МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС

"ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ"

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

ФАКУЛЬТЕТ СИСТЕМНИХ ДОСЛІДЖЕНЬ

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

КУРСОВА РОБОТА

З дисципліни: «БАЗИ ДАНИХ»

На тему: “Ігротека”

Виконали: студенти 3-го курсу

гр. КА-33

Німець Анастасія

Лєднікова Анна

Прийняла:

Яковлева Т.В.

Київ – 2016

**ЗМІСТ**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ,СКОРОЧЕНЬ І ТЕРМІНІВ 3

АНОТАЦІЯ 4

ВСТУП 5

1. ПОСТАНОВКА ЗАДАЧІ 6

2. АРХІТЕКТУРА ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ АСОД 7

2.1 DFD0 – діаграма 8

2.2 DFD1 – діаграма 9

2.3 DFD2 – діаграма 10

2.4 Матриця подій 11

2.5 Бізнес правила 13

2.6 Інфологічне проектування 13

2.7 Датологічне проектування 14

3 ОПИС ПРОГРАМИ 15

ВИСНОВОК 30

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 31

ДОДАТКИ 32

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ**

БД – База Даних

АСОД – Автоматизована Система Обробки Даних

ІС – Інформаційна система

**АНОТАЦІЯ**

В даній роботі розроблена і описана інформаційна система «Ігротека». Метою створення цього проекту була автоматизація роботи з клієнтами, співробітниками та товарами, реалізація основних функцій, таких як реєстрація клієнтів та додавання нових товарів, перегляд уже існуючих з можливістю швидкого пошуку за допомогою фільтрів, тощо. Також оператори можуть переглядати різноманітну інформацію: істнуючі кімнати, послуги, список поточних замовлень ігротеки та його клієнтів або за весь період, список працівників, посад, вільних посад тощо.

Програмне забезпечення має інтуїтивно зрозумілий, простий та зручний інтерфейс та призначена для персоналу, що має навички роботи з комп’ютером, зокрема з операційною системою Windows. Передбачена можливість входу до програмного продукту з правами адміністратора або оператора.

**ВСТУП**

Створена автоматизована система обробки даних є достатньо ефективною і надає основну функціональність для нормальної роботи підприємства. Крім операцій пов’язаних з рухом товарів, вона надає інструменти для відстеження зайнятості персоналу в певний час, вільних на даний момент посад, відображення кар’єри працівників на підприємстві тощо. Користувацький інтерфейс надає можливості зручного пошуку, фільтрації та вибору інформації. Також він контролює цілісність системи, наприклад те, що людина може бути найнята лише на вільну посаду, несуперечність дат контрактів щодо співробітника та посади тощо і не дозволяє користувачу зробити помилку.

1. **ПОСТАНОВКА ЗАДАЧІ**

Реалізована в цьому проекті автоматизована система обробки даних призначена для використання адміністратором ігротеки та його працівниками.

Метою створення проекту є збереження даних про структуру підприємства, його клієнтів та різні замовлення: на закупівлю товарів, продаж, списування, тощо. Все це дає можливість полегшення роботи працівників та прозоре й зручне відстеження необхідної інформації.

Область функціонування даного програмного продукту, яку охоплює дана АСОД, включає в себе:

* Реєстрацію працівників, клієнтів, організацій.
* Можливість перегляду усіх замовлень, виконаних клієнтами чи організаціями.
* Збереження та перегляд кімнат та послуг.
* Реєстрація персоналу.
* Внесення посад, назначення персоналу на посади.
* Перегляд даних про персонал, посади.
* Можливість передення персоналу з посади на посаду.

Для коректної роботи програми необхідно виконання відповідних вимог до технічних і програмних засобів.

1. **АРХІТЕКТУРА ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ АСОД**

Створена АСОД передбачена для використання співробітниками ігротеки, що працюють з персоналом або товарами.

Дані, що цікавлять користувачів АСОД це:

* Дані про клієнтів;
* Дані про резюме;
* Дані про інтерв’ю;
* Дані про вільні посади;
* Дані про товари;
* Дані про замовлення;

Функції оператора персоналу:

- Переглядати списки всіх працівників, людей, медичних довідок, резюме, інтерв’ю, досягнень, відсутностей;

Функції адміністратора персоналу, крім функцій оператора, включають в себе:

* Зареєструвати нову людину, якщо вона є дієздатною;
* Наймати та звільняти працівників;
* Переводити з посади на посаду.

Функції оператора руху товарів:

* Переглядати список усіх клієнтів, товарів, типів товарів, рецептів, замовлень

Функції адміністратора руху товарів, крім функцій оператора, включають в себе:

* Створювати нові товари, типи товарів, кімнати, послуги;
* Додавати контрагентів;
* Створювати замовлення;
  1. **DFD0 – діаграма**

Діаграма потоків даних нульового рівня зображена на рис. 2.1.

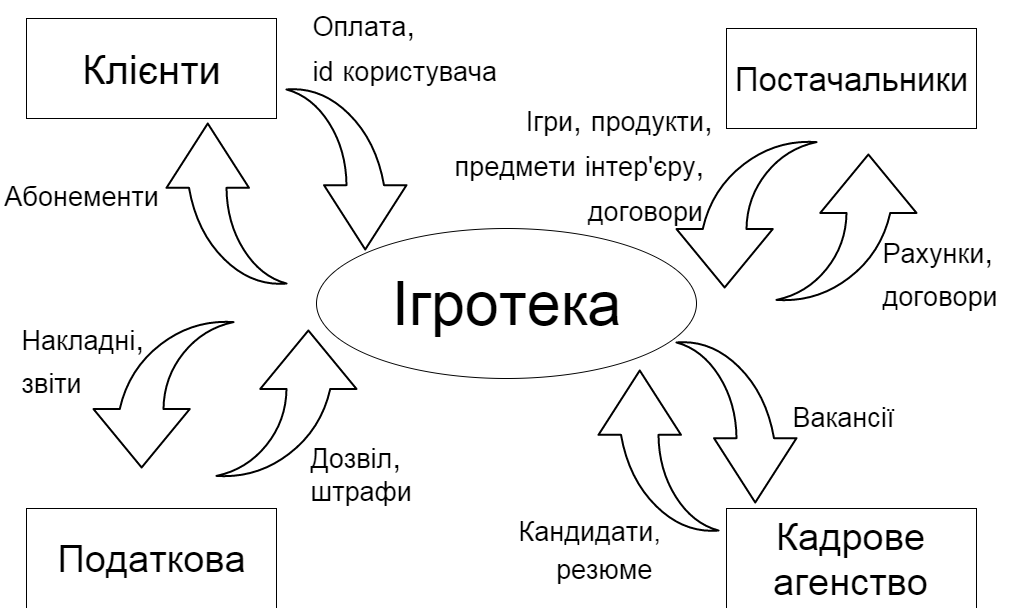


Рис. 2.1. DFD0 – діаграма

* 1. **DFD1 – діаграма**

На рис. 2.2 та 2.3 зображені діаграми потоків даних першого рівня для підсистем «Робота з клієнтами» та «Робота з товарами» відповідно.

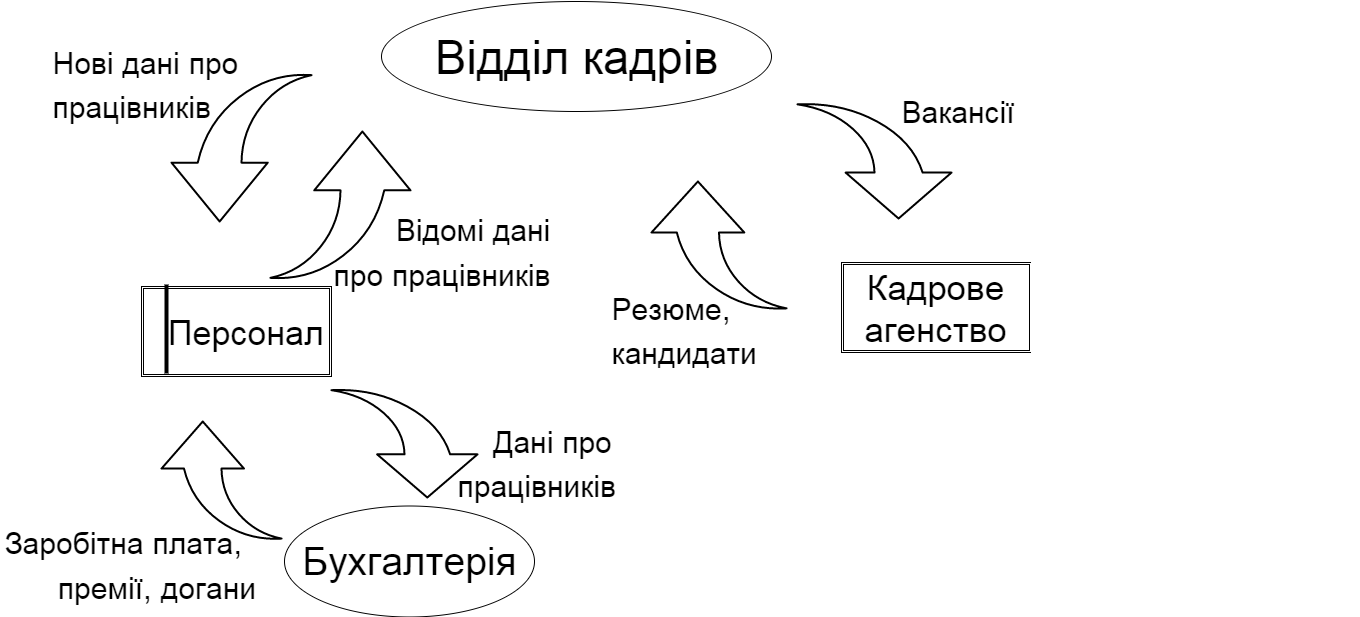


Рис. 2.2. DFD1 – діаграма. «Робота з клієнтами».

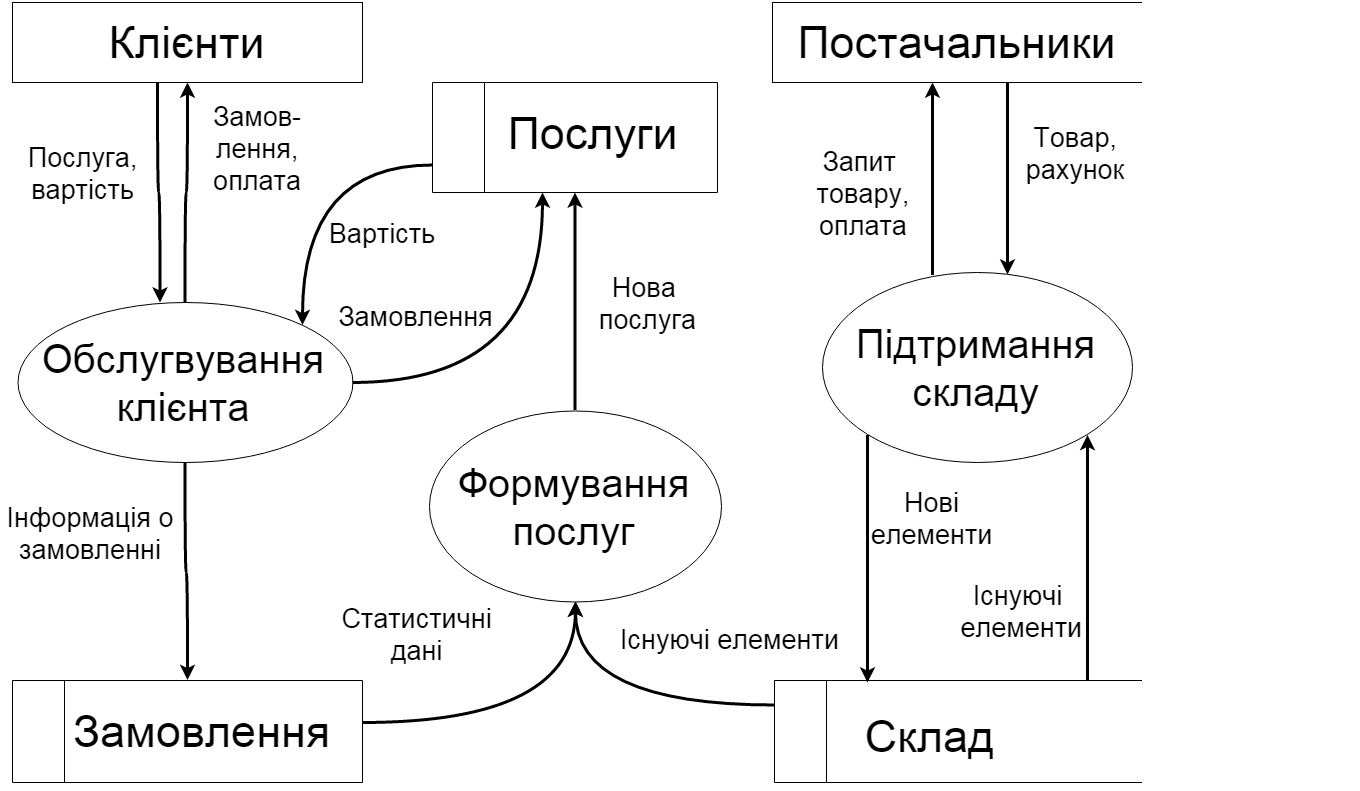


Рис. 2.3. DFD1 – діаграма. «Робота з товарами».

* 1. **DFD2 – діаграма**

На рис. 2.4 та 2.5 зображені діаграми потоків даних другого рівня для підсистем «Робота з клієнтами» та «Робота з товарами» відповідно.

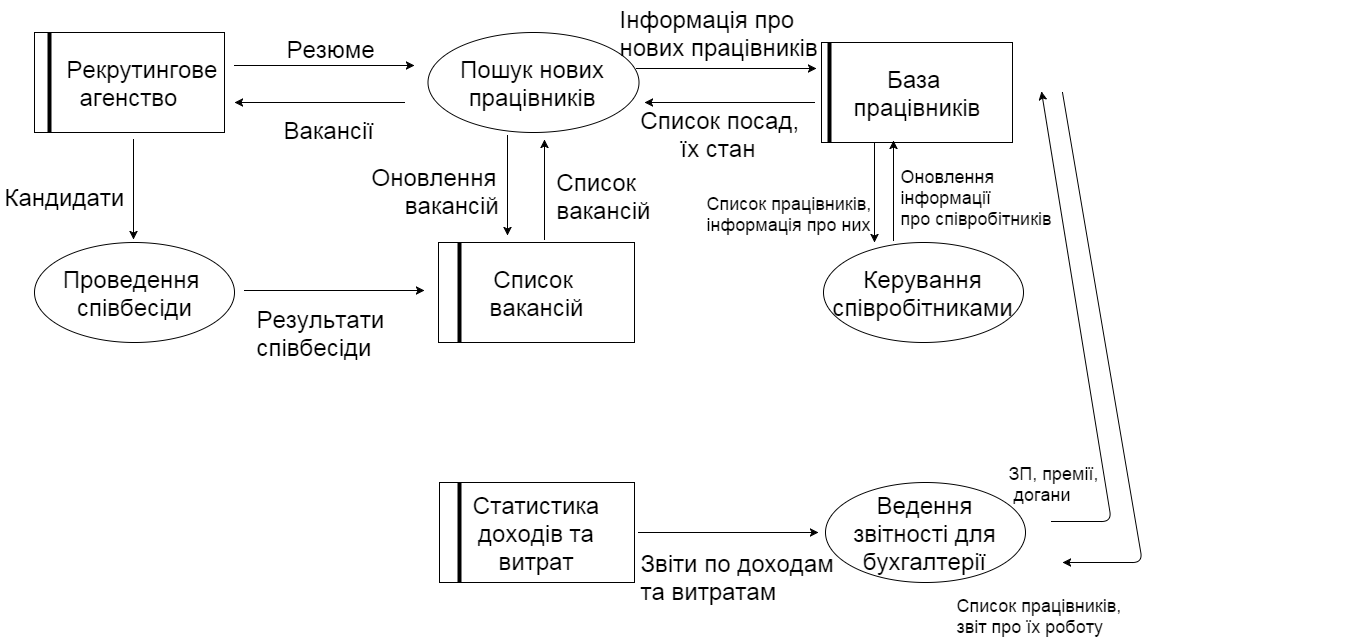


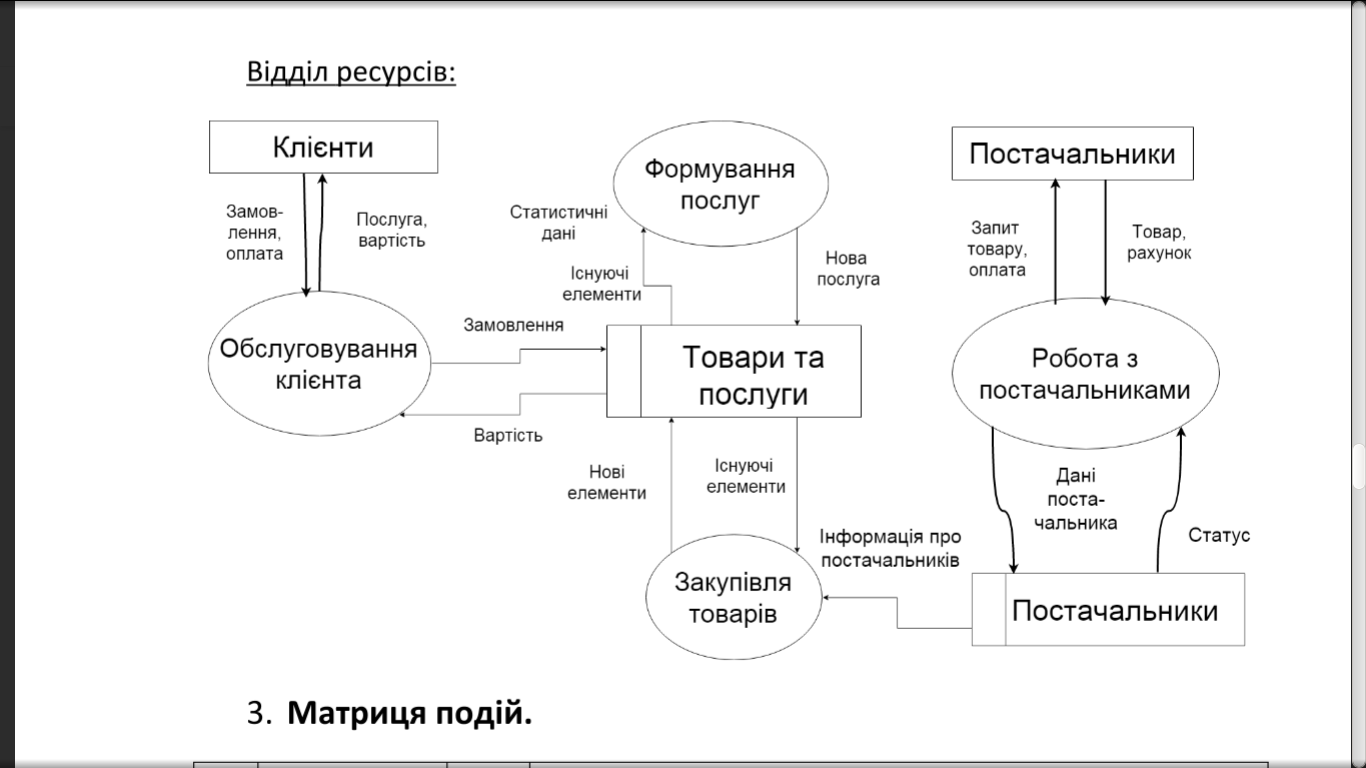
Рис. 2.4 DFD2 – діаграма. «Робота з клієнтами».

Рис. 2.5. DFD2 – діаграма. «Робота з товарами».

* 1. **Матриця подій**

Матриці подій двох підсистем відображають події, що можуть статися та реакцію системи на них.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Опис події | Тип | Умова | Реакція |
| 1. | Додавання нової посади | N |  | Додавання інформації про посаду в БД. Якщо співробітника на дану посаду немає, внесення її до списку вакансій та передача його до рекрутингового агенства. |
| 2. | Прийом нового працівника | N | Працівник є в БД | Додавання інформації про нову посаду працівника в БД. Видалення посади із списка вакансій. |
| Працівника не має в БД | Внесення працівника в БД. Видалення посади із списка вакансій. |
| 3. | Звільнення працівника | N |  | Внесення відповідної інформації до БД. Якщо на дану посаду потребується працівник, внесення її до списку вакансій та передача його до рекрутингового агенства. |
| 4. | Зміна графіку роботи працівника | N |  | Внесення відповідної інформації до БД. Якщо потрібно, знайти заміну. |
| 5. | Нарахування заробітної плати, премії, винесення догани | N |  | Врахування доходів та розходів підприємства; кількості робочих годин працівника, його посади, виконання обов’язків. |

Табл. 2.1. Матриця подій для підсистеми «Персонал»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Опис події | Тип | Реакція ІС на цю подію |
| 1. | Реєстрація контрагента | N | Занесення даних про контрагента в БД |
| 2. | Реєстрація товару на складі | N | Занесення даних про товар в БД |
| 3. | Редагування даних про товари на складі | N | Перевірка наявності товару на складі. Якщо товару немає, то перехід до п. 2. Якщо є, редагування інформації про товар. |
| 4. | Закупівля товарів | N | Перевірка існування контрагента в списку контрагентів в БД. Якщо контрагент не внесений до списку, то перехід до п. 1.  Перевірка наявності товару на складі. Якщо товару немає, то перехід до п. 2. Якщо є, перехід до п. 3.  Внесення документу про закупівлю товарів в БД. |
| 5. | Додавання нової послуги | N | Занесення даних про нову послугу в БД. |
| 6. | Видалення послуги | N | Якщо послуга є, видалення цієї послуги (робиться неактивною). |
| 7. | Розміщення клієнта у кімнатах | N | Якщо є вільні місця у кімнаті, то розміщуємо клієнта та зменшуємо кількість вільних місць, якщо немає, пропонуємо альтернативу та повторюємо п.7.  Збільшуємо лічильник обраної послуги (абонементу) |
| 8. | Розрахунок клієнта | N | Розрахунок клієнта з урахуванням знижки.  Збільшення кількості вільних місць у кімнаті, в якій знаходився клієнт. |

Табл. 2.2. Матриця подій для підсистеми «Рух товарів»

* 1. **Бізнес правила**

Бізнес правила:

1. Постачальниками товарів можуть бути тільки юридичні особи з відповідною ліцензією.

2. Під час прийому товарів від постачальника обов’язково перевіряється і фіксується наявність сертифікатів якості.

3. У разі ушкодження майна клієнтом він зобов’язується відшкодувати його вартість.

4. Кожен рік всі працівники мають проходити обов’язковий медогляд.

5. Працівниками ресторану можуть бути тільки особи, що досягли 18-річного віку.

7. Оплата абонементу з фіксованим часом здійснюється перед користуванням послугами ігротеки.

8. Якщо клієнт, що придбав абонемент з фіксованим часом, покидає ігротеку раніше зазначеного в абонементі часу, кошти йому не повертаються.

9. Оплата абонементу з нефіксованим часом здійснюється постфактум.

10. Реєстрація тільки дієздатних осіб.

**2.6 Інфологічне проектування**

На основі вивчення предметної області були виділені такі основні сутності:

1. Контрагент

2. Фізична особа

3. Юридична особа

4. Документ

5. Тип документу

6. Документ про рух товару

7. Товар

8. Тип товару

9. Стандартний

10. Послуга

11. Знижка

12. Знижка\_Документ\_про\_рух\_товару

13. Кімната

14. Співбесіда

15. Посада

16. Персонал

17. Досягнення

18. Тип досягнення

19. Штатний розклад

20. Персонал на посаді

21. Відсутність на роботі

22. Тип відсутності

23. Графік роботи

24. Персонал\_на\_посаді\_Графік\_роботи

Зв'язки між сутностями продемонстровані на інфологічній моделі системи «Ігротека» у вигляді логічної ER-діаграми (рис. 2.4), виконаної за допомогою програми ERwin Data Modeler.

**2.7 Датологічне проектування**

Датологічна модель представлена на рис. 2.4 у вигляді фізичної ER-діаграми.

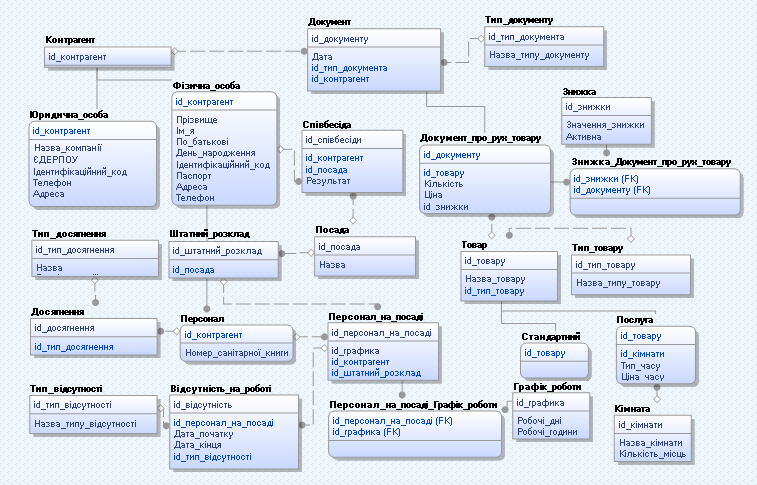


Рис. 2.4. Фізична ER-діаграма

1. **ОПИС ПРОГРАМИ**

БД було створено у СУБД MSSQL Server 2014 з огляду на її зручність та сумісність з іншими продуктами Windows, зокрема Visual Studio. Інтерфейс програми виконано в середовищі розробки Visual Studio 2013 Professional мовою програмування C#.

3.1. Інтерфейс для відділу кадрів

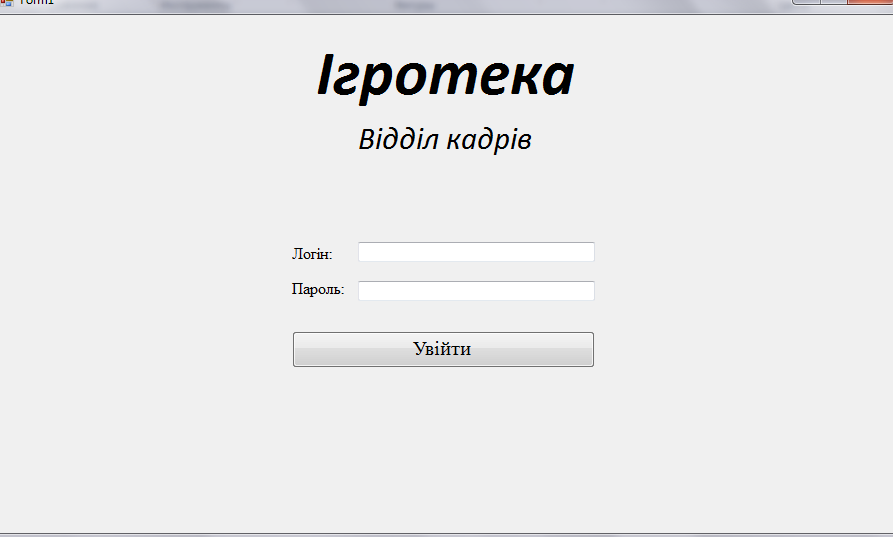
При запуску програми відкривається вікно авторизації (рис. 3.1.1). В ньому слід увести ім’я користувача та пароль. 

Рис. 3.1.1. Вікно авторизації

Якщо пароль або логін користувача було введено неправильно, з’явиться вікно з повідомленням про помилку: “Неправильний логін або пароль”.

Якщо авторизація пройшла успішно, користувач бачить головне вікно пограми (Рис. 3.1.2).



Рис. 3.1.2. Головне вікно програми

Для того, щоб переглянути персонал, користувачу потрібно натиснути на кнопку “Переглянути” в пункті “Персонал”. Дана вкладка має наступний вигляд (Рис. 3.1.3):

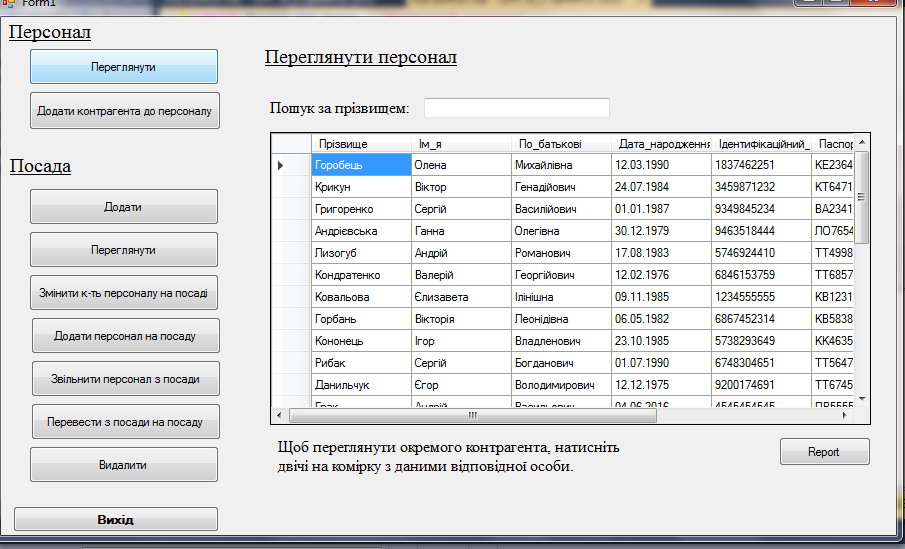


Рис. 3.1.3. Вкладка “Переглянути персонал”

Зверху присутнє поле для пошуку за прізвищем. Також можна роздрукувати список працівників, натиснувши кнопку “Report”.

Якщо натиснути на людину, відкриється нова вкладка з усіма даними про неї, включаючи посади, які вона займала (Рис. 3.1.4):

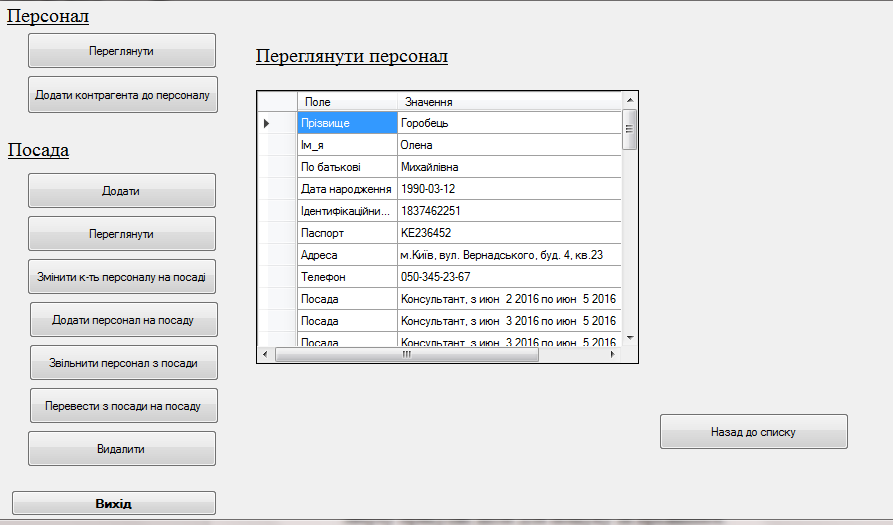


Рис. 3.1.4. Перегляд окремого працівника.

Для того, щоб додати працівника до бази даних, необхідно натиснути кнопку “Додати контрагента до персоналу” (Рис. 3.1.5). Варто зазначити, що для усіх полів виконується перевірка формату введених даних. Якщо формат невірний, користувачу буде виведено вікно з повідомленням про помилку. Усі поля мають бути заповнені.

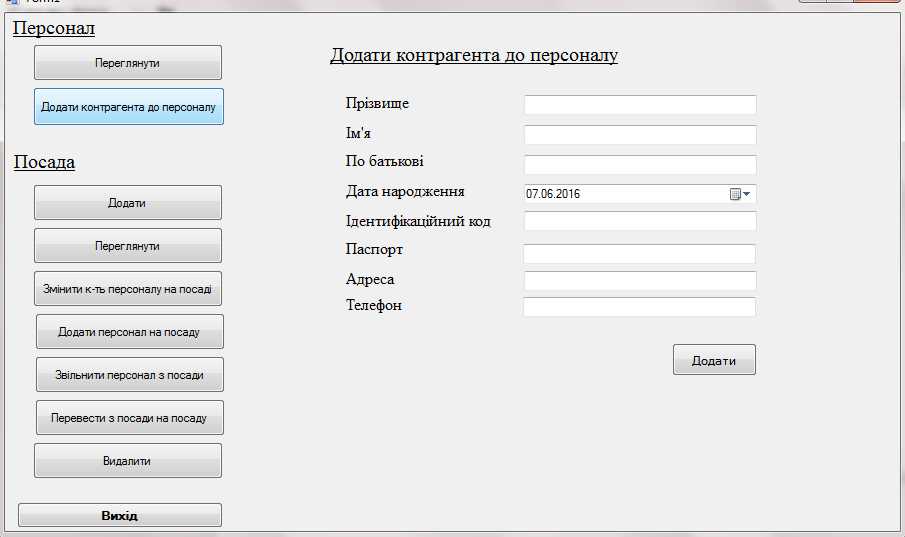


Рис. 3.1.5. Вкладка “Додати контрагента до персоналу”

Для того, щоб додати посаду, натисніть кнопку “Додати” у пункті “Посада” (Рис. 3.1.6). Для перегляду наявних посад, натисніть кнопку “Переглянути”. Доступна можливість переглянути посади станом на будь-який час. Для цього оберіть потрібну Вам дату, і натисніть кнопку “Переглянути” (Рис. 3.1.7).

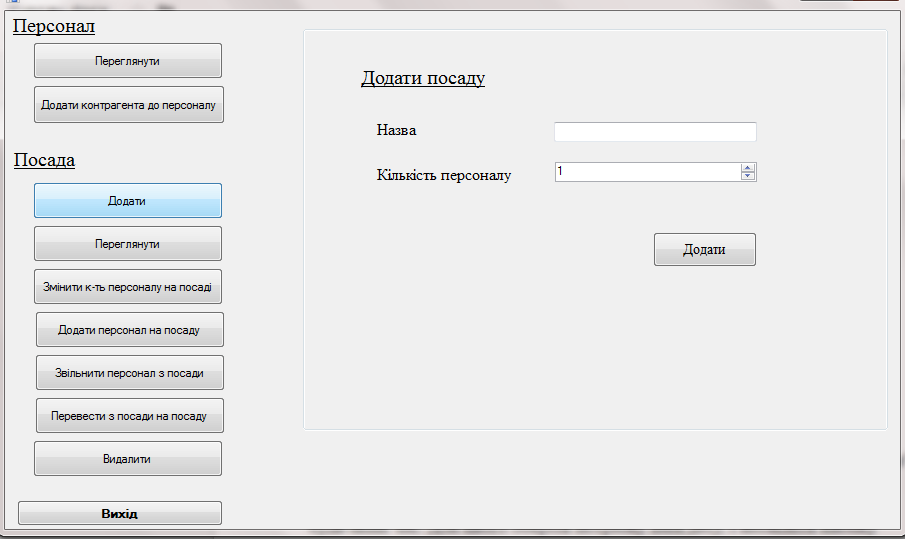


Рис. 3.1.6. Вкладка “Додати посаду”

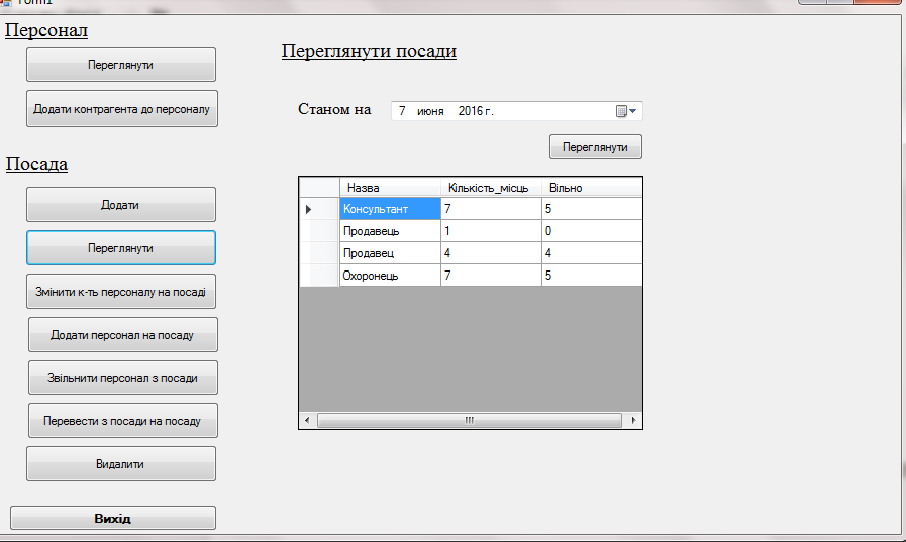


Рис. 3.1.7. Вкладка “Переглянути посади”

Варто зазначити, що наявність посад з однаковою назвою не допускається. Якщо користувач введе назву посади, що вже є у базі даних і яка не була закрита, буде виведено повіжомлення про помилку.

Для того щоб змінити кількість персоналу на вже наявній посаді, потрібно натиснути на кнопку “Змінити кількість персоналу на посаді”, обрати потрібну посаду зі списку, ввести нову кількість та натиснути кнопку “Змінити” (Рис. 3.1.8). Якщо на посаді в даний момент знаходиться кількість персоналу, що перевищує нову кількість, такі зміни внесено не буде (користувачу буде виведено повідомлення про помилку).

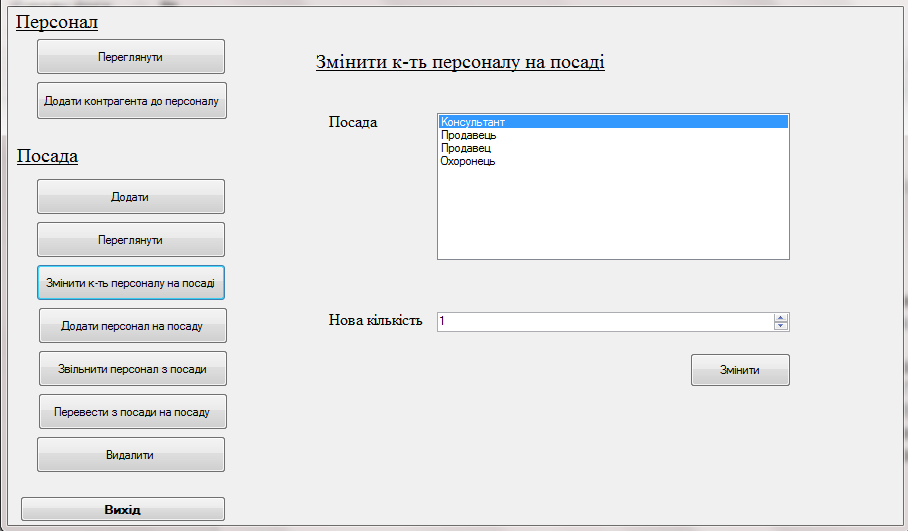


Рис. 3.1.8. Вкладка “Змінити кількість персоналу на посаді”

Для того, щоб додати персонал на посаду, натисніть кнопку “Додати персонал на посаду”. У верхньому вікні обрати зі списку потрібну людину, виділивши відповідний рядок. Нижче відображаються вільні на даний момент посади, серед яких потрібно обрати потрібну. Після цього натисніть кнопку “Додати” (Рис. 3.1.9).

Вгорі присутнє поле для пошуку за прізвищем. Якщо потрібної людини в списку немає, натисніть кнопку “Додати контрагента” та додайте її. Після цього програма повернеться до вкладки “Додати персонал на посаду”.

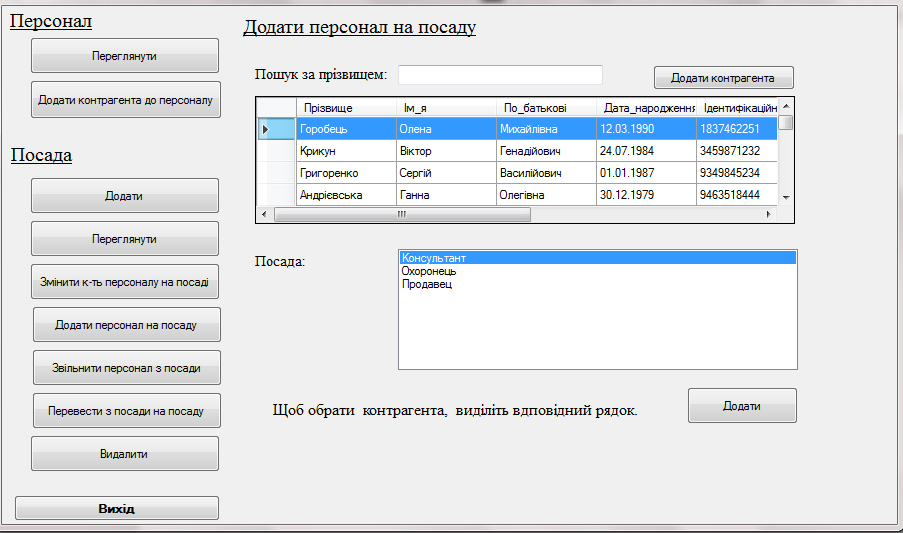


Рис. 3.1.9. Вкладка “Додати персонал на посаду”

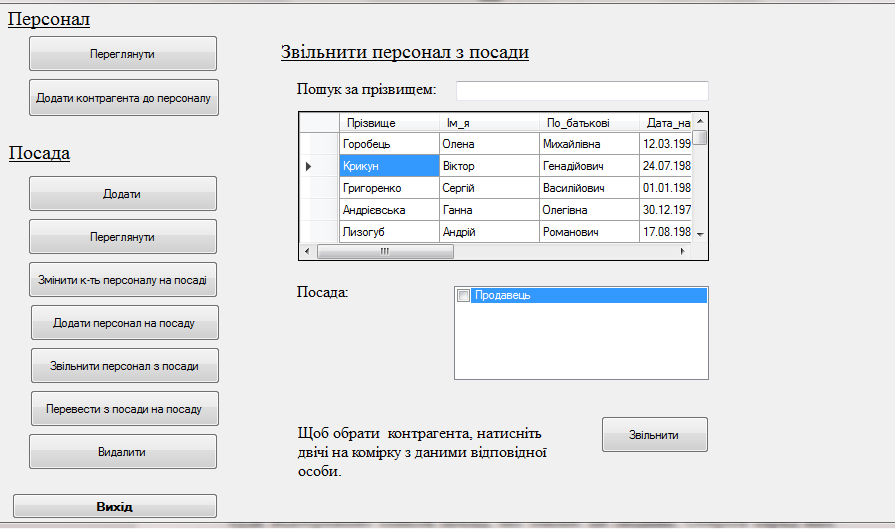
Для того, щоб звільнити персонал з посади, натисніть кнопку “Звільнити персонал з посади” (Рис. 3.1.10). У верхньому списку оберіть працівника, натиснувши двічі на комірку з даними відповідної особи. У разі, якщо людина жодних посад не займає, користувачу буде виведено вікно з наступною інформацією: “Посад немає”. В протилежному випадку, буде відображено список посад, які займає ця людина. Оберіть серед них потрібну та натисніть кнопку “Звільнити”. 

Рис. 3.1.10. Вкладка “Звільнити персонал з посади”

Користувач також може перевести працівника з посади на посаду. Для цього потрібно натиснути кнопку “Перевести з посади на посаду” (Рис. 3.1.11). Оберіть потрібну людину. Якщо вона жодних посад не займає, буде виведено повідомлення з наступною інформацією: “Посад немає”. В протилежному випадку, нижче буде відображено список посад, які даний працівник займає. Оберіть серед них потрібну, виберіть серед списку вільних посад нову посаду даного працівника та натисніть кнопку “Перевести”.

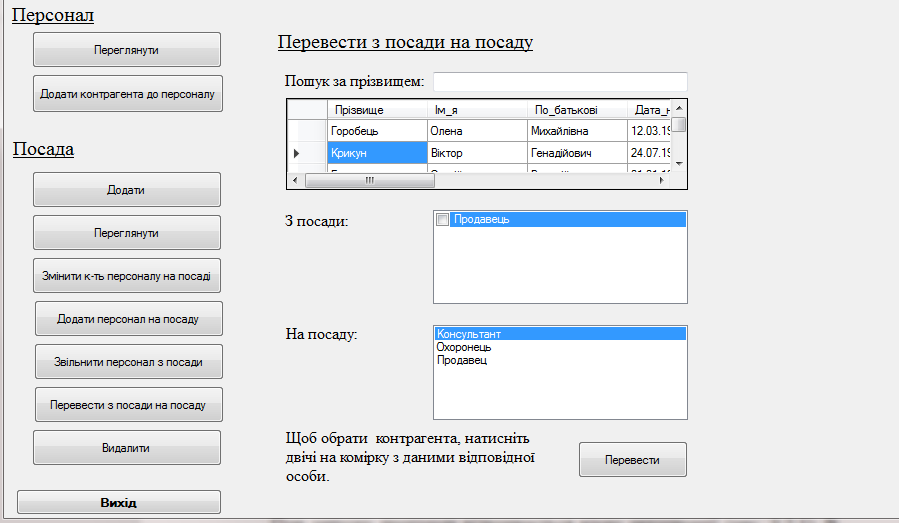


Рис. 3.1.11. Вкладка “Перевести з посади на посаду”

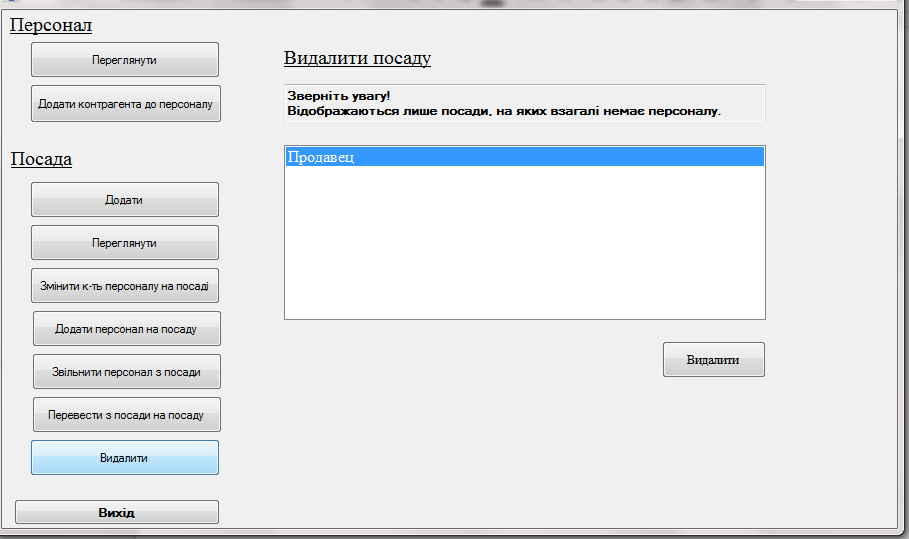
Для того, щоб закрити посаду, натисніть кнопку “Видалити” в пункті “Посада” (Рис. 3.1.12). Дана функція доступна лише для тих посад, на яких взагалі немає персоналу. У вкладці оберіть потрібну посаду та натисніть кнопку “Видалити”.

Рис. 3.1.12. Вкладка

“Видалити посаду”

Для виходу із системи, натисніть кнопку “Вийти”.

3.2. Інтерфейс для роботи з клієнтами та кімнатами

При запуску програми відкривається вікно авторизації (рис. 3.2.1). В ньому слід увести ім’я користувача та пароль.

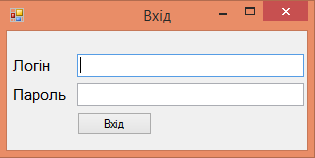


Рис. 3.2.1. Вікно авторизації

У разі введення невірного пароля або імені користувача виводиться повідомлення “Перевірьте правильність введеня логіну та паролю”. В разі успішного під’єднання, відкривається головне вікно програми (рис. 3.2.2.), але в залежності від акаунту (менеджер або адміністратор), під яким авторизувався юзер, деякі функції будуть доступні або ні. В останніх чотирьох вкладках редагувати таблиці має право лише адміністратор. На кожній вкладці представлена інформація, що відповідає заголовку вкладки.

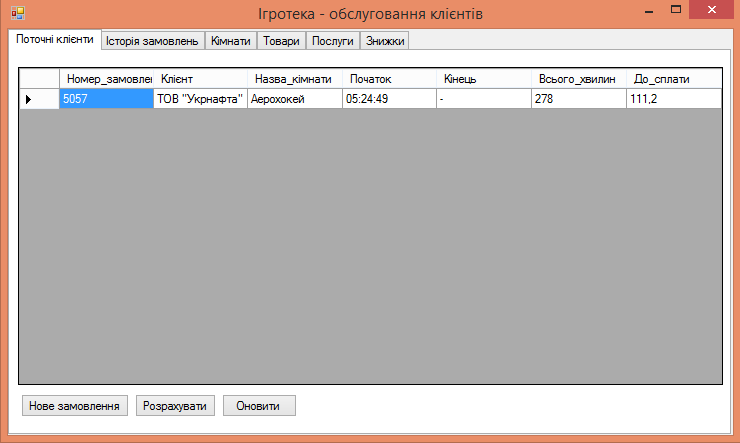


Рис. 3.2.2. Головне меню.

В обох частинах реалізована можливість додавання нового замовлення (рис. 3.2.3.).

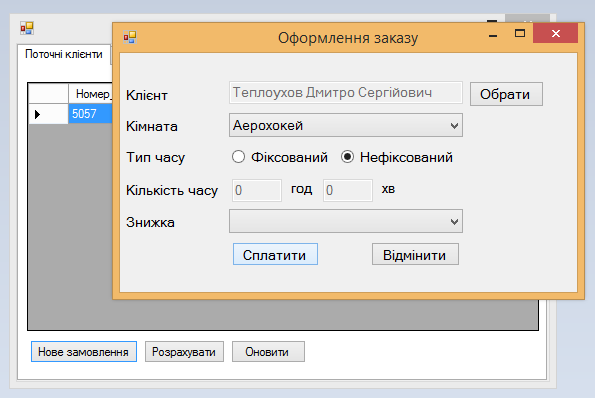


Рис. 3.2.3. Вікно «Додати замовлення».

Для того щоб його створити треба обрати існуючого клієнта (рис 2.3.4.), обрати кімнату та знижку з випадаючих списків, обрати тип часу. Якщо обран фіксований тип часу, то текстові поля для Кількості часу є доступними і користувач може ввести години та хвилини. Дозволений інтервал для годин - 0-12, для хвилин - 0-59. При некорректнотному вводі з’являється вікно с помилкою “Перевірьте введене значення! Воно повинно належити інтервалу від 0 по 12 ” або “Перевірьте введене значення! Воно повинно належити інтервалу від 0 по 59” для годин та хвилин відповідно.

Якщо натиснути “Сплатити”, то:

а) якщо якимось полям окрім “Знижка” не надані значення, то буде помилка "Не достатньо інформації!".

б) якщо час фіксований та все введено коректно, то відкроється повідомлення "До сплати: х грн"

в) якщо час нефіксований та все введено коректно, то відкроється повідомлення "Замовлення занесенне!", бо розраховуватися воно буде потім.

Обрати клієнта можна в новому вікні (рис 3.2.4.), викостовуючи текстове вікно для фільтрації за значенням та відвідні опції - за значенням. Якщо потрібний клієнт не знайден, його можно додати. При натисканні на кнопку “Додати нового клієнта” буде відкрито вікно для додавання того типу клієнта, який був обран при фільтруванні.

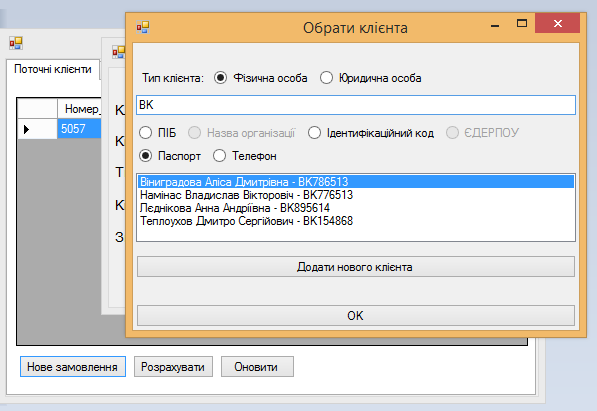


Рис. 3.2.4. Вікно «Обрати клієнта».

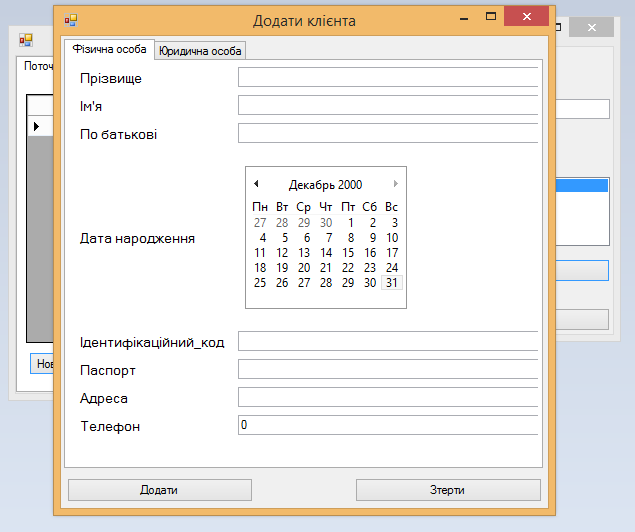


Рис. 3.2.5. Вікно «Додати фізичну особу».

При заповненні кожне поле перевіряється на корректність вводу та при неправильному з’являється повідомлення під полем. Якщо користувач залишить поле некоректно введеним, додаток видасть помилку, але він зможе продовжити редагувати інші поля.

Якщо всі поля введені і введені корректно, то користувач додається, але додатково на сервері перевіряється наявність цього користувача за паспортом й ідентифікаційним кодом для фізичних особ та за ЄДЕРПОУ та ідентифікаційним кодом для юридичних осіб.

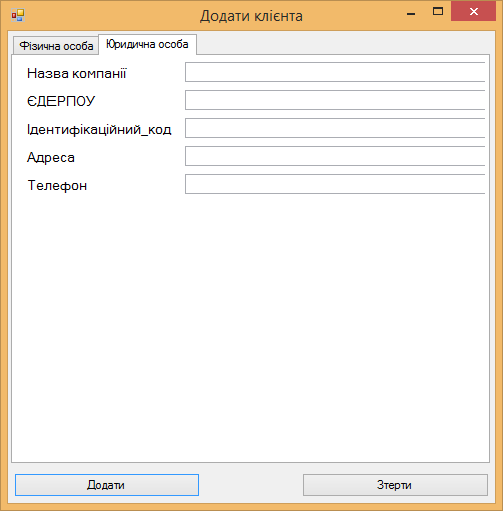


Рис. 3.2.6. Вікно «Додати юридичну особу».

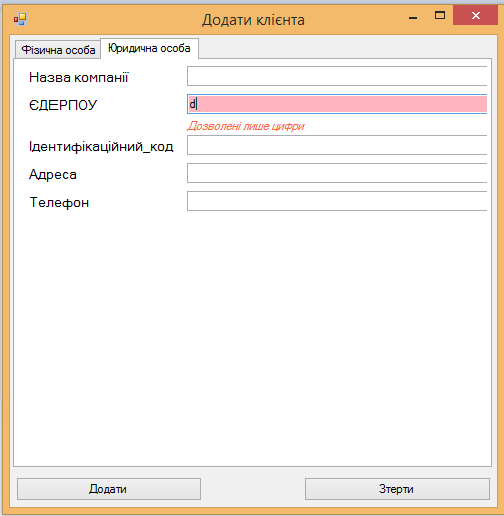


Рис. 3.2.7. Вікно «Додати юридичну особу» з помилкою.

Вкладка “Історія замовлень” містить замовлення за весь час, але лише розраховані, тобто якщо зроблено замовлення з нефіксованим часом та воно ще активне, то воно відображатися буде лише після розрахунку. У цій вкладці можна переглянути, роздрукувати або зберегти дані.

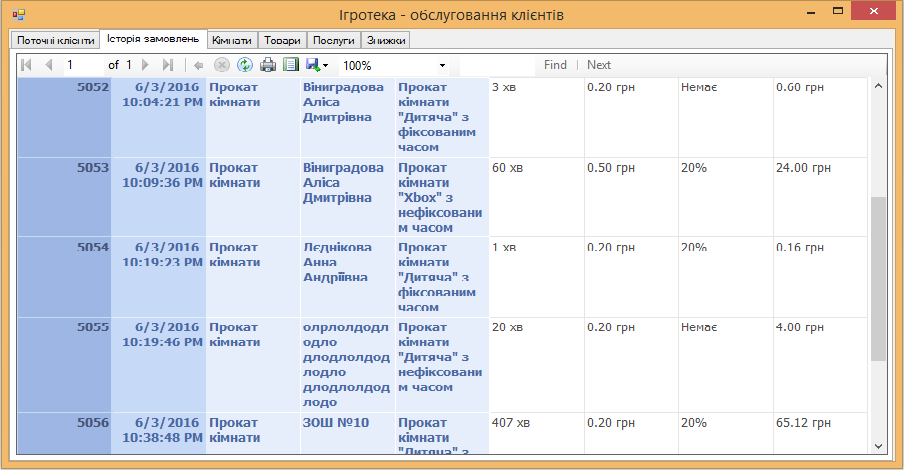


Рис. 3.2.8. Вкладка “Історія замовлень”

Наступні чотири вкладки, Кімнати, Товари, Послуги та Знижки, використовуються за єдиним принципом: відображується таблиця, можна редагувати в таблиці не ключові стовпці (primary та foreign key не можна), а за допомогою кнопок - додавати нові елементи та редагувати foreign key поля.

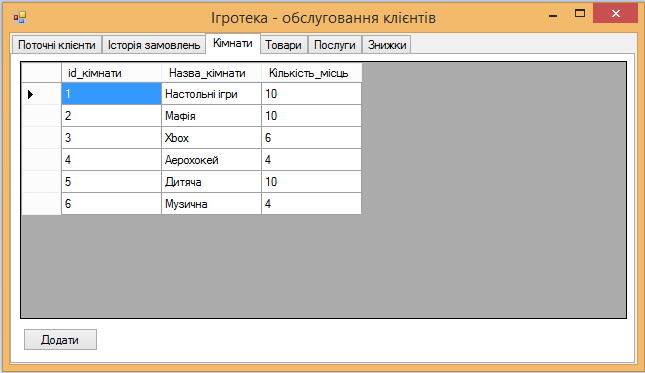


Рис. 3.2.9. Вкладка “Кімната”.

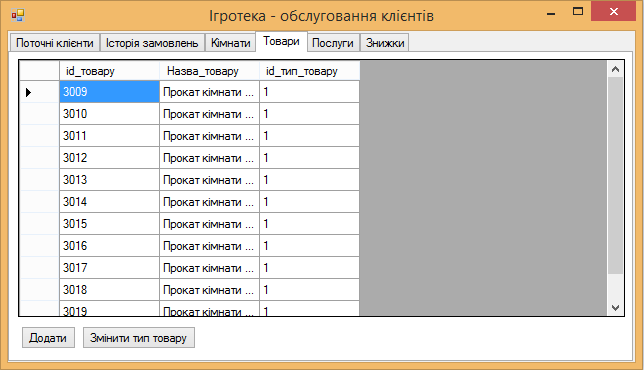


Рис. 3.2.10. Вкладка “Товари”.

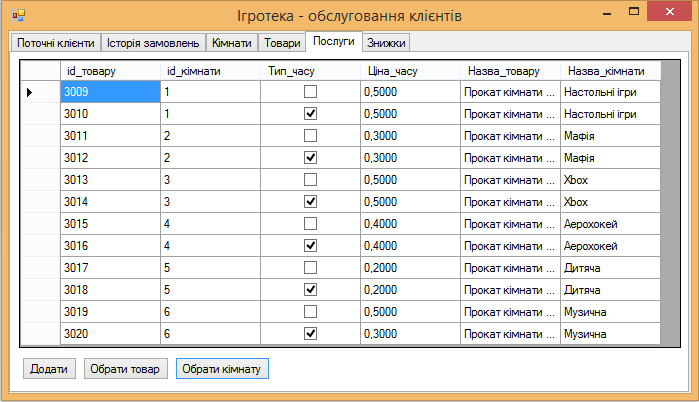


Рис. 3.2.11. Вкладка “Послуги”.

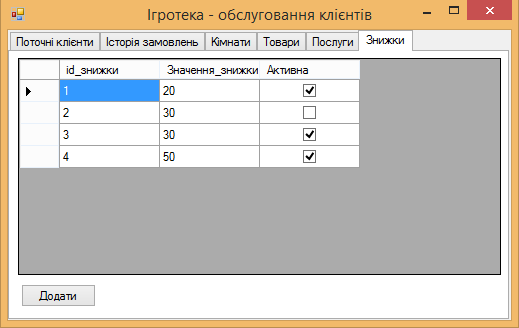


Рис. 3.2.12. Вкладка “Знижка”.

**ВИСНОВОК**

Отже, в ході виконання даного курсового проекту була спроектована та реалізована база даних засобами MS SQL Server 2014 та розроблено користувацький інтерфейс до цієї БД засобами Visual Studio 2013 Professional мовою програмування C#. Створений продукт забезпечує виконання основних функцій системи «Ігротека».

Передбачені можливості роботи з правами адміністратора та оператора. Їм доступні наступні функції:

Функції операторів:

* Перегляд даних таблиць;
* Перегляд звітів;
* Створення замовлень

Функції адміністраторів, крім функцій операторів, включають в себе:

* Можливість додавати нові дані до таблиць;
* Виконання збережених процедур;

При проектуванні БД була передбачена обробка виключних ситуацій. Інтерфейс програми є інтуїтивно зрозумілим і зручним.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. [Інтернет ресурс]: https://msdn.microsoft.com/en-s/
2. Мартин Грубер. Понимание SQL. – М., 1993.
3. Вибрані лекції по курсу «Бази даних та інформаційні системи», http://www.belani.narod.ru/1/main.htm.
4. Кузнецов С.Д. Основы современных баз данных.
5. Виейра Роберт. Программирование баз данных Microsoft SQL Server 2005. Базовый курс : пер. с англ. / Р. Виейра. - М. : Диалектика, 2008. - 832 с. - (Программистам от программистов).

**ДОДАТКИ**

**Додаток 1**. Процедури та функції для кадрів

ALTER PROCEDURE [dbo].[NewPost]

@name nvarchar(200),

@count int

AS

INSERT INTO Посада(Назва) VALUES(@name);

DECLARE @posada\_id int;

SELECT @posada\_id = id\_посада FROM Посада WHERE Назва = @name;

INSERT INTO Штатний\_розклад(id\_посада, Кількість) VALUES(@posada\_id, @count);

ALTER PROCEDURE [dbo].[DeletePost]

@name nvarchar(200)

AS

UPDATE Посада SET Дата\_закриття\_посади = GETDATE() WHERE Назва = @name AND Дата\_закриття\_посади IS NULL;;

ALTER PROCEDURE [dbo].[DeletePersonalFromPost]

@ic nvarchar(MAX),

@postname nvarchar(MAX)

AS

DECLARE @id\_agent int;

SELECT @id\_agent = id\_контрагент FROM Фізична\_особа WHERE Ідентифікаційний\_код = @ic;

UPDATE Персонал\_на\_посаді SET Дата\_звільнення = GETDATE() WHERE id\_контрагент = @id\_agent;

ALTER PROCEDURE [dbo].[ChangePersonalCount]

@name nvarchar(200),

@new\_count int

AS

DECLARE @find\_posada\_id int;

SELECT @find\_posada\_id = id\_посада FROM Посада WHERE Назва = @name AND Дата\_закриття\_посади IS NULL;

DECLARE @find\_r\_id int;

SELECT @find\_r\_id = id\_штатний\_розклад FROM Штатний\_розклад WHERE id\_посада = @find\_posada\_id;

--Якщо кількість не змінилась

DECLARE @old\_count int;

SELECT @old\_count = Кількість FROM Штатний\_розклад WHERE id\_штатний\_розклад = @find\_r\_id

IF (@old\_count = @new\_count) RETURN;

--Якщо на посаді більше людей

DECLARE @working int;

SELECT @working = Персоналу\_на\_посаді FROM dbo.Персонал\_на\_посаді\_id\_view WHERE id\_штатний\_розклад = @find\_r\_id

IF (@new\_count < @working)

BEGIN

RAISERROR('На даний момент на посаді знаходиться більше персоналу. Для внесення цієї зміни спочатку звільніть персонал.', 16, 1)

RETURN

END

UPDATE Посада SET Дата\_закриття\_посади = GETDATE() WHERE id\_посада = @find\_posada\_id;

INSERT INTO Посада(Назва) VALUES(@name);

DECLARE @posada\_id int;

SELECT @posada\_id = id\_посада FROM Посада WHERE Назва = @name AND (Дата\_закриття\_посади IS NULL);

INSERT INTO Штатний\_розклад(id\_посада, Кількість) VALUES(@posada\_id, @new\_count);

DECLARE @new\_r\_id int;

SELECT @new\_r\_id = id\_штатний\_розклад FROM Штатний\_розклад WHERE id\_посада = @posada\_id;

UPDATE Персонал\_на\_посаді SET Дата\_звільнення = GETDATE() WHERE id\_штатний\_розклад = @find\_r\_id;

DECLARE @персонал\_на\_посаді\_ID int = 0;

DECLARE @n int

DECLARE @i int = 0;

SELECT TOP(1) @n = id\_персонал\_на\_посаді FROM Персонал\_на\_посаді ORDER BY id\_персонал\_на\_посаді DESC;

-- Цикл

WHILE (1 = 1)

BEGIN

--Умова виходу

IF @i = @n BREAK;

-- Наступний персонал на посаді

SELECT TOP(1) @персонал\_на\_посаді\_ID = id\_персонал\_на\_посаді

FROM Персонал\_на\_посаді

WHERE id\_персонал\_на\_посаді > @i

ORDER BY id\_персонал\_на\_посаді

DECLARE @persID INT

DECLARE @pers\_r\_ID INT

SELECT @persID = id\_контрагент, @pers\_r\_ID = id\_штатний\_розклад FROM Персонал\_на\_посаді WHERE id\_персонал\_на\_посаді = @персонал\_на\_посаді\_ID

--Процедура

IF (@pers\_r\_ID = @find\_r\_id) EXEC dbo.AddPersonalToPost @persID, @new\_r\_id;

SET @i = @персонал\_на\_посаді\_ID

END

ALTER PROCEDURE [dbo].[AddPersonalToPostByIC]

@ic varchar(MAX),

@postname varchar(MAX)

AS

DECLARE @id\_agent int;

DECLARE @id\_r int;

SELECT @id\_agent = id\_контрагент FROM Фізична\_особа WHERE Ідентифікаційний\_код = @ic;

SELECT @id\_r = id\_штатний\_розклад FROM Штатний\_розклад, Посада WHERE Штатний\_розклад.id\_посада = Посада.id\_посада AND Назва = @postname AND Дата\_закриття\_посади IS NULL;

DECLARE @numb int = 0;

SELECT @numb = COUNT(\*) FROM Персонал WHERE id\_контрагент = @id\_agent;

IF (@numb = 0) INSERT INTO Персонал(id\_контрагент) VALUES (@id\_agent);

PRINT @id\_r;

EXEC AddPersonalToPost @id\_agent, @id\_r;

ALTER PROCEDURE [dbo].[AddPersonalToPost]

@id\_agent int,

@id\_r int

AS

INSERT INTO Персонал\_на\_посаді(id\_контрагент, id\_штатний\_розклад) VALUES(@id\_agent, @id\_r);

ALTER PROCEDURE [dbo].[AddContragentToPersonal]

@Name nvarchar(MAX),

@Surname nvarchar(MAX),

@SecondName nvarchar(MAX),

@BirthDate nvarchar(30),

@IC nvarchar(MAX),

@Passport nvarchar(MAX),

@Adress nvarchar(MAX),

@PhoneNumber nvarchar(MAX)

AS

DECLARE @i int = 0;

SELECT TOP(1) @i = id\_контрагент FROM Контрагент ORDER BY id\_контрагент DESC;

SET @i = @i + 1;

BEGIN TRY

INSERT INTO Фізична\_особа(Прізвище, Ім\_я, По\_батькові, Дата\_народження, Ідентифікаційний\_код, Паспорт, Адреса, Телефон, id\_контрагент)

VALUES (@Name, @Surname, @SecondName, convert(date,@BirthDate,100), @IC, @Passport, @Adress, @PhoneNumber, @i);

END TRY

BEGIN CATCH

RAISERROR('Дана особа вже є у базі даних.', 16, 1);

RETURN

END CATCH

INSERT INTO Контрагент(id\_контрагент) VALUES(@i);

ALTER FUNCTION [dbo].[SeePostWithDate] (@date datetime)

RETURNS TABLE

AS

RETURN

(

SELECT pos.Назва, s.Кількість AS Кількість\_місць, s.Кількість - ISNULL(p.Персоналу\_на\_посаді,0) AS Вільно

FROM Посада AS pos

INNER JOIN Штатний\_розклад AS s

ON pos.id\_посада = s.id\_посада

LEFT JOIN PersOnPostWithDate(@date) AS p

ON s.id\_штатний\_розклад = p.id\_штатний\_розклад

WHERE (@date >= pos.Дата\_відкриття\_посади) AND (@date <= pos.Дата\_закриття\_посади OR pos.Дата\_закриття\_посади IS NULL)

GROUP BY pos.Назва, s.Кількість, p.Персоналу\_на\_посаді

);

ALTER FUNCTION [dbo].[SeePersonal] (@pers\_ic varchar(MAX))

RETURNS @RT\_ResultFunction TABLE

(

Поле varchar(20),

Значення varchar(200)

)

AS

BEGIN

DECLARE @pers\_id int;

SELECT @pers\_id = id\_контрагент FROM Фізична\_особа WHERE Ідентифікаційний\_код = @pers\_ic;

DECLARE @surname varchar(MAX);

DECLARE @name varchar(MAX);

DECLARE @s\_n varchar(MAX);

DECLARE @birth date;

DECLARE @ic varchar(MAX);

DECLARE @pas varchar(MAX);

DECLARE @adr varchar(MAX);

DECLARE @phone varchar(MAX);

SELECT @surname = Прізвище FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

SELECT @name = Ім\_я FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

SELECT @s\_n = По\_батькові FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

SELECT @birth = Дата\_народження FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

SELECT @ic = Ідентифікаційний\_код FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

SELECT @pas = Паспорт FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

SELECT @adr = Адреса FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

SELECT @phone = Телефон FROM Фізична\_особа WHERE id\_контрагент = @pers\_id;

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Прізвище', @surname);

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Ім\_я', @name);

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('По батькові', @s\_n);

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Дата народження', @birth);

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Ідентифікаційний код', @ic);

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Паспорт', @pas);

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Адреса', @adr);

INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Телефон', @phone);

DECLARE @персонал\_на\_посаді\_ID int = 0;

DECLARE @n int

DECLARE @i int = 0;

SELECT TOP(1) @n = id\_персонал\_на\_посаді FROM Персонал\_на\_посаді ORDER BY id\_персонал\_на\_посаді DESC;

-- Цикл

WHILE (1 = 1)

BEGIN

--Умова виходу

IF @i = @n BREAK;

-- Наступний персонал на посаді

SELECT TOP(1) @персонал\_на\_посаді\_ID = id\_персонал\_на\_посаді

FROM Персонал\_на\_посаді

WHERE id\_персонал\_на\_посаді > @i

ORDER BY id\_персонал\_на\_посаді

DECLARE @pers\_r\_ID INT;

DECLARE @kontr\_ID INT;

DECLARE @d\_b datetime;

DECLARE @d\_e datetime;

SELECT @pers\_r\_ID = id\_штатний\_розклад, @kontr\_ID = id\_контрагент ,@d\_b = Дата\_назначення, @d\_e = Дата\_звільнення FROM Персонал\_на\_посаді

WHERE id\_персонал\_на\_посаді = @персонал\_на\_посаді\_ID;

--Процедура

IF (@kontr\_ID = @pers\_id)

BEGIN

DECLARE @pos varchar(MAX)

SELECT @pos = Назва FROM Штатний\_розклад, Посада WHERE id\_штатний\_розклад = @pers\_r\_ID AND Штатний\_розклад.id\_посада = Посада.id\_посада;

IF (@d\_e IS NULL) INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Посада', @pos+', з ' + CONVERT(VARCHAR, CAST(@d\_b AS DATE), 100));

ELSE INSERT INTO @RT\_ResultFunction(Поле, Значення) VALUES('Посада', @pos+', з ' +CONVERT(VARCHAR, CAST(@d\_b AS DATE), 100)+' по '+CONVERT(VARCHAR, CAST(@d\_e AS DATE), 100));

END

SET @i = @персонал\_на\_посаді\_ID

END

RETURN

END

ALTER FUNCTION [dbo].[PersOnPostWithDate] (@date datetime)

RETURNS TABLE

AS

RETURN

(

SELECT id\_штатний\_розклад, COUNT(id\_штатний\_розклад) AS Персоналу\_на\_посаді

FROM Персонал\_на\_посаді

WHERE (@date >= Дата\_назначення) AND (@date <= Дата\_звільнення OR Дата\_звільнення IS NULL)

GROUP BY id\_штатний\_розклад

);

ALTER FUNCTION [dbo].[GetPosts] (@ic varchar(MAX))

RETURNS TABLE

AS

RETURN

(

SELECT Назва FROM Фізична\_особа, Персонал\_на\_посаді, Штатний\_розклад, Посада

WHERE Ідентифікаційний\_код = @ic AND (Дата\_звільнення IS NULL) AND Персонал\_на\_посаді.id\_контрагент=Фізична\_особа.id\_контрагент

AND Персонал\_на\_посаді.id\_штатний\_розклад=Штатний\_розклад.id\_штатний\_розклад AND Штатний\_розклад.id\_посада=Посада.id\_посада

);

**Додаток 2**. Процедура та функції для товарів

CREATE FUNCTION roomAvailable(@room\_id int) RETURNS BIT

AS

BEGIN

DECLARE @available BIT;

DECLARE @max\_clients INT;

SET @max\_clients = (SELECT Кількість\_місць FROM Кімната WHERE id\_кімнати = @room\_id);

DECLARE @now\_clients INT;

SET @now\_clients = (SELECT COUNT(\*) FROM Документ\_про\_рух\_товару\_повний

WHERE Кількість IS NULL OR

(Кількість IS NOT NULL

AND DATEDIFF(MINUTE, Дата, CURRENT\_TIMESTAMP) < Кількість)

AND id\_товару IN (SELECT id\_товару FROM Послуга

WHERE id\_кімнати = @room\_id))

IF (@now\_clients<@max\_clients)

set @available = 1

ELSE SET @available = 0

RETURN @available

END

GO

CREATE PROCEDURE sum\_of\_doc

@doc\_id int,

@total money OUTPUT

AS

SELECT @total = Кількість \* Ціна \* (1 - CONVERT(float, ISNULL(Значення\_знижки,0))/100)

FROM Документ\_про\_рух\_товару LEFT JOIN Знижка

ON Документ\_про\_рух\_товару.id\_знижки = Знижка.id\_знижки

WHERE id\_документу = @doc\_id;

RETURN

GO

CREATE FUNCTION orderSum(@doc\_id int) RETURNS MONEY

AS

BEGIN

DECLARE @total MONEY

set @total= (select Кількість \* Ціна\*(1-ISNULL(Значення\_знижки,0)/100)

FROM Документ\_про\_рух\_товару JOIN Знижка ON

Знижка.id\_знижки = Документ\_про\_рух\_товару.id\_знижки

WHERE id\_документу = @doc\_id)

RETURN @total

END

GO

CREATE FUNCTION orderSum(@doc\_id int) RETURNS MONEY

AS

BEGIN

DECLARE @total MONEY;

DECLARE @price MONEY;

DECLARE @minutes INT;

DECLARE @priceOff INT;

SET @minutes = (select Кількість FROM Документ\_про\_рух\_товару WHERE id\_документу = @doc\_id);

SET @price = (select Ціна FROM Документ\_про\_рух\_товару WHERE id\_документу = @doc\_id);

SET @priceOff = (SELECT CASE

WHEN Документ\_про\_рух\_товару.id\_знижки IS NULL THEN 0

ELSE ISNULL(Значення\_знижки,0)

END

FROM Документ\_про\_рух\_товару LEFT JOIN Знижка ON

Знижка.id\_знижки = Документ\_про\_рух\_товару.id\_знижки

WHERE id\_документу = @doc\_id);

SET @total = @minutes\*@price\*(1-CONVERT(float,@priceOff)/100);

RETURN @total

END

GO

**Додаток 3**. Тригери для кадрів

ALTER TRIGGER [dbo].[Контагент\_unique]

ON [dbo].[Фізична\_особа]

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS(SELECT Ідентифікаційний\_код FROM Фізична\_особа WHERE Ідентифікаційний\_код IN (SELECT Ідентифікаційний\_код FROM Inserted) GROUP BY Ідентифікаційний\_код HAVING count(\*) > 1)

BEGIN

RAISERROR('Дана особа вже є у базі даних.', 16, 1)

ROLLBACK TRAN

END

END

ALTER TRIGGER [dbo].[Посада\_unique]

ON [dbo].[Посада]

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS(SELECT Назва, Дата\_закриття\_посади FROM Посада WHERE Назва IN (SELECT Назва FROM Inserted) AND (Дата\_закриття\_посади IS NULL) GROUP BY Назва, Дата\_закриття\_посади HAVING count(\*) > 1)

BEGIN

RAISERROR('Дана посада вже є у базі даних.', 16, 1)

ROLLBACK TRAN

END

END

**Додаток 4**. Тригери для товарів

CREATE TRIGGER Занесення\_Документу\_про\_рух\_товару

ON Документ\_про\_рух\_товару\_повний

INSTEAD OF INSERT

AS

BEGIN

DECLARE @client\_id INT;

DECLARE @doc\_id INT;

SELECT @client\_id = id\_контрагент FROM inserted;

IF ((SELECT DISTINCT id\_контрагент FROM Документ\_про\_рух\_товару\_повний

WHERE id\_контрагент = @client\_id) IS NOT NULL)

IF (SELECT TOP(1) Кількість FROM Документ\_про\_рух\_товару\_повний

WHERE id\_контрагент = @client\_id ORDER BY Дата DESC) IS NULL

THROW 60000, 'Вже є замовлення для цього клієнта!', 1 ;

ELSE

BEGIN

SELECT TOP(1) @doc\_id = id\_документу FROM Документ

WHERE id\_контрагент = @client\_id ORDER BY Дата DESC;

IF (SELECT DATEDIFF(minute, Дата, CURRENT\_TIMESTAMP) FROM Документ

WHERE id\_документу = @doc\_id) < (SELECT Кількість

FROM Документ\_про\_рух\_товару

WHERE id\_документу = @doc\_id)

THROW 60000, 'Вже є замовлення для цього клієнта! Воно ще дійсне', 1 ;

END

DECLARE @room\_id INT

SET @room\_id = (SELECT TOP(1) id\_кімнати FROM Послуга WHERE id\_товару = (select id\_товару from inserted))

IF dbo.roomAvailable(@room\_id) = 0

THROW 60000, 'Кімната заповнена! Оберить іншу', 1;

INSERT INTO Документ OUTPUT INSERTED.id\_документу VALUES (CURRENT\_TIMESTAMP, (select id\_тип\_документа from inserted),(select id\_контрагент from inserted));

SELECT @doc\_id = SCOPE\_IDENTITY();

INSERT INTO Документ\_про\_рух\_товару(id\_документу,id\_товару,Кількість,Ціна,id\_знижки) VALUES (@doc\_id, (select id\_товару from inserted),

(select Кількість from inserted),(select Ціна from inserted), (select id\_знижки from inserted));

END;

GO

CREATE TRIGGER create\_kontragent\_for\_jure

ON Юридична\_особа

INSTEAD OF INSERT AS

BEGIN

DECLARE @agent\_id INT

INSERT INTO Контрагент DEFAULT VALUES;

SELECT @agent\_id = SCOPE\_IDENTITY();

INSERT INTO Юридична\_особа VALUES (@agent\_id,

(select Назва\_компанії from inserted), (select ЄДЕРПОУ from inserted),

(select Ідентифікаційний\_код from inserted), (select Телефон from inserted), (select Адреса from inserted));

END

GO

CREATE TRIGGER create\_kontragent\_for\_physic

ON Фізична\_особа

INSTEAD OF INSERT AS

BEGIN

DECLARE @agent\_id INT

INSERT INTO Контрагент DEFAULT VALUES;

SELECT @agent\_id = SCOPE\_IDENTITY();

INSERT INTO Фізична\_особа VALUES (@agent\_id,

(select Прізвище from inserted), (select Ім\_я from inserted), (select По\_батькові from inserted),

(select День\_народження from inserted), (select Ідентифікаційний\_код from inserted),

(select Паспорт from inserted), (select Телефон from inserted), (select Адреса from inserted));

END

GO

**Додаток 5**. Представлення для товарів

CREATE VIEW Документ\_про\_рух\_товару\_повний AS

SELECT Документ.id\_документу, CONVERT(datetime2(0), Дата) AS Дата, id\_тип\_документа, id\_контрагент,

id\_товару, Кількість, Ціна, id\_знижки FROM Документ JOIN Документ\_про\_рух\_товару

ON Документ.id\_документу=Документ\_про\_рух\_товару.id\_документу;

GO

CREATE VIEW Поточні\_замовлення\_кімнат AS

SELECT

Д.id\_документу AS Номер\_замовлення,

CASE

WHEN Ф.id\_контрагент IS NOT NULL THEN CONCAT(Прізвище, N' ', Ім\_я, N' ' , По\_батькові)

WHEN Ю.id\_контрагент IS NOT NULL THEN Назва\_компанії

END AS Клієнт, Кімната.Назва\_кімнати,

CONVERT(time(0), Дата) AS Початок,

CASE

WHEN Кількість IS NULL THEN N'-'

ELSE CONVERT(varchar(10), CONVERT(time(0), DATEADD(minute, Кількість, Дата)))

END AS Кінець,

CASE

WHEN Кількість IS NULL THEN DATEDIFF(MINUTE, Дата, CURRENT\_TIMESTAMP)

ELSE Кількість

END AS Всього\_хвилин,

CASE

WHEN Кількість IS NULL

THEN DATEDIFF(MINUTE, Дата, CURRENT\_TIMESTAMP) \* Д.ціна \*(1-ISNULL(Значення\_знижки,0)/100)

ELSE Кількість\*Ціна\*(1-CONVERT(float,ISNULL(Значення\_знижки,0))/100)

END AS До\_сплати

FROM Документ\_про\_рух\_товару\_повний AS Д

LEFT JOIN Фізична\_особа AS Ф ON Д.id\_контрагент=Ф.id\_контрагент

LEFT JOIN Юридична\_особа AS Ю ON Д.id\_контрагент=Ю.id\_контрагент

LEFT JOIN Послуга ON Послуга.id\_товару=Д.id\_товару

LEFT JOIN Кімната ON Кімната.id\_кімнати = Послуга.id\_кімнати

LEFT JOIN Знижка ON Знижка.id\_знижки = Д.id\_знижки

WHERE Кількість IS NULL OR

(Кількість IS NOT NULL AND DATEDIFF(MINUTE, Дата, CURRENT\_TIMESTAMP) < Кількість);

GO

CREATE VIEW Історія\_замолень AS

SELECT

Документ.id\_документу,

CONVERT(datetime2(0), Дата) AS Дата,

Назва\_типу\_документу,

CASE

WHEN Ф.id\_контрагент IS NOT NULL THEN CONCAT(Прізвище, N' ', Ім\_я, N' ' , По\_батькові)

WHEN Ю.id\_контрагент IS NOT NULL THEN Назва\_компанії

END AS Клієнт,

Назва\_товару,

CONCAT(CONVERT(varchar(10),Кількість), ' хв') AS Кількість\_часу,

CONCAT(CONVERT(varchar(10),Ціна), N' грн') AS Ціна,

ISNULL(Текст\_знижки, 'Немає') AS Знижка,

CONCAT(CONVERT(varchar(10),dbo.orderSum(Документ\_про\_рух\_товару.id\_документу)), N' грн') AS Всього

FROM Документ JOIN Документ\_про\_рух\_товару

ON Документ.id\_документу=Документ\_про\_рух\_товару.id\_документу

LEFT JOIN Тип\_документу ON Тип\_документу.id\_тип\_документа = Документ.id\_тип\_документа

LEFT JOIN Фізична\_особа AS Ф ON Документ.id\_контрагент=Ф.id\_контрагент

LEFT JOIN Юридична\_особа AS Ю ON Документ.id\_контрагент=Ю.id\_контрагент

LEFT JOIN Товар ON Товар.id\_товару = Документ\_про\_рух\_товару.id\_товару

LEFT JOIN Знижка ON Знижка.id\_знижки = Документ\_про\_рух\_товару.id\_знижки

LEFT JOIN Знижка\_текст ON Знижка\_текст.id\_знижки = Документ\_про\_рух\_товару.id\_знижки

WHERE Кількість IS NOT NULL;

GO

CREATE VIEW Знижка\_текст AS

SELECT id\_знижки, CONCAT(Значення\_знижки, '%') AS Текст\_знижки FROM Знижка WHERE Активна = 1;

**Додаток 6**. Представлення для персоналу

CREATE VIEW dbo.Персонал\_на\_посаді\_id\_view

AS

SELECT id\_штатний\_розклад, COUNT(id\_штатний\_розклад) AS Персоналу\_на\_посаді

FROM Персонал\_на\_посаді

GROUP BY id\_штатний\_розклад;

GO

CREATE VIEW dbo.Посада\_view

AS

SELECT pos.Назва, s.Кількість AS Кількість\_місць, s.Кількість - ISNULL(p.Персоналу\_на\_посаді,0) AS Вільно

FROM Посада AS pos

INNER JOIN Штатний\_розклад AS s

ON pos.id\_посада = s.id\_посада

LEFT JOIN Персонал\_на\_посаді\_id\_view AS p

ON s.id\_штатний\_розклад = p.id\_штатний\_розклад;

GO

CREATE VIEW dbo.Вільні\_посади

AS

SELECT Назва

FROM Посада\_view

WHERE Вільно>0

GROUP BY Назва;

GO

**Додаток 7**. Код головної форми товарів

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

using Microsoft.Reporting.WinForms;

using System.Text.RegularExpressions;

namespace WindowsFormsApplication4

{

public partial class MainForm : Form

{

SqlCommand cmd;

string sql;

DataTable dt;

public MainForm()

{

InitializeComponent();

if (Program.admin)

{

((Control)this.tabPage3).Enabled = true;

((Control)this.tabPage4).Enabled = true;

((Control)this.tabPage5).Enabled = true;

((Control)this.tabPage6).Enabled = true;

}

else

{

((Control)this.tabPage3).Enabled = false;

((Control)this.tabPage4).Enabled = false;

((Control)this.tabPage5).Enabled = false;

((Control)this.tabPage6).Enabled = false;

}

}

private DataTable GetData(string table\_name, string filter)

{

string sql = "SELECT \* FROM " + table\_name + " " + filter + ";";

SqlCommand cmd = new SqlCommand(sql, Program.cnn);

DataTable dt = new DataTable();

dt.Load(cmd.ExecuteReader());

return dt;

}

private void refresh\_currentClients()

{

currentClients.DataSource = GetData("Поточні\_замовлення\_кімнат", "");

currentClients.Refresh();

}

private void refresh(DataGridView item, string table\_name)

{

item.DataSource = GetData(table\_name, "");

item.Refresh();

}

private void MainForm\_Load(object sender, EventArgs e)

{

// TODO: This line of code loads data into the 'ИгротекаDataSet.Історія\_замолень' table. You can move, or remove it, as needed.

this.Історія\_замоленьTableAdapter.Fill(this.ИгротекаDataSet.Історія\_замолень);

//currentClientDataGrid

refresh\_currentClients();

refresh(roomGrid, "Кімната");

roomGrid.Columns[0].ReadOnly = true;

refresh(productGrid, "Товар");

productGrid.Columns[0].ReadOnly = true;

productGrid.Columns[2].ReadOnly = true;

sql = "SELECT Послуга.id\_товару, Назва\_товару, Послуга.id\_кімнати, Назва\_кімнати, Тип\_часу, Ціна\_часу FROM Послуга LEFT JOIN Кімната ON Кімната.id\_кімнати = Послуга.id\_кімнати LEFT JOIN Товар ON Товар.id\_товару = Послуга.id\_товару";

cmd = new SqlCommand(sql, Program.cnn);

dt = new DataTable();

dt.Load(cmd.ExecuteReader());

serviceGrid.DataSource = dt;

serviceGrid.Columns[0].ReadOnly = true;

serviceGrid.Columns[1].ReadOnly = true;

serviceGrid.Columns[2].ReadOnly = true;

serviceGrid.Columns[3].ReadOnly = true;

refresh(priceOffGrid, "Знижка");

priceOffGrid.Columns[0].ReadOnly = true;

this.reportViewer1.RefreshReport();

}

private void newOrderButton\_Click(object sender, EventArgs e)

{

var f = new CreateOrder();

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

this.MainForm\_Load(sender,e);

}

}

private void reportViewer1\_Load(object sender, EventArgs e)

{

}

private void paymentButton\_Click(object sender, EventArgs e)

{

int id;

if (currentClients.CurrentRow.Cells["Кінець"].Value.Equals("-"))

{

id = int.Parse(currentClients.CurrentRow.Cells[0].Value.ToString());

sql = "UPDATE Документ\_про\_рух\_товару " +

"SET Кількість = DATEDIFF(MINUTE, " +

"(SELECT Дата FROM Документ WHERE id\_документу = " + id + "), CURRENT\_TIMESTAMP) " +

"WHERE id\_документу = " + id + ";";

SqlCommand cmd = new SqlCommand(sql, Program.cnn);

int i = cmd.ExecuteNonQuery();

if (i > 0)

{

float total = 0;

using (cmd = new SqlCommand("sum\_of\_doc", Program.cnn))

{

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add("@doc\_id", SqlDbType.Int);

cmd.Parameters.Add("@total", SqlDbType.Money).Direction = ParameterDirection.Output;

cmd.Parameters["@doc\_id"].Value = id;

cmd.ExecuteNonQuery();

total = (float)Convert.ToDecimal(cmd.Parameters["@total"].Value);

}

MessageBox.Show("До сплати: " + total.ToString() + " грн", "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.DialogResult = System.Windows.Forms.DialogResult.OK;

this.MainForm\_Load(sender, e);

}

}

else { MessageBox.Show("Вже розрахований", "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information); }

}

private void refreshButton\_Click(object sender, EventArgs e)

{

refresh\_currentClients();

}

private void reportViewer1\_ReportRefresh(object sender, CancelEventArgs e)

{

this.MainForm\_Load(sender, e);

}

private void roomGrid\_CellEndEdit(object sender, DataGridViewCellEventArgs e)

{

}

private void roomGrid\_CellValidating(object sender, DataGridViewCellValidatingEventArgs e)

{

}

private void roomGrid\_CellEnter(object sender, DataGridViewCellEventArgs e)

{

}

private void roomGrid\_CellValueChanged(object sender, DataGridViewCellEventArgs e)

{

string field = "";

switch (e.ColumnIndex)

{

case 0: { Console.Write("bad"); break; }

case 1:

{

if (!Regex.IsMatch(roomGrid.CurrentCell.EditedFormattedValue.ToString(), @"[є-їа-яА-ЯЄ-Ї-' 0-9]\*"))

{

MessageBox.Show("Лише літери, цифри та - й ' !", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

roomGrid.CurrentCell.Value = "";

}

field = "Назва\_кімнати"; break;

}

case 2:

{

if (!Regex.IsMatch(roomGrid.CurrentCell.EditedFormattedValue.ToString(), @"^[0-9]+$"))

{

MessageBox.Show("Тільки додатні числа!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

roomGrid.CurrentCell.Value = 0;

}

field = "Кількість\_місць"; break;

}

}

string sql = "UPDATE Кімната SET " + field + "= '" + roomGrid.CurrentCell.Value +

"' WHERE id\_кімнати = " + roomGrid.CurrentRow.Cells[0].Value + ";";

cmd = new SqlCommand(sql, Program.cnn);

try { cmd.ExecuteNonQuery(); }

catch { MessageBox.Show("Перевірьте значення!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error); }

}

private void addRoomButton\_Click(object sender, EventArgs e)

{

var f = new AddRoom();

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

refresh(roomGrid, "Кімната");

}

}

private void addProductButton\_Click(object sender, EventArgs e)

{

var f = new AddProduct();

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

refresh(productGrid, "Товар");

}

}

private void changeProductTypeButton\_Click(object sender, EventArgs e)

{

var f = new ChangeFromList("Тип\_товару", "Назва\_типу\_товару", "id\_тип\_товару");

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

sql = "UPDATE Товар SET id\_тип\_товару = " + f.id +

" WHERE id\_товару = " + productGrid.CurrentRow.Cells[0].Value + ";";

refresh(productGrid, "Товар");

}

}

private void addPriceOffButton\_Click(object sender, EventArgs e)

{

var f = new addPriceOff();

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

refresh(priceOffGrid, "Знижка");

}

}

private void priceOffGrid\_CellValueChanged(object sender, DataGridViewCellEventArgs e)

{

string field = "";

switch (e.ColumnIndex)

{

case 0: { Console.Write("bad"); break; }

case 1:

{

if (!Regex.IsMatch(priceOffGrid.CurrentCell.EditedFormattedValue.ToString(), @"^[0-9]+$"))

{

MessageBox.Show("Лише літери, цифри та - й ' !", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

roomGrid.CurrentCell.Value = "";

}

field = "Назва\_кімнати"; break;

}

}

string sql = "UPDATE Знижка SET Значення\_знижки =" + priceOffGrid.CurrentRow.Cells[1].EditedFormattedValue.ToString() + ", Активна = ";

if (priceOffGrid.CurrentRow.Cells[2].FormattedValue.Equals(true)) { sql += "1 "; }

else { sql += "0 "; }

sql += "WHERE id\_знижки = " + priceOffGrid.CurrentRow.Cells[0].Value.ToString() + ";";

SqlCommand cmd = new SqlCommand(sql, Program.cnn);

try

{

int i = cmd.ExecuteNonQuery();

}

catch { MessageBox.Show("Перевірьте значення!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error); }

}

private void button9\_Click(object sender, EventArgs e)

{

var f = new ChangeFromList("Кімната", "Назва\_кімнати", "id\_кімнати");

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

string sql = "UPDATE Послуга SET id\_кімнати =" + f.id +

" WHERE id\_товару = " + serviceGrid.CurrentRow.Cells[0].Value.ToString() + ";";

SqlCommand cmd = new SqlCommand(sql, Program.cnn);

try

{

int i = cmd.ExecuteNonQuery();

}

catch { MessageBox.Show("Перевірьте значення!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error); }

refresh(serviceGrid, "Послуга");

sql = "SELECT Товар.id\_товару, Назва\_товару, Кімната.id\_кімнати, Назва\_кімнати, Тип\_часу, Ціна\_часу FROM Послуга LEFT JOIN Кімната ON Кімната.id\_кімнати = Послуга.id\_кімнати LEFT JOIN Товар ON Товар.id\_товару = Послуга.id\_товару";

cmd = new SqlCommand(sql, Program.cnn);

dt = new DataTable();

dt.Load(cmd.ExecuteReader());

serviceGrid.DataSource = dt;

serviceGrid.Columns[0].ReadOnly = true;

serviceGrid.Columns[1].ReadOnly = true;

serviceGrid.Columns[2].ReadOnly = true;

serviceGrid.Columns[3].ReadOnly = true;

}

}

private void button8\_Click(object sender, EventArgs e)

{

var f = new ChangeFromList("Товар", "Назва\_товару", "id\_товару");

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

string sql = "UPDATE Послуга SET id\_товару =" + f.id +

" WHERE id\_кімнати = " + serviceGrid.CurrentRow.Cells[2].Value.ToString() + ";";

SqlCommand cmd = new SqlCommand(sql, Program.cnn);

try

{

int i = cmd.ExecuteNonQuery();

}

catch { MessageBox.Show("Дублікати!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error); }

refresh(serviceGrid, "Послуга");

sql = "SELECT Товар.id\_товару, Назва\_товару, Кімната.id\_кімнати, Назва\_кімнати, Тип\_часу, Ціна\_часу FROM Послуга LEFT JOIN Кімната ON Кімната.id\_кімнати = Послуга.id\_кімнати LEFT JOIN Товар ON Товар.id\_товару = Послуга.id\_товару";

cmd = new SqlCommand(sql, Program.cnn);

dt = new DataTable();

dt.Load(cmd.ExecuteReader());

serviceGrid.DataSource = dt;

serviceGrid.Columns[0].ReadOnly = true;

serviceGrid.Columns[1].ReadOnly = true;

serviceGrid.Columns[2].ReadOnly = true;

serviceGrid.Columns[3].ReadOnly = true;

}

}

private void button7\_Click(object sender, EventArgs e)

{

var f = new AddService();

f.ShowDialog();

if (f.DialogResult == System.Windows.Forms.DialogResult.OK)

{

refresh(serviceGrid, "Послуга");

}

}

}

}

**Додаток 8**. Код головної форми персоналу

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BD

{

public partial class Form1 : Form

{

private string ConnectionString;

private System.Data.SqlClient.SqlConnection Connection = new System.Data.SqlClient.SqlConnection();

private string ic;

private string ic2;

private string ident\_cont;

private Boolean add\_pers = false;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

// TODO: This line of code loads data into the 'bD\_3DataSet7.Фізична\_особа' table. You can move, or remove it, as needed.

this.фізична\_особаTableAdapter1.Fill(this.bD\_3DataSet7.Фізична\_особа);

// TODO: This line of code loads data into the 'bD\_3DataSet6.Вільні\_посади' table. You can move, or remove it, as needed.

this.вільні\_посадиTableAdapter.Fill(this.bD\_3DataSet6.Вільні\_посади);

// TODO: This line of code loads data into the 'bD\_3DataSet4.Фізична\_особа' table. You can move, or remove it, as needed.

this.фізична\_особаTableAdapter.Fill(this.bD\_3DataSet4.Фізична\_особа);

// TODO: This line of code loads data into the 'bD\_3DataSet3.Посада\_total\_free' table. You can move, or remove it, as needed.

this.посада\_total\_freeTableAdapter.Fill(this.bD\_3DataSet3.Посада\_total\_free);

// TODO: This line of code loads data into the 'bD\_3DataSet1.Посада\_view' table. You can move, or remove it, as needed.

this.посада\_viewTableAdapter.Fill(this.bD\_3DataSet1.Посада\_view);

// TODO: This line of code loads data into the 'bD\_3DataSet.Посада' table. You can move, or remove it, as needed.

this.посадаTableAdapter.Fill(this.bD\_3DataSet.Посада);

this.reportViewer.RefreshReport();

this.reportViewer.RefreshReport();

this.reportViewer.RefreshReport();

this.reportViewer.RefreshReport();

}