**Зміст**

Зміст 2

Анотація 3

Вступ 4

1. Постановка задачі 5
2. Діаграми потоків даних 6

2.1. DFD-0 6

2.2. DFD-1 7

2.3. DFD-2 9

2.4. Матриця подій 10

2.5. Бізнес план 11

2.6. ER-діаграма (логічна) 11

2.7. ER-діаграма (фізична) 12

3. Керівництво користувача 13

Текст програми 26

**Анотація**

В даній роботі розроблена та описана інформаційна система «Супермаркет. Відділ продажу. Відділ кадрів».

Мета створення цього проекту – оптимізувати та автоматизувати роботу менеджерів супермаркету, полегшити їх взаємодію з робітниками супермаркету.

Програмне забезпечення (ПЗ) має інтуїтивно зрозумілий та простий інтерфейс і призначене для персоналу, який має елементарні навички роботи з комп’ютером та офісним програмним забезпеченням.

**Вступ**

На сьогоднішній день створення автоматизованих інформаційних систем є однією з домінуючих сфер розвитку нового інформаційного суспільства. У будь-якій сфері людської діяльності необхідна інформаційна система.

Мета інформатизації та комп’ютеризації – підвищити ефективність роботи. Але просте обладнання робочого місця „розумною” технікою не завжди призводить до підвищення робочих можливостей. Часто з’являється необхідність мати ефективну інформаційну систему, яка розроблена спеціально під бізнес-правила конкретного підприємства, розрахована на виконання не тільки загальних функцій, а й функцій специфічних для даного закладу. Завданням даної курсової роботи було створення такої системи для менеджерів супермаркету.

1. Постановка задачі

Реалізована в цьому проекті автоматизована система обробки даних призначена для використання менеджерами при організації роботи в супермаркеті.

Створення програми має на меті полегшити роботу менеджерів відділу продажу, автоматизувати взаємодію менеджерів з робітниками підприємства.

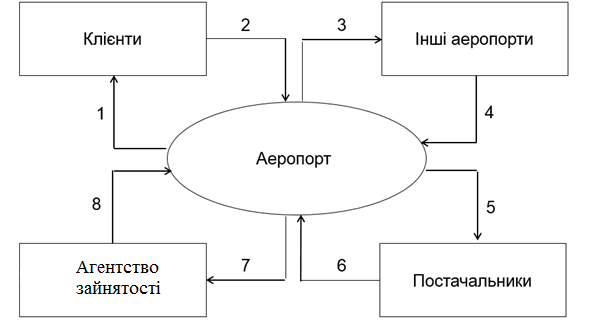
В даному курсовому проекті були реалізовані основні функції інформаційної системи, яка б сприяла оптимізації роботи менеджерів. Зокрема:

* Прийняття на роботу нового працівника підприємства;
* Звільнення працівника;
* Зміна робочого графіку працівника;
* MORE ;

Для коректної роботи програми необхідне виконання відповідних вимог до технічних і програмних засобів, а саме:

* доступ до інтернету;
* процесор з архітектурою x86-64 (Intel з підтримкою EM64T, AMD з підтримкою AMD64);
* оперативна пам’ять 1024 Мб та вище;
* жорсткий диск 40Гб та вище­­;
* SVGA-відеокарта

1. **Діаграми потоків даних** 
   1. **DFD-0**

****

**Опис потоків даних на DFD-0**

1 – Квитки, інформація про розклад

2 – Бронювання, замовлення та оплата квитків

3 – Узгодження польотів від нашого аеропорту, можливість аварійної посадки на іншому аеропорту

4 – Узгодження польотів до нашого аеропорту, можливість прийняти аварійну посадку

5 – Замовлення матеріальної бази

6 – Виконання замовлення

7 – Вакансії

8 – Резюме, кандидати.

**2.2. DFD-1**

19

22

Інші аеропорти

Клієнти

20

23



18

21

17

16

Розклад

Літаки

15

7

14

13

6

Вакансії

Склад

5

8

9

10

3

4

Постачальники

Агентство зайнятості

12

1

2

11

**Опис потоків даних на DFD-1**

1 – Замовлення, гроші.

2 – Товари.

3 – Фіксування нових матеріалів\товарів на складі.

4 – Інформація про наявні матеріали на складі.

5 – Запит на необхідні для ремонту матеріали.

6 – Видача матеріалів.

7 – Запит на працівника з необхідними вміннями.

8 – Працівник.

9 – Інформація про необхідних працівників.

10 – Нові працівники.

11 – Вакансії.

12 – Кандидати.

13 – Інформація про всі літаки та їх технічний стан.

14 – Оновлення інформації про літаки та їх технічний стан.

15 – Інформація про доступні для посадки літаки.

16 – Формування розкладу польотів

17 – Можливість здійснити аварійну посадку

18 – Можливість прийняти аварійну посадку

19 – Узгодження польотів до нашого аеропорту

20 – Узгодження польотів від нашого аеропорту

21 – Розклад польотів

22 – Квитки, інформація про розклад

23 – Бронювання, замовлення та оплата квитків

­­­

**2.3. DFD-2**

Призначені роботи

Літаки

Великі

задачі

Завдання проектів

Атомарні роботи

Персонал

Вакансії

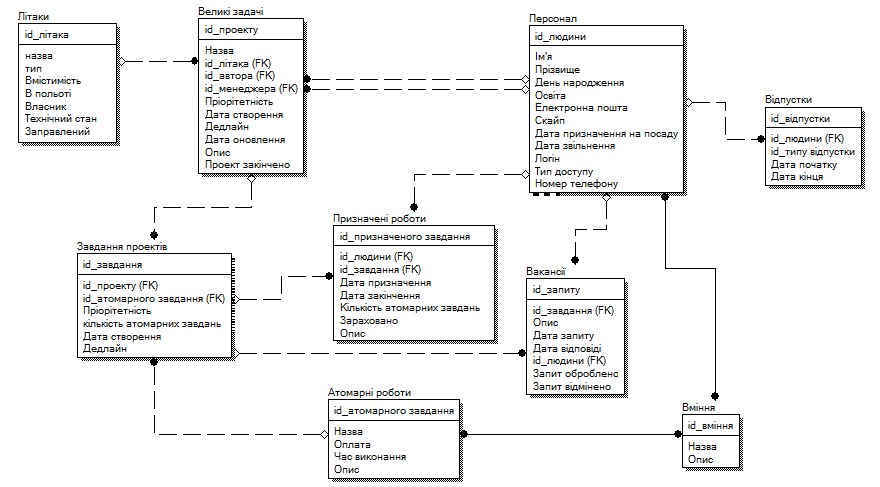
**2.4. Матриця подій**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Опис події** | **Реакція системи на подію** | **Тип події** |
| **1.** | Створення проекту/великої задачі | Надати користувачу форму для створення проекту про закупку; перевірити правильність введення даних; додати задачу у БД Великі задачі. | N |
| **2.** | Розбиття задачі на завдання | Надати користувачу форму для створення нової задачі на основі атомарних робіт. Перевірити правильність введення даних. Якщо необхідної атомарної роботи нема, то вона додається до БД. Додати задачу у БД Завдання проектів. | N |
| **3.** | Призначення працівникам завдань | Надати користувачу форму для вибору працівника з відповідними вміннями і призначення йому завдання. Додати призначену задачу у БД Призначені роботи | N |
| **4.** | Зарахування виконаного завдання | Ввести необхідні зміни до БД Призначені завдання | N |
| **5.** | Звільнення робітника із завдання | Ввести необхідні зміни до БД Призначені завдання | N |
| **6.** | Створення запиту до відділу кадрів | Надати користувачу форму для створення запиту до відділу кадрів. Додати запис у БД Вакансії | N |
| **7.** | Створення нової атомарної роботи | Надати користувачу форму для створення атомарного завдання та вибору необхідних вмінь. Додати запис у БД Атомарні роботи | N |
| **8.** | Оновлення інформації про літак | Ввести необхідні зміни до БД Літаки | N |

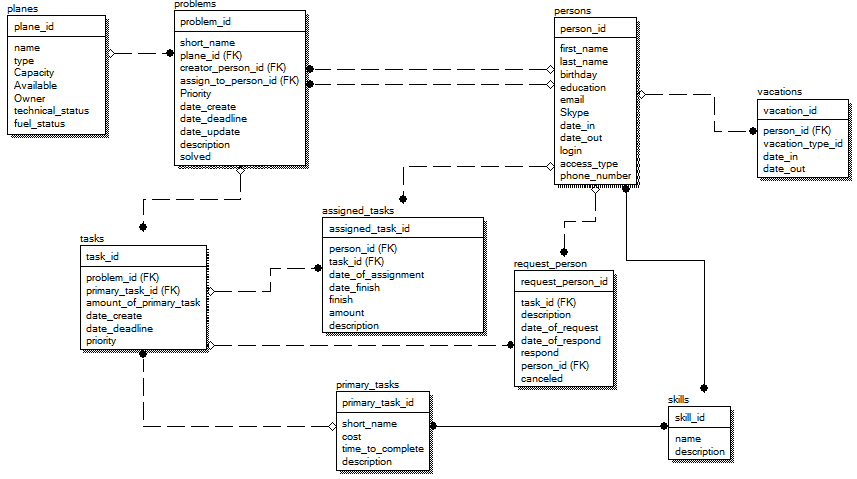
**2.5.** **Бізнес правила**

1. Відділ займається розподіленням робіт тільки між працівниками відділу обслуговування літаків.
2. Призначати роботу можна тільки працівникам, які мають необхідні для даної роботи вміння.
3. Призначати роботу можна тільки працівникам, які не у відпустці і не на лікарняному.
4. Відділ надсилає запит до відділу кадрів тільки у випадку, якщо людини з необхідними вміннями немає.
5. Працівнику оплачується тільки та робота, яка була виконана (зарахована менеджером).
6. Працівника можна звільнити тільки з того завдання, яке ще не було зарахованим менеджером.
7. У разі необхідності певних матеріалів для виконання роботи, працівник звертається до відділу матеріального забезпечення.

**2.6. ER – діаграма**

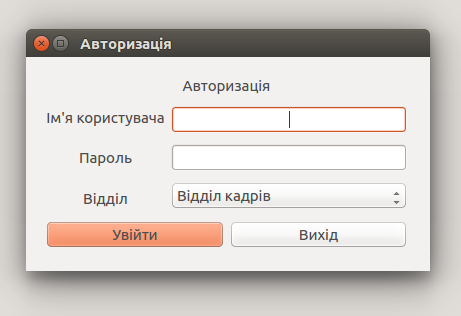
Логічний рівень

**2.7. ER – діаграма**

Фізичний рівень

1. **Керівництво користувача**

Рівень доступу до бази даних залежить від того, під яким логіном заходить користувач. Є 2 типи доступу: operator – для працівників, admin – для менеджерів. В даній роботі реалізовано інтерфейс як для типу доступу admin, так і для operator. При запуску програми з’являється вікно авторизації.



Тут користувач вводить свої логін та пароль. Якщо логін чи пароль введені невірно, то появляється вікно помилки, що повідомляє про це.

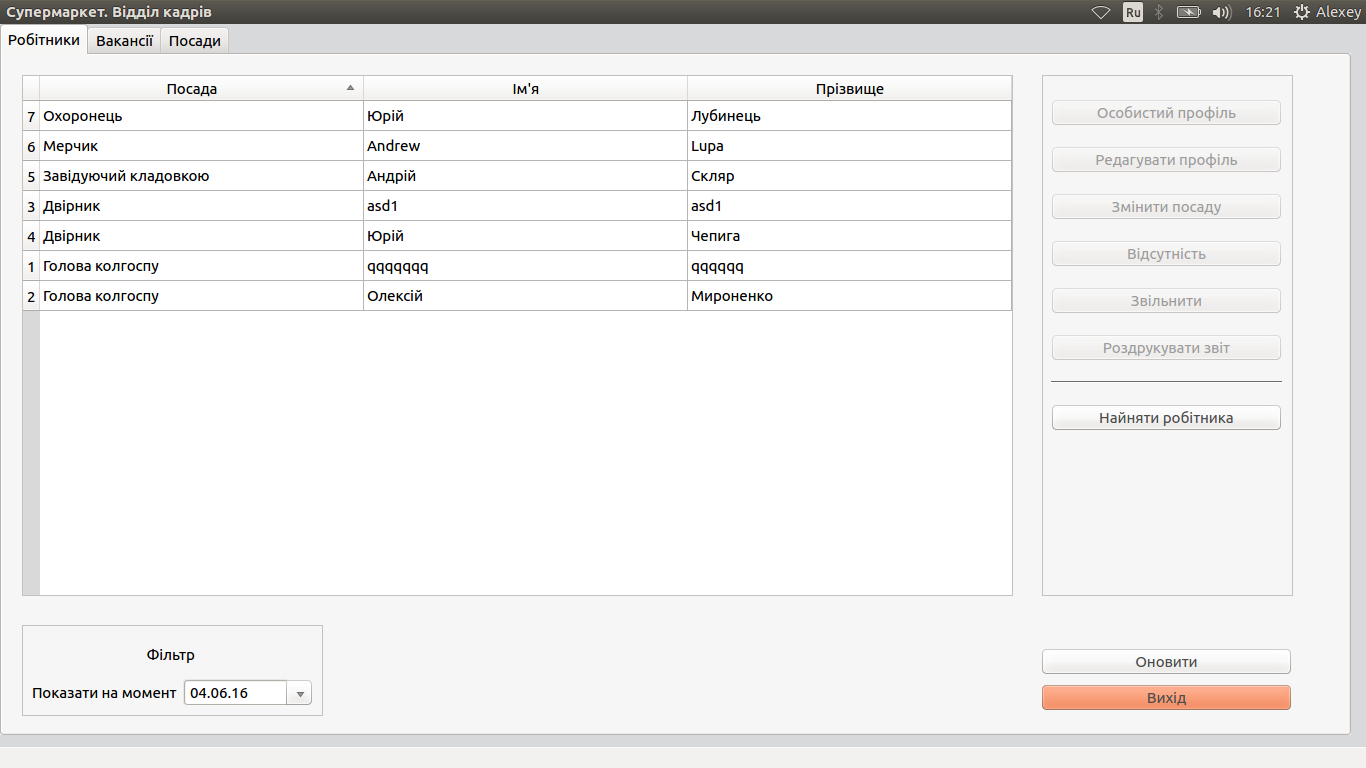
Після того, як менеджер введе вірно свої логін та пароль користувач потрапляє до свого особистого кабінету свого відділу. Відділ вибирається в селекті у вікні авторизації.

**3.1. Відділ кадрів**

У верхній частині вікна розміщені 3 вкладки — 3 основних частини відділу кадрів. Можна переходити із однієї частини в іншу, зробивши 1 клік по відповідній вкладці.

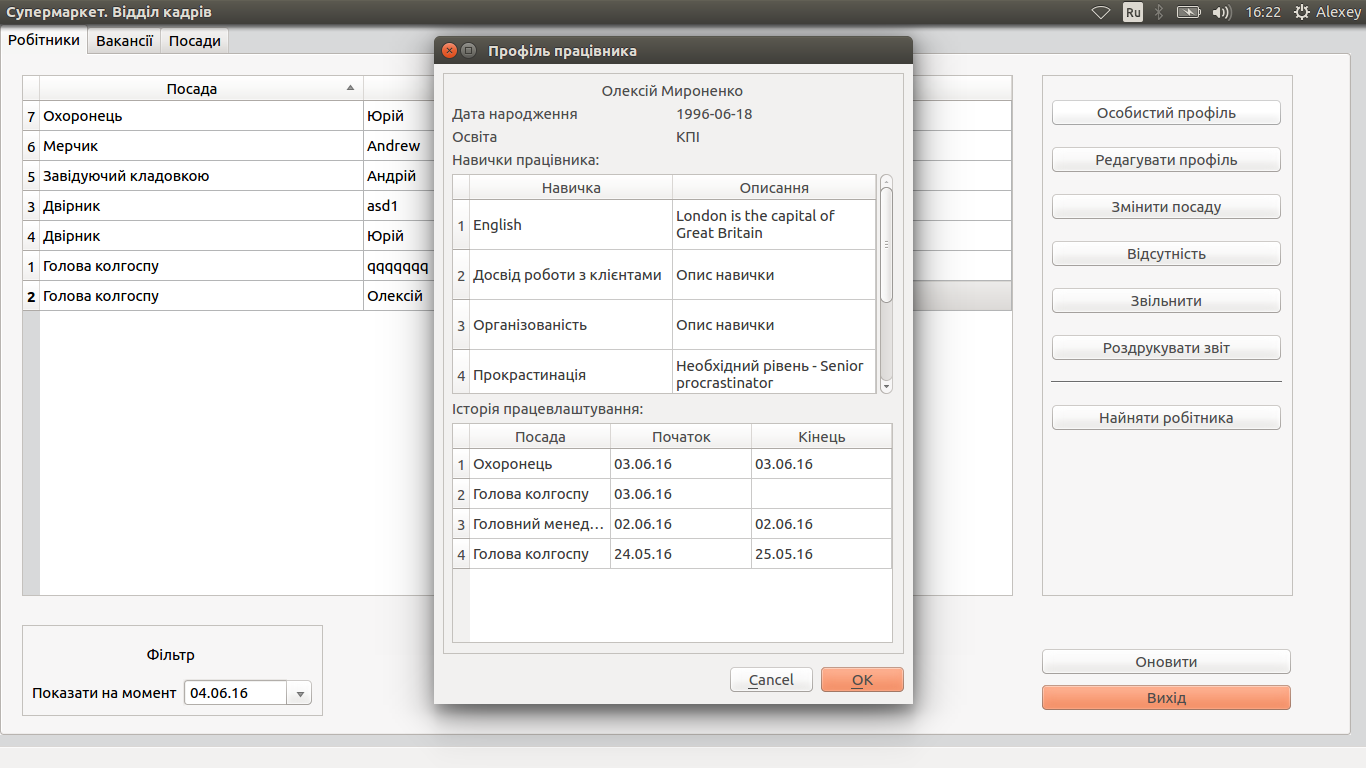
Опишемо основний функціонал програми:

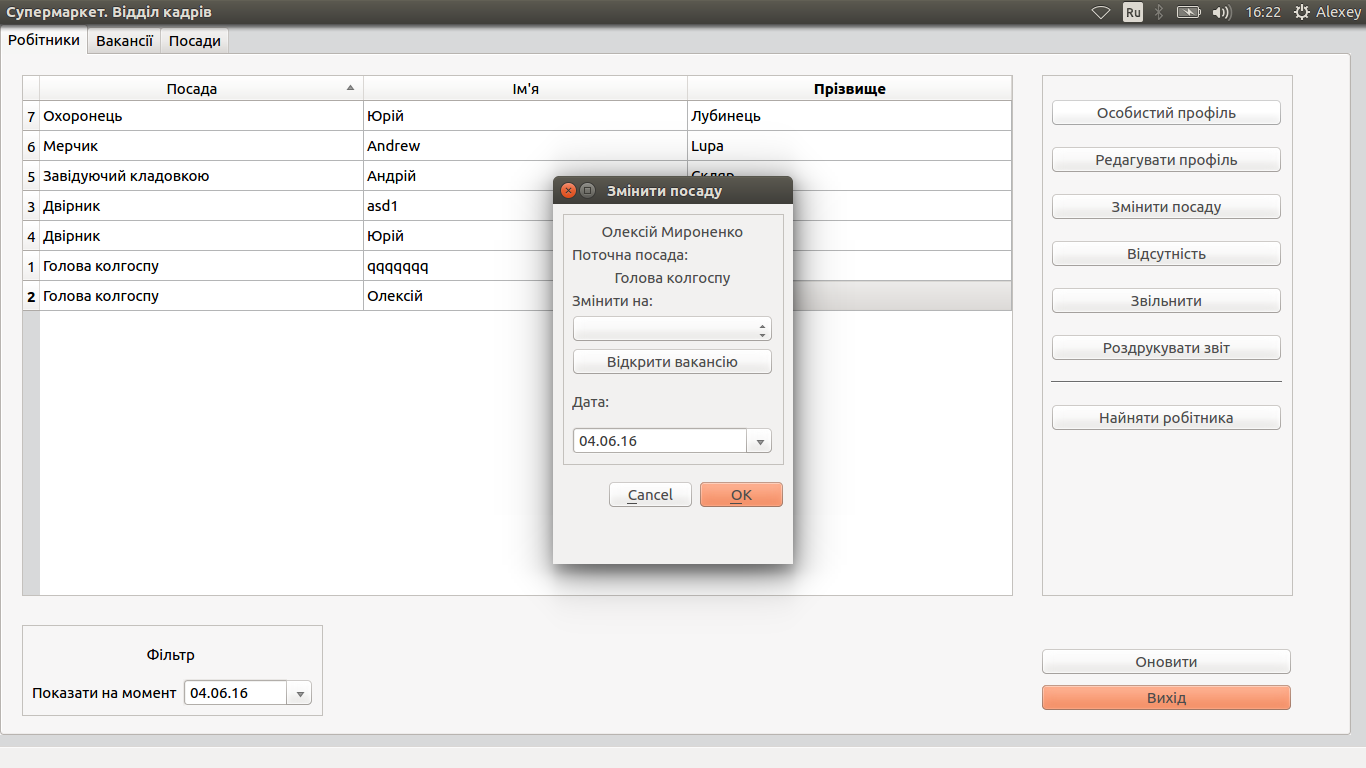
**Вікно поточних робітників супермаркету —** перша вкладка. Фільтр, розміщений в нижньому лівому кутку дає можливість переглядати історію.

****

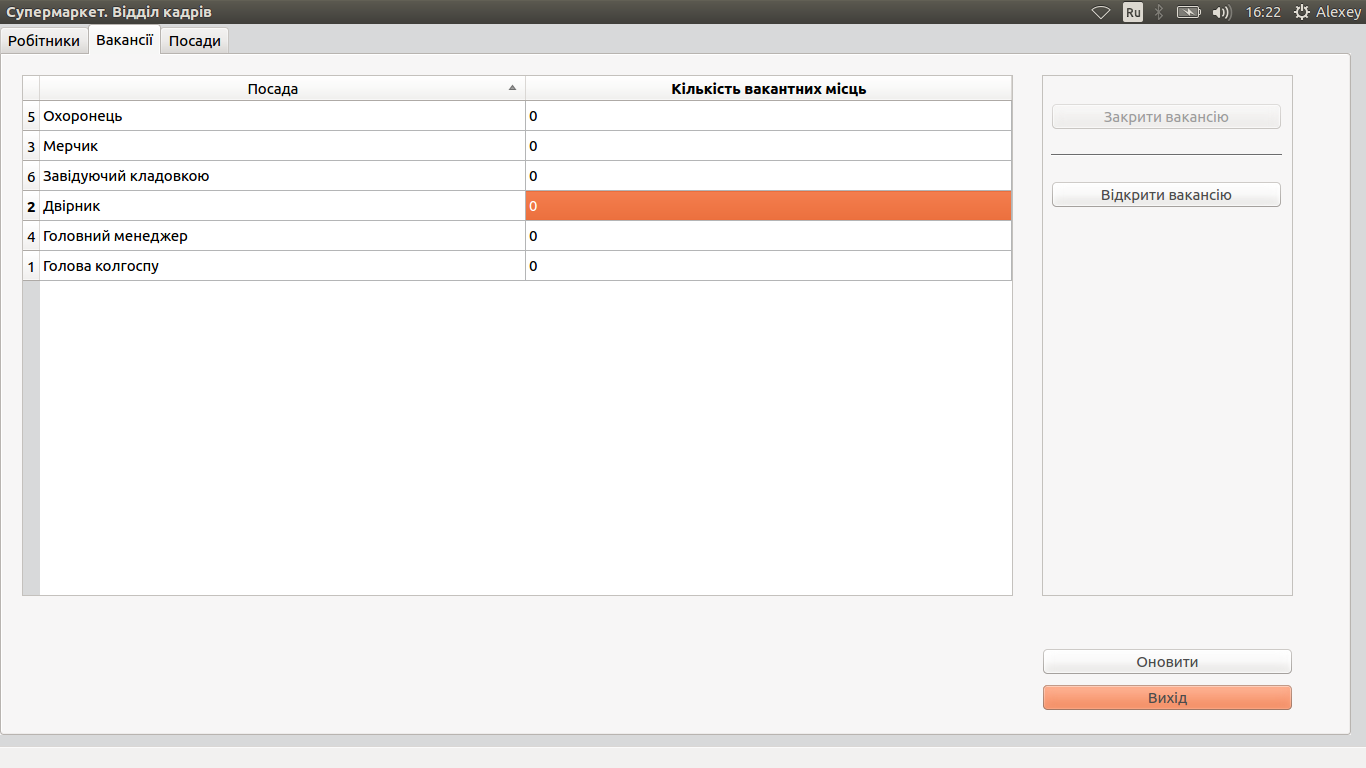
При натисканні по рядку працівника стають доступні наступні функції:

* Переглянути особистий профіль працівника з історією працевлаштувань, особистими навичками, персональною інформацією
* Редагувати дані про працівника, перелічені в пункті вище
* Змінити посаду робітника
* Переглянути і редагувати графік відсутності працівника на роботі
* Звільнити працівника з посади
* Роздрукувати особистий профіль працівника (з інформацією з пункту 1) у вигляді документу

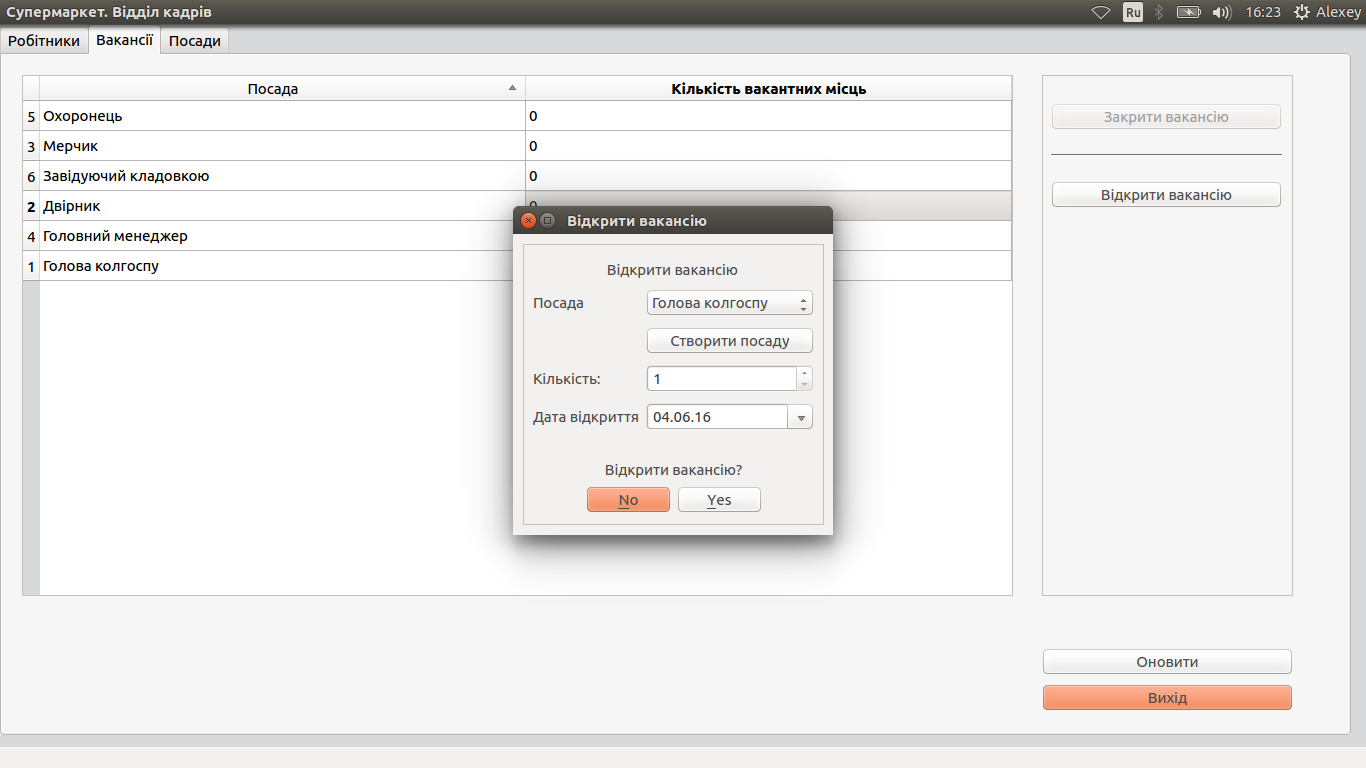
****

****

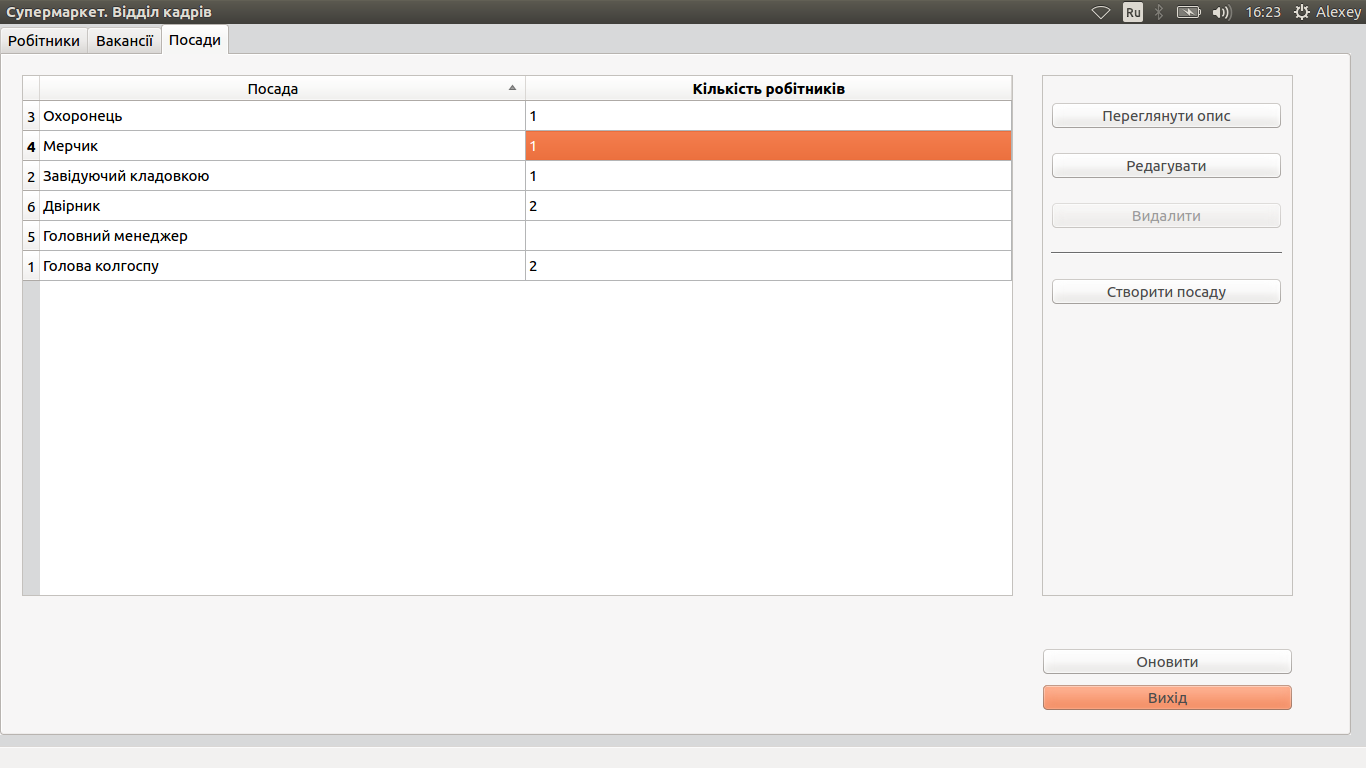
**Вікно поточних вакансій** — друга вкладка**.**



Вибравши посаду, можна створити вакансію на неї, або, якщо це можливо, закрити вакансію.



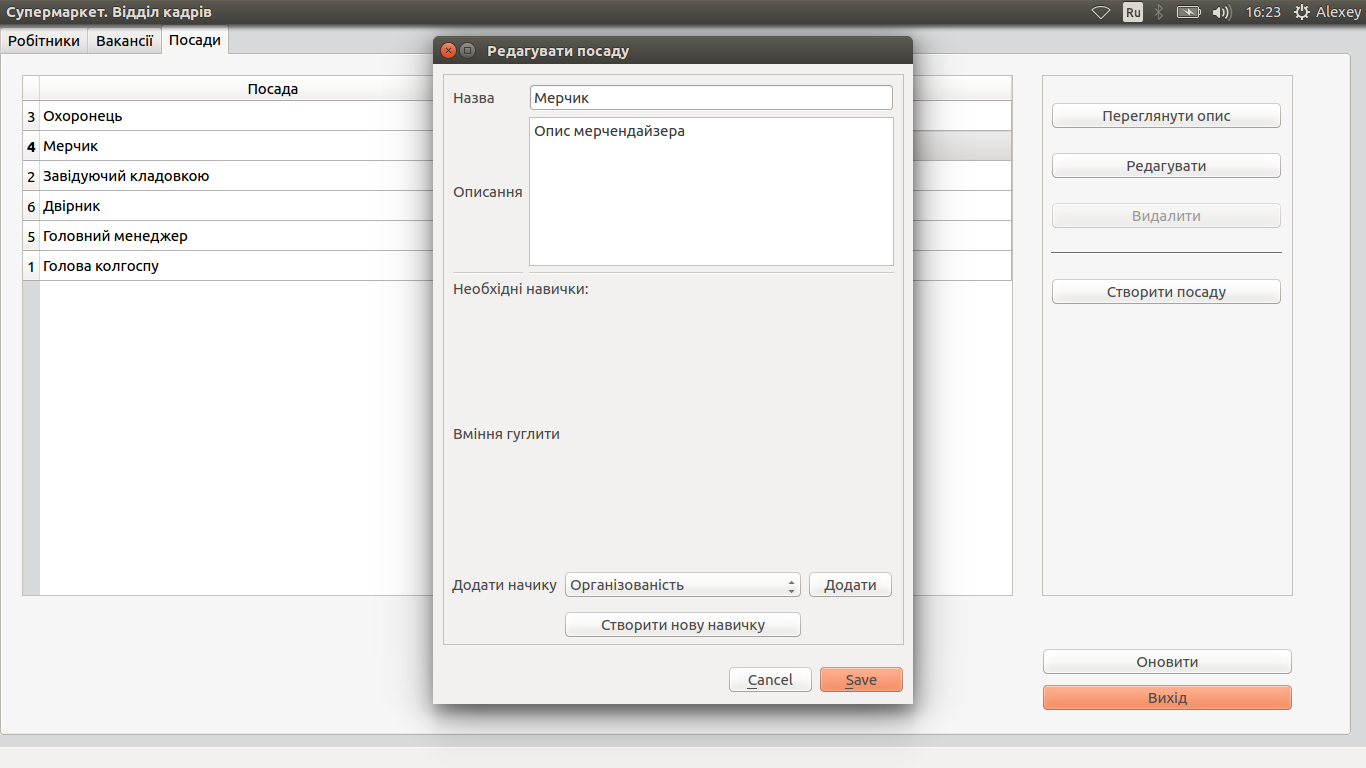
**Вікно посад підприємства** — третя вкладка.



У таблиці наведено перелік існуючих посад, а також кількість працівників на кожній з них.

При виборі посади, стають доступними наступні функції:

* переглянути детальну інформацію про посаду — опис, необхідні навички
* редагувати посаду — редагувати дані, перелічені в в пункті вище
* видалити посаду (можливо лише за відсутності працівників на ній)



**Текст програми**

Для розробки інформаційної системи було використано СУБД Postgres, яка дозволяє легко адмініструвати базу даних, зокрема створювати таблиці, тригери, процедури. Сама програма була реалізована на мові програмування – С++, що дозволяє користуватись даною програмою на будь-якій операційній системі.

***Користувач***

-- admin

CREATE USER admin WITH PASSWORD 'admin';

GRANT ALL PRIVILEGES ON DATABASE db\_work TO admin;

GRANT ALL ON SCHEMA "Myronenko\_O" TO admin;

-- hr\_operator

CREATE USER hr\_operator WITH PASSWORD 'hr\_operator';

GRANT ALL PRIVILEGES ON DATABASE db\_work TO hr\_operator;

GRANT ALL ON SCHEMA "Myronenko\_O" TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".absence\_type TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".absences TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".neccessary\_skills TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".person TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".personal\_skills TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".skills TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".staff TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".timeschedule TO hr\_operator;

GRANT ALL ON TABLE "Myronenko\_O".vacancies TO hr\_operator;

GRANT SELECT ON TABLE "Myronenko\_O".positions TO hr\_operator;

***Процедура найму працівника на посаду***

CREATE OR REPLACE FUNCTION "Myronenko\_O".add\_person\_on\_position

(IN id\_person integer, IN id\_position integer, IN startDate VARCHAR)

RETURNS void AS

$BODY$

DECLARE

opened\_vac\_by\_position integer;

position\_exists boolean;

new\_vac\_id integer;

last\_fired date;

BEGIN

IF EXISTS (

SELECT "Id\_staff"

FROM "Myronenko\_O".staff

WHERE

"Id\_person" = id\_person AND

(date\_out IS NULL OR

date\_out > to\_date(startDate, 'YYYY-MM-DD'))

) THEN RAISE EXCEPTION 'person is already working';

END IF;

IF NOT EXISTS

(SELECT \*

FROM "Myronenko\_O".positions

WHERE

"Id\_position" = id\_position AND

exists = TRUE)

THEN RAISE EXCEPTION 'position does not exist';

END IF;

SELECT sum(type) INTO opened\_vac\_by\_position

FROM "Myronenko\_O".vacancies

WHERE "Id\_position" = id\_position

GROUP BY "Id\_position";

IF opened\_vac\_by\_position <= 0

THEN RAISE EXCEPTION 'no vacancies on this position';

END IF;

SELECT nextval('inc\_primary'::regclass) INTO new\_vac\_id;

INSERT INTO "Myronenko\_O".vacancies ("Id\_vacancy", type, date, "Id\_position")

VALUES (new\_vac\_id, -1, to\_date(startDate, 'YYYY-MM-DD'), id\_position);

INSERT INTO "Myronenko\_O".staff ("Id\_person", "Id\_vacancy", date\_in)

VALUES (id\_person, new\_vac\_id, to\_date(startDate, 'YYYY-MM-DD'));

END;

$BODY$

LANGUAGE plpgsql;

***Процедура видалення посади***

CREATE OR REPLACE FUNCTION "Myronenko\_O".delete\_position

(IN id\_position integer)

RETURNS void AS

$BODY$

DECLARE

staff\_count\_by\_position integer;

opened\_vac\_count\_bu\_posit integer;

iterator integer;

BEGIN

SELECT count(\*) into staff\_count\_by\_position

FROM

"Myronenko\_O".staff staff

join "Myronenko\_O".vacancies vacancies

on staff."Id\_vacancy" = vacancies."Id\_vacancy"

WHERE

staff.date\_out IS NOT NULL AND

vacancies."Id\_position" = id\_position;

IF staff\_count\_by\_position > 0

THEN RAISE EXCEPTION 'impossible action';

END IF;

SELECT SUM(type) INTO opened\_vac\_count\_bu\_posit

FROM "Myronenko\_O".vacancies

WHERE "Id\_position" = id\_position;

SELECT 0 INTO iterator;

WHILE iterator < opened\_vac\_count\_bu\_posit

LOOP

iterator := iterator + 1;

SELECT "Myronenko\_O".delete\_last\_vacancy(id\_position, current\_date);

END LOOP;

UPDATE "Myronenko\_O".positions

SET exists = FALSE

WHERE "Id\_position" = id\_position;

END;

$BODY$

LANGUAGE plpgsql;

***Процедура звільнення працівника***

CREATE OR REPLACE FUNCTION "Myronenko\_O".fire\_staff

(IN id\_staff integer, IN fire\_date VARCHAR)

RETURNS void AS

$BODY$

DECLARE

date\_start date;

BEGIN

IF NOT EXISTS

(SELECT \*

FROM "Myronenko\_O".staff

WHERE

"Id\_staff" = id\_staff)

THEN RAISE EXCEPTION 'staff does not exist';

END IF;

SELECT staffDoc.date\_in INTO date\_start from (

SELECT date\_in

FROM "Myronenko\_O".staff

WHERE

"Id\_staff" = id\_staff AND

date\_out IS NULL

) AS staffDoc;

IF date\_start > to\_date(fire\_date, 'YYYY-MM-DD')

THEN RAISE EXCEPTION 'Error. Staff cant be fired before work start.';

END IF;

UPDATE "Myronenko\_O".staff

SET date\_out = to\_date(fire\_date, 'YYYY-MM-DD')

WHERE

"Id\_staff" = id\_staff AND

date\_out IS NULL;

END;

$BODY$

LANGUAGE plpgsql VOLATILE;