могут повторяться но только тогда когда их создали разные пользователи), GroupJoin – обязательный список id тех пользователей, которые подали заявку на вступление в группу, но еще не получили ответ, GroupMembers – обязательный список id тех пользователей, которые подали заявку и вступили в группу, MaxPeople – обязательное максимальное количество участников группы от 3 до 15 (задаться при создании группы и может быть изменено в бедующем), Icon – иконка группы в виде битовой последовательности. При создании группы в нее автоматически заноситься 2 участника, 1 – сам хозяин группы, 2 – невидимый вездесущий администратор, нужно это для того что бы администратор на правах члена группы мог ее просматривать и изменять для предотвращения неприятных ситуаций, оба этих участника владеют одинаковыми правами, а все остальные считаются второстепенными гостями. Messages – строка хранящая все номера сообщений в группе.

На изображении №3.6 представлена таблица ChatsInfo.



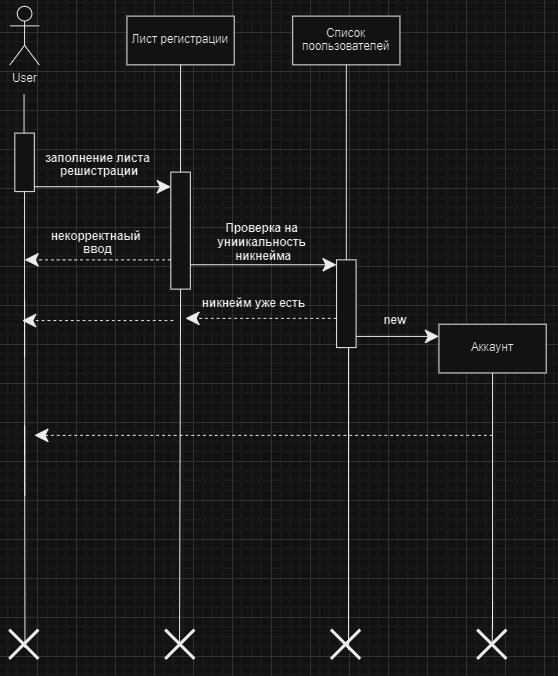
Изображении №3.6 – “таблица ChatsInfo”

Описание полей: Id - обязательный уникальный индекс каждого сообщения, WhoSay – индекс пользователя отправившего сообщение, WhenSay – время отправки сообщения, WhatSay – содержимое сообщения, FriendId и GroupId обязателен один из этих индексов, он будет указывать какой группе или каким пользователям принадлежит сообщение.

* 1. Проектирование последовательности взаимодействия

При проектировании последовательности взаимодействия объектов приложения была использована UML-диаграмма последовательности.

Диаграмма последовательности регистрации в приложении представлена на изображении №3.7.



Изображении №3.7 – “Диаграмма последовательности”

На диаграмме последовательности объекты представлены в виде вертикальных линий, а сообщения между объектами - в виде стрелок, направленных от одного объекта к другому. Каждое сообщение имеет свойство времени жизни, которое указывает, сколько времени занимает передача сообщения между объектами. Диаграмма последовательности позволяет описать не только последовательность сообщений между объектами, но и условия ветвления и циклов в процессе взаимодействия.

3.5 Вывод к разделу

Из представленных таблиц баз данных, структуры проекта, а также при внимательном рассмотрении диаграммы последовательности можно приблизительно понять, как работает программа внутри. Данная программа представляет собой социальную сеть, которая хранит различные типы информации о своих пользователях. Это включает в себя не только основные данные пользователей, но и их изображения, список друзей, группы, в которые они входят, а также сообщения, которые они отправляют и получают.

Таблицы баз данных демонстрируют, как структурирована и организована информация.

Диаграмма последовательности иллюстрирует пошаговый процесс выполнения операции регистрации внутри системы. Благодаря диаграмме мы видим работу определенного функционала программы и из каких блоков оно состоит. Каждый из этих этапов можно проследить на диаграмме, что помогает лучше понять внутреннюю логику работы программы.

Таким образом, рассмотрение таблиц баз данных, структуры проекта и диаграммы последовательности позволяет получить целостное представление о работе социальной сети. Это знание помогает разработчикам и аналитикам эффективно работать над улучшением и оптимизацией программы, обеспечивая ее надежность и удобство для пользователей.

1. Реализация программного средства
   1. Реализация паттернов репозиторий и Unit of Work

Реализация класса MyRepasitory представлена в листинге 4.1 – интерфейс, 4.2 – сам класс, 4.3 – пример использования.

using SuperCat.MyObjects;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SuperCat

{

public interface IMyRepository<T>

{

IEnumerable<T> GetAll();

T Get(int id);

void Create(T item);

void Update(T item);

void Delete(int id);

void Dek(int id);

}

}

Листинг 4.1 – “Интерфейс IMyRepasitory”

В этом интерфейсе описаны основные методы которые должен выполнять репозиторий.

using Microsoft.EntityFrameworkCore;

using Microsoft.VisualBasic.ApplicationServices;

using SuperCat.MyObjects;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SuperCat

{

public class MyRepository<T> : IMyRepository<T> where T : class

{

private readonly DbContext context;

private readonly DbSet<T> dbSet;

public MyRepository(DbContext context)

{

this.context = context;

dbSet = context.Set<T>();

}

public IEnumerable<T> GetAll()

{

return dbSet.AsNoTracking().ToList();

}

public T Get(int id)

{

return dbSet.Find(id) ?? dbSet.First();

}

public void Add(T item)

{

dbSet.Add(item);

}

public void Create(T item)

{

dbSet.Add(item);

context.SaveChanges();

}

public void Update(T item)

{

context.Entry(item).State = EntityState.Modified;

}

public void Updated(T item)

{

context.Entry(item).State = EntityState.Modified;

context.SaveChanges();

}

public void Delete(int id)

{

T item = dbSet.Find(id)!;

if (item != null)

dbSet.Remove(item);

context.SaveChanges();

}

public void Dek(int id)

{

T item = dbSet.Find(id)!;

if (item != null)

dbSet.Remove(item);

}

}

}

Листинг 4.2 – “Класс MyRepasitory”

В классе представлена реализация методов, которые мы ранее описали в интерфейсе. Эти методы позволят отделить некоторые операции над базой данных в отдельный класс в примере приведен код с использованием метода выдающие все элементы таблицы и последующей работы с полученными данными.

private UserInfo GetUser()

{

return rep.GetAll().Where(us => us.Nikname.Equals(niknameBox.Text)).ToList().First();

}

private void LogInAccount(object sender, RoutedEventArgs e)

{

try

{

string pass = GetPassword();

if (pass == "")

throw new Exception("Takih net");

if (!HelpWork.VerifyPassword(passwordBox.Password, pass))

throw new Exception("Ne tot parol");

NavigationService.Navigate(new MyList(GetUser()));

}

catch

{

ErrorLog.Visibility = Visibility.Visible;

}

}

Листинг 4.3 – “Пример использования MyRepasitory”

В этом примере мы наблюдаем за авторизацией где человек вводит свой никнейм, мы с помощью репозитория извлекаем и ищем аккаунт с подобным никнеймом и если такой был найден то проверяем введённый пароль а если не найден то выводит ошибку.

Также есть паттерн Unity of work, 4.4 – интерфейс, 4.5 – класс, 4.6 – пример реализации.

using Microsoft.VisualBasic.ApplicationServices;

using SuperCat.MyObjects;

namespace SuperCat.Paterns

{

internal interface IMyUnitOfWork:IDisposable

{

IMyRepository<UserInfo> UserRepository { get; }

void Save();

}

}

Листинг 4.4 – “интерфейс IMyUnitOfWork”

Описываем репозиторий и метод Save.

using Microsoft.EntityFrameworkCore;

using Microsoft.VisualBasic.ApplicationServices;

using SuperCat.MyObjects;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

namespace SuperCat.Paterns

{

internal class MyUnitOfWork : IMyUnitOfWork

{

private readonly DbContext \_context;

public MyRepository<UserInfo> userRepository;

private bool disposed = false;

public MyUnitOfWork(DbContext context)

{

\_context = context;

userRepository = new MyRepository<UserInfo>(\_context);

}

public IMyRepository<UserInfo> UserRepository

{

get

{

if (userRepository == null)

userRepository = new MyRepository<UserInfo>(\_context);

return userRepository;

}

}

public void Save()

{

\_context.SaveChanges();

}

public virtual void Dispose(bool disposing)

{

if (!this.disposed)

{

if (disposing)

{

\_context.Dispose();

}

}

this.disposed = true;

}

public void Dispose()

{

Dispose(true);

GC.SuppressFinalize(this);

}

}

}

Листинг 4.5 – “Класс MyUnitOfWork”

В классе мы реализуем метод и создаем все необходимые поля для работы патерна.

private void DoSave(Label block, TextBox box, Button button)

{

change = true;

work.userRepository.Update(user);

work.Save();

button.Visibility = Visibility.Visible;

block.Visibility = Visibility.Visible;

if (box != null)

{

block.Content = box.Text;

box.Visibility = Visibility.Hidden;

}

}

Листинг 4.6 – “Пример использования MyUnitOfWork”

В этом примере мы используем Unit of work для обновления элемента и таблицы БД и сохраняем эти изменения.

* 1. Реализация хеширования паролей

Хеширования пароля в моем проекте происходит с использованием солей и встроенной библиотеки Cryptography и методом Rfc2898DeriveBytes.

Листинг 4.7 – метод для хеширования пароля, Листинг 4.8 – метод для проверки пароля.

public static string HashPassword(string password)

{

byte[] salt;

new RNGCryptoServiceProvider().GetBytes(salt = new byte[16]);

var pbkdf2 = new Rfc2898DeriveBytes(password, salt, 10000);

byte[] hash = pbkdf2.GetBytes(20);

byte[] hashBytes = new byte[36];

Array.Copy(salt, 0, hashBytes, 0, 16);

Array.Copy(hash, 0, hashBytes, 16, 20);

string savedPasswordHash = Convert.ToBase64String(hashBytes);

return savedPasswordHash;

}

Листинг 4.7 – метод для хеширования пароля

В методе мы создаем соль – рандомно сгенерированную последовательность определенной длинны, после с помощью метода прогоняем полученный пароль через хеш функцию 10000 раз, в итоге мы соединяем соли определенной длинны и хеш определенной длинны тем самым получая пароль для взлома которого, надо знать, где соль, а где хеш, сколько символов соль, какой хеш функцией вычислялся хеш и сколько раз через нее проходили. В итоге получаем надежно сохранённый пароль надежный.

public static bool VerifyPassword(string enteredPassword, string storedHash)

{

byte[] hashBytes = Convert.FromBase64String(storedHash);

byte[] salt = new byte[16];

Array.Copy(hashBytes, 0, salt, 0, 16);

var pbkdf2 = new Rfc2898DeriveBytes(enteredPassword, salt, 10000);

byte[] hash = pbkdf2.GetBytes(20);

for (int i = 0; i < 20; i++)

if (hashBytes[i + 16] != hash[i])

return false;

return true;

}

Листинг 4.8 – метод для проверки пароля

Мы не переводим пароль назад в нормальный вид, мы хешируем новую строку. Это проще так как мы делаем то же самое что и при хешировании, но с уже составленной солью, которую мы берем из таблицы базы данных зная сколько символов она занимает, это безопаснее так как за хешированный код ни в одной операции не становиться известен. В итоге мы можем без рисков убедиться, что пароли совпадают или не совпадают не трогая хеш.

4.3 Вывод к разделу

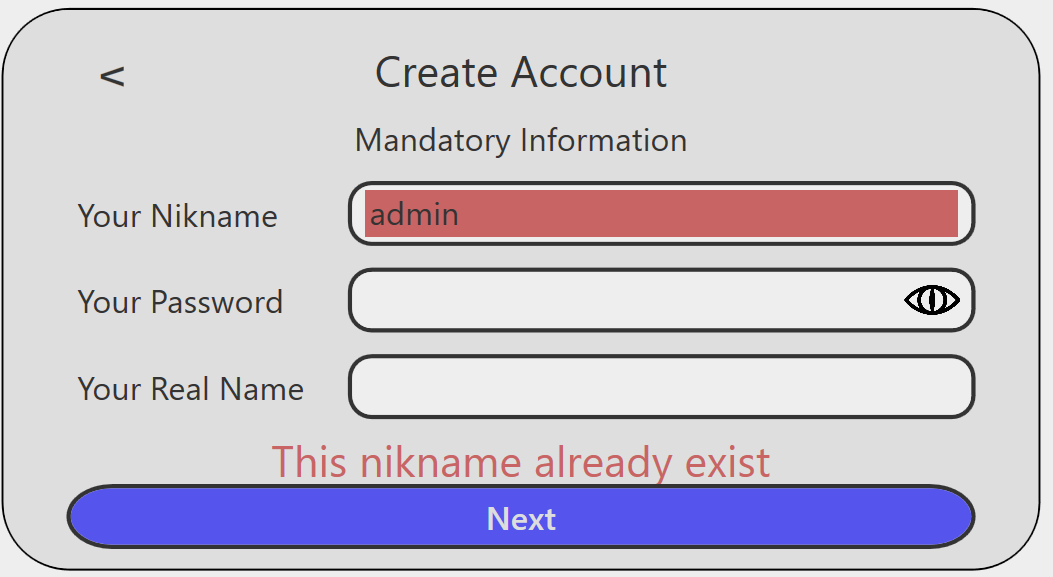
В Проекте использовались паттерны, которые упрощают понимание кода, делают программу проще и безопаснее. Это безусловно важные пункты, так как они помогут при разработке и чтении кода. Также хеширование является чуть ли не обязательной функцией в любом приложении, где есть аккаунты так как, например, при утечки информации из моего приложения, злоумышленники не смогут получать доступы к аккаунтам.

1. Тестирование, проверка работоспособности и анализ полученных результатов

5.1 Тестирование авторизации и регистрации

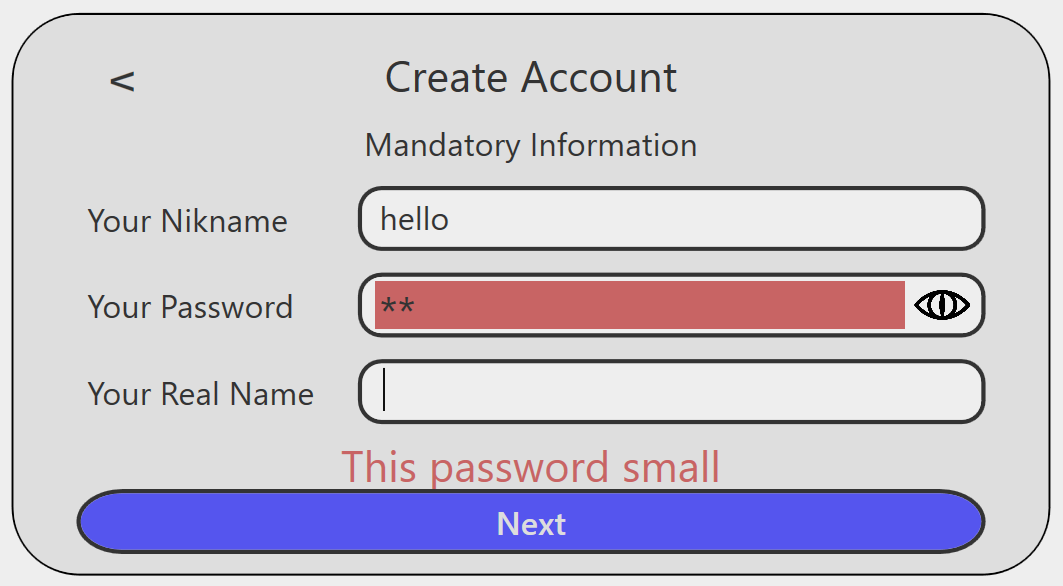
В момент авторизации и регистрации возможны ситуации, когда пользователь вводит некорректные данные, например, неверный пароль, незарегистрированный логин, некорректный адрес электронной почты и т.д. Такие исключения обрабатываются программным средством, оповещая пользователя о некорректных значениях.

На изображении 5.1 приведен пример обработки регистрации с уже существующим ником.



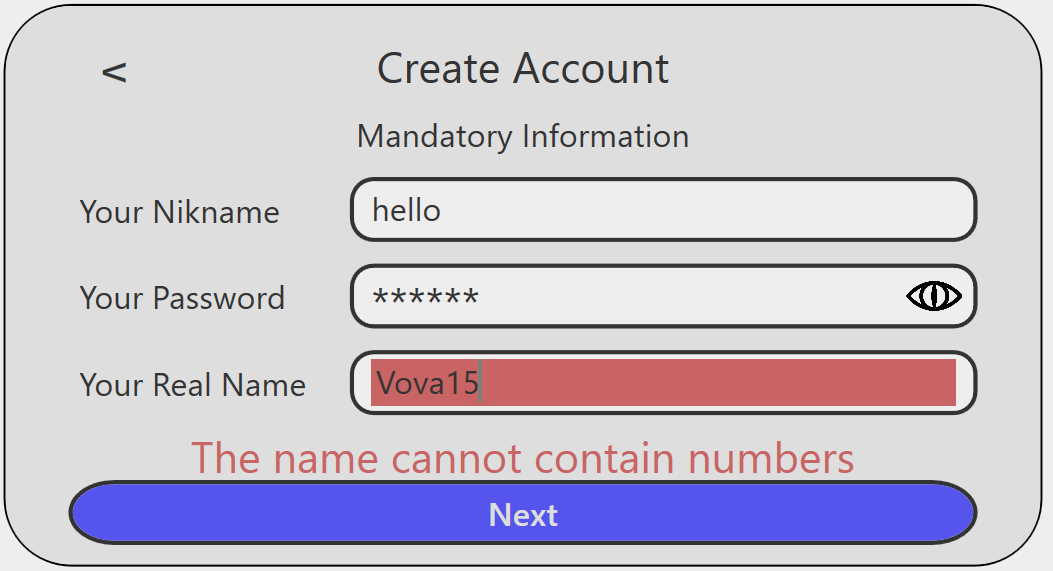
Изображение 5.1 – “предупреждения о некорректном никнейме”

На изображении 5.2 приведен пример обработки регистрации с некорректным паролем.



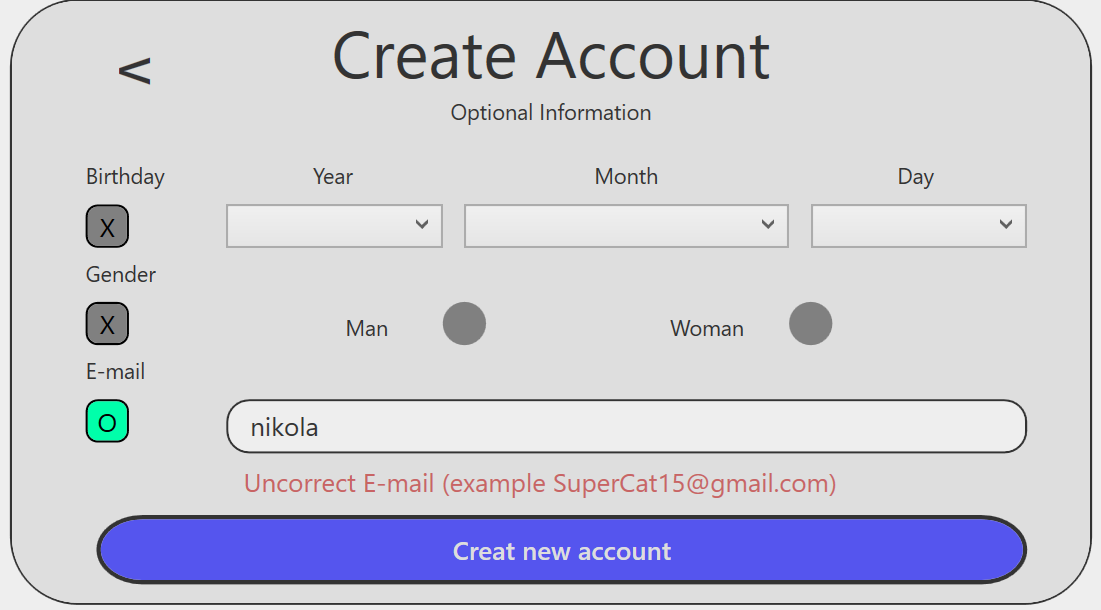
Изображение 5.2 – “предупреждения о некорректном пароле”

На изображении 5.3 приведен пример обработки регистрации с некорректным паролем.



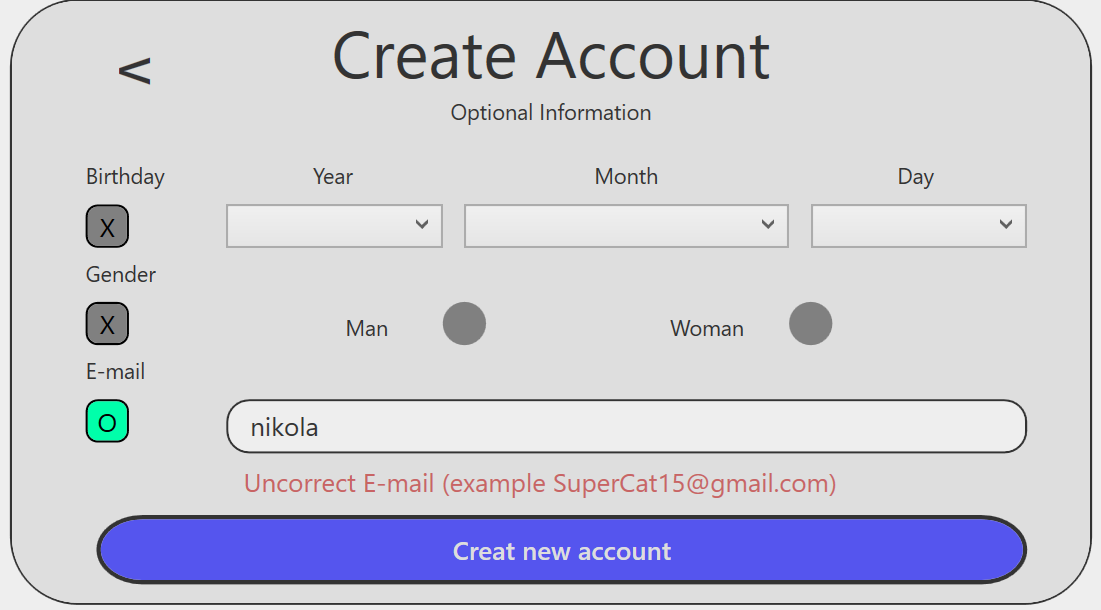
Изображение 5.3 – “предупреждения о некорректном реальном имени”

На изображении 5.4 приведен пример обработки регистрации с некорректным паролем.



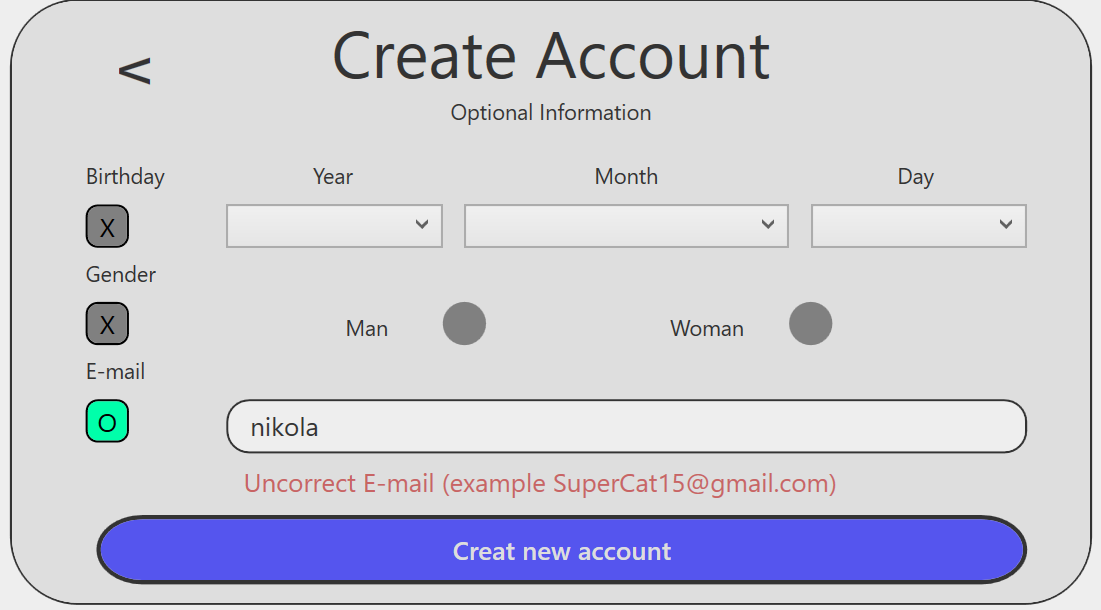
Изображение 5.4 – “предупреждения о некорректном адресе почты”

На изображении 5.5 приведен пример обработки авторизации без пароля.



Изображение 5.5 – “предупреждения о некорректной авторизации”

На изображении 5.6 приведен пример обработки авторизации без пароля.

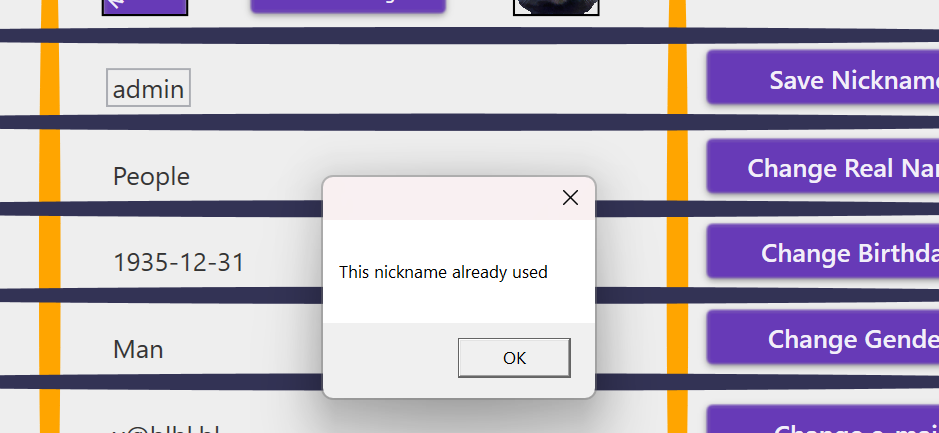


Изображение 5.6 – “предупреждения о некорректной авторизации”

В итоге все поля имеют валидацию и при возникновении проблем пользователь получает сообщение и может исправить то, что нужно.

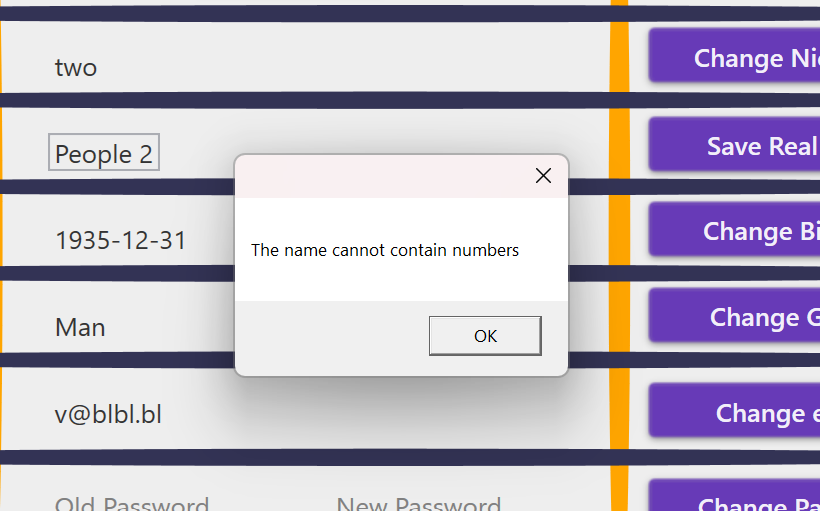
5.2 Тестирование изменения личной информации

На изображении 5.6 приведен пример обработки некорректного изменения никнейма в аккаунте.



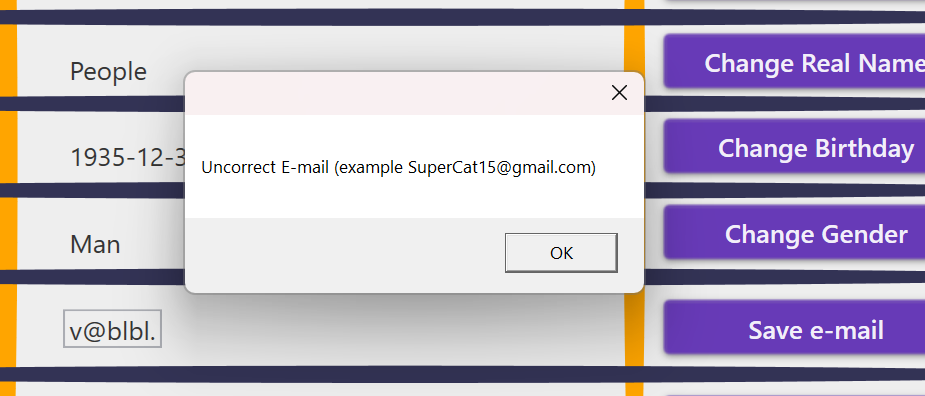
Изображение 5.6 – “предупреждения о некорректном изменении ника”

На изображении 5.7 приведен пример обработки некорректного изменения реального имени в аккаунте.



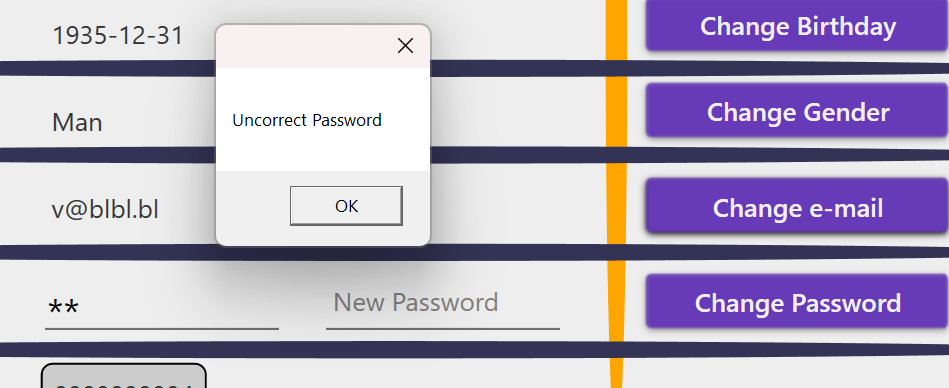
Изображение 5.7 – “предупреждения о некорректном реальном имени”

На изображении 5.8 приведен пример обработки некорректного изменения электронной почты.



Изображение 5.8 – “предупреждения о некорректной электронной почте”

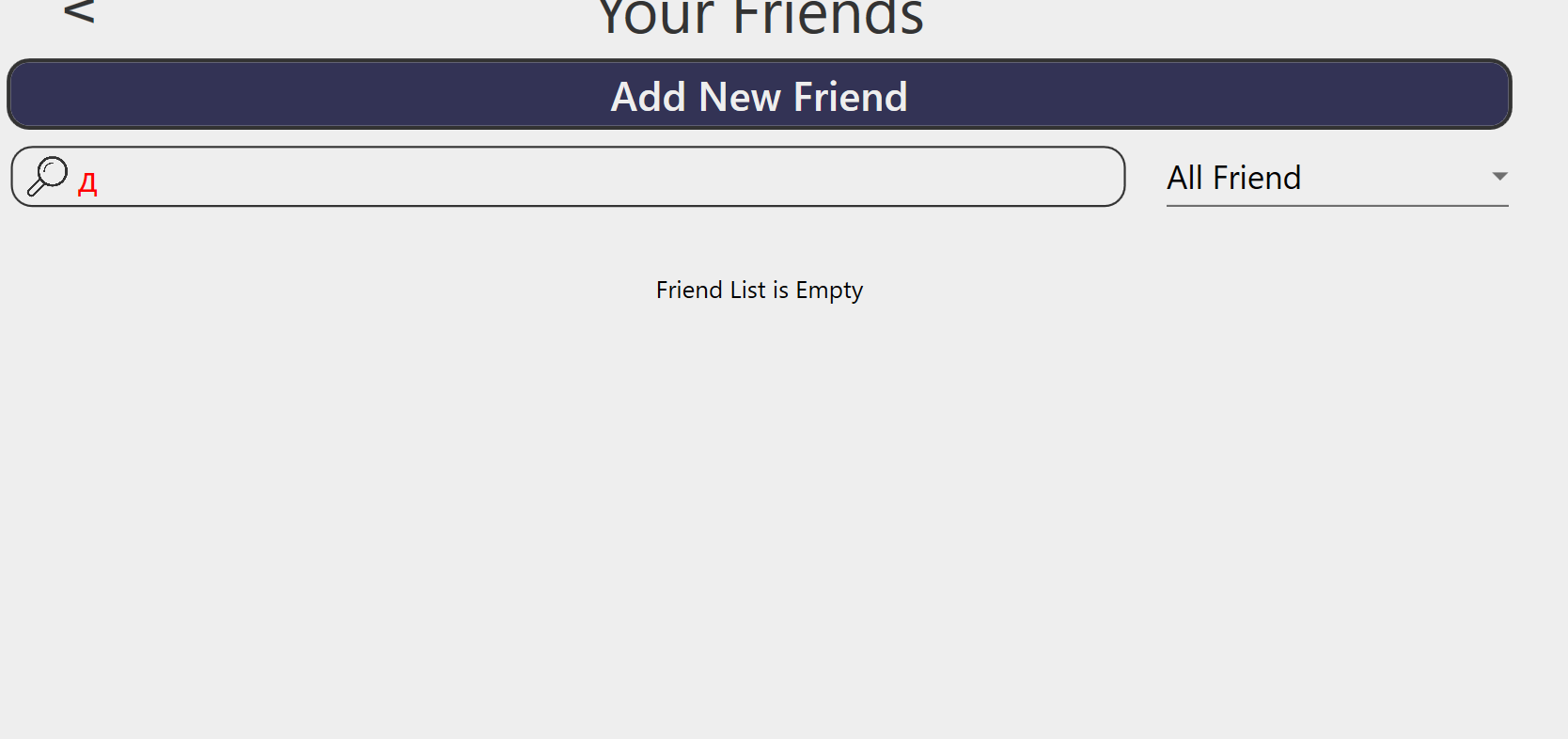
На изображении 5.9 приведен пример обработки некорректного изменения пароля.



Изображение 5.9 – “предупреждения о некорректном пароле”

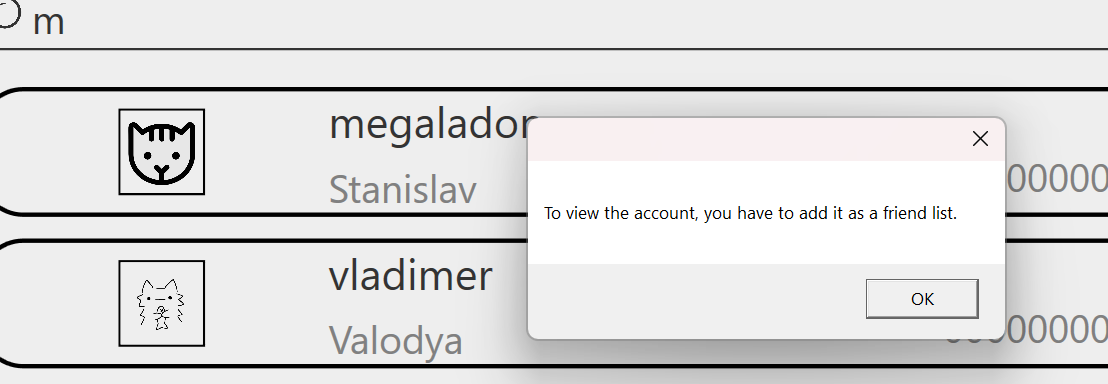
5.3 Тестирование списка друзей

На изображении 5.10 обработка поиска аккаунтов которых нету.



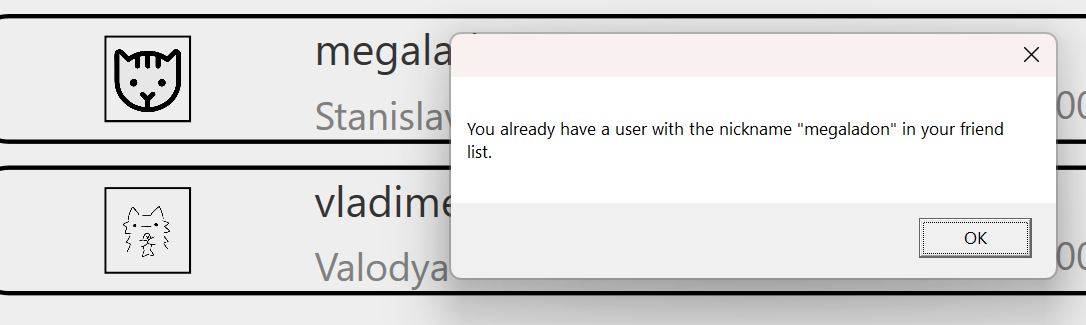
Изображение 5.10 – “вывод пустого списка”

На изображении 5.11 просмотр аккаунта которого нет в друзьях.



Изображение 5.11 – “предупреждения о невозможности просмотра”

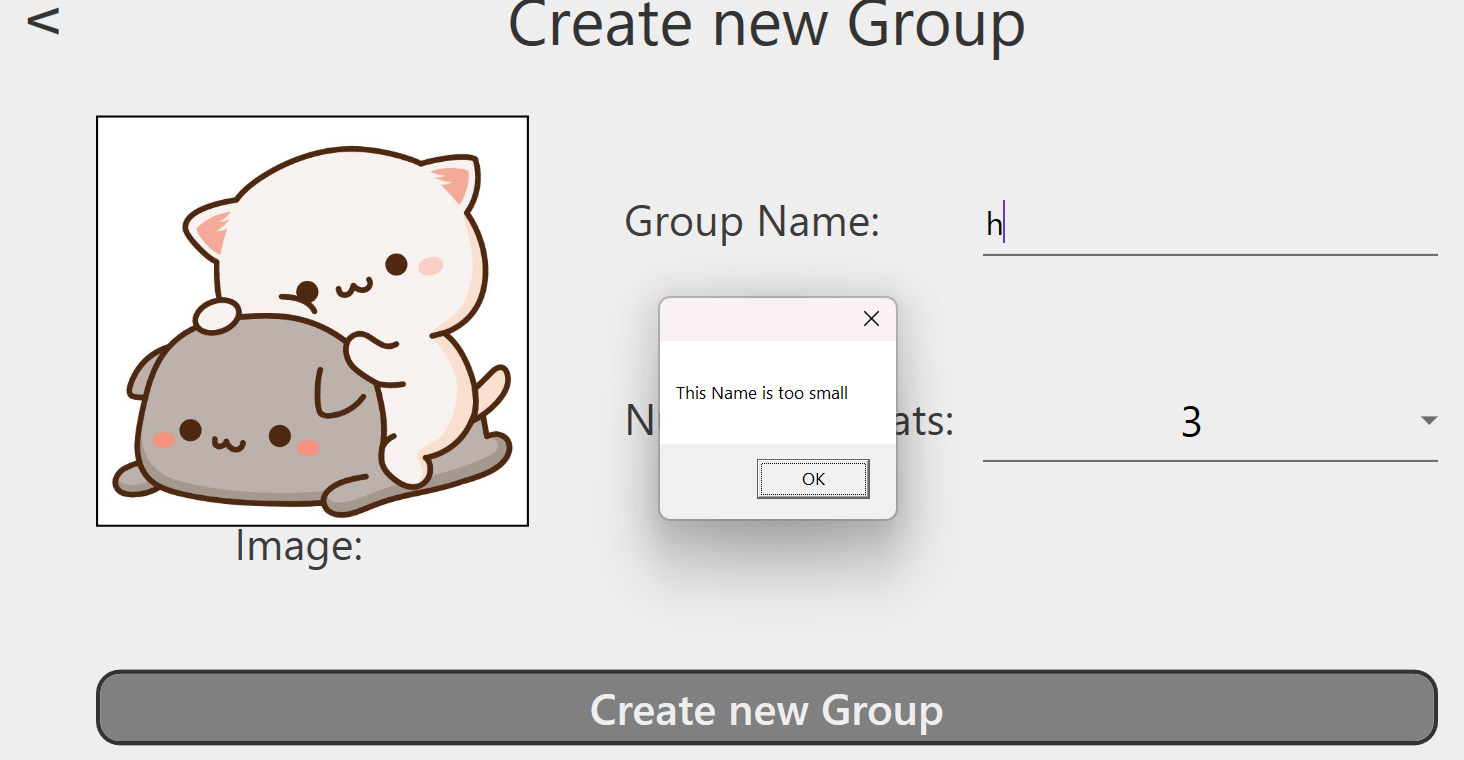
На изображении 5.12 попытка отправить запрос на дружбу другу.



Изображение 5.12 – “предупреждения о невозможности отправки запроса”

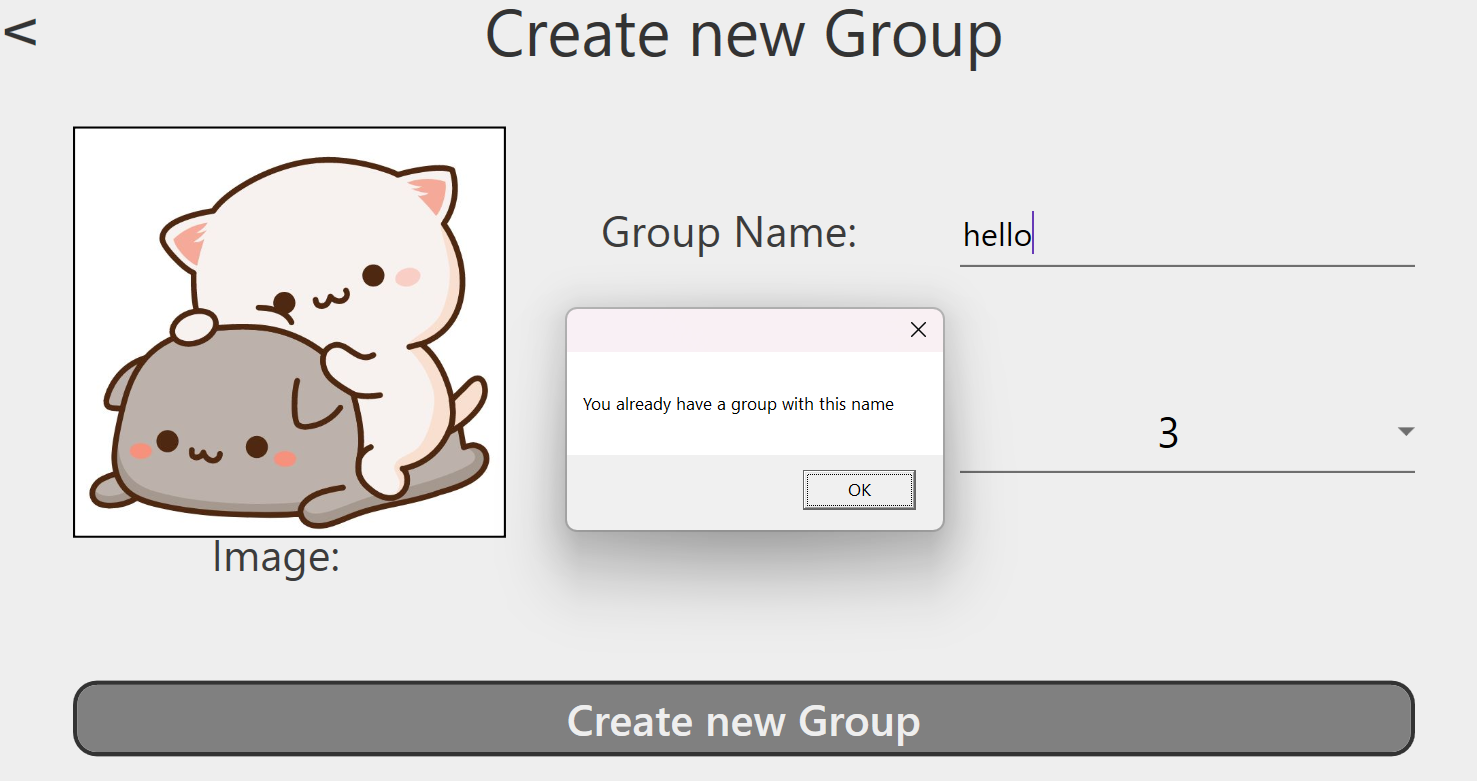
5.4 Тестирование списка групп

На изображении 5.13 попытка создать группу с некорректным именем.



Изображение 5.13 – “предупреждения о некорректном имени”

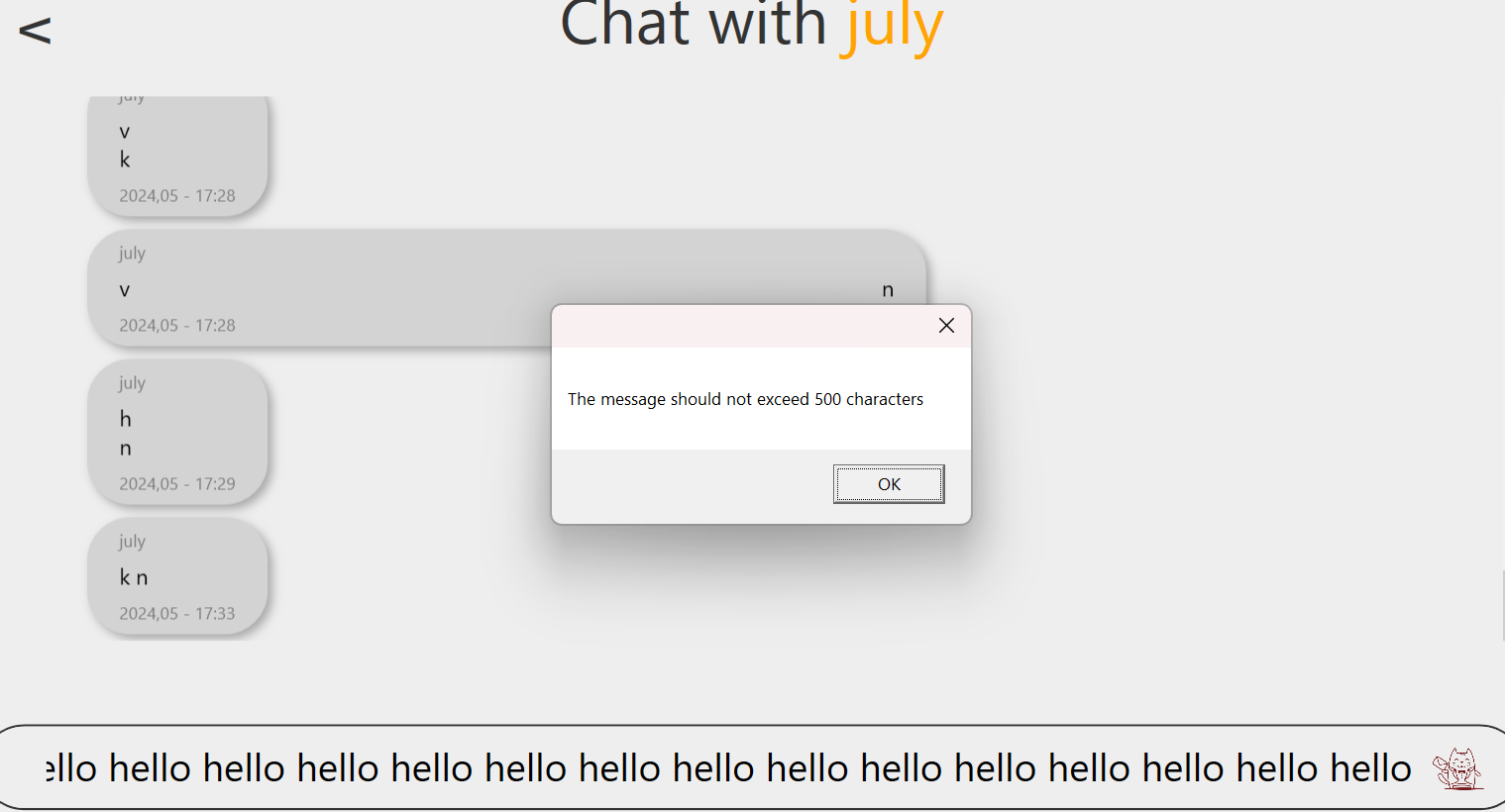
На изображении 5.13 попытка создать группу с уже существующим именем именем.



Изображение 5.13 – “предупреждения о повторном имени”

5.5 Тестирование чата

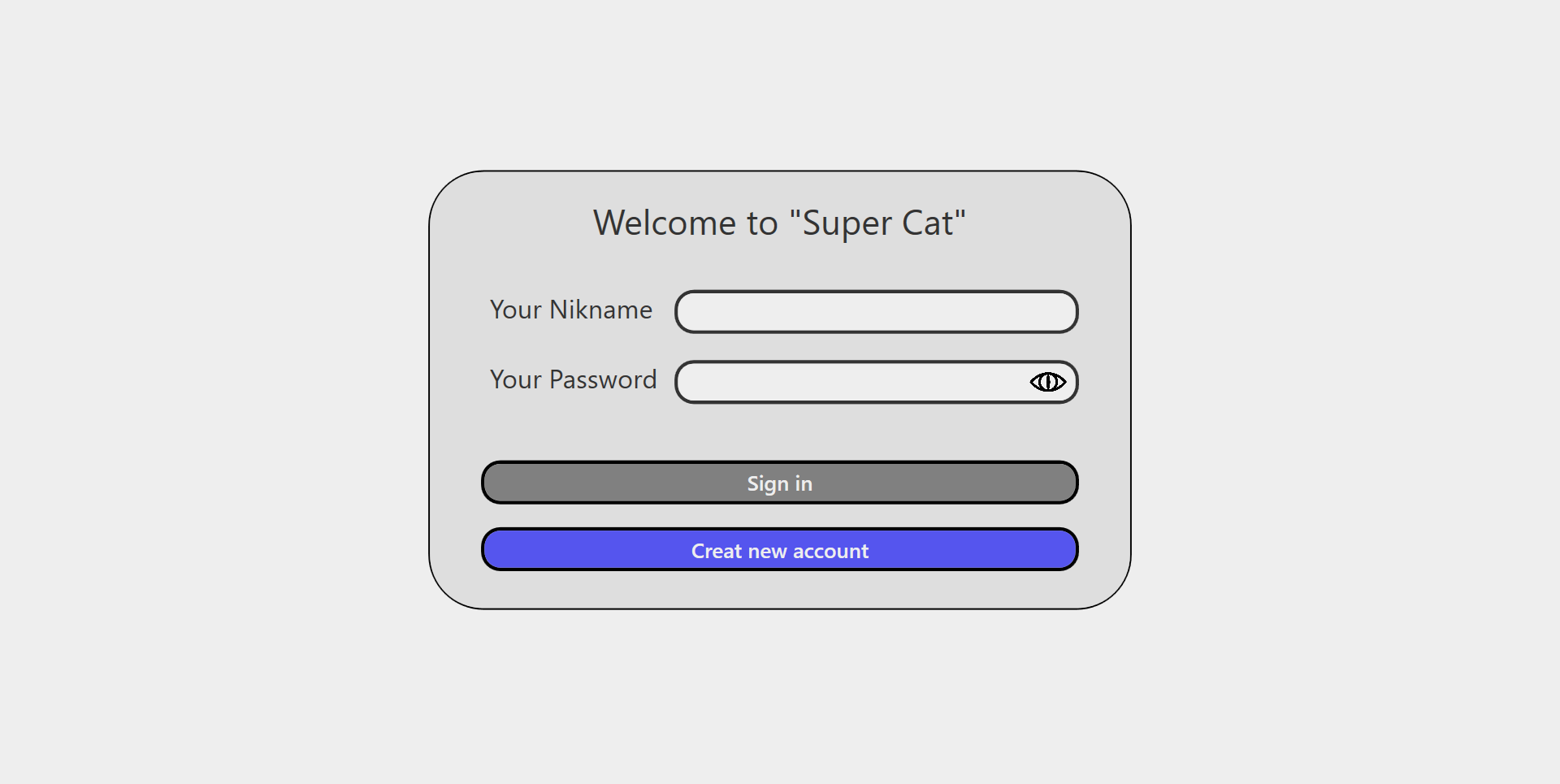
На изображении 5.13 попытка отправить слишком большое сообщение.



Изображение 5.13 – “предупреждения о повторном имени”

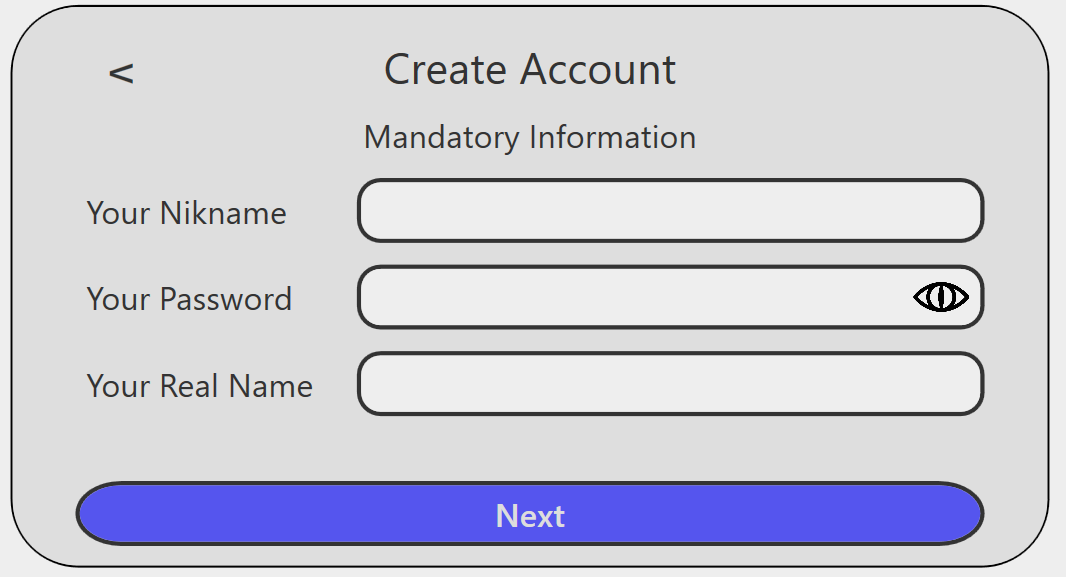
1. Руководство по установке и использованию

При запуске программы “Super cat” нас приветствует окно авторизации в которой есть поля Nickname и Password которые можно ввести если у нас есть аккаунт, при введении информации раз блокируется кнопка войти при нажатии на которую идет поиск аккаунта в базе данных, если у нас нету аккаунта то на странице есть кнопка регистрации при нажатии на которую мы можем создать аккаунт. Лист авторизации на изображении №6.1.



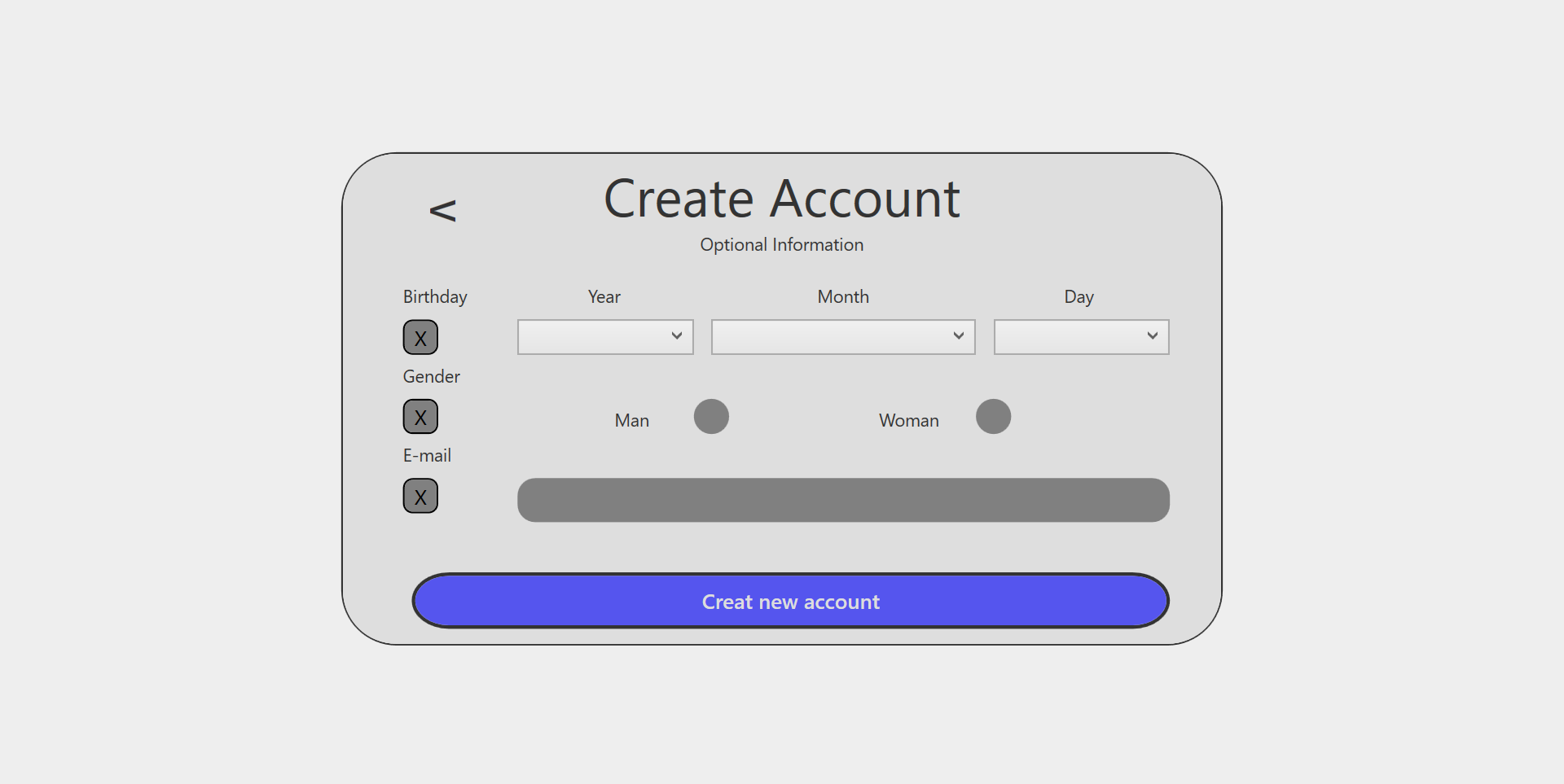
Изображение №6.1 – “Страница авторизации”

Регистрация разбита на два блока. Первый – обязательные параметры, на которые мы попадаем при регистрации, и там мы должны вписать такие параметры как Nickname, Password и Real Name. Все параметры имеют различные ограничения поэтому мы не можем продолжить регистрацию пока не введем корректную информацию. В случае некорректной информации нам выдаст соответствующее ошибке предупреждение, если же все введено правильно, то мы можем перейти к необязательном элементам, так же есть возможность вернуться к авторизации. Лист обязательных элементов для регистрации на изображении №6.2



Изображение №6.2 – “Страница обязательной регистрации”

После введения корректных данных мы переходим к необязательным элементам таким как Birthday, Gender и Email. Мы сами выбираем какие поля мы хотим заполнить есть проверка на корректность введенного e-mail. Мы можем как просто создать аккаунт, так и вернуться к обязательным элементам. Лист необязательных элементов для регистрации на изображении №6.3.



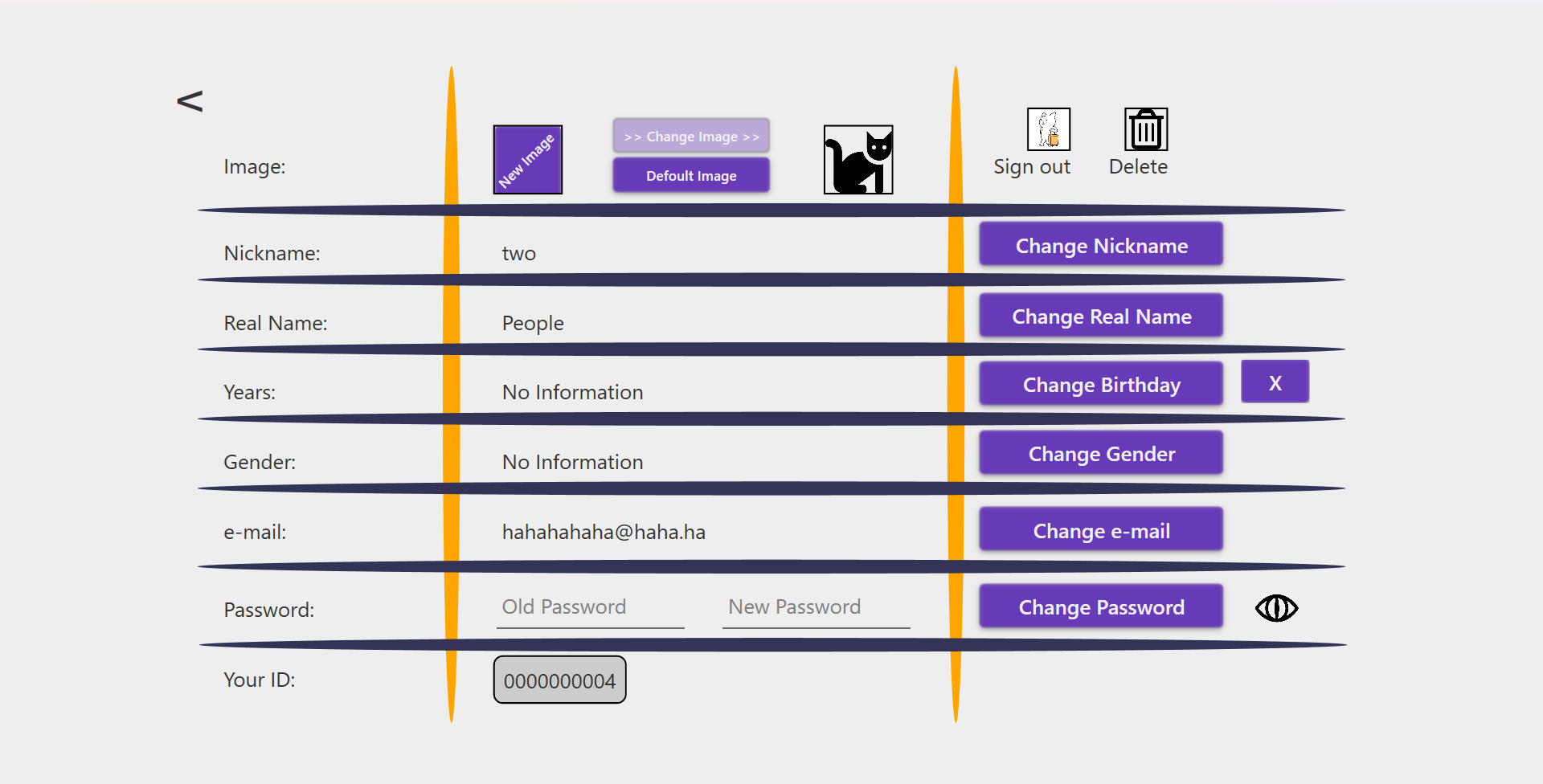
Изображении №6.3 – “Страница необязательных элементов”

После авторизации или регистрации мы попадаем на страницу аккаунта, где мы можем перейти в настройки, добавить изображение, просмотреть друзей и группы. Эту страницу можно будет изменить через настройки, а также на эту страницу будут попадать друзья, но друзья не будут иметь возможность изменять аккаунт. Изменения аккаунта доступно только хозяину и админу, но админ не может добавлять картинки. Лист информации аккаунта пользователя на изображении №6.4.



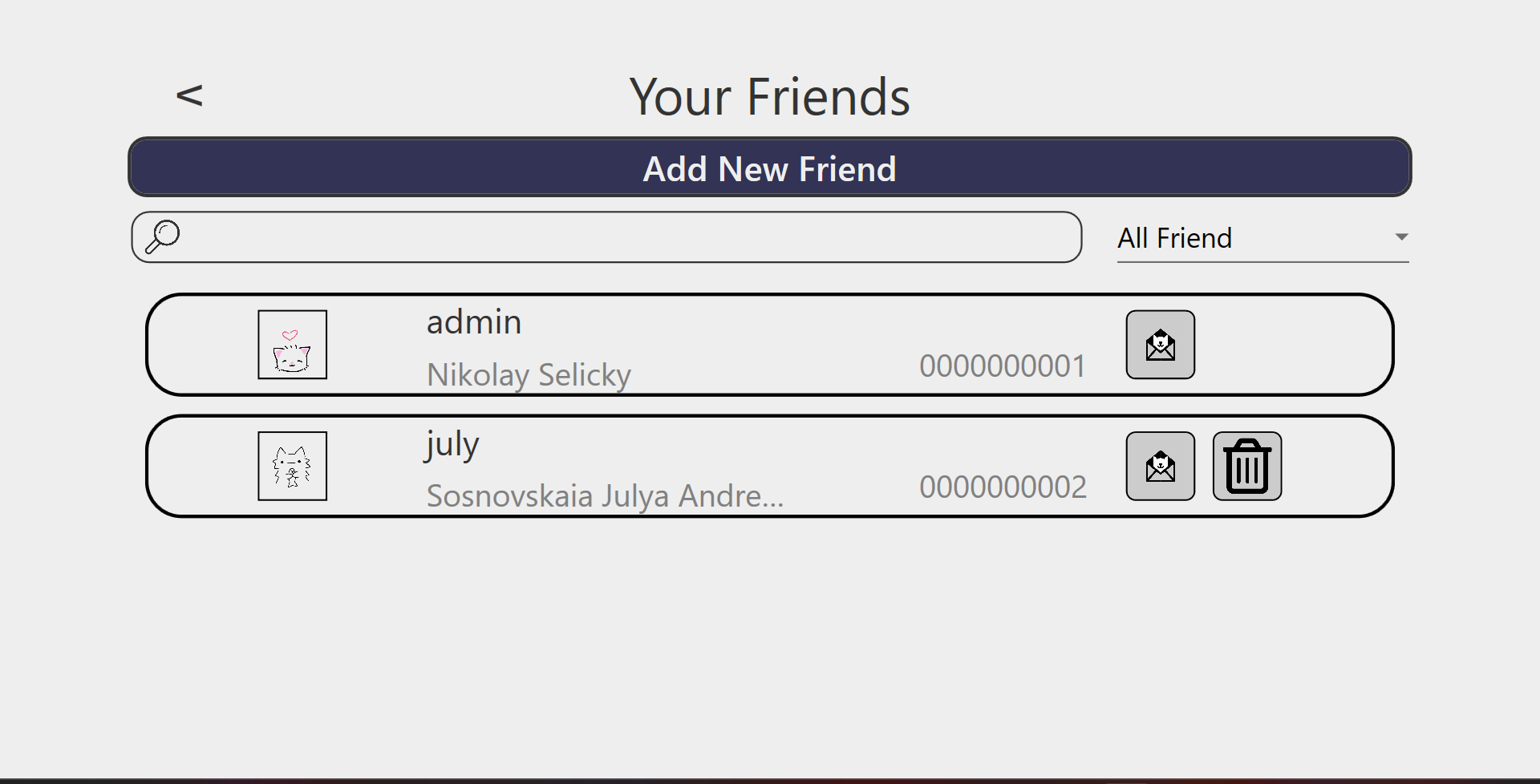
Изображение №6.4 – “Лист аккаунта”

При нажатии на шестеренку мы попадаем на страницу настройки, где мы можем менять все поля, каждое изменение должны быть валидно мы можем поменять иконку на дефолтную или выбрать из компьютера. Для изменения пароля нужно повторно ввести пароль, так же можно посмотреть свой Id. Лист настройки аккаунта пользователя на изображении №6.5.

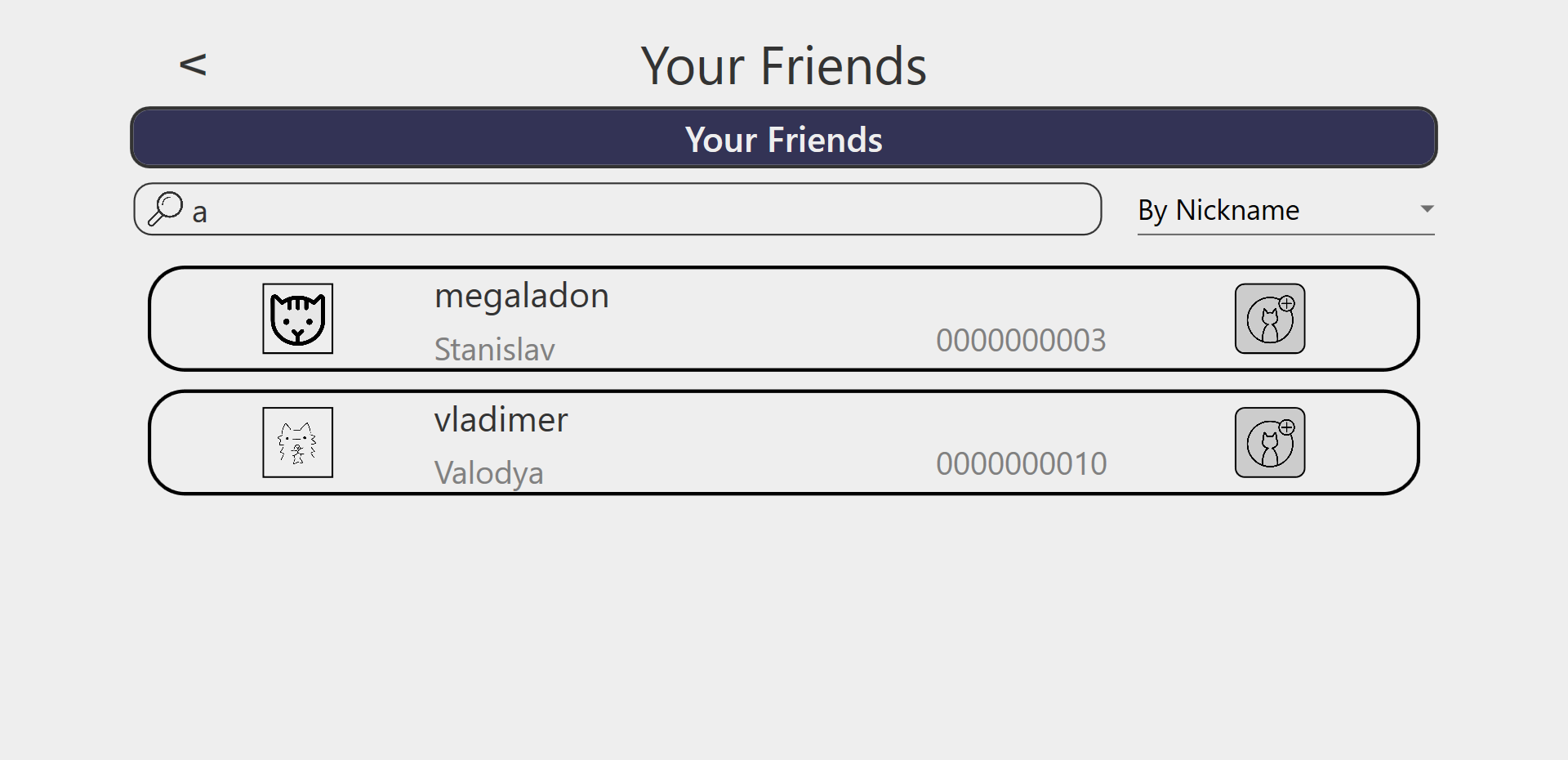


Изображение №6.5 – “Окно настроек”

Из страницы аккаунта мы можем просмотреть друзей, там отображается все друзья, которым мы отправляли запрос либо когда нам отправили запрос. Мы можем отфильтровать только те на которые мы отправили запрос, которые нам отправили запрос, те кто подтвердили запрос и по всем друзьям. Так же есть поле для поиска и если ввести пустую строку то выдаст всех друзей, а если что то ввести то найдет всех у кого в имени есть эта подстрока. Так же есть кнопка для добавления новых друзей. Также у всех аккаунтах в друзьях есть администратор, которому можно отправить жалобы и предложения. Лист списка друзей на изображении №6.6 и лист добавления друзей на изображении №6.7.



Изображение №6.6 – “Окно списков друзей”

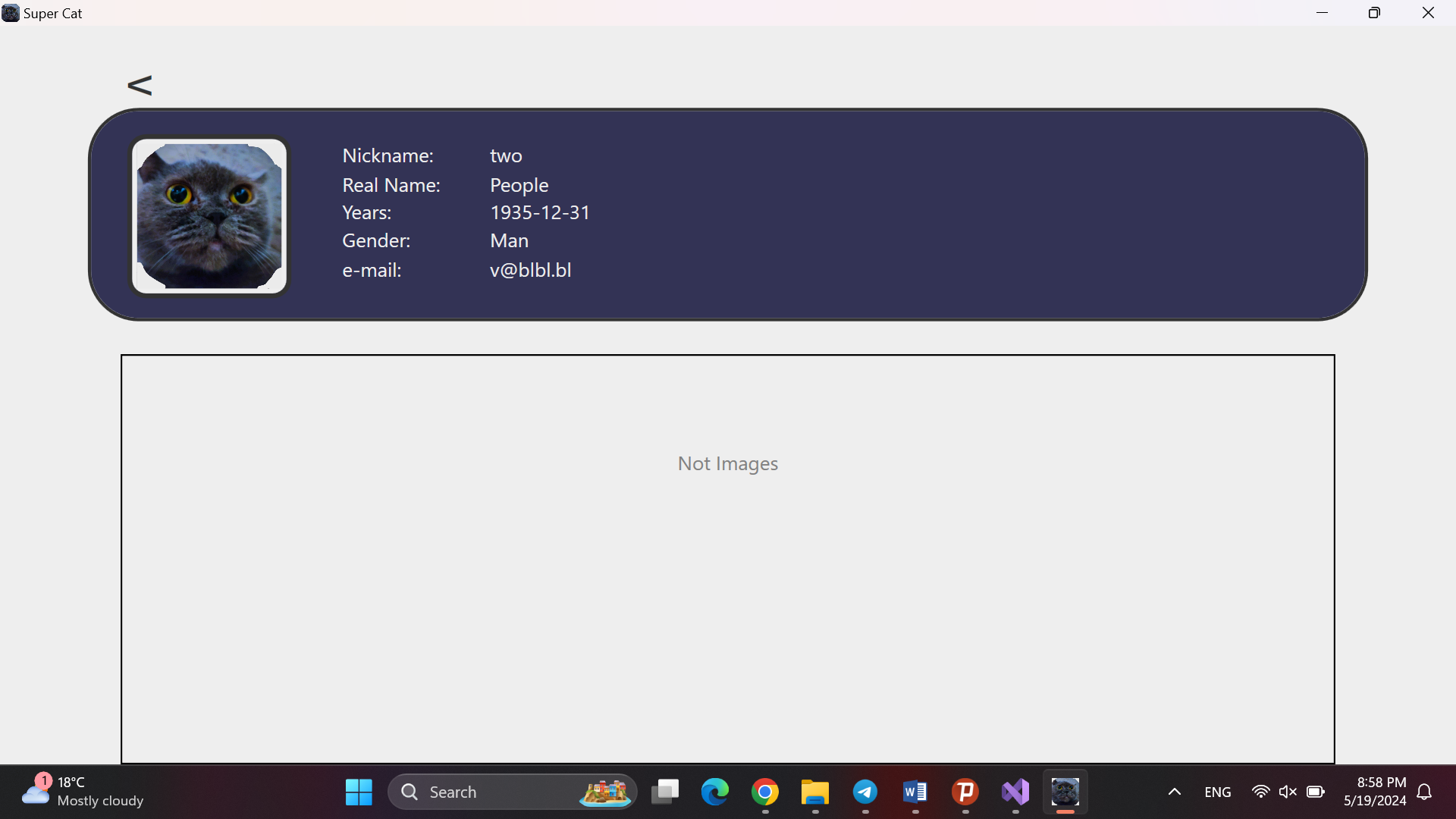


Изображение №6.7 – “Окно добавления друзей”

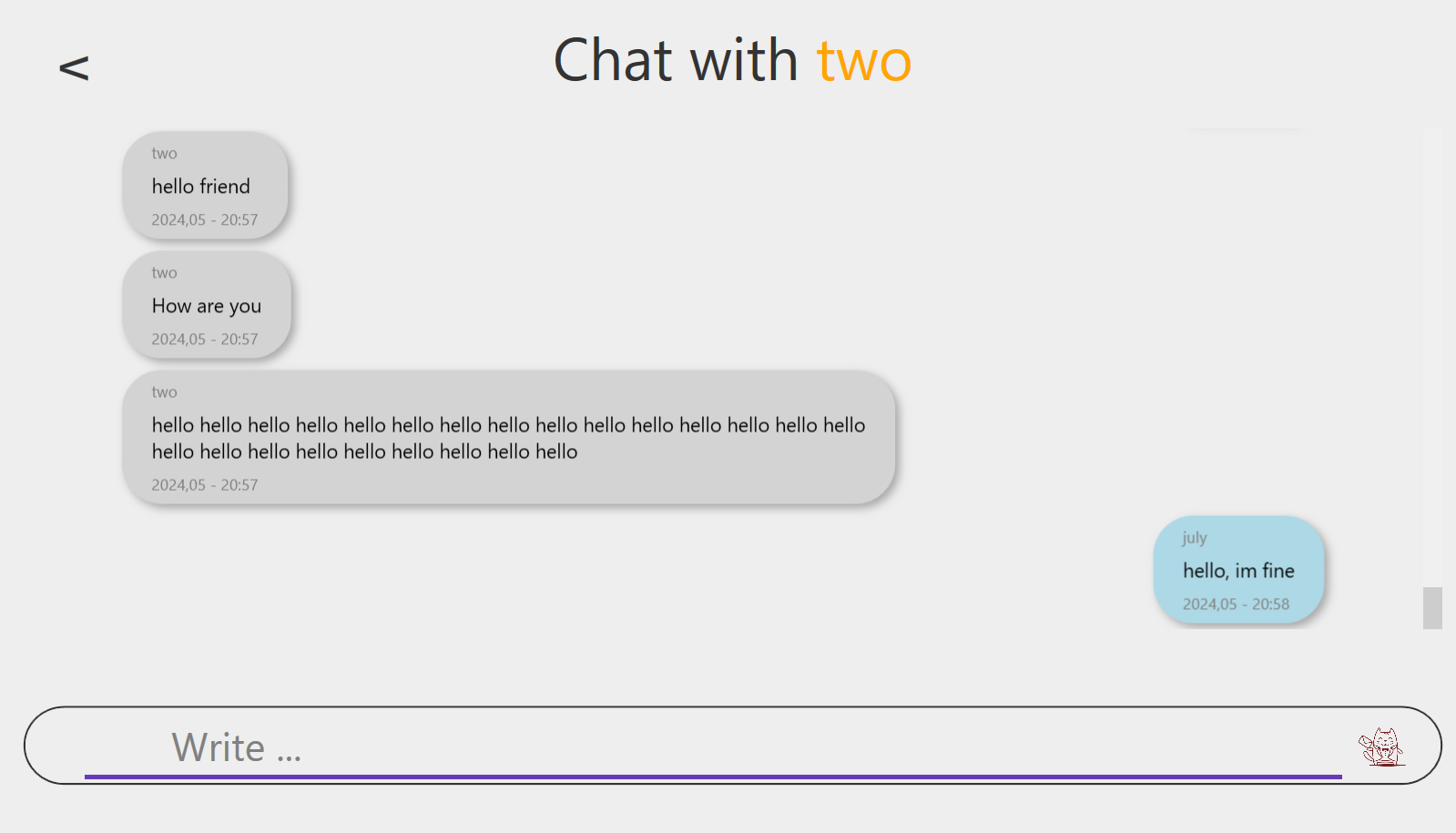
В случае если мы отправили запрос и другой человек принял его, мы можем или мы приняли предложение на дружбу от другого аккаунта, то у нас появляются 2 дополнительные возможности:

* просмотр аккаунта друга;
* открыть чат с другом.

Лист просмотра аккаунта друга на изображении 6.8, лист чата с другом на 6.9.

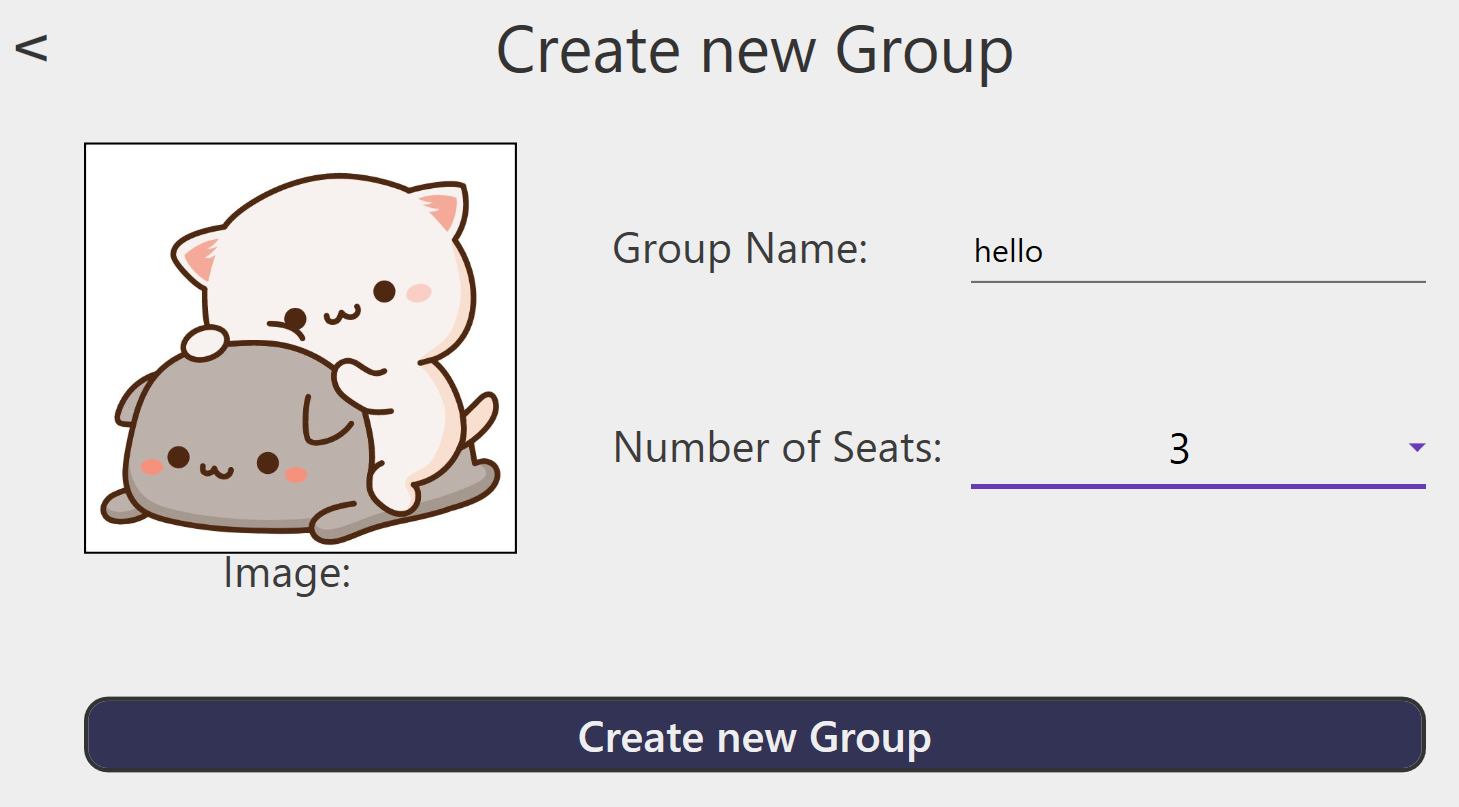


Изображение 6.8 – “Просмотр аккаунта друга”



Изображение 6.9 – “Чат с другом”

Так же в приложении поддерживаться группы, страницы которых такие же как и в чатах, за исключением того что группу нужно создавать. Создание идет в отдельной странице, где мы можем выбирать количество участников, название группы и иконку для группы. А после создания владелец и администратор могут изменить поля группы в этом же окне. Лист создания и обновления группы на изображении 6.10.



Изображение 6.10 – “Создание группы”

На странице создания группы кнопка создания заблокирована пока не будет введено корректное название. Таким названием является от 4 до 20 символов, и у человека не должно быть групп с таким именем.

**Заключение**

В ходе выполнения курсового проект была разработана социальная сеть дизайн которой должен прийти по вкусу любителям кошек.

Проведен анализ аналогов и сформулированы требования к разработанному программному средству.

Созданы все необходимые объекты структуры и базы данных, необходимых для корректной работы приложения

Реализованы паттерны для работы с базами данных и функции хеширования пароля для обеспечения безопасности.

Разработана структура базы данных, обеспечивающая удобное хранение изменение поиск и удаление информации.

Проведено тестирование основных функций и представлены варианты использования предложения, демонстрирующие работоспособность приложения.

Таким образом, в результате выполнения курсового проекта было разработано программное средство, обладающее всеми необходимыми функциональными возможностями для создания изменения аккаунта и общения с другими людьми.

# **Список использованных источников**

1. Microsoft Learn [Электронный ресурс] – https://learn.microsoft.com/ru-ru/– Дата доступа 27.03.2024
2. Metanit [Электронный ресурс] - <https://metanit.com/> – Дата доступа 20.04.2024
3. Poe [Электронный ресурс] – <https://poe.com/>– Дата доступа 10.05.2024

# **Приложение А**

Диаграмма использования

