

Міністерство освіти і науки України
Національний університет «Львівська політехніка»



Звіт
з проекту
з дисципліни «Організація баз даних»
на тему: «Система пропуску - QRBadge»

Виконавці: ст. гр. КІ-203

Ярмола Ю. Ю.

Ширий Б.І.

Гапонова Д. І.

Бокало П. М.

Щирба Д. В.

Чаус Б. В.

Миценко О. С.

Перевірила:

Колодчак О. М.

Львів 2023

ЗМІСТ

1.ВСТУП	2
2. ОГЛЯД ІСНУЮЧИХ РІШЕНЬ	4
3. ОПИС ПРОЕКТУ	5
3.1 QRBadge	5
3.2 Опис назви	5
3.3 Переваги проекту системи пропуску - “QRBadge”	6
3.4 Цільова аудиторія	6
3.5 Розробка функціоналу програми та телеграм бота	7
3.5.1 Функціонал програми	7
3.5.2 Функціонал телеграм бота	11
3.6 Функціонал екрана доступу	15
3.7 Програмна реалізація	16
3.7.1 Загальні відомості	16
3.7.2 Програма адміністрування	16
3.7.3 Програма допуску	23
3.8 Дизайн програми	26
3.8.1 Загальні відомості	26
3.8.2 Програма адміністрування	26
3.8.3 Програма Допуску	29
3.9 База Даних	30
3.9.1 Діаграма	30
3.9.2 Таблиці	31
3.9.3 Операції з даними	31
3.10. Апаратне забезпечення	33
4.ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ	35
4.1 Пропонована(стартова) ціна:	36
4.2 Аналіз витрат	36
5. ВИКОНАВЦІ	37
7. ЛІТЕРАТУРНІ ДЖЕРЕЛА	39
ДОДАТОК 1	40

1.ВСТУП

1.1 Тема: Система пропуску - QRBadge

1.2 Мета: Надати послуги безпеки, дозволяючи або обмежуючи доступ до певних ресурсів будь-якою стороною або фізичною особою.

1.3 Актуальність: Сучасне суспільство все більше стикається з потребою забезпечення безпеки та обмеження доступу до різних ресурсів для різних користувачів. Зокрема, це стосується бізнесу, де необхідно обмежувати доступ до конфіденційної інформації, або деяких приміщень, де необхідно контролювати вхід та вихід працівників та гостей. Також, система пропуску може бути корисною в інших сферах життя, наприклад, урядовій, освітній або медичній сферах, де також можуть виникати потреби в обмеженні доступу до ресурсів.

1.4 Проблематика: Проблематика забезпечення безпеки та обмеження доступу до різних ресурсів може бути надзвичайно складною в різних сферах життя, включаючи бізнес, уряд та освіту. Ось декілька причин, чому це може бути проблемою:

- Несанкціонований доступ: Безпека може бути порушена, коли особа або сторона отримує доступ до ресурсів, на які вони не мають права. Це може призвести до витоку конфіденційної інформації та порушення правил компанії або установи.
- Помилковий доступ: У деяких випадках може виникати проблема з помилковим доступом, коли особа або сторона отримує доступ до ресурсів, на які вони не мають повного права. Це може

призвести до неправильної обробки даних та надання некоректної інформації.

- Викрадення даних: Особиста інформація, що зберігається в ресурсах, може бути викрадена та використана несанкціонованою стороною. Це може призвести до крадіжки конфіденційних даних та порушення правил захисту персональних даних.
- Необхідність миттєвого доступу: У деяких випадках, наприклад в аварійних ситуаціях, потрібен швидкий доступ до певних ресурсів. Це може бути складно забезпечити в разі, якщо система контролю доступу не є достатньою чи складною в використанні.
- Контроль доступу в реальному часі: Коли створюється система контролю доступу, важливо мати змогу змінювати рівні доступу у реальному часі. Це може бути проблемою, якщо система не може швидко оновлювати та змінювати рівні доступу.

Усі ці фактори можуть стати перешкодою для забезпечення повної безпеки та контролю доступу до ресурсів. Відсутність ефективної системи контролю доступу може призвести до втрати конфіденційної інформації, надмірного витрачання ресурсів та порушення внутрішніх правил компанії або установи.

Розробка системи пропуску, такої як QRBadge, може бути важливим кроком у розв'язанні цієї проблематики. QRBadge може надати різним установам та компаніям ефективний інструмент для контролю доступу до різних ресурсів, що дозволить їм забезпечити безпеку та ефективність своєї діяльності.

2. ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

Існує багато різних систем контролю доступу та забезпечення безпеки, які використовуються у різних сферах діяльності. Деякі з них:

- Ключ-карти: це фізичні ключі, які дозволяють доступ до певних приміщень або об'єктів. Ці ключі можуть бути електронними або механічними.
- Кодові замки: це системи контролю доступу, що вимагають введення коду для отримання доступу до приміщень або об'єктів.
- Біометричні системи: ці системи використовують фізичні риси людини, такі як відбиток пальця, розпізнавання обличчя або розпізнавання голосу, для контролю доступу.
- Системи RFID: це системи, що використовують бездротову технологію для ідентифікації та контролю доступу до об'єктів.
- Системи контролю доступу з використанням мобільних пристроїв: ці системи дозволяють контролювати доступ до приміщень або ресурсів за допомогою мобільного телефону або іншого мобільного пристрою.

Кожна з цих систем має свої переваги та недоліки, і вибір системи залежить від потреб користувачів. Система пропуску QRBadge може бути альтернативою до існуючих систем контролю доступу, оскільки вона використовує технологію QR-кодів, яка є швидкою та зручною для користувачів. Крім того, QRBadge може бути легко інтегрованим з іншими системами, що робить його ефективним рішенням для забезпечення безпеки та контролю доступу до різних ресурсів.

3. ОПИС ПРОЕКТУ

3.1 QRBadge

QRBadge - це система пропуску, яка використовує QR-коди для ідентифікації користувачів та перевірки їх прав на вхід у певний об'єкт або зону.

QRBadge складається з двох компонентів: програмного забезпечення та фізичних QR-кодів. Програмне забезпечення QRBadge дозволяє створювати та керувати пропусками, встановлювати права доступу для користувачів, а також отримувати звіти про вхід та вихід користувачів. Фізичні QR-коди встановлюються на вхідних точках та інших місцях контролю доступу.

3.2 Опис назви

Назва "QRBadge" складається з двох слів. QR означає "Quick Response", що походить від швидкого зчитування QR-кодів. Badge в перекладі з англійської означає "жетон" або "бейдж", тобто фізичний елемент, що можна прикріпити до одягу для ідентифікації. Отже, QRBadge поєднує в собі швидкість та зручність QR-кодів з фізичною ідентифікацією, що відображається в назві системи.

3.3 Переваги проекту системи пропуску - “QRBadge”

- Відсутність необхідності, для працівників, скачувати додаткове програмне забезпечення;
- Простота інтерфейсу;
- Відсутність необхідності додатково навчати адміністраторів;
- Ми власноруч встановлюємо програмне та апаратне забезпечення;
- Надаємо гарантію;
- Надаємо онлайн консультації у разі виникнення питань чи проблем;
- Надійність програмного забезпечення;
- Надійність апаратного забезпечення;
- Адаптація до мови клієнта;
- Незначна вартість у порівнянні з конкурентами.

3.4 Цільова аудиторія

Географічні

Регіон: Європа;

Клімат: не важливо;

Населення: міста населенням > 200000;

ЗМІ: соціальні мережі.

Економічні

Дохід: середній та високий;

Клієнтура: організації та компанії, учбові заклади, тощо

Демографічні

Стать: чоловіки/жінки; Вік: 18+;

Сімейний статус: не важливо;

Освіта: закінчення середньої школи.

3.5 Розробка функціоналу програми та телеграм бота

3.5.1 Функціонал програми

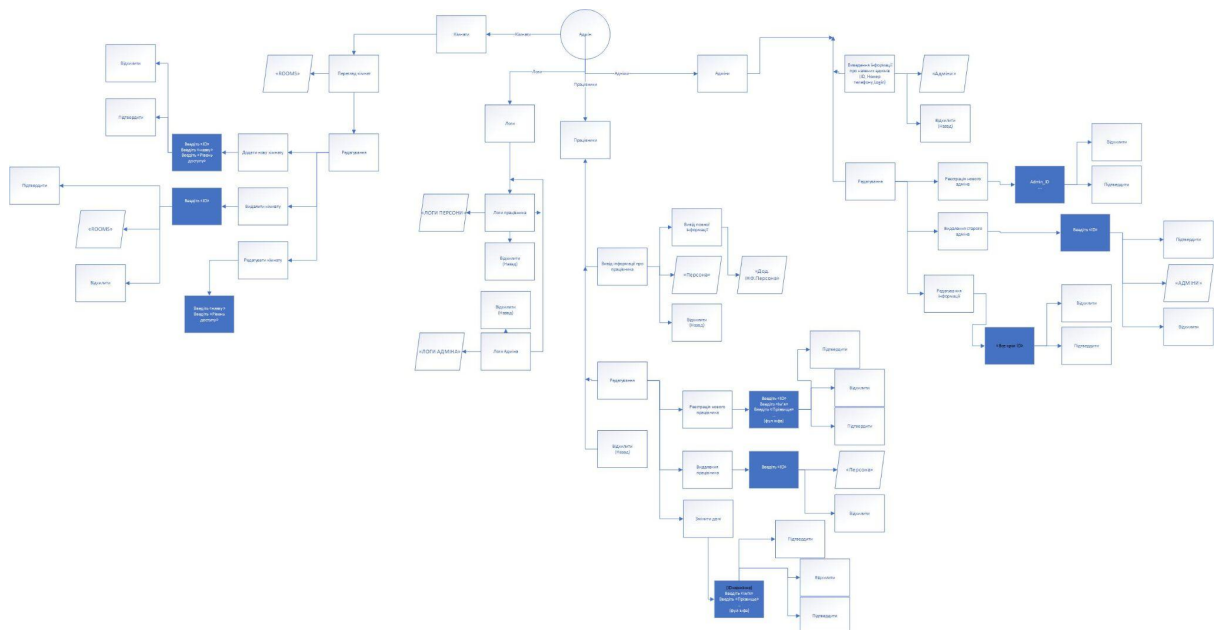


Рис. 3.5.1.1 Функціонал програми

Як видно з рисунку, програма має 4 вітки : кімнати, логи, адміни, працівники.

Гілка “Кімнати” виглядає так:

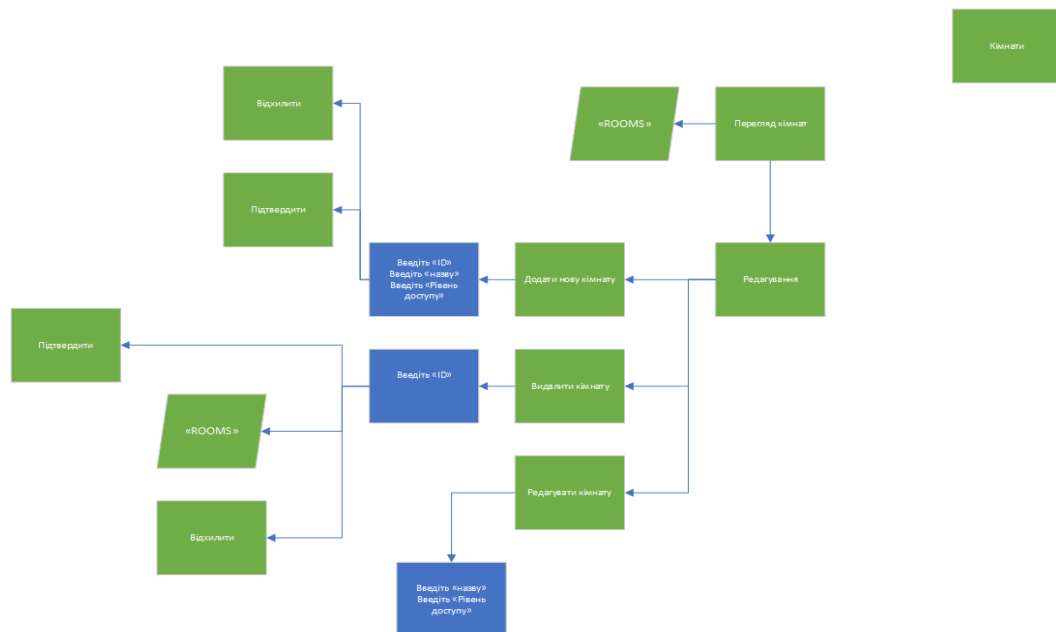


Рис. 3.5.1.2 Гілка “Кімнати”

Натиснувши кнопку “Кімнати”, користувач має 2 опції - “Перегляд кімнат” і “Редагування”

1. “Кімнати” - виведе на екран інформацію про доступні кімнати. Тут кімнати - це ті кімнати до яких підключена система пропуску.
2. “Редагування” - надасть можливість додавати, редагувати і видаляти кімнати.

Гілка “Логи” виглядає так:

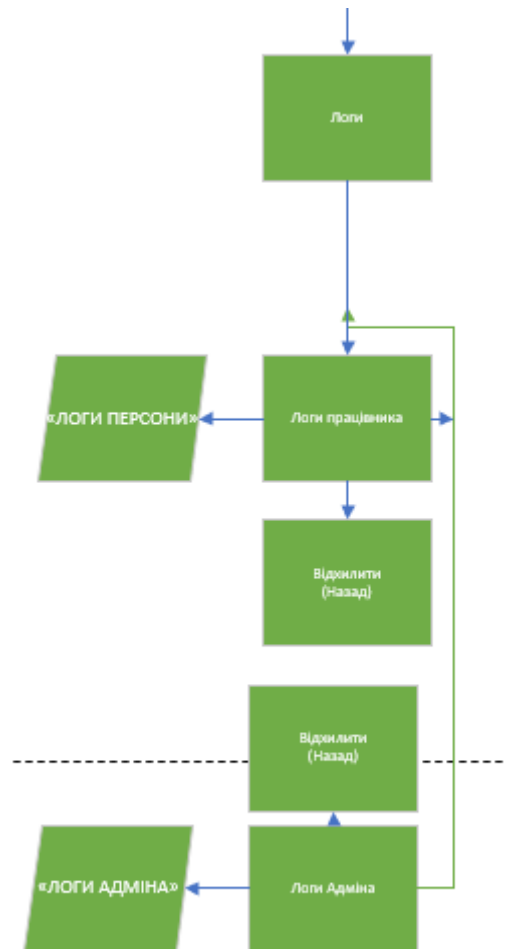


Рис. 3.5.1.2 Гілка “Логи”

Натиснувши кнопку “Логи” користувач побачить 2 опції - “Логи працівника” та “Логи адміна”

1. “Логи працівника” - виведе інформацію безпосередньо про працівника компанії, тобто про людину яка постійно верифікується.
2. “Логи адміна” - виведе інформацію про адміністраторів, наприклад час входження в акаунт, для того щоб відслідковувати їх роботу.

Гілка “Адміні” має такий вигляд :

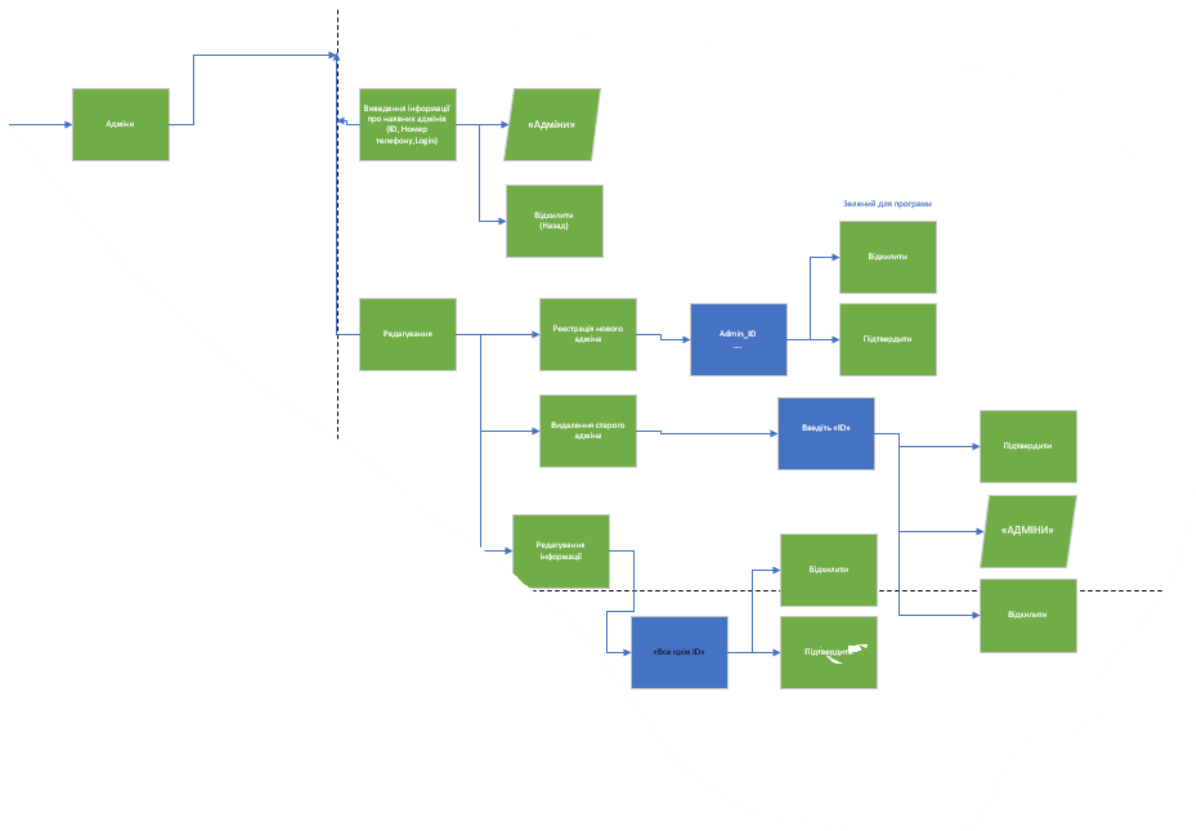


Рис. 3.5.1.3 Гілка “Адміні”

Натиснувши кнопку “Адміні”, користувач отримує варіанти : “Виведення інформації” та “Редагування”

1. “Виведення інформації” - виведе інформацію про наявних адмінів.
2. “Редагування” - дозволяє видалити, додати чи редагувати адміна.

The diagram is divided into two main horizontal sections by a dashed line, representing the State (top) and the Citizen (bottom).

Top Section (State):

- Приветствие (Greeting):** A green rectangle at the top left.
- Ввод информации про гражданина (Input of citizen information):** A green rectangle below the greeting.
- Ввод личной информации (Input of personal information):** A green rectangle to the right of the input box.
- «Персона» (Person):** A green parallelogram below the personal information input.
- «Дод. ВН.Персона» (Additional personal data):** A green parallelogram to the right of the person box.
- Вывод (Output):** A green rectangle below the personal information input.

Bottom Section (Citizen):

- Регистрация (Registration):** A green rectangle on the left.
- Входные данные (Input data):** A green rectangle below registration.
- Выводы и рекомендации (Outputs and recommendations):** A green rectangle to the right of the input data.
- Выводы, «ID» (Outputs, ID):** A blue rectangle to the right of the outputs and recommendations.
- «Персона» (Person):** A green parallelogram to the right of the ID box.
- Выводы (Outputs):** A green rectangle below the person box.
- Выводы (Outputs):** A green rectangle at the bottom right.
- «ID персона» (ID person):** A blue rectangle below the outputs and recommendations.
- Выводы и рекомендации (Outputs and recommendations):** A green rectangle to the right of the ID person box.
- Выводы (Outputs):** A green rectangle below the ID person box.

Flow of Information and Data:

- Information Flow (Green Arrows):**
 - From **Приветствие** to **Ввод информации про гражданина**.
 - From **Ввод информации про гражданина** to **Ввод личной информации**, **«Персона»**, and **Вывод**.
 - From **Ввод личной информации** to **«Дод. ВН.Персона»**.
 - From **Вывод** to **Регистрация**.
 - From **Регистрация** to **Входные данные**.
 - From **Входные данные** to **Выводы и рекомендации** and **Выводы, «ID»**.
 - From **Выводы и рекомендации** to **«Персона»** and **Выводы**.
 - From **Выводы, «ID»** to **«Персона»** and **Выводы**.
 - From **«Персона»** to **Выводы**.
 - From **Выводы** to **Выводы (Outputs)**.
- Data Flow (Blue Arrows):**
 - From **Входные данные** to **«ID персона»**.
 - From **«ID персона»** to **Выводы и рекомендации** and **Выводы (Outputs)**.

Рис. 3.5.1.2 Гілка “Працівники”

Натиснувши кнопку “Працівники”, користувач отримує варіанти : “Виведення інформації” та “Редагування”

1. “Виведення інформації” - виводить головну інформацію про працівника, а також дає можливість вивести детальнішу інформацію.
2. “Редагування” - дозволяє видаляти працівника та редагувати його дані.

3.5.2 Функціонал телеграм бота

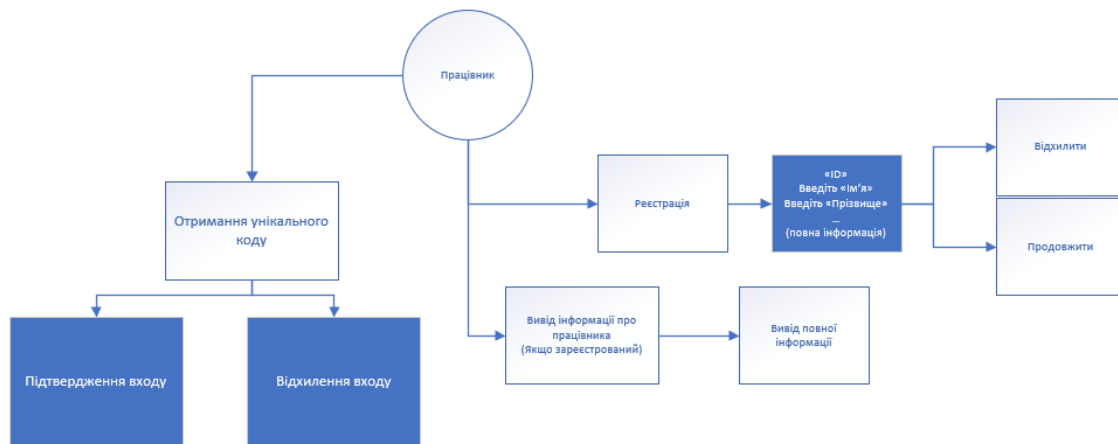


Рис. 3.5.2.1 - Функціонал телеграм бота

Перша гілка (Реєстрація) - В даній гілці відбувається реєстрація нового акаунту працівника. Отримання всієї інформації про нього. При потребі можна змінювати кількість та види інформації яка запитується у працівника. Після успішної реєстрації працівник додається у базу даних, та очікує підтвердження(верифікацію) свого акаунту.

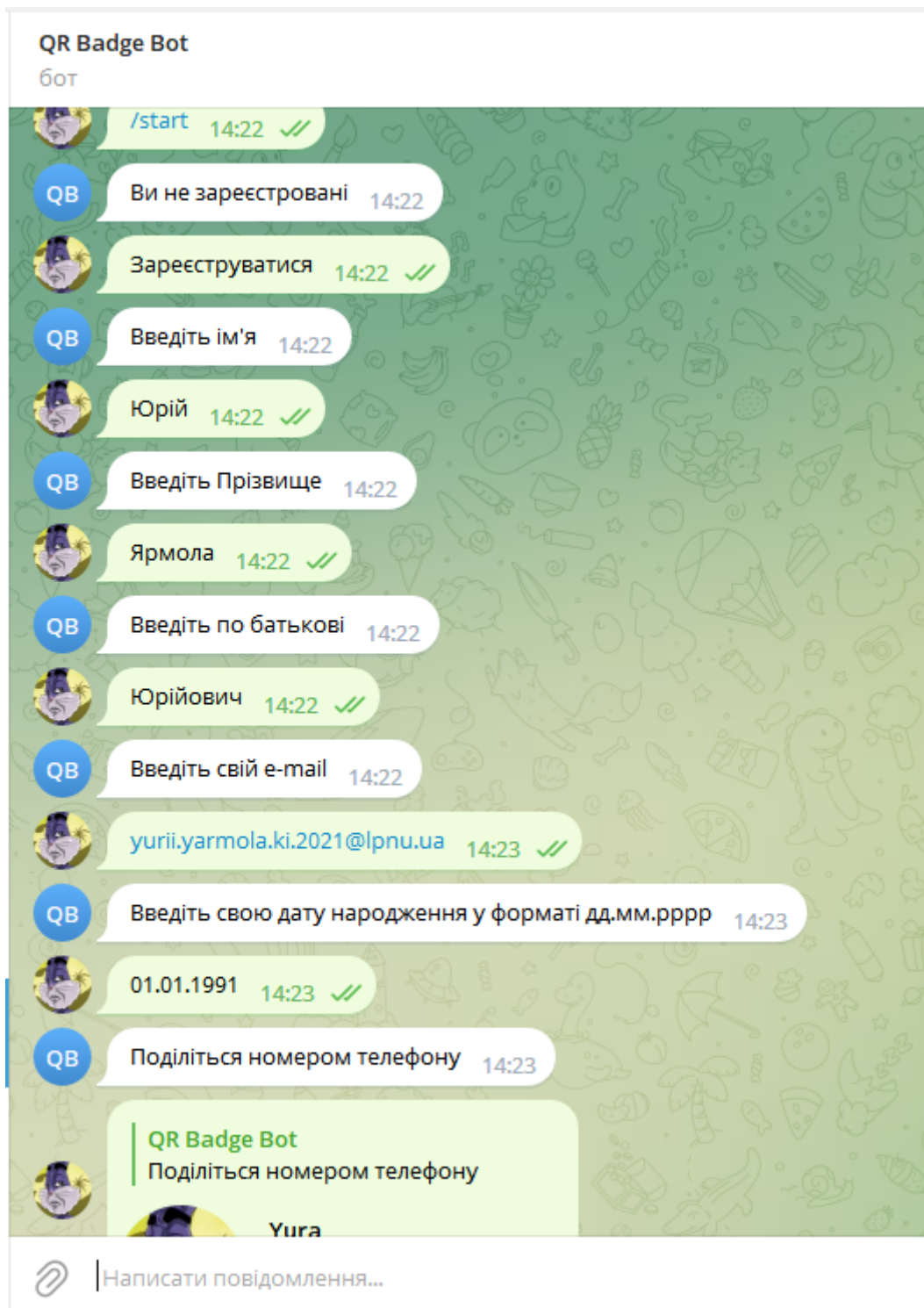


Рис. 3.5.2.2

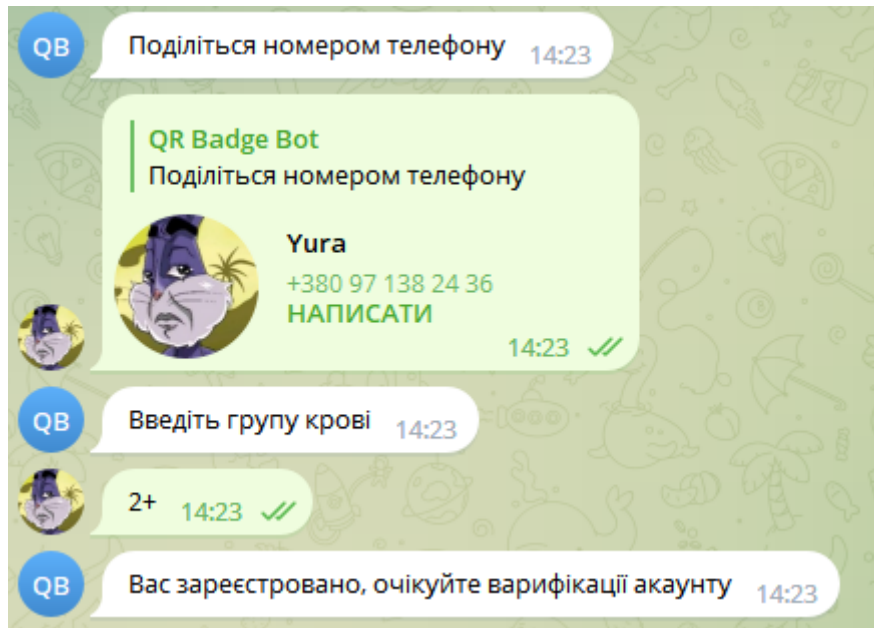


Рис. 3.5.2.3

Рис. 3.4.2.2 - 3.4.2.3 - Реєстрація нового співробітника/гостя

Друга гілка (Отримання інформації про акаунт) - для перевірки свого статусу та рівня пропуску працівник завжди може отримати інформацію про свій профіль.

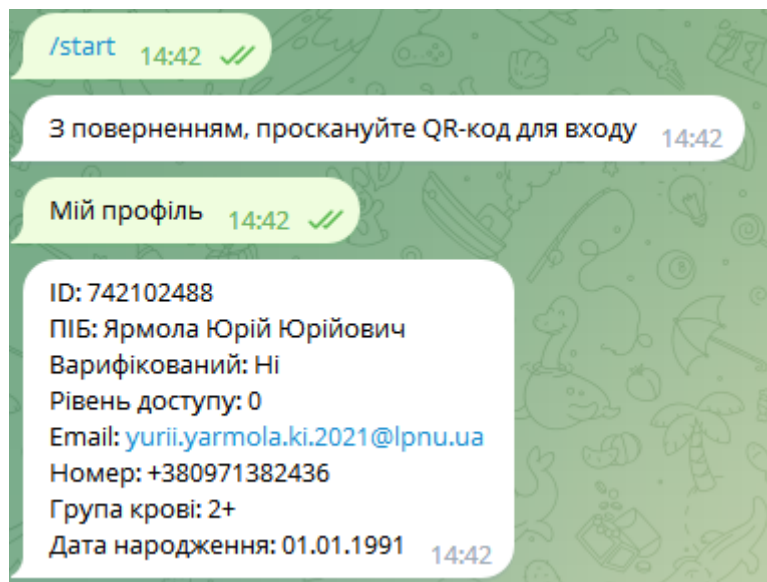


Рис. 3.5.2.4 - Демонстрація роботи кнопки “Мій профіль”

Третя гілка (вхід) - для входу користувачу потрібно просканувати спеціальний QR-код та натиснути start. Після цього програма телеграм боту перевіряє всі дані користувача, а саме чи не записаний він у чорний список, чи верифікований його акаунт та чи достатній рівень пропуску для конкретних дверей. Якщо всі вище перелічені умови є вірними для користувача, то користувачу повідомляється щоб він підтвердив на екрані дверей що вхід дозволено, якщо ж одна з умов недостатня для входу, користувачу повідомляється чому саме вхід для нього є забороненим.

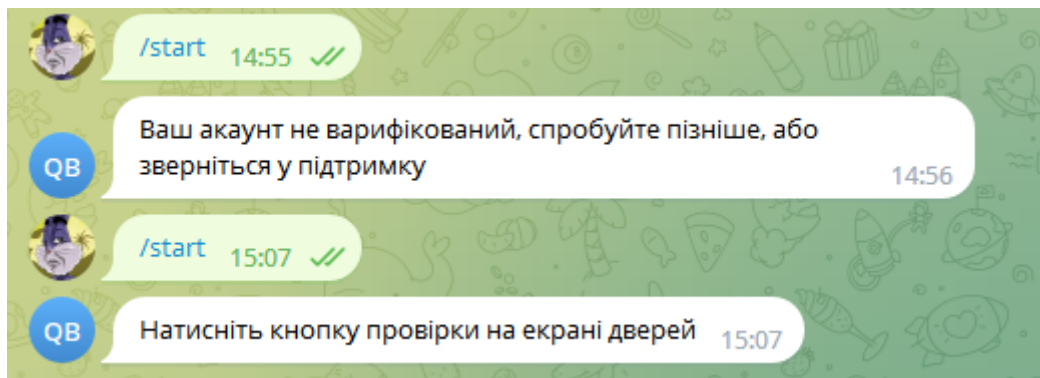


Рис. 3.5.2.5 - Демонстрація різних сценаріїв з входом

Чому саме телеграм? Все просто, ніхто не любить встановлювати додаткових 1000 програм для кожного випадку життя, а телеграм чудовий месенджер який дозволяє реалізувати власного бота, та є практично у всіх(1+млрд завантажень у Google Play). Також це робить систему відкриття дешевшою та мультиплатформенною.

Алгоритм телеграм бота реалізований на мові Python що дозволяє робити легкі зміни та реалізацію додаткових функцій без великих затрат. Запити до бази даних відбуваються прямо в алгоритмі за допомогою бібліотеки MySQL та екрановані від SQL ін'єкцій. Основними командами для запитів до бази даних є SELECT, INSERT, JOIN. Також використання алгоритму мовою Python дозволяє запускати його на відносно дешевих серверах, або ж прямо на сервері компанії за кілька хвилин. Для відлагодження та вивчення причин гіпотетичних збоїв, використовується логування подій.

3.6 Функціонал екрана доступу

Коли користувач хоче пройти в певні двері, він спочатку сканує QR код на екрані, після чого, якщо такий користувач зареєстрований, він може натиснути кнопку “Відчинити” і двері відчиняються на певний час. Якщо ж користувач не зареєстрований, то двері не відчиняються.

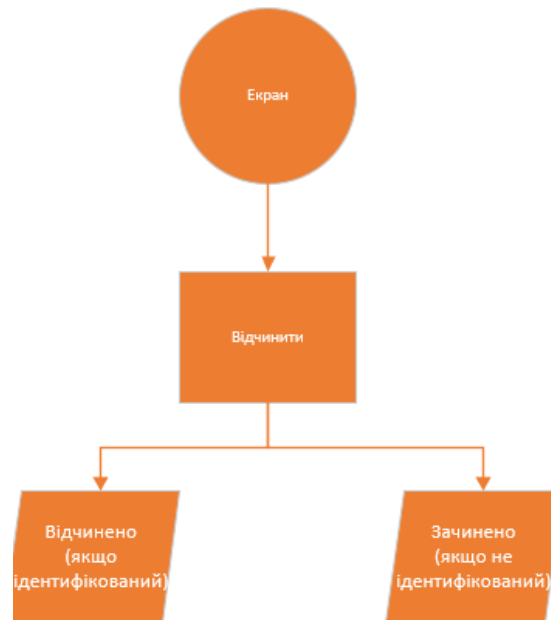


Рис. 3.6.1 Реалізація екрана доступу

3.7 Програмна реалізація

3.7.1 Загальні відомості

Для створення даних програм використовувалися технології WPF та C# .NET Framework 4.8, також технології роботи з базами даних ADO.NET. Використана архітектура баз даних – MySQL, для чого потрібно було використати бібліотеку MySql.Data.MySqlClient.

3.7.2 Програма адміністрування

Вхід в програму відбувається через вхід в акаунт адміна, залежно від виду адміна, йому буде дозволено здійснювати різні дії. Реалізацією передбачено «Нульовий адмін», який буде зареєстрований при встановленні обладнанні та зможе редагувати «побічних адмінів», також він може реєструвати додаткових адмінів його типу. Такий тип адмінів передбачається для керівних посад. Сам вхід відбувається через SecureString, окремий репозиторій, який імплементується у модель адміна, використовуючи об'єкти ICommand для реалізації функціонал кнопок та полів (текстових та паролельних). Загалом, після входу в акаунт відбувається перехід на вікно MainView, оскільки вхід реалізований окремо від основного функціонала заради безпеки. Отож, користувач побачить вікно, яку зображене на рисунку 3.7.2.1.

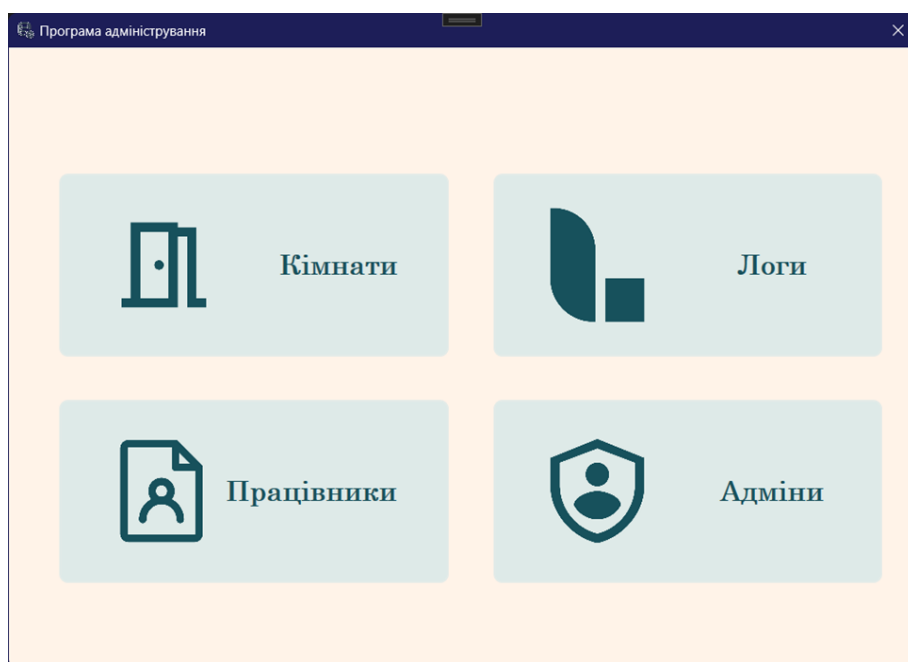


Рис. 3.7.2.1. Сітка головного меню

Загалом, основне вікно (Window) організоване у вигляді незалежних сіток (Grid), відповідно на цьому рисунку ми бачимо основну сітку, звідки можна вибрати потрібний для користувача відділ програми та відредагувати базу даних. Далі будемо збирати роботу програми на прикладі розділу “Працівники”. Отож, коли ми вибрали

відділ, ми побачимо відповідне вікно на рисунку 3.7.2.2, вибравши першу опцію зможемо побачити список всіх працівників на рисунку 3.7.2.3.

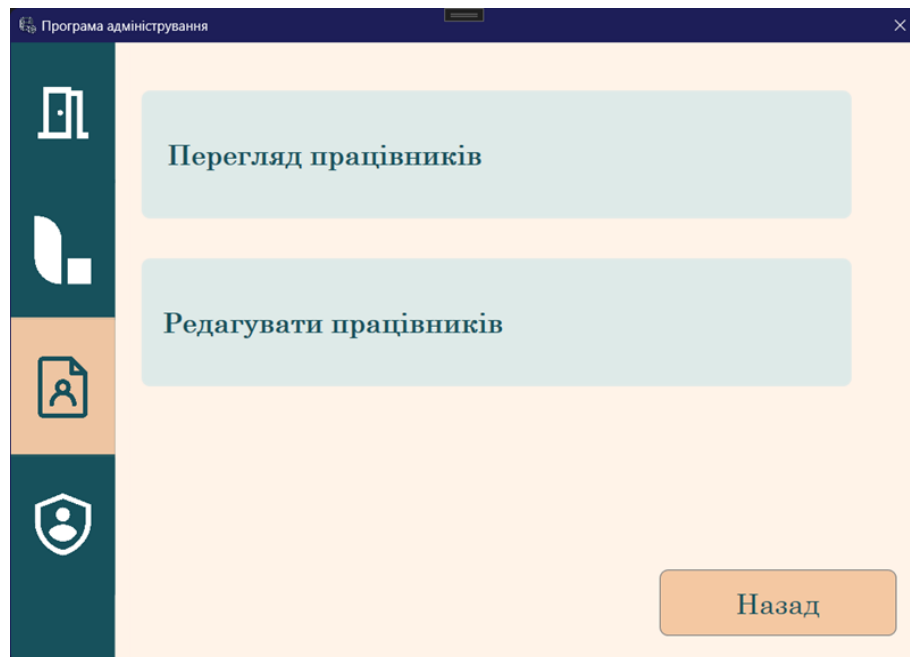


Рис. 3.7.2.2. Меню вибору опцій взаємодії з таблицею працівників.



Рис. 3.7.2.3. Показ даних таблиці працівників.

Також поглянувши на рисунок 2, можемо побачити, що у нас є опція редагування працівників, натиснувши на яку ми відкриємо сітку на рисунку 4.

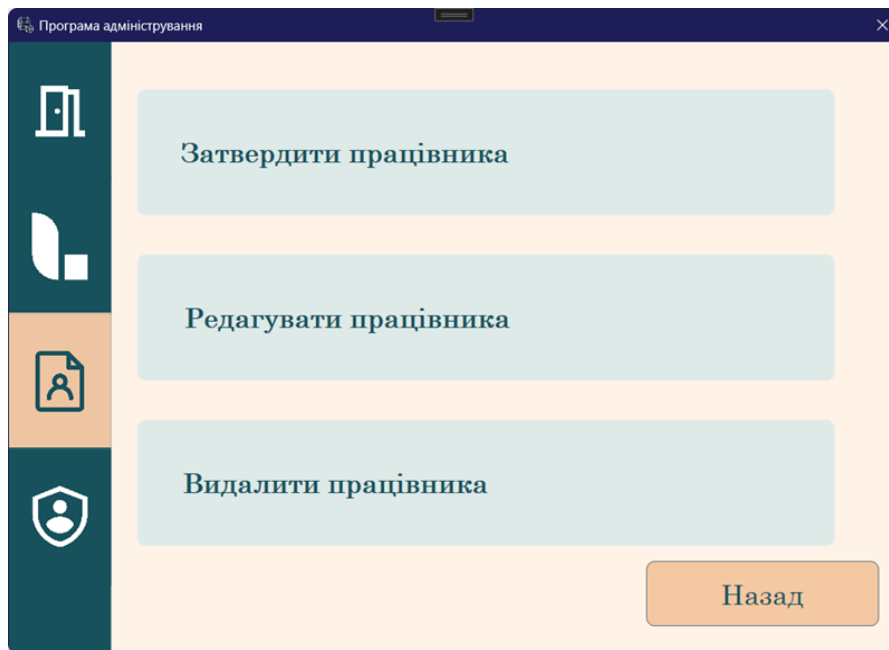


Рис. 3.7.2.4. Головне меню редагування працівників.

Підемо знизу до верху, отже якщо натиснути “видалити працівника”, то ми перемістимось на наступну сітку, що зображена на рисунку 3.6.2.5.

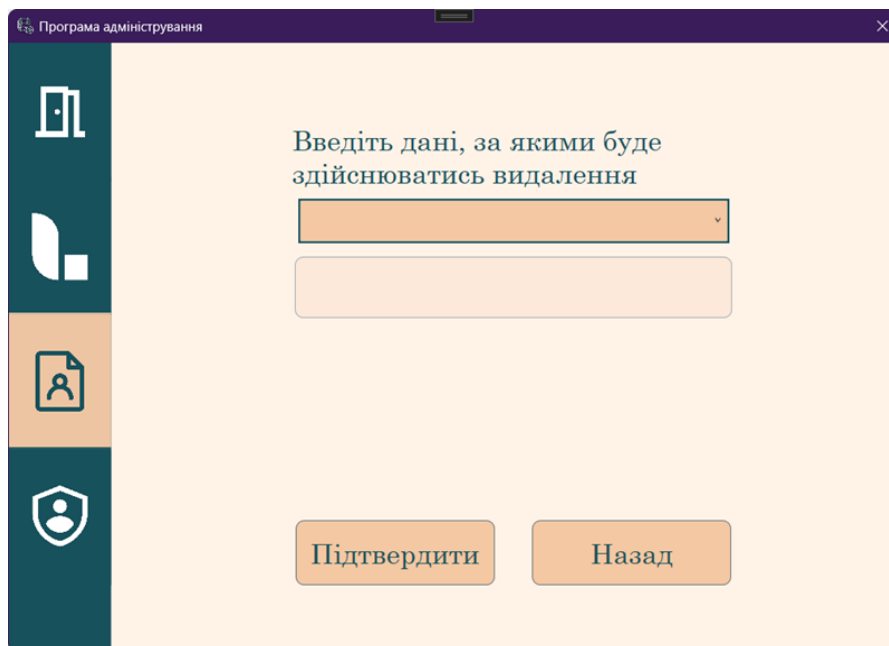
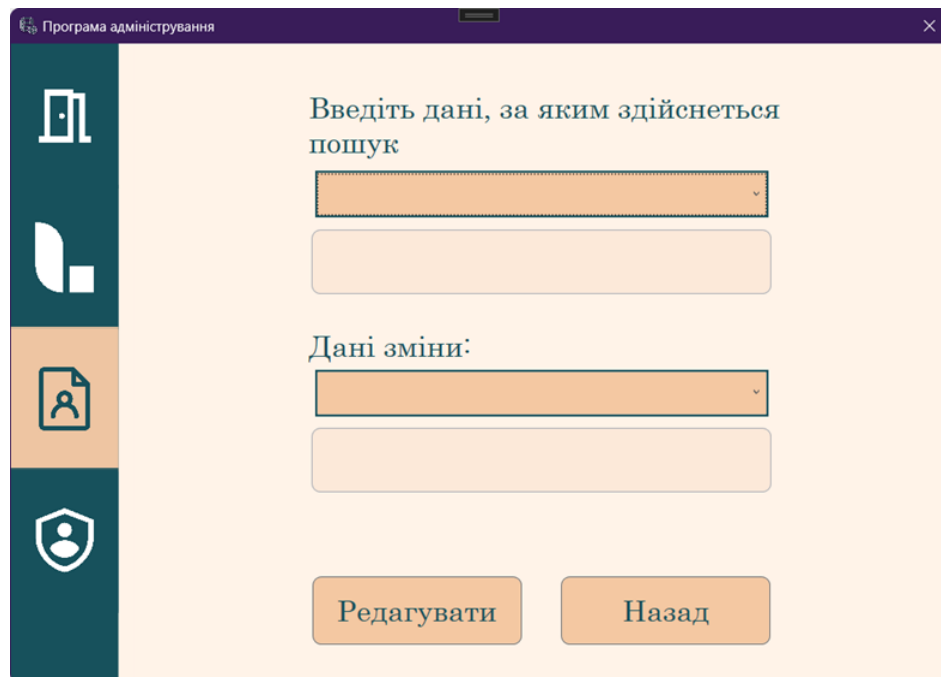


Рис. 3.7.2.5. Вид видалення працівників.

Як бачимо, у цій сітці є комбо та текст бокси, першим ми вибираємо ознаку (одну колонку з двох таблиць працівників), а в друге вводимо відповідне значення за яким буде здійснюватися видалення.

Якщо на рисунку 3.7.2.4 вибрати “Редагування працівників”, то появиться сітка, яка наведена на рисунку 3.7.2.6.



The screenshot shows a web application window titled "Програма адміністрування" (Administration Program). On the left is a dark blue sidebar with four icons: a building, a bar chart, a person with a document, and a shield. The main area has a light orange background. It contains two sections for data entry. The first section, labeled "Введіть дані, за яким здійснеться пошук" (Enter data for search), has a dropdown menu and a text input field. The second section, labeled "Дані зміни:" (Change data:), also has a dropdown menu and a text input field. At the bottom are two buttons: "Редагувати" (Edit) and "Назад" (Back).

Рис. 3.7.2.6. Вид редагування працівника.

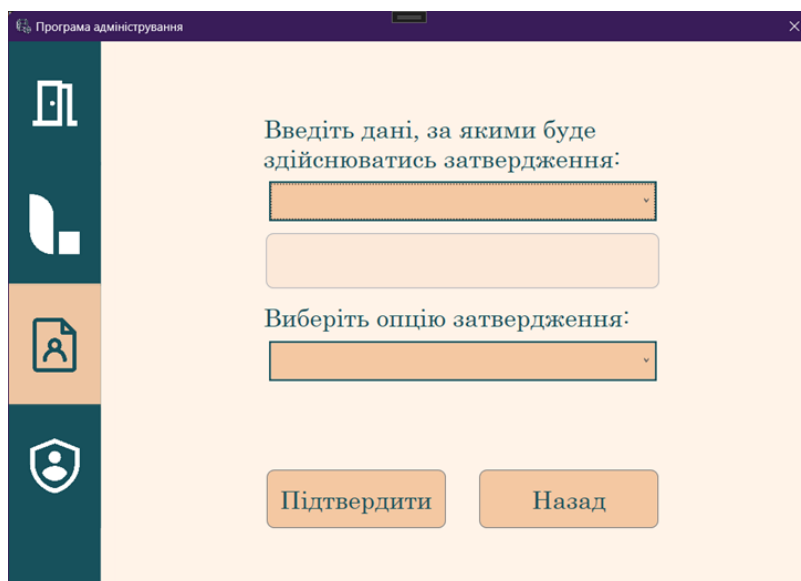
Першою парою полів ми знаходимо необхідних працівників, а другою – виконуємо редагування їх ознак, для більшої наочності навів код методу виконання редагування працівника у лістингу 1.

Лістинг 1. Метод виконання редагування працівника.

```
private void editWorkerActionButton_Click(object sender, RoutedEventArgs e)
{
    string input = ((workerInputEditComboBox.SelectedIndex == 0) || (workerInputEditComboBox.SelectedIndex == 3))
        ? workerInputEditTextBox.Text : ("\" + workerInputEditTextBox.Text + "\"),
        output = ((workerOutputEditComboBox.SelectedIndex == 0) || (workerOutputEditComboBox.SelectedIndex == 3))
        ? workerOutputEditTextBox.Text : ("\" + workerOutputEditTextBox.Text + "\");
    try
    {
        rowsAffected = 0;
        DataTable table = new DataTable();
        MySqlConnection connection = new MySqlConnection(ConnectionString);
        {
            connection.Open();
            MySqlDataAdapter adapter = new MySqlDataAdapter("SELECT CHAT_ID FROM " +
                $" {(workerInputEditComboBox.SelectedIndex > 3 ? "PersonsAdditionalInfo" : "Persons")} " +
                $"WHERE {comboBoxWorkerOptions[workerInputEditComboBox.SelectedIndex]} = {input}", connection);
            adapter.Fill(table);
            foreach (DataRow row in table.Rows)
            {
                MySqlCommand command = new MySqlCommand("UPDATE " +
                    $" {(workerOutputEditComboBox.SelectedIndex > 3 ? "PersonsAdditionalInfo" : "Persons")} " +
                    $"SET {comboBoxWorkerOptions[workerOutputEditComboBox.SelectedIndex]} = {output} " +
                    $"WHERE CHAT_ID = {(int)row["CHAT_ID"]}", connection);
                command.ExecuteNonQuery();
                rowsAffected++;
            }
            connection.Close();
        }
        workerInfoEditTextBox.Text = rowsAffected > 0 ? (rowsAffected + " з працівників було змінено.") : "Жодного працівника не було змінено.";
    }
    catch (Exception) { workerInfoEditTextBox.Text = "Виникла помилка."; }
}
```

Як бачимо, програма зразу працює з текстовими та комбо полями, а також використовує адаптори бази даних для роботи взаємодії з таблицею.

Відповідно, остання опція розділу – це «Затвердити працівника», оскільки реєстрування працівника відбувається у Telegram боті, то у програмі можна лише його затвердити. На рисунку 3.7.2.7 наведено відповідну сітку.



Програма адміністрування

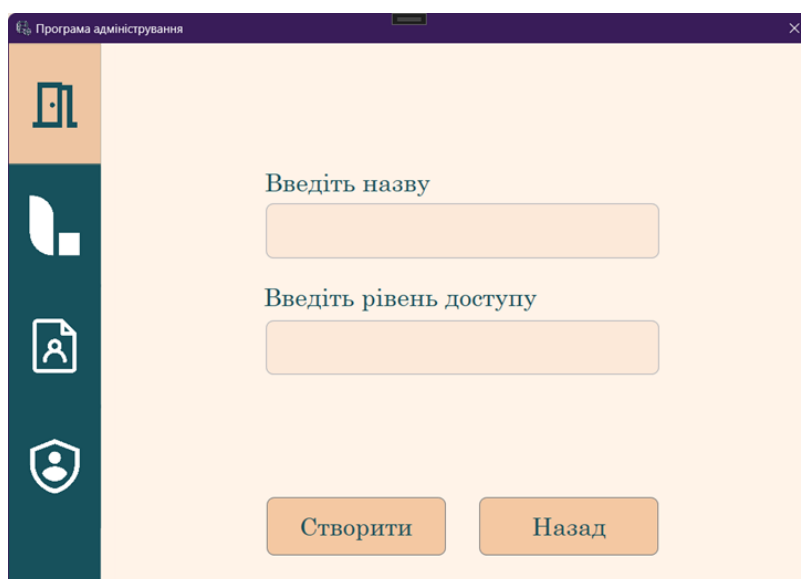
Введіть дані, за якими буде здійснюватись затвердження:

Виберіть опцію затвердження:

Підтвердити Назад

Рис. 3.7.2.7. Вид затвердження працівника.

Отож, бачимо, що затвердження ми здійснюємо першими боксами та потім обираємо чи затвердити працівника, чи спростувати. Також у програмі можна створювати двері або реєструвати адміна, розглянемо сітку створення дверей, яка зображена на рисунку 3.7.2.8.



Програма адміністрування

Введіть назву

Введіть рівень доступу

Створити Назад

Рис. 3.7.2.8. Вид створення нових дверей.

Таким чином, чи коротко розглянули роботу програми адміністрування, решта більша частина функціоналу схожа на розглянуту.

Загалом, програма адміністрування дозволяє працювати з такими об'єктами та виконувати такі дії які зображені у таблиці 3.7.2.1.

Таблиця 3.7.2.1. Можливі дії за допомогою програми адміністрування.

Об'єкт	Дії
Двері	Переглянути
	Створити
	Редагувати
	Видалити
Працівник	Переглянути
	Затвердити
	Редагувати
	Видалити
<u>Адмін</u>	Переглянути
	Зареєструвати
	Редагувати
	Видалити
<u>Чориний список</u>	Переглянути його
	Добавити в нього
	Видалити з нього
<u>Логи</u>	Переглянути історію

Частина у таблиці 3.7.2.1, яка підкреслена зеленим кольором, - це дії які може здійснювати лише «Нульовий адмін». Як бачимо, це всі операції що пов'язані з роботою інших адмінів.

3.7.3 Програма допуску

Ця програма буде пропускати людей через двері, за наявності зареєстрованого затвердженого працівника, що не знаходиться у чорному списку та має достатній рівень доступу. Початковий вигляд та вигляд при створенні QR коду, зображено на рисунку 3.7.3.1.

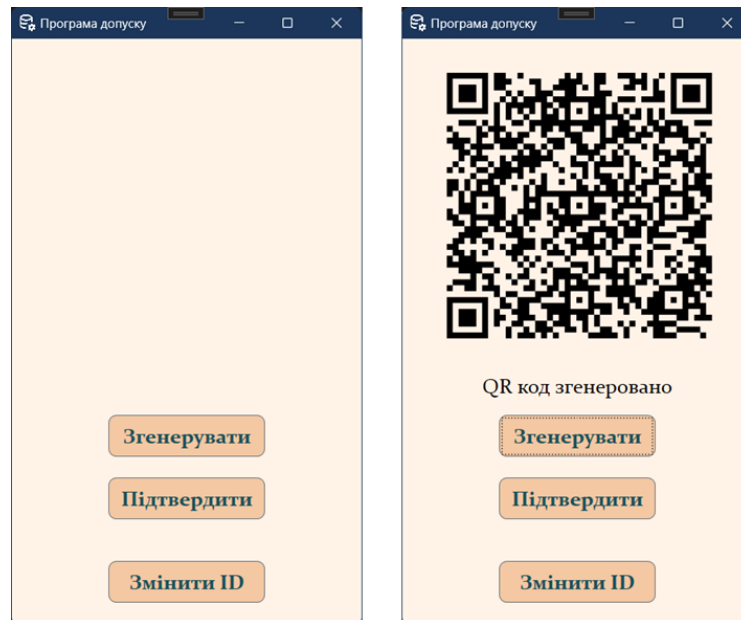


Рис. 3.7.3.1. Початковий вигляд та вигляд з генерованим кодом.

Відповідно, якщо не працівник не відповідає вищезазначеним умовам, то у доступі йому буде відмовлено, якщо навпаки, то прохід буде підтверджено та двері відчиняться. Приклад першого випадку зображено ліворуч на рисунку 3.7.3.2, а приклад другого випадку зображено на тому ж рисунку праворуч.

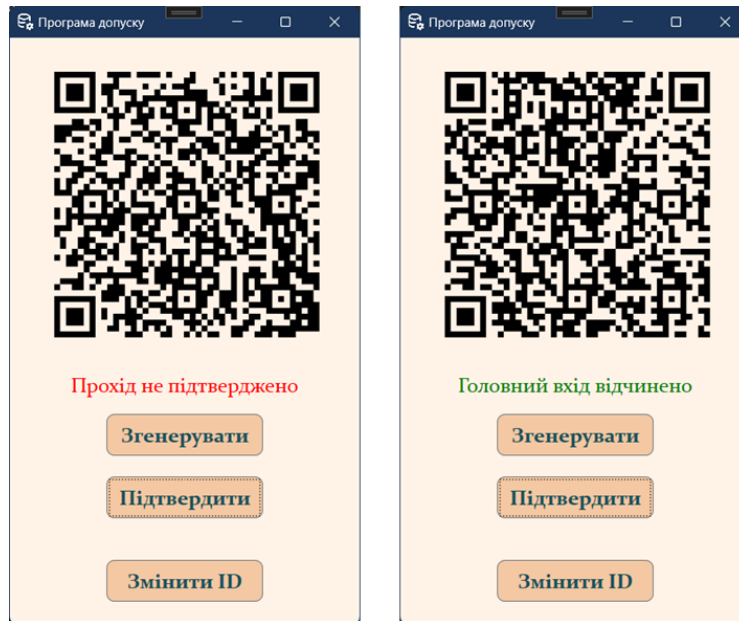


Рис. 3.7.3.2. Відхилення та підтвердження допуску.

Також у разі необхідності перепрошивки прив'язки до дверей, натиснувши на кнопку “Змінити ID”, ми перемістимося на сітку зміни прошивки дверей. Після підтвердження перепрошивки ми отримаємо повідомлення про успішне чи провальне її виконання, як, відповідно, зображено ліворуч та праворуч на рисунку 3.7.3.3.

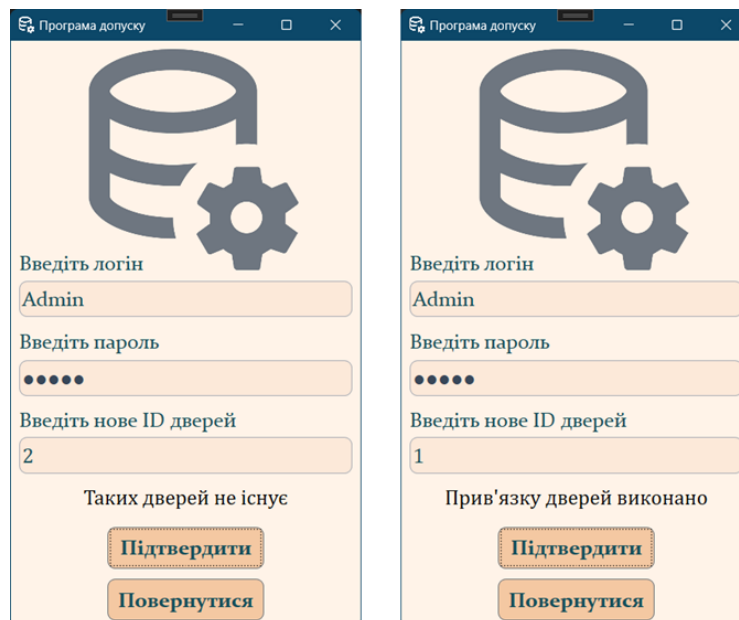


Рис. 3.7.3.3. Спроби виконання прив'язки дверей.

Більше одного QR коду генерувати не можна, як це зображено ліворуч на рисунку 3.7.3.4, згенерувати новий код можна лише після виконання роботи з наявним. Також, праворуч на рисунку наведено: що станеться з QR кодом, якщо операційна система перейде у режим сну або заблокується.

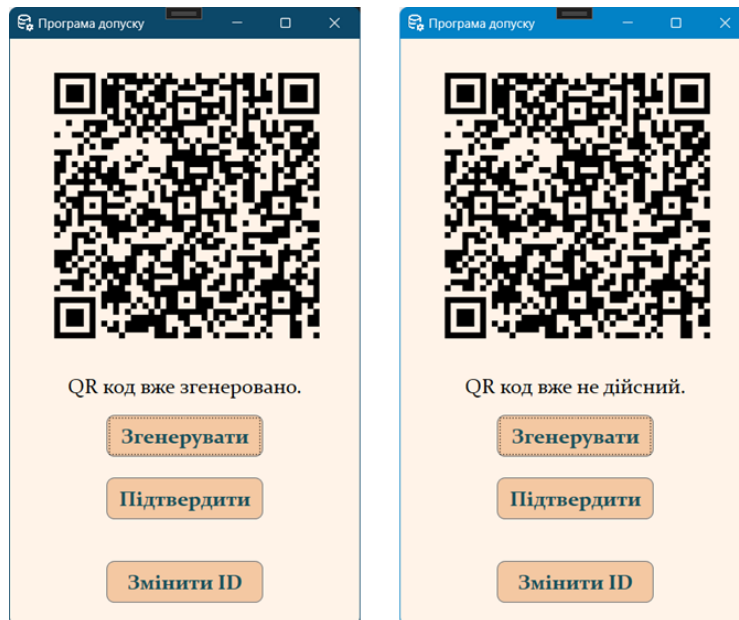


Рис. 3.7.3.4. Інші випадки роботи з програмою.

3.8 Дизайн програми

3.8.1 Загальні відомості

Для розробки зовнішнього вигляду програми використовувалися векторний онлайн-сервіс розробки інтерфейсів та прототипування Figma, також сервіс для створення кольорових схем Adobe Color, бібліотека вільно розповсюджуваних шрифтів Google Fonts.

3.8.2 Програма адміністрування

За допомогою сервісу розробки інтерфейсів Figma було зроблено макет вікон програми. Кольори підбрані так, щоб користувачу було приємно взаємодіяти з програмою без зайвих зусиль. Фоновий колір FFF3E8 не надто яскравий і на його фоні чітко можна розібрати текст. Поля для вводу даних користувача зроблені у яскравішому відтінку. Колір тексту контрастний, підібраний так щоб виділятися на фоновому кольорі.

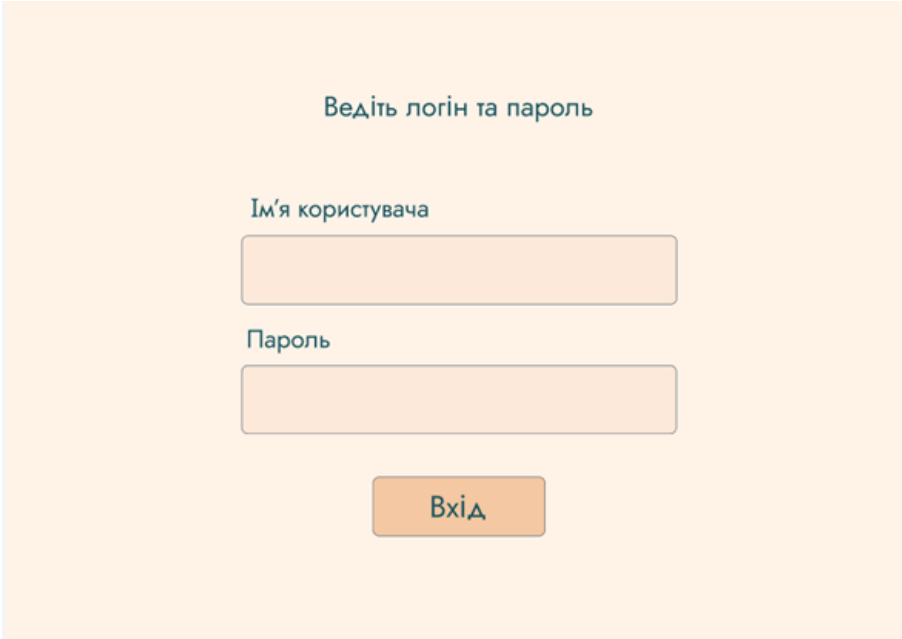
A login form mockup on a light orange background. At the top, the text "Ведіть логін та пароль" is centered in a dark teal font. Below it, the label "Ім'я користувача" is centered above a light orange rectangular input field. Underneath, the label "Пароль" is centered above another light orange rectangular input field. At the bottom, a dark orange button with the text "Вхід" in white is centered.

Рис. 3.8.2.1. Вікно входу.

У основному меню колір 4 основних кнопок більш блакитний, але не надто яскравий, для того щоб текст на фоні кнопок виділявся. Також на кнопку кожної категорії в меню було додано векторне зображення для полегшення розуміння користувачем позначення кнопок.

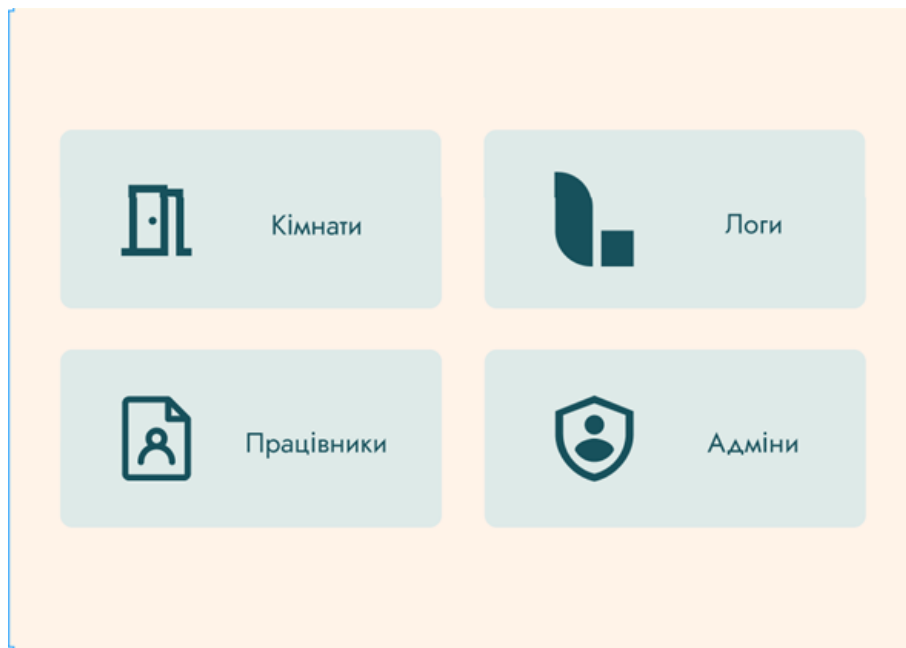


Рис. 3.8.2.2. Сітка головного меню.

При переході до будь якої з категорій в головному меню збоку з'являється бокова панель, яка інформує про відділ програми в якому знаходиться користувач. Також кнопка для повернення назад має яскравіший колір для кращого розпізнавання користувачем, проте текст на ній досі досить легко читається.

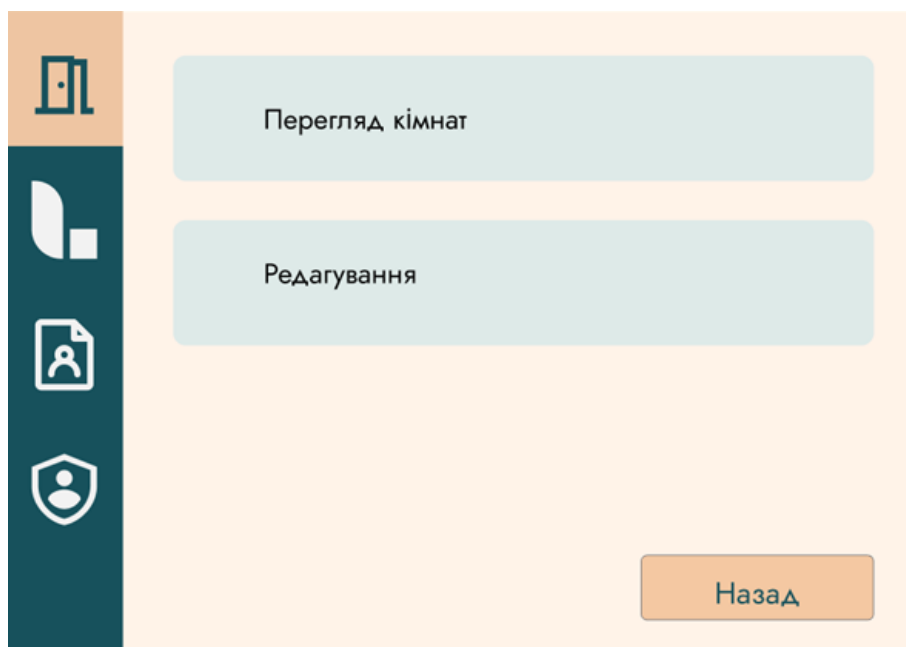


Рис. 3.8.2.3. Меню вибору опцій взаємодії з таблицею працівників

Поля для введення, кнопки в меню для переходу між відділами програми та кнопки для підтвердження чи скасування відрізняються за кольором та розміром що полегшує їх розрізнення та спрощує роботу користувача.

Також було зроблено векторну іконку, що репрезентує застосунок.



Рис.3.8.2.4. Іконка програми.

3.8.3 Програма Допуску

Ця програма була зроблена зі схожим дизайном на програму адміністрування, з застосуванням тих самих кольорів. Кнопки зроблено відповідно до функціоналу програми.

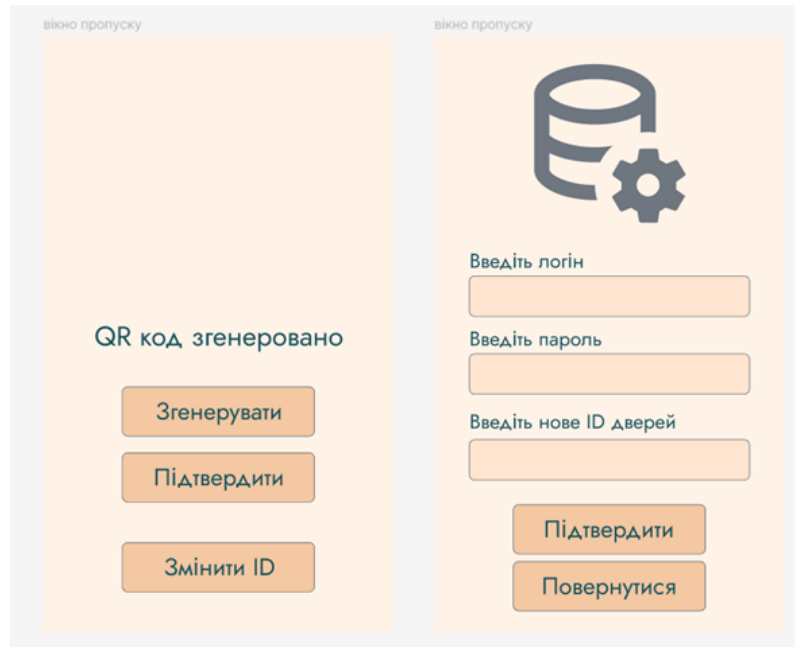


Рис. 3.8.3.1 Дизайн програми.

Також на екрані перепрошивки прив'язки до дверей у верхній частині екрану було додано векторну іконку програми у сірому кольорі.



Рис. 3.8.3.2 Іконка програми.

3.9 База Даних

В сучасному світі, де технології стають все більш розвиненими, заміна паперової документації є однією з найбільш актуальних та нагальних проблем. Одним з розумних варіантів для розв'язання цієї проблеми є використання баз даних, які можуть забезпечити доступ до інформації у будь-який час та місце, а також зменшити витрати на зберігання та обробку даних. Більш того, використання цієї технології може значно збільшити ефективність роботи та зменшити кількість помилок. Таким чином, застосування баз даних - це важлива складова в сучасному управлінні документацією та дозволяє забезпечити успішну роботу проекту.

3.9.1 Діаграма

Загальна будова нашої бази даних - у вигляді діаграми:

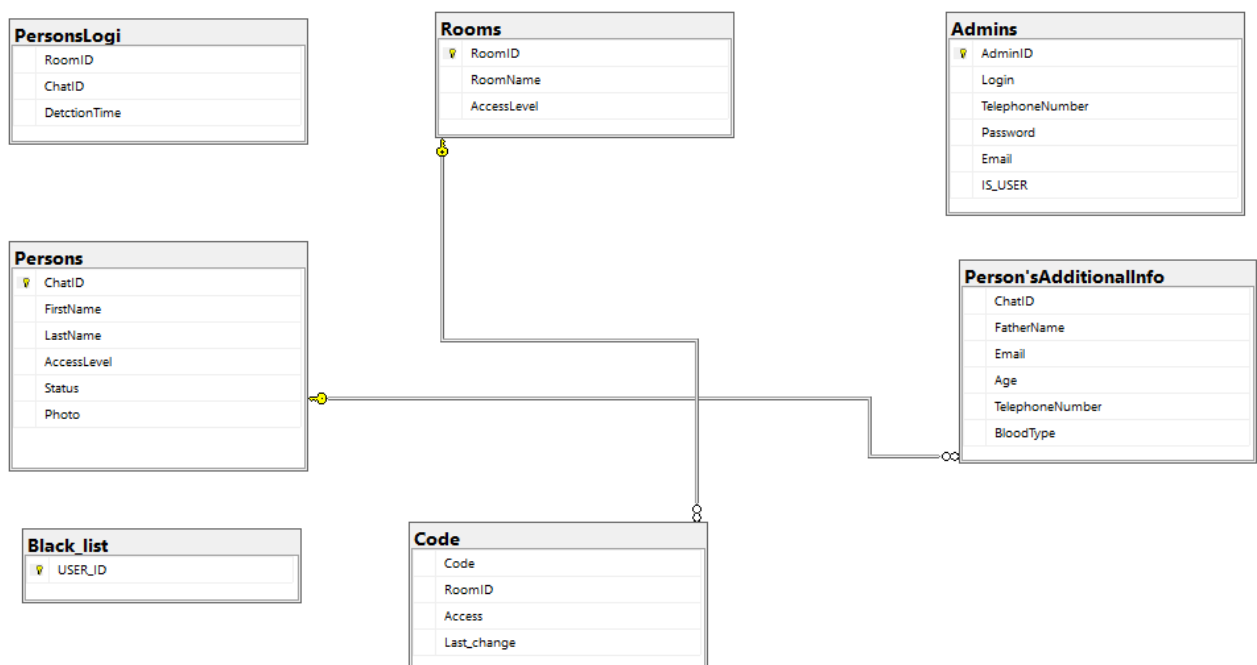


Рис. 3.9.1.1 Діаграма бази Даних.

3.9.2 Таблиці

Код описується в **ДОДАТКУ 1**

Admins - основні дані про адміністраторів

Persons - основні дані про працівників

PersonsAdditionalInfo - додаткові дані про працівників

PersonsLogins - логи, коли працівники заходили в кімнати

black_list - чорний список працівників

Rooms - кімнати з певним рівнем доступу

Code - код та внесені до нього зміни

3.9.3 Операції з даними

Код описується в **Додатку 1**

GetAdminById(admin_id INT): Повертає інформацію про адміністратора по заданому AdminID.

GetPersonById(person_id INT): Повертає інформацію про користувача по заданому CHAT_ID.

GetRoomById(room_id INT): Повертає інформацію про кімнату по заданому RoomID.

GetAllPersons(): Повертає інформацію про всіх користувачів.

UpdateAdminTelephone(admin_id INT, new_telephone INT): Оновлює номер телефону заданого адміністратора.

UpdatePersonLastName(person_id INT, new_last_name NVARCHAR(50)): Оновлює прізвище заданого користувача.

UpdateRoomAccessLevel(room_id INT, new_access_level INT): Оновлює рівень доступу до заданої кімнати.

InsertPerson(chat_id INT, first_name NVARCHAR(50), last_name NVARCHAR(50), access_level INT, status INT, photo BLOB): Вставляє нового користувача в таблицю Persons.

InsertRoom(room_name NVARCHAR(50), access_level INT): Вставляє нову кімнату в таблицю Rooms.

DeletePerson(person_id INT): Видаляє користувача з таблиці Persons.

EditPersonFirstName(person_id INT, new_first_name NVARCHAR(50)): Змінює ім'я користувача.

EditPersonStatus(person_id INT, new_status INT): Змінює статус користувача.

EditRoomName(room_id INT, new_room_name NVARCHAR(50)): Змінює назву кімнати.

EditRoomAccess(room_id INT, new_access_level INT): Змінює рівень доступу до кімнати.

EditAdminPassword(admin_id INT, new_password NVARCHAR(50)): Змінює пароль адміністратора.

3.10. Апаратне забезпечення

Наша команда планує закуповувати електронні замки і сенсорні екрани для взаємодії у німецької компанії Schlage.

Нижче наведені таблиці з технічними характеристиками цих пристроїв.

1. SCHLAGE CONTROL BE467F 150 € = 163\$ = 6160 грн



Таблиця 3.10.1

Тип:	електромагнітний
Напруга живлення:	12 В постійного струму
Сила утримування:	600 кг
Розміри:	250 x 47 x 26 мм
Матеріал корпусу:	алюміній
Інтерфейс:	NO / NC / COM

2. SCHLAGE DISPLAY SP367RR 30 € = 32.5\$ = 1232 грн



Таблиця 3.10.2

Розмір:	4 дюйми
Тип екрану:	IPS
Роздільна здатність:	1920 x 1080
Яскравість:	300 кд / м ²
Контрастність:	1000:1
Чутливість до дотику:	10 точок одночасного дотику
Інтерфейс:	HDMI, USB

4.ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ

Даний проект є економічно обґрунтованим, оскільки передбачає значну економію коштів та підвищення ефективності діяльності. Зокрема, впровадження автоматизованої системи контролю та управління витратами дозволить знизити витрати на енергопостачання та забезпечення безпеки промислового обладнання, зменшити кількість відходів та збільшити продуктивність працівників.

Ціна закупівлі складається з апаратного забезпечення, яке включає в себе сенсорні екранчики і магнітні замки до дверей, програмне забезпечення і підтримка Telegram бота та сервера БД.

Апаратне Забезпечення: 200\$ = 7500 грн на одну кімнату.

Технічне обслуговування: 300\$ = 11400 грн на місяць.

4.1 Пропонована(стартова) ціна:

Програмне забезпечення: 400\$ = 15000 грн

Апаратне Забезпечення: 350\$ = 13000 грн на одну кімнату.

Технічне обслуговування: 500\$ = 19000 грн на місяць.

У технічне обслуговування входить обслуговування ПЗ і апаратури + обслуговування бази даних та Telegram бота

4.2 Аналіз витрат

Крім того, впровадження новітніх технологій та автоматизація процесів дозволять підвищити якість продукції та конкурентоспроможність на ринку. В результаті цього зростуть обсяги продажів та прибуток компанії.

У плані фінансування проекту, його вартість буде повністю окуплена в межах трьох років завдяки заощадженням на витратах та збільшенню прибутку. Таким чином, можна стверджувати, що проект є економічно вигідним та має перспективи розвитку в майбутньому.

5. ВИКОНАВЦІ

Ярмола Юрій(Керівник групи) – реалізація телеграм бота та опис функціоналу телеграм бота.

Ширий Богдан – Програмна реалізація.

Гапонова Дарина – Дизайн програми.

Бокало Петро – Розробка функціоналу програми та телеграм бота.

Щирба Данило – Апаратне забезпечення(Hardware), економічне обґрунтування проекту

Чаус Богдан – Розробка функціоналу бази даних.

Миценко Олександр – Реалізація бази даних.

6. ВИСНОВОК

У цьому проекті було запропоновано систему пропуску QRBadge, яка надає послуги безпеки і обмежує доступ до певних ресурсів для різних користувачів. Це важливо для багатьох сфер життя, включаючи бізнес, уряд, освіту і медицину. Система пропуску QRBadge може допомогти уникнути несанкціонованого доступу, помилкового доступу та викрадення даних. Крім того, система пропуску QRBadge має можливість контролювати доступ в реальному часі та надавати швидкий доступ до ресурсів в аварійних ситуаціях. В цілому, система пропуску QRBadge є актуальною та корисною і може бути використана для забезпечення безпеки та обмеження доступу до різних ресурсів.

7. ЛІТЕРАТУРНІ ДЖЕРЕЛА

1. Джеймс Р. Грофф, Пол М. Вайнберью. SQL: повне керівництво. - 2008.
2. Діго С.М. Проектування та використання баз даних. - 2005.
3. Пасічник В.В. Організація баз даних та знань: підручник для ВНЗ/ В.В. Пасічник, В.А. Резніченко.-К.: Видавнича група BVH, 2006.-384с.

ДОДАТОК 1

Admins

```
CREATE TABLE `Admins` (  
    `AdminID` int NOT NULL AUTO_INCREMENT,  
    `Login` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT NULL,  
    `Password` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT  
NULL,  
    `IS_SUPER` tinyint(1),  
    `Email` varchar(50) NOT NULL,  
    `TelephoneNumber` varchar(50) NOT NULL,  
    PRIMARY KEY (`AdminID`)  
)
```

Persons

```
CREATE TABLE `Persons` (  
    `CHAT_ID` int NOT NULL,  
    `FirstName` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT  
NULL,  
    `LastName` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT  
NULL,  
    `AccessLevel` int NOT NULL,  
    `Status` int NOT NULL,  
    `Photo` blob,  
    PRIMARY KEY (`CHAT_ID`)  
)
```

PersonsAdditionalInfo

```
CREATE TABLE `PersonsAdditionalInfo` (  
    `CHAT_ID` int NOT NULL,
```



```

        `FatherName` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT
NULL,

        `Email` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT NULL,

        `TelephoneNumber` varchar(20) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci
NOT NULL,

        `BloodType` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT
NULL,

        `Age` varchar(50) NOT NULL,

        KEY `CHAT_ID` (`CHAT_ID`),

        CONSTRAINT `PersonsAdditionalInfo_ibfk_1` FOREIGN KEY (`CHAT_ID`) REFERENCES
`Persons` (`CHAT_ID`)
)

```

PersonsLogins

```

CREATE TABLE `PersonsLogins` (

        `RoomID` int NOT NULL,

        `CHAT_ID` int NOT NULL,

        `DetctionTime` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT
NULL

)

```

black_list

```

CREATE TABLE `black_list` (

        `USER_ID` int NOT NULL,

        PRIMARY KEY (`USER_ID`)

)

```

Rooms

```

CREATE TABLE `Rooms` (

        `RoomID` int NOT NULL AUTO_INCREMENT,

        `RoomName` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT
NULL,

```

```

        `AccessLevel` int NOT NULL,

        PRIMARY KEY (`RoomID`)

    )

```

Code

```

CREATE TABLE `Code` (

    `Code` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT NULL,

    `RoomID` int NOT NULL,

    `Access` int NOT NULL,

    `Last_change` varchar(20) NOT NULL,

    KEY `RoomID` (`RoomID`),

    CONSTRAINT `Code_ibfk_1` FOREIGN KEY (`RoomID`) REFERENCES `Rooms`
    (`RoomID`)

)

```

```

CREATE TABLE `Code` (

    `Code` varchar(50) CHARACTER SET utf8mb3 COLLATE utf8mb3_general_ci NOT NULL,

    `RoomID` int NOT NULL,

    `Access` int NOT NULL,

    `Last_change` varchar(20) NOT NULL,

    KEY `RoomID` (`RoomID`),

    CONSTRAINT `Code_ibfk_1` FOREIGN KEY (`RoomID`) REFERENCES `Rooms`
    (`RoomID`)

)

```

-- GetAdminById

```

CREATE FUNCTION GetAdminById(admin_id INT)
RETURNS TABLE (
    admin_id INT,
    telephone_number INT,
    login NVARCHAR(50),
    password NVARCHAR(50),
    is_super BOOL
)

```

```

AS $$
BEGIN
    RETURN QUERY SELECT * FROM admins WHERE AdminID = admin_id;
END;

-- GetPersonById
CREATE FUNCTION GetPersonById(person_id INT)
RETURNS TABLE (
    chat_id INT,
    first_name NVARCHAR(50),
    last_name NVARCHAR(50),
    access_level INT,
    status INT,
    photo BLOB
)
AS $$
BEGIN
    RETURN QUERY SELECT * FROM persons WHERE CHAT_ID = person_id;
END;

-- GetRoomById
CREATE FUNCTION GetRoomById(room_id INT)
RETURNS TABLE (
    room_id INT,
    room_name NVARCHAR(50),
    access_level INT
)
AS $$
BEGIN
    RETURN QUERY SELECT * FROM rooms WHERE RoomID = room_id;
END;

-- GetAllPersons
CREATE FUNCTION GetAllPersons()
RETURNS TABLE (
    chat_id INT,
    first_name NVARCHAR(50),
    last_name NVARCHAR(50),
    access_level INT,
    status INT,
    photo BLOB
)
AS $$
BEGIN
    RETURN QUERY SELECT * FROM persons;
END;

```

-- UpdateAdminTelephone

```
CREATE PROCEDURE UpdateAdminTelephone(admin_id INT, new_telephone INT)
AS $$
BEGIN
    UPDATE admins SET TelephoneNumber = new_telephone WHERE AdminID = admin_id;
END;
```

-- UpdatePersonLastName

```
CREATE PROCEDURE UpdatePersonLastName(person_id INT, new_last_name NVARCHAR(50))
AS $$
BEGIN
    UPDATE persons SET LastName = new_last_name WHERE CHAT_ID = person_id;
END;
```

-- UpdateRoomAccessLevel

```
CREATE PROCEDURE UpdateRoomAccessLevel(room_id INT, new_access_level INT)
AS $$
BEGIN
    UPDATE rooms SET AccessLevel = new_access_level WHERE RoomID = room_id;
END;
```

-- InsertPerson

```
CREATE PROCEDURE InsertPerson(chat_id INT, first_name NVARCHAR(50), last_name
NVARCHAR(50), access_level INT, status INT, photo BLOB)
AS $$
BEGIN
    INSERT INTO persons (CHAT_ID, FirstName, LastName, AccessLevel, Status, Photo) VALUES
(chat_id, first_name, last_name, access_level, status, photo);
END;
```

-- InsertRoom

```
CREATE PROCEDURE InsertRoom(room_name NVARCHAR(50), access_level INT)
AS $$
BEGIN
    INSERT INTO rooms (RoomName, AccessLevel) VALUES (room_name, access_level);
END;
```

-- DeletePerson

```
CREATE PROCEDURE DeletePerson(person_id INT)
AS $$
BEGIN
    DELETE FROM persons WHERE CHAT_ID = person_id;
```

END;

-- EditPersonFirstName

CREATE PROCEDURE EditPersonFirstName(person_id INT, new_first_name NVARCHAR(50))

AS \$\$

BEGIN

 UPDATE persons SET FirstName = new_first_name WHERE CHAT_ID = person_id;

END;

-- EditPersonStatus

CREATE PROCEDURE EditPersonStatus(person_id INT, new_status INT)

AS \$\$

BEGIN

 UPDATE persons SET Status = new_status WHERE CHAT_ID = person_id;

END;