

2023p.

Харківський національний університет радіоелектроніки

(назва вузу)

Факультет АКТ

Кафедра КІТАМ

Дисципліна Технології програмування комп'ютеризованих систем

Напрямок підготовки 6.050201 «Системна інженерія»

Курс: IV група: АКТСІ-19-1 семестр 4

ЗАВДАННЯ НА КУРСОВУ РОБОТУ

студентові Іщенко Михайлу Дмитровичу
(Прізвище, ім'я, по батькові)

1. Тема роботи «Розробка інформаційно-пошукової системи «Паспортний стіл»
2. Термін здачі студентом закінченої роботи _____
3. Вхідні дані до роботи Функція: розробка інформаційно-пошукової системи «Паспортний стіл». Перелік використаних програмних засобів: ОС Microsoft Windows 10, середовище розробки Microsoft Visual Studio 2022, мова програмування C#, MySQL Server 8.0.28. Технічне забезпечення: IBM – сумісний ПК з МП Core i5 і вище.
4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити)
4.1 Вступ. 4.2 Опис предметної області. 4.2.1 Бізнес-правила. 4.3 Проектування бази даних. 4.3.1 ER – моделювання. 4.3.2 Опис сутностей. 4.3.3 Опис зв'язків
4.4 Розробка схеми бази даних. 4.3.5 Опис таблиць бази даних 4.4 Вибір СУБД та середовища розробки 4.5 Висновки 4.6 Перелік джерел посилання 4.7 Додаток А. Текст програми. 4.8 Додаток Б. Посібник користувача
5. Дата видачі завдання: _____

КАЛЕНДАРНИЙ ПЛАН

| № | Назви етапів курсової роботи | Строк виконання етапів роботи | Примітка |
|---|--------------------------------------|-------------------------------------|-----------------|
| 1 | Отримання завдання на курсову роботу | | <i>Виконано</i> |
| 2 | Аналіз предметної області | | <i>Виконано</i> |
| 3 | Проектування бази даних | | <i>Виконано</i> |
| 4 | Опис таблиць | | <i>Виконано</i> |
| 5 | Вибір СУБД та середи розробки | | <i>Виконано</i> |
| 6 | Розробка Web-форм | | <i>Виконано</i> |
| 7 | Оформлення пояснювальної записки | | <i>Виконано</i> |
| 8 | Захист курсової роботи | | <i>Виконано</i> |

Студент _____
(підпис)

Керівник _____
(підпис)

«____» _____ 2023 р.

РЕФЕРАТ

Іщенко М.Д. Розробка програмного забезпечення інформаційно-пошукової системи «Паспортний стіл». Курсова робота з дисципліни «Технології програмування КС». Пояснювальна записка – Харків: ХНУРЕ. – 2023. – 61 с.

Розглядаються питання розробки програмного забезпечення інформаційно-пошукової системи «Паспортний стіл». Вхідними даними інформаційно-пошукової системи є команди, які надають користувачі. В залежності від натиснутої користувачем кнопки виконується той чи інший запит, який взаємодіє з базою даних. Усі введені дані зберігаються у реляційній базі даних MySQL. Програмне забезпечення реалізоване у середовищі Microsoft Visual Studio 2022.

Ключові слова: C#, .NET, SHA256, MySQL, Windows Form, ODBC, Microsoft Visual Studio 2022, Passport desk.

Ishchenko M.D. Software development of the information retrieval system "Passport desk". Course work on the discipline "CS programming technologies". Explanatory note – Kharkiv: KNURE. - 2023. – 61 с.

The paper considers the issues of software development of the information retrieval system "Passport desk". The input data of the information retrieval system are commands provided by users. Depending on the button pressed by the user, a particular query is executed that interacts with the database. All entered data is stored in a MySQL relational database. The software is implemented in the Microsoft Visual Studio 2022 environment.

Keywords: C#, .NET, SHA256, MySQL, Windows Form, ODBC, Microsoft Visual Studio 2022, Passport desk.

ЗМІСТ

| | |
|--|----|
| Перелік умовних позначень | 6 |
| Вступ..... | 7 |
| 1 Опис предметної області | 8 |
| 1.1 Бізнес-правила..... | 9 |
| 2 Проектування бази даних | 11 |
| 2.1 ER – моделювання | 11 |
| 2.2 Опис сутностей | 11 |
| 2.3 Опис зв'язків | 12 |
| 2.4 Розробка схеми бази даних | 13 |
| 2.5 Опис таблиць бази даних | 14 |
| 3 Вибір СУД та середовища розробки | 17 |
| Висновки | 19 |
| Перелік джерел посилань | 20 |
| Додаток А. Текст програми. | 21 |
| Додаток Б. Посібник користувача. | 50 |

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ, ПОЗНАЧЕНЬ, ТЕРМІНІВ

ODBC – Open Database Connectivity – ODBC stands for Open Database Connectivity, and it is a programming interface that allows applications to access and manage databases from different database management systems using a common set of functions and syntax.

ER-діаграма – модель даних, що дозволяє описувати концептуальні схеми предметної області.

SQL – Structured Query Language – декларативна мова програмування, що застосовується для створення, модифікації та управління даними в реляційній базі даних.

ВСТУП

Розробка інформаційно-пошукової системи є важливою задачею в цифрову еру. Оскільки обсяг інформації, доступної нам, зростає експоненційно, стає все складніше управляти та знаходити необхідну інформацію вчасно та ефективно. Розробка інформаційно-пошукової системи може допомогти вирішити цю проблему та зробити процес пошуку інформації більш зручним та швидким.

У даній курсовій роботі буде розглянуто процес розробки інформаційно-пошукової системи "Паспортний стіл". Основною метою системи є забезпечення зручного та швидкого пошуку необхідної інформації про населення та інші соціальні статистичні дані. В системі буде реалізовано функції пошуку за критеріями, фільтрування та сортування результатів пошуку. Буде врахована можливість введення та збереження даних про населення, що дозволить оновлювати та поповнювати базу даних.

Розробка інформаційно-пошукової системи "Паспортний стіл" має значний потенціал для застосування у соціальній сфері та управлінні містом, дозволяючи збільшити ефективність та точність збору та аналізу соціальних даних. В даній роботі буде описано процес розробки та функціональні можливості системи.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

Паспортний стіл – це спеціалізована область, яка забезпечує надання послуг щодо виготовлення, оформлення, пошуку та оновлення документів, що ідентифікації людини у країні або за її межами . Такі документи включають в себе паспорти, візи, страхові поліси та інші документи, які можуть бути потрібні для ідентифікації.

Паспортний стіл може бути частиною державного органу, такого як «Департамент міграційної служби» або «Державний департамент з паспортних справ», або відділом консульської установи, що знаходиться за кордоном. Також паспортний стіл може бути організований у складі туристичної агенції, яка спеціалізується на наданні послуг туристам, які планують подорожувати за кордоном.

Паспортний стіл забезпечує надання допомоги клієнтам у підготовці всіх необхідних документів, заповненні форм та подачі заяв на отримання паспортів та віз. Він також забезпечує відповідну інформацію про вимоги до документів.

Оскільки паспортний стіл має велике значення для підготовки документів, пов'язаних з міжнародними подорожами, його робота повинна відповідати високим стандартам якості. Тому паспортний стіл зазвичай має професійний персонал, який має глибокі знання про вимоги до документів та інші нюанси, які повинні бути відомі клієнтам.

У залежності від юрисдикції та місцезнаходження паспортного столу, в нього можуть входити різні типи послуг. Наприклад, деякі паспортні столи можуть надавати послуги з оформлення документів для виїзду за кордон, а інші можуть бути спеціалізовані на виготовленні паспортів, віз та інших документів для резидентів країни. В окремих випадках паспортний стіл або окремий його функціонал може бути цифровізованим, що дозволить зменшити обсяг черг у

відповідних закладах та спростити доступ до необхідної інформації зі зменшенням паперового оберту.

1.1 Бізнес-правила

Дана інформаційна система реалізує наступні правила роботи з вхідними в неї даними:

- 1) клієнт може сам створити свій обліковий запис;
- 2) не можна створити акаунт, з логіном який вже зареєстрований;
- 3) статус «адміністрації» присвоюється потрібному акаунту з бази даних, на програмному рівні;
- 4) будь який «id» не може бути від'ємним;
- 5) щоб зробити «insert» значень у таблиці «International_passport», «Place_of_registration», «Place_of_birth» обов'язково повинно бути значення у таблиці «Passport» із первинним ключем, який би відповідав вторинному ключу;
- 6) таблиця «Passport» у якості первинного ключа використовує ідентифікаційний номер;
- 7) поле «id_passport» таблиці «Passport» має обмеження у 9 символів, що відповідає розміру ідентифікаційного номеру;
- 8) кожна таблиця має первинний та/або вторинний ключ за допомогою яких може відбуватися будь яка доступна модифікація даних;
- 9) поле «gender» таблиці «Passport» приймає 2 значення М-чоловік та W-жінка відповідно;
- 10) будь які поля дат у базі даних використовують тип DATE з наступним прикладом оформлення уууу.mm.dd, використання іншого формату може привести до помилки;
- 11) поле «type_international_passport» таблиці «International_passport» приймає значення Р, що значить паспорт. У майбутньому функціонал програми

може бути розширений, внаслідок чого буде додано необхідну кількість типів документів;

12) поле «country_code» використовує значення класифікації держав світу відповідно;

13) поле «verified_by_tax_service» таблиці «International_passport» приймає значення 1, тобто документ пройшов перевірку державною податковою службою, або 0 – відповідно не пройшов;

14) поле «Password» таблиці «User» використовує 256-бітний алгоритм безпечного хешування.

2 ПРОЕКТУВАННЯ БАЗИ ДАНИХ

2.1 ER-моделювання

ER-модель є моделлю даних, яка дозволяє описувати концептуальні схеми, використовуючи узагальнені конструкції блоків. Це мета-модель даних, який є засобом опису моделей даних.

Модель "сутність-зв'язок" є одним з найзручніших інструментів уніфікованого представлення даних, який може бути використаний для створення всіх існуючих моделей даних (ієрархічна, мережева, реляційна, об'єктна). Ця модель є результатом систематичного процесу, який описує та визначає деяку предметну область і візуалізує її у вигляді компонентів (сутностей) та зв'язків між ними. Сутності можуть мати різні властивості (атрибути), які характеризують їх. Створені для представлення цих компонентів, атрибутів і зв'язків діаграми називаються сутність-зв'язок діаграмами.

У випадку реляційної бази даних, ER-модель може бути реалізована у вигляді таблиць, де кожен рядок є одним екземпляром сутності, а деякі поля вказують на індекси в інших таблицях, що є показниками фізичної реалізації зв'язків між сутностями.

2.2 Опис сутностей

З урахуванням проведеного аналізу предметної області були виділені наступні сутності, представлені в табл. 2.1.

Таблиця 2.1 – Опис сутностей бази даних

| Назва сутностей | Опис сутностей |
|------------------------|--|
| Passport | Зберігає інформацію про ідентифікаційний документ громадян України |
| International_passport | Зберігає інформацію про міжнародний ідентифікаційний документ |
| Place_of_registration | Зберігає інформацію про місце реєстрації громадян України |
| Place_of_birth | Зберігає інформацію про місце народження громадян України |
| User | Зберігає логіни та паролі акаунтів |

2.3 Опис зв'язків

Опис зв'язків між сутностями, які наведені у табл. 2.1, представлений у табл. 2.2.

Таблиця 2.2 – Опис зв'язків між сутностями бази даних

| Назва сутності | Найменування зв'язку | Назва сутності | Тип зв'язку |
|------------------------|----------------------|----------------|-------------|
| Place_of_birth | належить | Passport | 1:1 |
| Place_of_registration | належить | Passport | 1:1 |
| International_passport | належить | Passport | 1:1 |

Детальний опис зв'язку приведений нижче:

- а) один документ про місце народження відповідає одному паспорту
- б) один документ про місце реєстрації відповідає одному паспорту
- в) один міжнародний паспорт може відповідати одному паспорту

Модель сутність-зв'язок (ER-модель) розробленої бази даних представлена на рис. 2.1

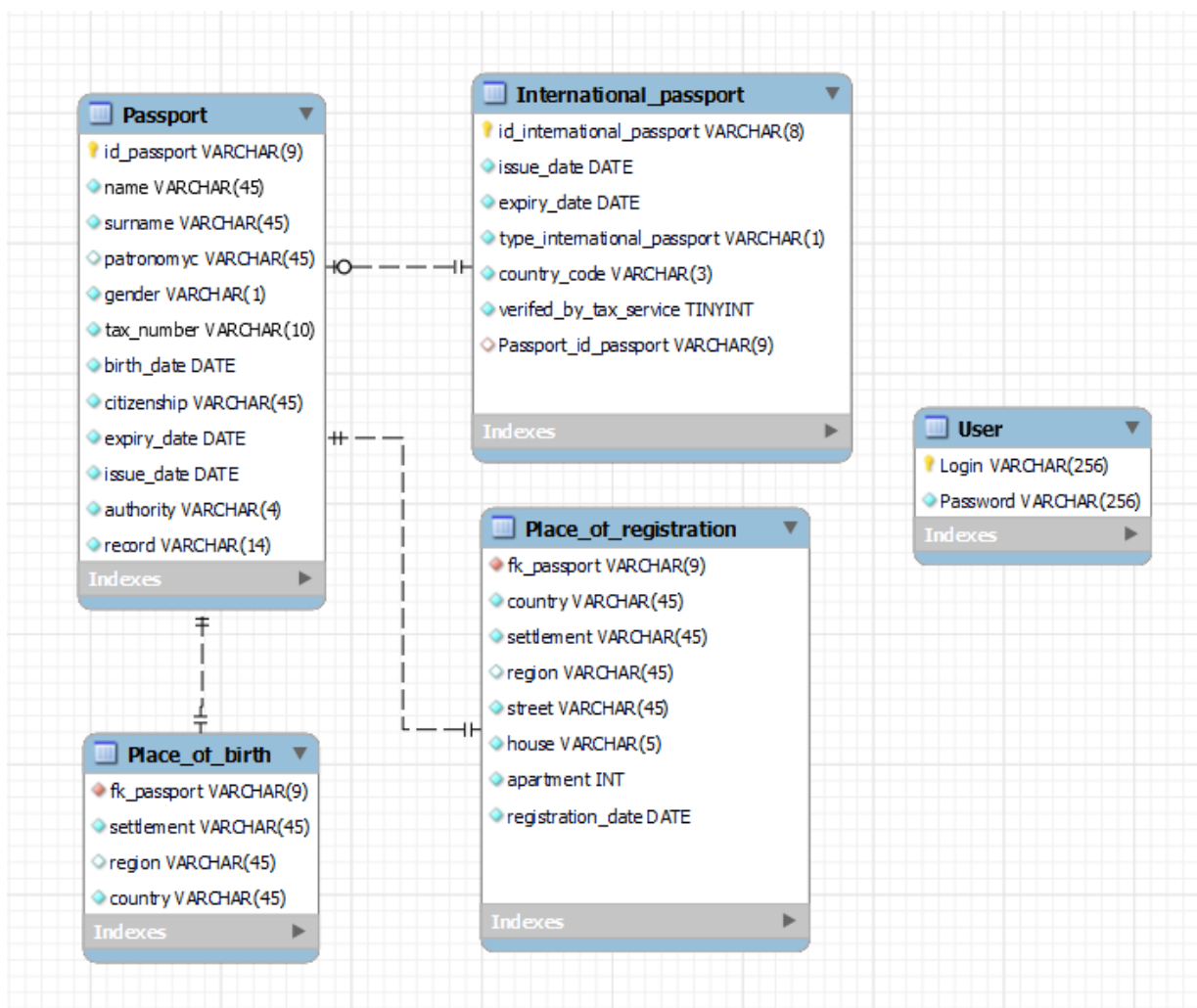


Рисунок 2.1 – ER-модель бази даних «Паспортний стіл»

2.4 Розробка схеми бази даних

Реляційна модель даних – логічна модель даних. Вперше була запропонована британським ученим співробітником компанії IBM Едгаром Франком Коддом (E. F. Codd) в 1970 році в статті «A Relational Model of Data for Large Shared Data Banks». В даний час ця модель є фактичним стандартом, на який орієнтуються практично всі сучасні комерційні системи керування базами даних (СКБД).

Можна провести аналогію між елементами реляційної моделі даних і елементами моделі «сутність-зв'язок». Реляційні відносини відповідають наборам сутностей, а кортежі – сутностям. Тому, як і в моделі «сутність-зв'язок»,

стовпці в таблиці, що представляє реляційне відношення, називають атрибутами.

Кожен атрибут визначений на домені, тому домен можна розглядати як множина допустимих значень даного атрибуту. Кілька атрибутів одних відношень і навіть атрибути різних відношень можуть бути визначені на одному і тому ж домені.

Переваги реляційної моделі:

- простота і доступність для розуміння користувачем. Єдиною використовуваною інформаційною конструкцією є «таблиця»;
- суворі правила проектування, які базуються на математичному апараті;
- повна незалежність даних. Зміни в прикладній програмі при зміні реляційної БД мінімальні;
- для організації запитів і написання прикладного ПЗ немає необхідності знати конкретну організацію БД у зовнішній пам'яті.

Недоліки реляційної моделі:

- не завжди предметна область може бути представлена у вигляді «таблиць»;
- в результаті логічного проектування з'являється множина «таблиць». Це призводить до труднощів розуміння структури даних;
- БД займає відносно багато зовнішньої пам'яті;
- відносно низька швидкість доступу до даних.

Опис таблиць бази даних представлений у табл.°2.3.

2.5 Опис таблиць бази даних

У таблиці 2.3 представлені описи усіх таблиць бази даних

Таблиця 2.3 – Опис таблиць бази даних

| Назва таблиці | Назва атрибуту | Тип даних | Розмір | NULL | Ключ |
|-----------------------|-------------------|-----------|--------|------|------|
| Passport | id_passport | VARCHAR | 9 | NN | PK |
| | name | VARCHAR | 45 | NN | |
| | surname | VARCHAR | 45 | NN | |
| | patronymc | VARCHAR | 45 | N | |
| | gender | VARCHAR | 1 | NN | |
| | tax_number | VARCHAR | 10 | NN | |
| Passport | birth_date | DATE | | NN | |
| | citizenship | VARCHAR | 45 | NN | |
| | expiry_date | DATE | | NN | |
| | issue_date | DATE | | NN | |
| | authority | VARCHAR | 4 | NN | |
| | record | VARCHAR | 4 | NN | |
| Place_of_birth | fk_passport | VARCHAR | 9 | NN | FK |
| | settlement | VARCHAR | 45 | NN | |
| | region | VARCHAR | 45 | N | |
| | country | VARCHAR | 45 | NN | |
| Place_of_registration | fk_passport | VARCHAR | 9 | NN | FK |
| | country | VARCHAR | 45 | NN | |
| | settlement | VARCHAR | 45 | NN | |
| | region | VARCHAR | 45 | N | |
| | street | VARCHAR | 45 | NN | |
| | house | VARCHAR | 5 | NN | |
| | apartment | INT | | NN | |
| | registration_date | DATE | | NN | |
| User | Login | VARCHAR | 256 | NN | PK |
| | Password | VARCHAR | 256 | NN | |

Продовження таблиці 2.3

| | | | | | |
|------------------------|-----------------------------|---------|---|----|----|
| International_passport | id_international_passport | VARCHAR | 8 | NN | PK |
| | issue_date | DATE | | NN | |
| | expiry_date | DATE | | NN | |
| | type_international_passport | VARCHAR | 1 | NN | |
| | country_code | VARCHAR | 3 | NN | |
| | verified_by_tax_service | TINYINT | | NN | |
| | Passport_id_passport | VARCHAR | 9 | N | FK |

3 ВИБІР СУБД ТА СЕРЕДОВИЩА РОЗРОБКИ

Для створення бази даних була використана середовище розробки MySQL Server з конектором ODBC.

MySQL Workbench – інструмент для візуального проєктування баз даних, що інтегрує проєктування, моделювання, створення та експлуатацію БД в єдине безшовне оточення для системи баз даних MySQL і є наступником DBDesigner 4 від FabForce.

MySQL Workbench був першим сімейством продуктів, який був доступний в двох варіантах. Щоб залучити розробників в основну команду розробки, комерційна стандартна версія програми (англ. Standard Edition) пропонується поверх вільної версії (англ. Community Edition), що розповсюджується під ліцензією GNU GPL. «Community Edition» є повнофункціональним продуктом, що володіє всіма основними можливостями комерційного варіанту. Будучи основою для всіх майбутніх релізів, він буде отримувати користь від усіх майбутніх зусиль, прикладених для розвитку продукту. «Standard Edition» розширює «Community Edition» серією модулів і плагінів, що дозволяють оптимізувати робочий процес і, тим самим, заощадити час і уникнути помилок.

Для створення програми використовувалось середовище Microsoft VisualStudio 2022. Microsoft VisualStudio 2022 – це продукт фірми Майкрософт, який містить інтегроване середовище розробки програмного забезпечення та низку інших інструментальних засобів. Цей продукт дає змогу розробляти як консольні програми, так і програми з графічним інтерфейсом, включно з підтримкою технології Windows Forms, а також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight.

Програма була написана мовою C#. Це сучасна об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET. C# дозволяє розробникам створювати безліч типів безпечних і надійних програм, які потребує екосистема.

Мова C# була вибрана тому, що – це проста і в той же час потужна мова програмування, яка дозволяє розробникам створювати багатофункціональні програми. Вона об'єднує в собі кращі ідеї сучасних мов програмування Java, Visual Basic, C ++, і т.д. Через великого розмаїття синтаксичних конструкцій і можливості працювати з платформою .NET, C# дозволяє швидше, ніж будь-яка інша мова, розробляти програмні рішення.

ВИСНОВКИ

У процесі виконання курсової роботи було розроблено й створено базу даних «Паспортний стіл». На основі зібраної інформації системи було обрано фізичну реалізацію типу реляційна модель. У реляційній моделі досягається більш високий рівень абстракції даних, ніж в ієрархічній або мережевій. Вона майже немає недоліків.

Після створення моделі було розроблено бізнес-правила, які накладають обмеження та пояснюють деякі аспекти бази даних.

Наступним етапом була розробка ER моделі бази даних. Було виявлено усі можливі сутності, що характерні для цієї предметної області. Проаналізувавши сутності було встановлено зв'язки між ними та для кожної сутності обрано певні атрибути, що відповідають їм.

Для зручного та комфортного використання бази даних було створено програму. Ця програма дозволяє шукати, додавати або видаляти дані з бази даних шляхом заповнення відповідних форм та натиснення кнопок.

Завдання на курсову роботу виконано в повному обсязі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Боуман, Дж., Емерсон, С., Дарновський, М.. Практичний посібник SQL: пер. с англ. Видавничий дім "Вільямс", 2002. 352 с.
2. Шилдт Герберт. С# 4.0: Повне керівництво / Герберт Шилдт. - Вільямс, 2011. - 1056 с.
3. ДСТУ 3008:2015. Документація. Звіти в сфері науки і техніки. Структура і правила оформлення. Державний стандарт України, 2016. 26 с.
4. Конноллі, Т., Бегт, К., Страчан, А. Бази даних: проектування, реалізація і супровід. Теорія і практика: пер. з англ. Видавничий дім «Вільямс», 2003. 1436 с.
5. Методичні вказівки до курсового проектування з дисципліни «Технології програмування комп'ютеризованих систем» для студентів усіх форм навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Системна інженерія» / Упоряд.: О.М. Цимбал, А.І. Бронніков – Харків: ХНУРЕ. – 2019. - 30 с.
6. Ріхтер Джеффри. CLR via C#. Програмування на платформі Microsoft.NET Framework 4.5 на мові C#. 4-е видання / Джеффри Ріхтер. - Пітер, 2013. - 896 с.

ДОДАТОК А

ТЕКСТ ПРОГРАМИ

Міністерство освіти і науки України

ЗАТВЕРДЖУЮ

Керівник курсової роботи

проф. каф. КІТАМ

_____ Цимбал О.М.

(підпис, дата)

ІНФОРМАЦІЙНО-ПОШУКОВА СИСТЕМА «ПАСПОРТНИЙ СТИЛ»

Текст програми

АРКУШ ЗАТВЕРДЖЕННЯ

ГЮІК.508100.0110 – 01 12 01 – ЛУ

Студент групи _____АКТСІ-19-1_____

(назва групи)

_____Іщенко М.Д.

(підпис, дата)

Харків 2023

Міністерство освіти і науки України

ЗАТВЕРДЖУЮ

ГЮІК.508100.0110 – 01 12 01 – ЛУ

ІНФОРМАЦІЙНА СИСТЕМА «ПАСПОРТНИЙ СТІЛ»

Текст програми

ГЮІК.508100.0110– 01 12 01

Аркушів 28

Харків 2023

ΓΙΟΙΚ.508100.0110– 01 12 01

Program.cs

```
using System;
using System.Windows.Forms;

namespace PassprotDeskApp
{
    internal static class Program
    {
        [STAThread]

        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new LoginForm());
        }
    }
}
```


ГЮИК.508100.0110– 01 12 01

LoginForm.cs

```

using System;
using System.Text;
using System.Windows.Forms;
using System.Security.Cryptography;
using System.Data.Odbc;

namespace PassprotDeskApp
{
    public partial class LoginForm : Form
    {
        public LoginForm()
        {
            InitializeComponent();

            private void button1_Click(object sender, EventArgs e)
            {
                Boolean security = true;
                string loginUser = textBox1.Text;
                string tryPassword = "";
                string passwordUser = "";
                SHA256 shaPasswordUser = SHA256.Create();
                byte[] b = Encoding.ASCII.GetBytes(textBox2.Text);
                byte[] hash = shaPasswordUser.ComputeHash(b);
                StringBuilder sb = new StringBuilder();
                foreach (var a in hash)
                    sb.Append(a.ToString("X2"));

                passwordUser = Convert.ToString(sb);

                string query = "SELECT `password` FROM passport_desk.`User` WHERE `login`=?" ;

                try
                {
                    using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
                    {
                        DbConnection.Open();
                        using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                        {
                            command.Parameters.AddWithValue("@login", textBox1.Text);
                            OdbcDataReader DbReader = command.ExecuteReader();
                            while (DbReader.Read())
                            {
                                tryPassword = DbReader[0].ToString();
                            }
                        }
                    }

                    query = "";
                }
            }
        }
    }
}

```

ГЮІК.508100.0110– 01 12 01

```

        catch (OdbcException exp)
        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }

        if ((loginUser == "admin") && (passwordUser.ToLower() == tryPassword))
        {
            security = false;

            Home homewindow = new Home(security);
            homewindow.Show();
            this.Hide();
        }
        else if((loginUser != "admin") && (passwordUser.ToLower() == tryPassword))
        {
            Home homewindow = new Home(security);
            homewindow.Show();
            this.Hide();
        }
        else
        {
            MessageBox.Show("Дані не підходять", "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }

    private void simpleButton1_Click(object sender, EventArgs e)
    {
        Application.Exit();
    }

    private void label4_Click(object sender, EventArgs e)
    {
        Sign_Up signUpWindow = new Sign_Up();
        signUpWindow.Show();
        this.Hide();
    }
}

```

ГЮИК.508100.0110– 01 12 01

Sign_Up.cs

```

using System;
using System.Data.Odbc;
using System.Windows.Forms;

namespace PassprotDeskApp
{
    public partial class Sign_Up : Form
    {
        public Sign_Up()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            string query = "INSERT INTO passport_desk.`User` (`login`,`password`) VALUES
(?, SHA2(?,256));";
            try
            {
                using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
                {
                    DbConnection.Open();
                    using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                    {
                        command.Parameters.AddWithValue("@login", textBox1.Text);
                        command.Parameters.AddWithValue("@password", textBox2.Text);
                        command.ExecuteNonQuery();
                    }
                }
                query = "";
                Form ifrm = Application.OpenForms[0];
                ifrm.Show();
                this.Hide();
            }
            catch (OdbcException exp)
            {
                MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            }
        }

        private void simpleButton1_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
    }
}

```

ГЮИК.508100.0110– 01 12 01

Home.cs

```

using System;
using System.Collections.Generic;
using System.Data.Odbc;
using System.Windows.Forms;

namespace PassprotDeskApp
{
    public partial class Home : Form
    {
        Boolean security;
        public Home()
        {
            InitializeComponent();
        }
        public Home(Boolean flag)
        {
            InitializeComponent();
            security = flag;
            if (security==false)
            {
                simpleButton3.Visible = true;
                simpleButton4.Visible=true;
                simpleButton9.Visible = true;
                simpleButton11.Visible = true;
                simpleButton12.Visible = true;
                simpleButton13.Visible = true;

                insert_registration.Visible = true;
                simpleButton17.Visible = true;
            }
        }

        private void simpleButton1_Click_1(object sender, EventArgs e)
        {
            Application.Exit();
        }

        //-----Select international passport
        PASSPORT-----

        private void simpleButton2_Click(object sender, EventArgs e)
        {
            dataGridView1.Rows.Clear();
            string query = "SELECT * FROM international_passport WHERE ";
            string query_set = "";
            Boolean flag = false;

            if(textEdit1.Text != "" || textEdit2.Text != "" || textEdit3.Text != "" ||
            textEdit4.Text != "" || textEdit5.Text != "" || textEdit6.Text != "")
            {
                if (textEdit1.Text != "")
                {

```

ГЮИК.508100.0110– 01 12 01

```

        query_set = "SET @id_international_passport =\"" +
textEdit1.Text.Replace(" ", "") + "\"";
        query += ("id_international_passport=@id_international_passport ");
        flag = true;
    }

    if (textEdit2.Text != "" && flag == false)
    {
        query_set = "SET @issue_date =\"" + textEdit2.Text.Replace(" ", "") +
"\\"";
        query += ("issue_date=@issue_date ");
        flag = true;
    }
    else if (textEdit2.Text != "" && flag == true)
    {
        query_set += ", @issue_date =\"" + textEdit2.Text.Replace(" ", "") +
"\\"";
        query += ("AND issue_date=@issue_date ");
    }

    if (textEdit3.Text != "" && flag == false)
    {
        query_set = "SET @expiry_date =\"" + textEdit3.Text.Replace(" ", "") +
"\\"";
        query += ("expiry_date=@expiry_date ");
        flag = true;
    }
    else if (textEdit3.Text != "" && flag == true)
    {
        query_set += ", @expiry_date =\"" + textEdit3.Text.Replace(" ", "") +
"\\"";
        query += ("AND expiry_date=@expiry_date ");
    }

    if (textEdit4.Text != "" && flag == false)
    {
        query_set = "SET @type_international_passport =\"" +
textEdit4.Text.Replace(" ", "") + "\"";
        query += ("type_international_passport = @type_international_passport
");
        flag = true;
    }
    else if (textEdit4.Text != "" && flag == true)
    {
        query_set += ", @type_international_passport =\"" +
textEdit4.Text.Replace(" ", "") + "\"";
        query += ("AND type_international_passport =
@type_international_passport ");
    }

    if (textEdit5.Text != "" && flag == false)
    {
        query_set = "SET @country_code =\"" + textEdit5.Text.Replace(" ", "") +
"\\"";

```

ГЮИК.508100.0110– 01 12 01

```

        query += ("country_code=@country_code ");
        flag = true;
    }
    else if (textEdit5.Text != "" && flag == true)
    {
        query_set += ", @country_code =\"" + textEdit5.Text.Replace(" ", "") +
"\\"";
        query += ("AND country_code=@country_code ");
    }

    if (textEdit6.Text != "" && flag == false)
    {
        query_set = "SET @verified_by_tax_service =\"" + textEdit6.Text.Replace("
", "") + "\\"";
        query += ("verified_by_tax_service=@verified_by_tax_service ");
        flag = true;
    }
    else if (textEdit6.Text != "" && flag == true)
    {
        query_set += ", @verified_by_tax_service =\"" + textEdit6.Text.Replace("
", "") + "\\"";
        query += ("AND verified_by_tax_service=@verified_by_tax_service ");
    }

    if (textEdit19.Text != "" && flag == false)
    {
        query_set = "SET @fk_passport =\"" + textEdit19.Text.Replace(" ", "") +
"\\"";
        query += ("fk_passport=@fk_passport ");
        flag = true;
    }
    else if (textEdit19.Text != "" && flag == true)
    {
        query_set += ", @fk_passport =\"" + textEdit19.Text.Replace(" ", "") +
"\\"";
        query += ("AND fk_passport=@fk_passport ");
    }

    query_set += ";";
    query += ";";
    try {

        using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
        {

            DbConnection.Open();
            OdbcCommand DbCommand = DbConnection.CreateCommand();
            DbCommand.CommandText = query;
            OdbcCommand DbCommand2 = DbConnection.CreateCommand();
            DbCommand2.CommandText = query_set;
            DbCommand2.ExecuteNonQuery();
            OdbcDataReader DbReader = DbCommand.ExecuteReader();
            List<string[]> data = new List<string[]>();

```

ГЮИК.508100.0110– 01 12 01

```

        DateTime date;
        while (DbReader.Read())
        {
            data.Add(new string[7]);

            data[data.Count - 1][0] = DbReader[0].ToString();
            date = (DateTime)DbReader[1];
            data[data.Count - 1][1] = date.ToString("yyyy.MM.dd");
            date = (DateTime)DbReader[2];
            data[data.Count - 1][2] = date.ToString("yyyy.MM.dd");
            data[data.Count - 1][3] = DbReader[3].ToString();
            data[data.Count - 1][4] = DbReader[4].ToString();
            data[data.Count - 1][5] = DbReader[5].ToString();
            data[data.Count - 1][6] = DbReader[6].ToString();
        }

        foreach (string[] s in data)
        {
            dataGridView1.Rows.Add(s);
        }
        DbReader.Close();
    }

    flag = false;
    query = "";
    query_set = "";
}
catch (OdbcException exp)
{
    MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
return;
}

//-----Select Passsport-----
private void simpleButton5_Click(object sender, EventArgs e)
{
    dataGridView2.Rows.Clear();
    string query = "SELECT * FROM passport WHERE ";
    string query_set = "";
    Boolean flag = false;

    if (textEdit7.Text != "" || textEdit8.Text != "" || textEdit9.Text != "" ||
    textEdit10.Text != "" || textEdit11.Text != "" || textEdit12.Text != ""
    || textEdit13.Text != "" || textEdit14.Text != "" || textEdit15.Text != ""
    || textEdit16.Text != "" || textEdit17.Text != "" || textEdit18.Text != "" )
    {
        if (textEdit7.Text != "")
        {
            query_set = "SET @id_passport =\"" + textEdit7.Text.Replace(" ", "") +
            "\"";

            query += ("id_passport=@id_passport ");

```

ГЮИК.508100.0110–01 12 01

```

        flag = true;
    }

    if (textEdit9.Text != "" && flag == false)
    {
        query_set = "SET @surname =\"" + textEdit9.Text.Replace(" ", "") + "\"";
        query += ("surname=@surname ");
        flag = true;
    }
    else if (textEdit9.Text != "" && flag == true)
    {
        query_set += ", @surname =\"" + textEdit9.Text.Replace(" ", "") + "\"";
        query += ("AND surname=@surname ");
    }

    if (textEdit8.Text != "" && flag == false)
    {
        query_set = "SET @name =\"" + textEdit8.Text.Replace(" ", "") + "\"";
        query += ("name=@name ");
        flag = true;
    }
    else if (textEdit8.Text != "" && flag == true)
    {
        query_set += ", @name =\"" + textEdit8.Text.Replace(" ", "") + "\"";
        query += ("AND name=@name ");
    }

    if (textEdit10.Text != "" && flag == false)
    {
        query_set = "SET @patronymc =\"" + textEdit10.Text.Replace(" ", "") + "\"";
        query += ("patronymc = @patronymc ");
        flag = true;
    }
    else if (textEdit10.Text != "" && flag == true)
    {
        query_set += ", @patronymc =\"" + textEdit10.Text.Replace(" ", "") + "\"";
        query += ("AND patronomyc = @patronymc ");
    }

    if (textEdit11.Text != "" && flag == false)
    {
        query_set = "SET @gender =\"" + textEdit11.Text.Replace(" ", "") + "\"";
        query += ("gender=@gender ");
        flag = true;
    }
    else if (textEdit11.Text != "" && flag == true)
    {
        query_set += ", @gender =\"" + textEdit11.Text.Replace(" ", "") + "\"";
        query += ("AND gender=@gender ");
    }

    if (textEdit12.Text != "" && flag == false)

```


ГЮОК.508100.0110– 01 12 01

```

{
    query_set = "SET @tax_number =\"" + textEdit12.Text.Replace(" ", "") +
"\"";
    query += ("tax_number=@tax_number ");
    flag = true;
}
else if (textEdit12.Text != "" && flag == true)
{
    query_set += ", @tax_number =\"" + textEdit12.Text.Replace(" ", "") +
"\"";
    query += ("AND tax_number=@tax_number ");
}

if (textEdit13.Text != "" && flag == false)
{
    query_set = "SET @birth_date =\"" + textEdit13.Text.Replace(" ", "") +
"\"";
    query += ("birth_date=@birth_date ");
    flag = true;
}
else if (textEdit13.Text != "" && flag == true)
{
    query_set += ", @birth_date =\"" + textEdit13.Text.Replace(" ", "") +
"\"";
    query += ("AND birth_date=@birth_date ");
}

if (textEdit14.Text != "" && flag == false)
{
    query_set = "SET @citizenship =\"" + textEdit14.Text.Replace(" ", "") +
"\"";
    query += ("citizenship=@citizenship ");
    flag = true;
}
else if (textEdit14.Text != "" && flag == true)
{
    query_set += ", @citizenship =\"" + textEdit14.Text.Replace(" ", "") +
"\"";
    query += ("AND citizenship=@citizenship ");
}

if (textEdit15.Text != "" && flag == false)
{
    query_set = "SET @expiry_date =\"" + textEdit15.Text.Replace(" ", "") +
"\"";
    query += ("expiry_date=@expiry_date ");
    flag = true;
}
else if (textEdit15.Text != "" && flag == true)
{
    query_set += ", @expiry_date =\"" + textEdit15.Text.Replace(" ", "") +
"\"";
    query += ("AND expiry_date=@expiry_date ");
}
}

```

ГЮИК.508100.0110– 01 12 01

```

if (textEdit20.Text != "")
{
    query_set = "SET @fk_passport =\"" + textEdit20.Text.Replace(" ", "") +
"\\"";

    query += ("fk_passport=@fk_passport ");
    flag = true;
}

if (textEdit21.Text != "" && flag == false)
{
    query_set = "SET @settlement =\"" + textEdit21.Text.Replace(" ", "") +
"\\"";

    query += ("settlement=@settlement ");
    flag = true;
}
else if (textEdit21.Text != "" && flag == true)
{
    query_set += ", @settlement =\"" + textEdit21.Text.Replace(" ", "") +
"\\"";

    query += ("AND settlement=@settlement ");
}

if (textEdit22.Text != "" && flag == false)
{
    query_set = "SET @region =\"" + textEdit22.Text.Replace(" ", "") + "\"";
    query += ("region=@region ");
    flag = true;
}
else if (textEdit22.Text != "" && flag == true)
{
    query_set += ", @region =\"" + textEdit22.Text.Replace(" ", "") + "\"";
    query += ("AND region=@region ");
}

if (textEdit23.Text != "" && flag == false)
{
    query_set = "SET @country =\"" + textEdit23.Text.Replace(" ", "") +
"\\"";

if (textEdit16.Text != "" && flag == false)
{
    query_set = "SET @issue_date =\"" + textEdit16.Text.Replace(" ", "") +
"\\"";

    query += ("issue_date=@issue_date ");
    flag = true;
}
else if (textEdit16.Text != "" && flag == true)
{
    query_set += ", @issue_date =\"" + textEdit16.Text.Replace(" ", "") +
"\\"";

    query += ("AND issue_date=@issue_date ");
}

if (textEdit17.Text != "" && flag == false)
{

```

ГЮИК.508100.0110– 01 12 01

```

        query_set = "SET @authority =\"" + textEdit17.Text.Replace(" ", "") +
"\\"";

        query += ("authority=@authority ");
        flag = true;
    }
    else if (textEdit17.Text != "" && flag == true)
    {
        query_set += ", @authority =\"" + textEdit17.Text.Replace(" ", "") +
"\\"";

        query += ("AND authority=@authority ");
    }

    if (textEdit18.Text != "" && flag == false)
    {
        query_set = "SET @record =\"" + textEdit18.Text.Replace(" ", "") + "\"";
        query += ("record=@record ");
        flag = true;
    }
    else if (textEdit18.Text != "" && flag == true)
    {
        query_set += ", @record =\"" + textEdit18.Text.Replace(" ", "") + "\"";
        query += ("AND record=@record ");
    }

    query_set += ";";
    query += ";";
    try
    {
        using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
        {

            DbConnection.Open();
            OdbcCommand DbCommand = DbConnection.CreateCommand();
            DbCommand.CommandText = query;
            OdbcCommand DbCommand2 = DbConnection.CreateCommand();
            DbCommand2.CommandText = query_set;
            DbCommand2.ExecuteNonQuery();
            OdbcDataReader DbReader = DbCommand.ExecuteReader();
            List<string[]> data = new List<string[]>();
            DateTime date;
            while (DbReader.Read())
            {
                data.Add(new string[12]);

                data[data.Count - 1][0] = DbReader[0].ToString();
                data[data.Count - 1][1] = DbReader[1].ToString();
                data[data.Count - 1][2] = DbReader[2].ToString();
                data[data.Count - 1][3] = DbReader[3].ToString();
                data[data.Count - 1][4] = DbReader[4].ToString();
                data[data.Count - 1][5] = DbReader[5].ToString();
                date = (DateTime)DbReader[6];
                data[data.Count - 1][6] = date.ToString("yyyy.MM.dd");
                data[data.Count - 1][7] = DbReader[7].ToString();
            }
        }
    }

```

ГЮИК.508100.0110– 01 12 01

```

        date = (DateTime)DbReader[8];
        data[data.Count - 1][8] = date.ToString("yyyy.MM.dd");
        date = (DateTime)DbReader[9];
        data[data.Count - 1][9] = date.ToString("yyyy.MM.dd");
        data[data.Count - 1][10] = DbReader[10].ToString();
        data[data.Count - 1][11] = DbReader[11].ToString();
    }

    foreach (string[] s in data)
    {
        dataGridView2.Rows.Add(s);
    }
    DbReader.Close();

}

flag = false;
query = "";
query_set = "";

}
catch (OdbcException exp)
{
    MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}
return;
}

private void simpleButton6_Click(object sender, EventArgs e)
{
    dataGridView3.Rows.Clear();
    string query = "SELECT * FROM Place_of_birth WHERE ";
    string query_set = "";
    Boolean flag = false;

    if (textEdit20.Text != "" || textEdit21.Text != "" || textEdit22.Text != "" ||
    textEdit23.Text != "")
    {
        query += ("country = @country ");
        flag = true;
    }
    else if (textEdit23.Text != "" && flag == true)
    {
        query_set += ", @country =\"" + textEdit23.Text.Replace(" ", "") + "\"";
        query += ("AND country = @country ");
    }

    query_set += ";";
    query += ";";
    try
    {

```

ГЮИК.508100.0110– 01 12 01

```

        using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
        {

            DbConnection.Open();
            OdbcCommand DbCommand = DbConnection.CreateCommand();
            DbCommand.CommandText = query;
            OdbcCommand DbCommand2 = DbConnection.CreateCommand();
            DbCommand2.CommandText = query_set;
            DbCommand2.ExecuteNonQuery();
            OdbcDataReader DbReader = DbCommand.ExecuteReader();
            List<string[]> data = new List<string[]>();
            DateTime date;
            while (DbReader.Read())
            {
                data.Add(new string[4]);

                data[data.Count - 1][0] = DbReader[0].ToString();
                data[data.Count - 1][1] = DbReader[1].ToString();
                data[data.Count - 1][2] = DbReader[2].ToString();
                data[data.Count - 1][3] = DbReader[3].ToString();
            }

            foreach (string[] s in data)
            {
                dataGridView3.Rows.Add(s);
            }
            DbReader.Close();

        }

        flag = false;
        query = "";
        query_set = "";

    }
    catch (OdbcException exp)
    {
        MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
    }
}
return;
}
private void simpleButton7_Click(object sender, EventArgs e)
{
    dataGridView4.Rows.Clear();
    string query = "SELECT * FROM Place_of_registration WHERE ";
    string query_set = "";
    Boolean flag = false;
    if (textEdit24.Text != "" || textEdit25.Text != "" || textEdit26.Text != "" ||
textEdit27.Text != "" || textEdit28.Text != "" || textEdit29.Text != "" ||
        textEdit30.Text != "" || textEdit31.Text != "")
    {

```

ГЮИК.508100.0110– 01 12 01

```

if (textEdit24.Text != "")
{
    query_set = "SET @fk_passport =\"" + textEdit24.Text.Replace(" ", "") +
"\"";
    query += ("fk_passport=@fk_passport ");
    flag = true;
}

if (textEdit25.Text != "" && flag == false)
{
    query_set = "SET @country =\"" + textEdit25.Text.Replace(" ", "") +
"\"";
    query += ("country=@country ");
    flag = true;
}
else if (textEdit25.Text != "" && flag == true)
{
    query_set += ", @country =\"" + textEdit25.Text.Replace(" ", "") + "\"";
    query += ("AND country=@country ");
}

if (textEdit26.Text != "" && flag == false)
{
    query_set = "SET @settlement =\"" + textEdit26.Text.Replace(" ", "") +
"\"";
    query += ("settlement=@settlement ");
    flag = true;
}
else if (textEdit26.Text != "" && flag == true)
{
    query_set += ", @settlement =\"" + textEdit26.Text.Replace(" ", "") + "\"";
    query += ("AND settlement=@settlement ");
}

if (textEdit27.Text != "" && flag == false)
{
    query_set = "SET @region =\"" + textEdit27.Text.Replace(" ", "") + "\"";
    query += ("region = @region ");
    flag = true;
}
else if (textEdit27.Text != "" && flag == true)
{
    query_set += ", @region =\"" + textEdit27.Text.Replace(" ", "") + "\"";
    query += ("AND region = @region ");
}

if (textEdit28.Text != "" && flag == false)
{
    query_set = "SET @street =\"" + textEdit28.Text.Replace(" ", "") + "\"";
    query += ("street=@street ");
    flag = true;
}
else if (textEdit28.Text != "" && flag == true)

```

ГЮИК.508100.0110–01 12 01

```

        {
            query_set += ", @street =\"" + textEdit28.Text.Replace(" ", "") + "\"";
            query += ("AND street=@street ");
        }

        if (textEdit29.Text != "" && flag == false)
        {
            query_set = "SET @house =\"" + textEdit29.Text.Replace(" ", "") + "\"";
            query += ("house=@house ");
            flag = true;
        }
        else if (textEdit29.Text != "" && flag == true)
        {
            query_set += ", @house =\"" + textEdit29.Text.Replace(" ", "") + "\"";
            query += ("AND house=@house ");
        }

        if (textEdit30.Text != "" && flag == false)
        {
            query_set = "SET @apartment =\"" + textEdit30.Text.Replace(" ", "") +
"\"";

            query += ("apartment=@apartment ");
            flag = true;
        }
        else if (textEdit30.Text != "" && flag == true)
        {
            query_set += ", @apartment =\"" + textEdit30.Text.Replace(" ", "") +
"\"";

            query += ("AND apartment=@apartment ");
        }

        if (textEdit31.Text != "" && flag == false)
        {
            query_set = "SET @registration_date =\"" + textEdit31.Text.Replace(" ",
"") + "\"";

            query += ("registration_date=@registration_date ");
            flag = true;
        }
        else if (textEdit31.Text != "" && flag == true)
        {
            query_set += ", @registration_date =\"" + textEdit31.Text.Replace(" ",
"") + "\"";

            query += ("AND registration_date=@registration_date ");
        }

        query_set += ";";
        query += ";";
        try
        {
            using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
            {

```

ГЮИК.508100.0110– 01 12 01

```

        DbConnection.Open();
        OdbcCommand DbCommand = DbConnection.CreateCommand();
        DbCommand.CommandText = query;
        OdbcCommand DbCommand2 = DbConnection.CreateCommand();
        DbCommand2.CommandText = query_set;
        DbCommand2.ExecuteNonQuery();
        OdbcDataReader DbReader = DbCommand.ExecuteReader();
        List<string[]> data = new List<string[]>();
        DateTime date;
        while (DbReader.Read())
        {
            data.Add(new string[8]);

            data[data.Count - 1][0] = DbReader[0].ToString();
            data[data.Count - 1][1] = DbReader[1].ToString();
            data[data.Count - 1][2] = DbReader[2].ToString();
            data[data.Count - 1][3] = DbReader[3].ToString();
            data[data.Count - 1][4] = DbReader[4].ToString();
            data[data.Count - 1][5] = DbReader[5].ToString();
            data[data.Count - 1][6] = DbReader[6].ToString();
            date = (DateTime)DbReader[7];
            data[data.Count - 1][7] = date.ToString("yyyy.MM.dd");
        }

        foreach (string[] s in data)
        {
            dataGridView4.Rows.Add(s);
        }
        DbReader.Close();

    }

    flag = false;
    query = "";
    query_set = "";

    }
    catch (OdbcException exp)
    {
        MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
    }
    return;
}

//-----INSERT Place_of_birth PASSPORT-----
//-----

private void simpleButton12_Click(object sender, EventArgs e)
{
    if (textEdit20.Text != "" && textEdit21.Text != "" && textEdit22.Text != "" &&
    textEdit23.Text != "")
    {

```


ГЮИК.508100.0110– 01 12 01

```

        string query = "INSERT INTO passport_desk.Place_of_birth VALUES (?, ?, ?, ?)";
        try
        {
            using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
            {
                DbConnection.Open();
                using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                {
                    command.Parameters.AddWithValue("@fk_passport",
textEdit20.Text);
                    command.Parameters.AddWithValue("@settlement", textEdit21.Text);
                    command.Parameters.AddWithValue("@region", textEdit22.Text);
                    command.Parameters.AddWithValue("@country", textEdit23.Text);

                    command.ExecuteNonQuery();
                }
            }
            query = "";
        }
        catch (OdbcException exp)
        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
        return;
    } else if(textEdit20.Text != "" && textEdit21.Text != "" && textEdit22.Text ==
"" && textEdit23.Text != "")
    {
        string query = "INSERT INTO passport_desk.Place_of_birth VALUES
(?, ?, \"\", ?)";
        try
        {
            using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
            {
                DbConnection.Open();
                using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                {
                    command.Parameters.AddWithValue("@fk_passport",
textEdit20.Text);
                    command.Parameters.AddWithValue("@settlement", textEdit21.Text);
                    command.Parameters.AddWithValue("@country", textEdit23.Text);

                    command.ExecuteNonQuery();
                }
            }
        }
        query = "";
    }

```

ГЮИК.508100.0110– 01 12 01

```

        }
        catch (OdbcException exp)
        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
        return;
    }
}

//-----INSERT INTERNATIONAL PASSPORT-----
-----

private void simpleButton3_Click(object sender, EventArgs e)
{
    if (textEdit1.Text != "" && textEdit2.Text != "" && textEdit3.Text != "" &&
textEdit4.Text != "" && textEdit5.Text != "" && textEdit6.Text != ""
    && textEdit19.Text != "")
    {
        string query = "INSERT INTO passport_desk.International_passport VALUES
(?,?,?,?,?,?,?)";
        try
        {
            using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
            {
                DbConnection.Open();
                using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                {
                    command.Parameters.AddWithValue("@id_international_passport",textEdit1.Text);

                    command.Parameters.AddWithValue("@issue_date", textEdit2.Text);
                    command.Parameters.AddWithValue("@expiry_date", textEdit3.Text);
                    command.Parameters.AddWithValue("@type_international_passport",
textEdit4.Text);
                    command.Parameters.AddWithValue("@country_code",
textEdit5.Text);
                    command.Parameters.AddWithValue("@verified_by_tax_service",
textEdit6.Text);
                    command.Parameters.AddWithValue("@fk_passport",
textEdit19.Text);

                    command.ExecuteNonQuery();
                }
            }
        }
        query = "";
    }
}

```

ГЮИК.508100.0110– 01 12 01

```

        catch (OdbcException exp)
        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
        }
    }
    return;
}

//-----INSERT PASSPORT-----
//-----

private void simpleButton4_Click(object sender, EventArgs e)
{
    if (textEdit7.Text != "" && textEdit8.Text != "" && textEdit9.Text != "" &&
    textEdit10.Text == "" && textEdit11.Text != "" && textEdit12.Text != ""
    && textEdit13.Text != "" && textEdit14.Text != "" && textEdit15.Text != ""
    && textEdit16.Text != "" && textEdit17.Text != "" && textEdit18.Text != "")
    {
        string query = "INSERT INTO passport_desk.Passport VALUES
        (?, ?, ?, \"\", ?, ?, ?, ?, ?, ?, ?)";
        try
        {
            using (OdbcConnection DbConnection = new
            OdbcConnection("DSN=passport_desk"))
            {
                DbConnection.Open();
                using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                {
                    command.Parameters.AddWithValue("@id_passport", textEdit7.Text);
                    command.Parameters.AddWithValue("@name", textEdit8.Text);
                    command.Parameters.AddWithValue("@surname", textEdit9.Text);
                    command.Parameters.AddWithValue("@gender", textEdit11.Text);
                    command.Parameters.AddWithValue("@tax_number", textEdit12.Text);
                    command.Parameters.AddWithValue("@birth_date", textEdit13.Text);
                    command.Parameters.AddWithValue("@citizenship",
                    textEdit14.Text);
                    command.Parameters.AddWithValue("@expiry_date",
                    textEdit15.Text);
                    command.Parameters.AddWithValue("@issue_date", textEdit16.Text);
                    command.Parameters.AddWithValue("@authority", textEdit17.Text);
                    command.Parameters.AddWithValue("@record", textEdit18.Text);

                    command.ExecuteNonQuery();
                }
            }
            query = "";
        }
    }
    catch (OdbcException exp)

```

ГЮИК.508100.0110– 01 12 01

```

        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
        return;
    }
    else if (textEdit7.Text != "" && textEdit8.Text != "" && textEdit9.Text != "" &&
textEdit11.Text != "" && textEdit12.Text != ""
        && textEdit13.Text != "" && textEdit14.Text != "" && textEdit15.Text != ""
&& textEdit16.Text != "" && textEdit17.Text != "" && textEdit18.Text != ""
        && textEdit10.Text != "")
    {
        string query = "INSERT INTO passport_desk.Passport VALUES
(?,?,?,?,?,?,?,?,?,?,?,?,?)";
        try
        {
            using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
            {
                DbConnection.Open();
                using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                {
                    command.Parameters.AddWithValue("@id_passport", textEdit7.Text);
                    command.Parameters.AddWithValue("@name", textEdit8.Text);
                    command.Parameters.AddWithValue("@surname", textEdit9.Text);
                    command.Parameters.AddWithValue("@patronymic", textEdit10.Text);
                    command.Parameters.AddWithValue("@gender", textEdit11.Text);
                    command.Parameters.AddWithValue("@tax_number", textEdit12.Text);
                    command.Parameters.AddWithValue("@birth_date", textEdit13.Text);
                    command.Parameters.AddWithValue("@citizenship",
textEdit14.Text);
                    command.Parameters.AddWithValue("@expiry_date",
textEdit15.Text);
                    command.Parameters.AddWithValue("@issue_date", textEdit16.Text);
                    command.Parameters.AddWithValue("@authority", textEdit17.Text);
                    command.Parameters.AddWithValue("@record", textEdit18.Text);

                    command.ExecuteNonQuery();
                }
            }
            query = "";
        }
        catch (OdbcException exp)
        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
        return;
    }
}

```

ГЮІК.508100.0110– 01 12 01

```

    }

//-----INSERT Place_of_registration-----
//-----
//Треба додати IF
private void simpleButton15_Click(object sender, EventArgs e)
{
    if (textEdit24.Text != "" && textEdit25.Text != "" && textEdit26.Text != "" &&
textEdit27.Text != "" && textEdit28.Text != "" && textEdit29.Text != "" &&
textEdit30.Text != "" && textEdit31.Text != "")
    {
        string query = "INSERT INTO passport_desk.Place_of_registration VALUES
(?,?,?,?,?,?,?,?)";
        try
        {
            using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
            {
                DbConnection.Open();
                using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                {
                    command.Parameters.AddWithValue("@fk_passport",
textEdit24.Text);

                    command.Parameters.AddWithValue("@country", textEdit25.Text);
                    command.Parameters.AddWithValue("@settlement", textEdit26.Text);
                    command.Parameters.AddWithValue("@region", textEdit27.Text);
                    command.Parameters.AddWithValue("@street", textEdit28.Text);
                    command.Parameters.AddWithValue("@house", textEdit29.Text);
                    command.Parameters.AddWithValue("@apartment", textEdit30.Text);
                    command.Parameters.AddWithValue("@registration_date",
textEdit31.Text);

                    command.ExecuteNonQuery();
                }
            }
            query = "";
        }
        catch (OdbcException exp)
        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
        return;
    }
    else if(textEdit24.Text != "" && textEdit25.Text != "" && textEdit26.Text != ""
&& textEdit27.Text == "" && textEdit28.Text != "" && textEdit29.Text != ""
&& textEdit30.Text != "" || textEdit31.Text != "")
    {
        string query = "INSERT INTO passport_desk.Place_of_registration VALUES
(?,?,?,?,,\"\",?,?,?,?)";
    }
}

```

ГЮІК.508100.0110– 01 12 01

```

        try
        {
            using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
            {
                DbConnection.Open();
                using (OdbcCommand command = new OdbcCommand(query, DbConnection))
                {
                    command.Parameters.AddWithValue("@fk_passport",
textEdit24.Text);
                    command.Parameters.AddWithValue("@country", textEdit25.Text);
                    command.Parameters.AddWithValue("@settlement", textEdit26.Text);
                    command.Parameters.AddWithValue("@street", textEdit28.Text);
                    command.Parameters.AddWithValue("@house", textEdit29.Text);
                    command.Parameters.AddWithValue("@apartment", textEdit30.Text);
                    command.Parameters.AddWithValue("@registration_date",
textEdit31.Text);

                    command.ExecuteNonQuery();
                }
            }
            query = "";
        }
        catch (OdbcException exp)
        {
            MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
        return;
    }
}

//-----Видалення місця народження-----

private void simpleButton13_Click(object sender, EventArgs e)
{
    string query = "DELETE FROM passport_desk.Place_of_birth WHERE fk_passport = ?";
    try
    {
        using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
        {
            DbConnection.Open();
            using (OdbcCommand command = new OdbcCommand(query, DbConnection))
            {
                command.Parameters.AddWithValue("@fk_passport",
dataGridView3.SelectedRows[0].Cells[0].Value.ToString());

```

ГЮИК.508100.0110– 01 12 01

```

        command.ExecuteNonQuery();
    }

    }
    query = "";

    }
    catch (OdbcException exp)
    {
        MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
    return;
}

private void simpleButton17_Click(object sender, EventArgs e)
{
    string query = "DELETE FROM passport_desk.Place_of_registration WHERE
fk_passport = ?";
    try
    {
        using (OdbcConnection DbConnection = new
        OdbcConnection("DSN=passport_desk"))
        {
            DbConnection.Open();
            using (OdbcCommand command = new OdbcCommand(query, DbConnection))
            {
                command.Parameters.AddWithValue("@fk_passport",
dataGridView4.SelectedRows[0].Cells[0].Value.ToString());
                command.ExecuteNonQuery();
            }
        }
        query = "";
    }
    catch (OdbcException exp)
    {
        MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
    return;
}

private void simpleButton11_Click(object sender, EventArgs e)
{
    string query = "DELETE FROM passport_desk.International_passport WHERE
id_international_passport = ?";
    try
    {
        using (OdbcConnection DbConnection = new

```

ГЮИК.508100.0110– 01 12 01

```

OdbcConnection("DSN=passport_desk"))
    {
        DbConnection.Open();
        using (OdbcCommand command = new OdbcCommand(query, DbConnection))
        {
            command.Parameters.AddWithValue("@id_passport",
dataGridView1.SelectedRows[0].Cells[0].Value.ToString());
            command.ExecuteNonQuery();
        }

        query = "";
    }
    catch (OdbcException exp)
    {
        MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
    }
    return;
}

private void simpleButton9_Click(object sender, EventArgs e)
{
    string query_passport = "DELETE FROM passport_desk.Passport WHERE id_passport =
?";
    string query_birth = "DELETE FROM passport_desk.Place_of_birth WHERE fk_passport
= ?";
    string query_registration = "DELETE FROM passport_desk.Place_of_registration
WHERE fk_passport = ?";
    try
    {
        using (OdbcConnection DbConnection = new
OdbcConnection("DSN=passport_desk"))
        {
            DbConnection.Open();
            using (OdbcCommand command2 = new OdbcCommand(query_birth,
DbConnection))
            {
                command2.Parameters.AddWithValue("@fk_passport",
dataGridView2.SelectedRows[0].Cells[0].Value.ToString());
                command2.ExecuteNonQuery();
            }
            using (OdbcCommand command3 = new OdbcCommand(query_registration,
DbConnection))
            {
                command3.Parameters.AddWithValue("@fk_passport",
dataGridView2.SelectedRows[0].Cells[0].Value.ToString());
                command3.ExecuteNonQuery();
            }
            using (OdbcCommand command = new OdbcCommand(query_passport,

```


ГЮИК.508100.0110– 01 12 01

```
DbConnection))
    {
        command.Parameters.AddWithValue("@id_passport",
dataGridView2.SelectedRows[0].Cells[0].Value.ToString());
        command.ExecuteNonQuery();
    }
    query_passport = query_birth = query_registration = "";
}
catch (OdbcException exp)
{
    MessageBox.Show(exp.ToString(), "Помилка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
}
return;
}
}
```

ДОДАТОК Б

ПОСІБНИК КОРИСТУВАЧА

Міністерство освіти і науки України

ЗАТВЕРДЖУЮ

Керівник курсової роботи

проф. каф. КІТАМ

_____ Цимбал О.М.

(підпис, дата)

ІНФОРМАЦІЙНА СИСТЕМА «Паспортний стіл»

Керівництво користувача

АРКУШ ЗАТВЕРДЖЕННЯ

ГЮІК.508100.0110 – 01 12 01 – ЛУ

Студент групи _____ АКТСІ-19-1

(назва групи)

_____ Іщенко М.Д.

(підпис, дата)

Харків 2023

Міністерство освіти і науки України

ЗАТВЕРДЖУЮ

ГЮІК.508100.0110 – 01 12 01 – ЛУ

ІНФОРМАЦІЙНА СИСТЕМА «ПАСПОРТНИЙ СТІЛ»

Посібник користувача

ГЮІК.508100.0110– 01 12 01

Аркушів 11

Харків 2023

ЗМІСТ

| | |
|-------------------------------------|----|
| Вступ | 54 |
| 1. Призначення і умови використання | 54 |
| 2. Підготовка до роботи | 54 |
| 3. Опис програми | 55 |
| 4. Аварійні ситуації | 61 |

ВСТУП

Документ «Керівництво користувача» використовується для ознайомлення з основними функціями і можливостями програми.

Основна мета створеної програми оцифровувати основні функції паспортного столу, тобто створена програма допомагає користувачу або персоналу шукати, додавати та видаляти інформацію.

Для роботи з програмою користувач повинен мати базові навички володіння комп'ютером. Для отримання можливості використати весь функціонал програми, користувач повинен бути зареєстрованим.

Б.1 ПРИЗНАЧЕННЯ І УМОВИ ВИКОРИСТАННЯ

Розроблена програма призначена для того, щоб клієнти могли не виходячи з дому мати можливість отримати необхідні послуги для успішної ідентифікації, створена програма допомагає користувачу або персоналу шукати, додавати та видаляти інформацію.

Використання даної програми можливе лише при наявності персонального комп'ютера та встановленої програми з підключенням до серверу «MySQL Community Server», на якому розташовується база даних.

Б.2 ПІДГОТОВКА ДО РОБОТИ

Коректна робота програми можлива при наявності з'єднання з сервером «MySQL Community Server» та з розташованою на ньому базою даних «Паспортний стіл».

Для початку програми потрібно запустити файл з назвою програми та розширенням .exe.

Б.3 ОПИС ПРОГРАМИ

Програма запускається з вікна авторизації. Зовнішній вигляд вікна авторизації приведений на рис. Б.3.1.

The image shows a login window with a dark blue background. At the top right is a close button (X). In the center, there is a large white 'P' logo and the text 'LOG IN' in white. Below the logo, there are two input fields: 'Username' and 'Password'. At the bottom, there are two buttons: 'Log in' (blue) and 'Sign up' (white with blue text).

Рисунок Б.3.1 – Вікно авторизації

Користувач повинен ввести логін та пароль, натиснути кнопку «Log in» щоб увійти до свого кабінету. При введенні невірних даних програма поверне помилку, що дані невірні. Якщо користувач ще не має свого облікового запису, то потрібно натиснути на кнопку «Sign up», щоб відкрити вікно реєстрації, зовнішній вигляд показаний на рис. Б.3.2.

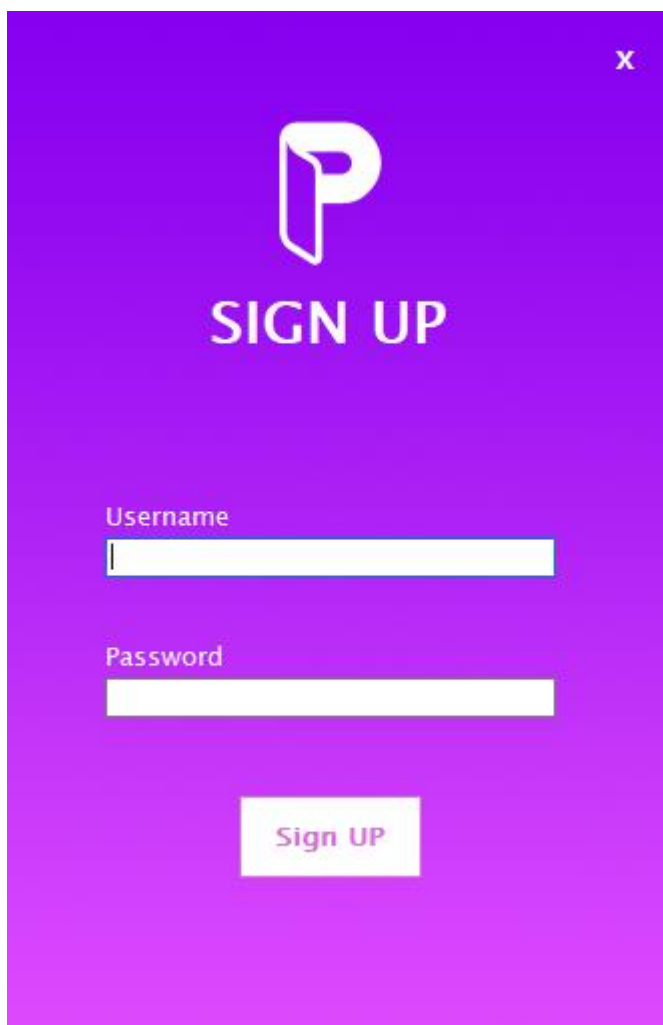


Рисунок Б.3.2 – Вікно авторизації

Користувач повинен заповнити усі поля для реєстрації, у противному випадку програма повертає помилку. При натисканні на кнопку «Sign UP» користувач дає згоду на обробку його персональних даних та програма заносить новостворений обліковий запис до бази даних, при успішному виконанні цієї операції програма повертає інтерфейс вікна авторизації, де клієнт може ввести дані новоствореного акаунту. Також, якщо користувач використовує логін, що вже є в базі даних програма поверне помилку.

Після авторизації користувач потрапляє на головне вікно, зовнішній вигляд якого залежить від статусу користувача. Звичайна форма відображена на рис. Б.3.3., адміністраторська на Б.3.4

Рисунок Б.3.3 – Вікно «Головне паспорт» для звичайного користувача

Рисунок Б.3.4 – Вікно «Головне паспорт» для користувача з адміністраторськими правами

У головному вікні розташовані 4 вкладки, кожна з яких відповідає конкретному документу та кнопка «Пошук» з текстовими полями для пошуку.

Кнопка «Закордонний паспорт» відправляє користувача до вікна, звичайна форма відображена на рис. Б.3.5., адміністраторська на Б.3.6. У цьому вікні можна переглянути інформацію про закордонний паспорт.

Також можна вийти з програми, натиснувши кнопку «X».

Рисунок Б.3.5 – Вікно «Головне закордонний паспорт» для звичайного користувача

Рисунок Б.3.6 – Вікно «Головне закордонний паспорт» для користувача з адміністраторськими правами

За допомогою кнопки «Місце народження» користувач відправляється до вікна обробки інформації відповідного документу, звичайна форма відображена на рис. Б.3.7., адміністраторська на Б.3.8.

The screenshot shows a web application interface. On the left, there is a vertical sidebar with four buttons: 'Паспорт', 'Закордонний паспорт', 'Місце народження', and 'Місце реєстрації'. The 'Місце народження' button is highlighted. The main area of the window is divided into two sections. The top section contains a table with four columns: 'ID_Паспорт', 'Населений пункт', 'Область', and 'Країна'. The table body is currently empty. The bottom section contains four input fields labeled 'ID_Паспорт', 'Населений пункт', 'Область', and 'Країна', followed by a blue button labeled 'Пошук'.

Рисунок Б.3.7 – Вікно «Головне місце народження» для звичайного користувача

This screenshot shows the same web application interface as Figure Б.3.7, but for an administrator user. The sidebar and the top table section are identical. However, the bottom section contains a different set of controls: the 'Пошук' button is at the top, and below it are two buttons, 'Додати' and 'Видалити', which are not present in the regular user version.

Рисунок Б.3.8 – Вікно «Головне місце народження» для користувача з адміністраторськими правами

За допомогою кнопки «Місце реєстрації» користувач відправляється до вікна обробки інформації відповідного документу. звичайна форма відображена на рис. Б.3.9., адміністраторська на Б.3.10.

Рисунок Б.3.9 – Вікно «Головне місце реєстрації» для звичайного користувача

Рисунок Б.3.10 – Вікно «Головне місце реєстрації» для користувача з адміністраторськими правами

Б.4 АВАРІЙНІ СИТУАЦІЇ

При введенні неіснуючих даних у вікно авторизації програма повертає помилку.

Якщо заповнені не усі обов'язкові поля при додаванні значення, програма повертає помилку.

Якщо унікальні поля при додаванні значення заповненні вже існуючими значеннями, програма повертає помилку.

Якщо формат дати відрізняється від уууу.мм.дд програма може повернути помилку.