Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ І. СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №3-4**

з дисципліни “Компоненти програмної інженерії”

“Проектування та створення архітектури програмного забезпечення і реляційної бази даних Національного реєстру”

**Тема**: Єдиний реєстр спеціальних бланків нотаріальних документів

Виконали

студенти 3 курсу

групи КП-71

Песчанський Даниїл

Кравчук Аркадій

Київ 2020

**Мета роботи:** Уточнення даних, інтерфейсів, алгоритму функціонування, архітектури програми. Розроблення технічного проекту, яке охоплює проектування системи, що включає проектування архітектури системи і детальне проектування.

**Завдання на лабораторну роботу:** Проаналізувати та спроектувати архітектуру програмного забезпечення для реалізації Національного реєстру відповідно до індивідуального варіанта завдання. Задокументувати результати проектування.

СТРУКТУРА СИСТЕМИ

Структурно система складається з ізольованих один від одного рівнів, представлену у вигляді ієрархічної структури (див рис. 1), кожна частина якої взаємодії лише із рівнем, що знаходиться нижче за ієрархією. Рівні взаємодіють між собою за допомогою програмних інтерфейсів (API), які надаються кожним рівнем.



*Рис 1. Архітектура системи Реєстру спеціальних бланків нотаріальних документів*

## Опис компонентів системи Реєстру спеціальних бланків нотаріальних документів:

* Користувацький інтерфейс

Представляє собою веб-додаток, завдяки якому користувач має змогу взаємодіяти з системою реєстру спеціальних бланків нотаріальних документів. Містить сторінки передбачені функціональними вимогами, які були перелічені при формуванні вимог до програмного забезпечення функціонального реєстру.

* Серверна частина

Сервер, який має доступ до БД реєстру і до якого має доступ веб-додаток. Структурно складається із наступних програмних модулів:

* + модуль авторизації;
  + модуль для роботи із спеціальними бланками нотаріальних документів у реєстрі ;
  + модуль роботи із профілями Реєстраторів та Адміністраторів;
  + модуль для роботи із логами про діяльність Реєстраторів.
* БД реєстру

Представляє собою сервер СКБД. Зберігає дані про спеціальні бланки нотаріальних документів, користувачів (Адміністраторів, та Реєстраторів) та логи змін, внесених до записів реєстраторами.

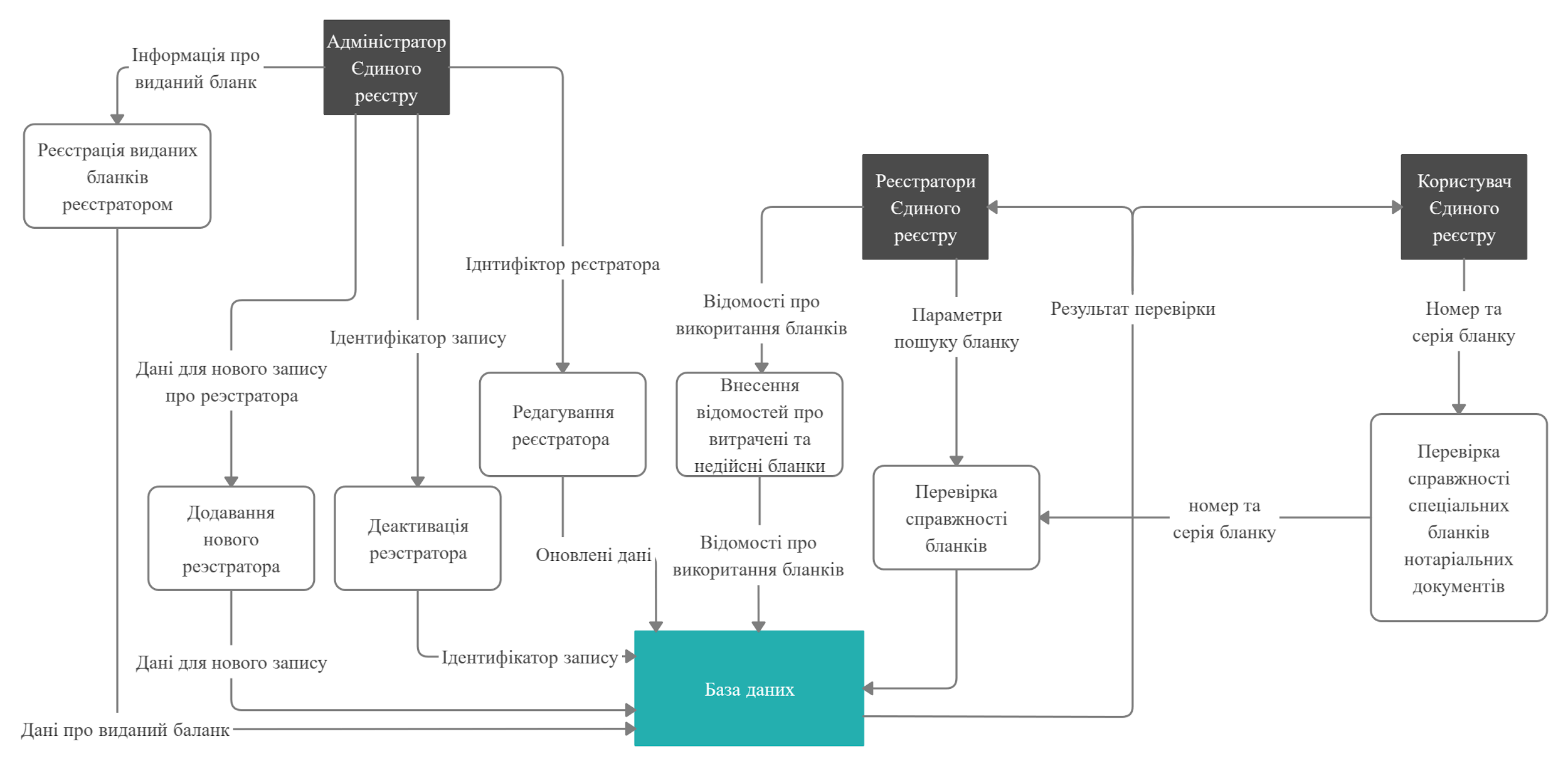
ФУНКЦІЇ І ЗАВДАННЯ, ЯКІ ВИРІШУЮТЬСЯ СИСТЕМОЮ

Система призначена для вирішення наступних завдань:

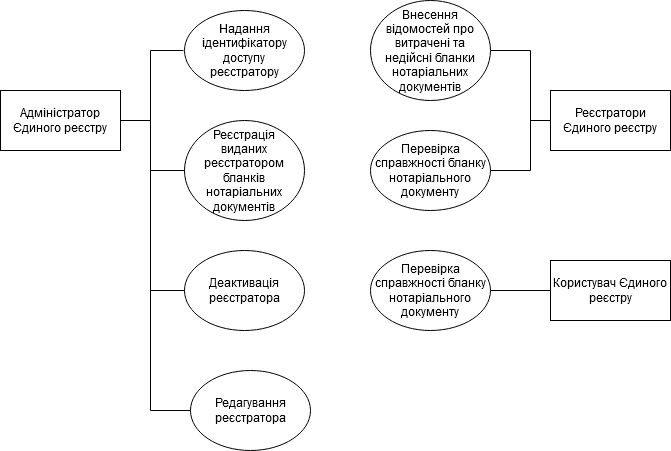
* Реєстрація відомостей про видачу нотаріусам спеціальних бланків (внесення, виправлення або видалення інформації про операції прийому-передачі бланків);
* Реєстрація відомостей про витрачання спеціальних бланків, а також відомостей про недійсні бланки (внесення, виправлення або видалення звітів про витрачання нотаріальних бланків);
* Перевірка справжності спеціальних бланків за даними РНБ для Користувачів;
* Статистичний аналіз відомостей РНБ щодо обліку постачання та звітності витрачання спеціальних бланків (формування та друк статистичних відомостей);
* Отримання інформації щодо кожного виданого та/або використаного спеціального бланка для Реєстраторів.

БІЗНЕС-ЛОГІКА ПРОГРАМИ

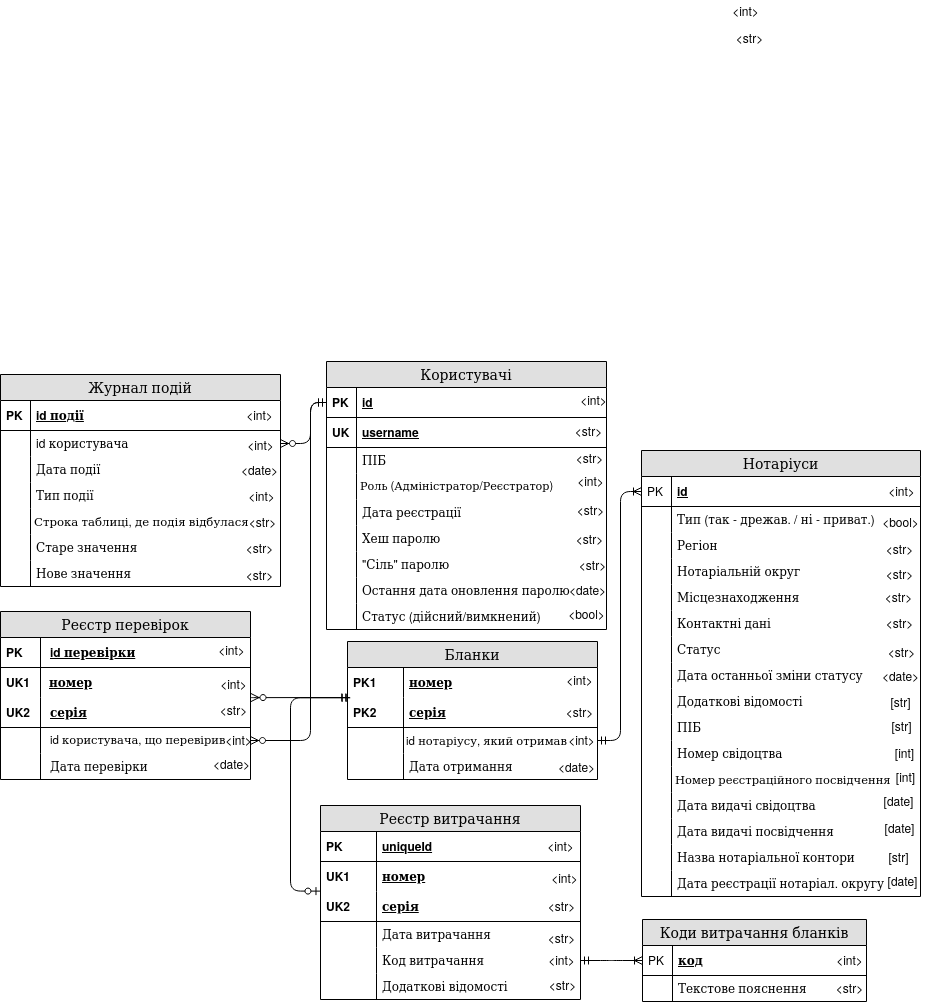
*Загальні сценарії* – нижче наведено схему інформаційних потоків, що відображає вхідні і вихідні дані, їх джерела, пункти призначення і зберігання.



*Сценарії використання* – нижче наведено модель запланованих функцій системи і околосистемних об'єктів, яка служить в якості угоди між замовником і розробниками.



СТРУКТУРА ДАНИХ



*Рис 2. ER-діаграма для БД розроблюваного реєстру*

Опис схеми БД Реєстру:

Таблиці:

* Користувачі – визначає усіх користувачів в системі, їх поточний стан та роль, відповідно до якої надаються певні функціональні можливості. Також містить дані, за якими проводиться авторизація та аутентифікація.
* Бланки – визначає сутність бланку за унікальним поєднанням серії та номеру бланку. Також зберігається дата видачі спеціального бланку нотаріусу. Має строге відношення один до одного з таблицею нотаріусів.
* Реєстр витрачання – визначає необхідну інформацію про витрачання бланків нотаріусом. Має відношення один до одного з таблицею бланки, але відношення є не строгим, адже бланки можуть бути зареєстровані в системі, але ще не використані. Також має зовнішній ключ: код витрачання, який посилається на відповідну таблицю.
* Коди витрачання бланків – визначає текстову відповідність для номера витрачання бланку. Використовується під час виводу інформації для полегшення сприйняття інформації.
* Нотаріуси – відображає сутність нотаріусу у системі. В цій таблиці будуть розташовані дані як про приватних, так і про державних нотаріусів. Тип записаного нотаріусу буде визначатися окремою булевою змінною. Поля, які не потрібні для даного типу нотаріуса будуть мати відсутнє значення (null). Спільні поля для двох типів нотаріусу мають зліва позначення типу даних у трикутних дужках <...>. До державного типу нотаріусу відносяться такі поля, як: назва нотаріальної контори та дата реєстрації нотаріального округу. Відповідно усі інші поля, у яких тип даних позначається [...] – відносяться до приватного нотаріусу. Якби розроблюваний реєстр мав би можливість інтегрування з існуючими реєстрами, то тоді б дану таблицю не довелося б зберігати, а дані можна було б отримувати з відповідного реєстру.
* Реєстр перевірок – зберігає інформацію про проведені перевірки спеціальних бланків адміністраторами та реєстраторами. Фіксується дата перевірки, та зовнішні ключі: серія і номер бланку та ідентифікатор користувача. Передбачена законодавством (див. пункт 3.3 – 3.6):  
  <https://zakon.rada.gov.ua/laws/show/z1043-09>.
* Журнал подій – призначена для зберігання змін даних в таблицях реєстру. Під зміною даних мається на увазі операції видалення, додавання та зміни записів у Реєстрі.

Типи полів:

На схемі визначені позначені у дужках <…>. int означає цілочисельне значення, bool означає булеве значення, str означає строкове значення, date означає значення типу дати.

Значення у дужках […] позначає можливу вілсутність данних.

Зв’язки між таблицями:

позначені стрілками, які визначають відношення: (не-) строго один до одного, (не-) строго один до багатьох.

Первинні та унікальні ключі:

На схемі визначені позначками “PK” та “UK” відповідно.

Тип СКБД:

Реляційна база даних PostgreSQL 12.2.

**SQL-скрипти бази даних та запитів до неї**

|  |
| --- |
| create\_tables.sql |
| CREATE TABLE IF NOT EXISTS blank  (  num int NOT NULL, -- номер  series varchar(2) NOT NULL, -- серія  notarius\_id bigint NOT NULL, -- id нотаріуса, що отримав бланки  date\_receiving date NOT NULL, -- дата отримання  CONSTRAINT blank\_pkey PRIMARY KEY (num, series ),  CONSTRAINT foreign\_key\_to\_notarius FOREIGN KEY (notarius\_id )  REFERENCES notarius (id) MATCH FULL  ON UPDATE RESTRICT  ON DELETE RESTRICT  )  CREATE TABLE TABLE IF NOT EXISTS code\_usages\_blank  (  code int NOT NULL, -- код  text\_representation text NOT NULL, -- текстове пояснення  CONSTRAINT code\_usages\_blank\_pkey PRIMARY KEY (code)  )  CREATE TABLE IF NOT EXISTS users  (  id serial NOT NULL, -- унікальний ідентифікатор користувача  name text NOT NULL, -- ПІБ користувача  role int NOT NULL, -- роль(Адміністратор/реєстратор)  date\_registration date NOT NULL, -- дата реєстрації  username text NOT NULL, -- логін користувача  pwd\_hash text NOT NULL, -- хеш паролю  pwd\_salt text NOT NULL, -- "сіль" паролю  date\_last\_update date NOT NULL, -- остання дата оновлення ідентифіка  status boolean NOT NULL, -- статус(дійсний/вимкнений)  CONSTRAINT users\_pkey PRIMARY KEY (id)  CONSTRAINT unq\_username UNIQUE(username)  )  CREATE TABLE IF NOT EXISTS notarius  (  id serial NOT NULL, -- унікальний ідентифікатор  type boolean NOT NULL, -- тип(приватний/державний)  status text NOT NULL, -- статус  date\_status\_update date NOT NULL, -- дата набуття статусу  num\_certificate text, -- номер свідоцтва  num\_card bigint, -- номер посвідчення  name text, -- ПІБ нотаріуса  name\_organization text, -- назва нотаріальної контори  region text NOT NULL, -- регіон  contacts text NOT NULL, -- контактні дані  notarius\_region text NOT NULL, -- нотаріальний округ  additional\_info text, -- додаткові відомості  date\_issue\_certificate date, -- дата видачі свідоцтва  date\_issue\_card date, -- дата видачі посвідчення  date\_reg\_region date, -- дата реєстрації нотаріального округу  location text NOT NULL, -- місцезнаходження  CONSTRAINT notarius\_pkey PRIMARY KEY (id)  )  CREATE TABLE IF NOT EXISTS usages\_register  (  id serial NOT NULL, -- унікальний ідентифікатор витрачання  num\_blank int NOT NULL, -- номер бланку  series\_blank text NOT NULL, -- серія бланку  date\_usage date NOT NULL, -- дата витрачання  code\_usage int NOT NULL, -- код витрачання  additional\_info text, -- додаткові відомості  CONSTRAINT usages\_register\_pkey PRIMARY KEY (id),  CONSTRAINT blank\_ukeys UNIQUE(num\_blamk, series\_blank),  CONSTRAINT foreign\_key\_to\_code\_usage FOREIGN KEY (code\_usage)  REFERENCES code\_usages\_blank (code) MATCH FULL  ON UPDATE RESTRICT  ON DELETE RESTRICT  )  CREATE TABLE IF NOT EXISTS verifications\_register  (  id serial NOT NULL, -- id перевірки  num\_blank int NOT NULL, -- номер бланку, що перевіряється  series\_blank text NOT NULL, -- серія бланку, що перевіряється  user\_id bigint NOT NULL, -- id користувача, що перевірив  date\_verification date NOT NULL, -- дата перевірки  CONSTRAINT verifications\_register\_pkey PRIMARY KEY (id),  CONSTRAINT blank\_ukeys UNIQUE(num\_blank, series\_blank)  )  CREATE TABLE IF NOT EXISTS journal\_actions  (  id serial NOT NULL, -- id події  user\_id bigint NOT NULL, -- id користувача  action\_date date NOT NULL, -- дата події  action\_type int NOT NULL, -- тип події  row\_affected text NOT NULL, -- строка таблиці де подія відбулася  old\_value text NOT NULL, -- старе значення  new\_value text NOT NULL, -- нове значення  CONSTRAINT journal\_actions\_pkey PRIMARY KEY (id),  CONSTRAINT foreign\_key\_to\_user\_id FOREIGN KEY (user\_id )  REFERENCES users (id) MATCH FULL  ON UPDATE RESTRICT  ON DELETE RESTRICT  ) |

**Приклади запитів до бази даних**

Приклад внесення даних про отримання бланків нотаріусом:

|  |
| --- |
| add\_blanks.sql |
| INSERT INTO blank(num, series, notarius\_id, date\_receiving) VALUES (234135, “AB”, 135135, CURRENT\_TIMESTAMP); |

Внесення інформації про використання бланка:

|  |
| --- |
| use\_blanks.sql |
| INSERT INTO usages\_register(num\_blank, series\_blank, date\_usage, code\_usage, additional\_info) VALUES (234135, “AB”, CURRENT\_TIMESTAMP, 20, NULL); |

Пошук бланків з певним кодом витрачання:

|  |
| --- |
| search\_blanks.sql |
| SELECT num\_blank, series\_blank FROM usages\_register WHERE code\_usage=32; |

Перевірка витрачання бланку за серією та номером:

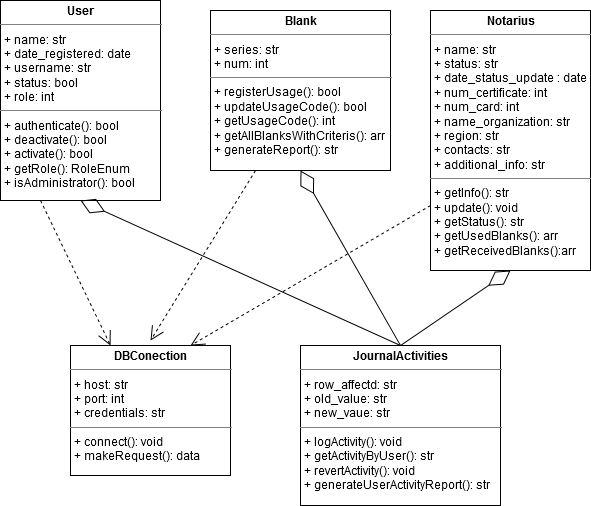
|  |
| --- |
| verification\_blanks.sql |
| INSERT INTO verifications\_register(num\_blank, series\_blank, user\_id, date\_verification) VALUES (234135, “AB”, 758, CURRENT\_TIMESTAMP);  SELECT \* FROM usages\_register WHERE num\_blank = 234135 AND series\_blank = “AB”; |

Дезактивація користувача за юзернеймом:

|  |
| --- |
| deactivate\_user.sql |
| UPDATE users SET status = false WHERE id = (SELECT id FROM users WHERE username = “some\_username”); |

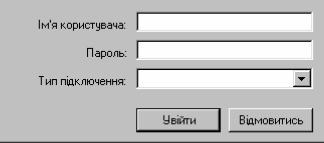
**Класи об’єктно-орієнтованого програмування**

Класи об’єктно-орієнтованого програмування, які будуть використані при розробці ПЗ Реєстру зазначені на діаграмі нижче:

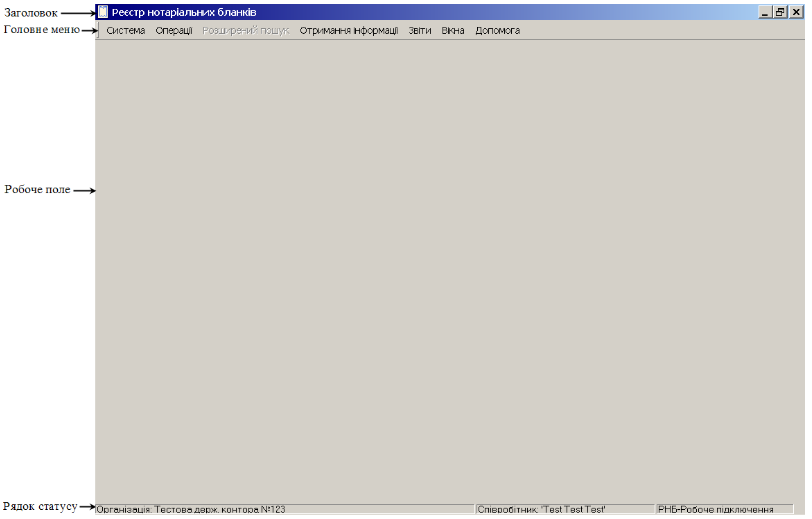


АРХІТЕКТУРА КОРИСТУВАЦЬКОГО ІНТЕРФЕЙСУ ПРОГРАМИ

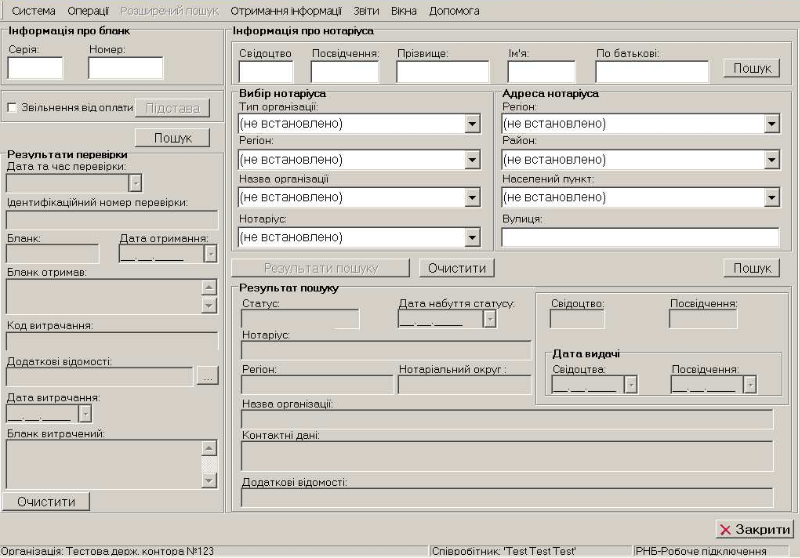
(основний функціонал)



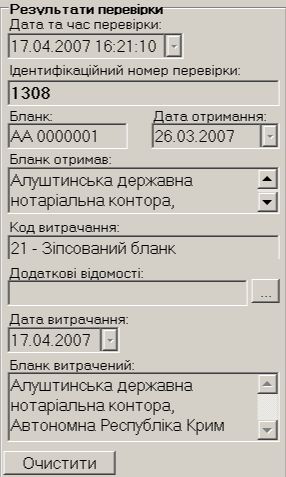
*рис. 2.1 вікно «Вхід в систему»*

**

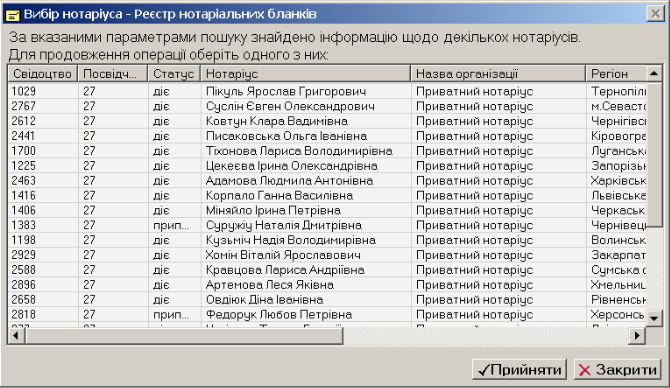
*рис. 2.2 вікно «Головне вікно програми»*

**

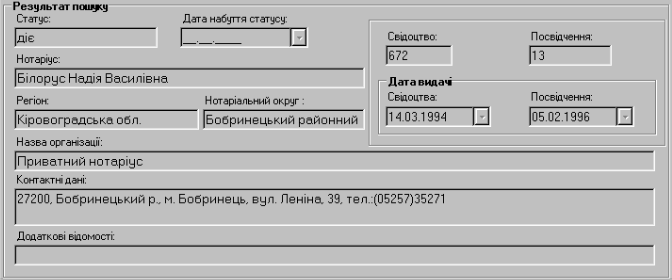
*рис. 2.3 вікно «Перевірка – Реєстр нотаріальних бланків»*

**

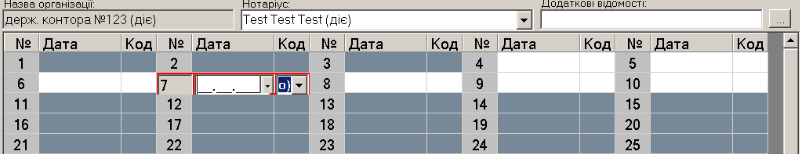
*рис 2.6 вікно «Результат перевірки справжності нотаріального бланка»*



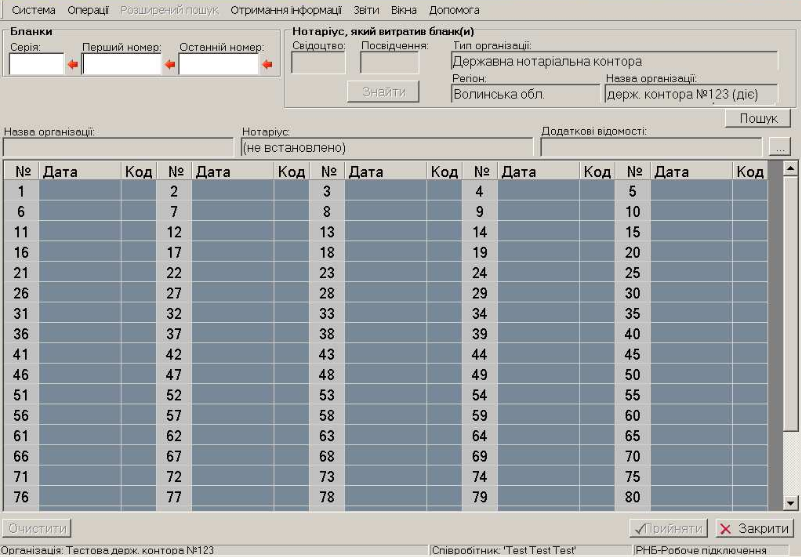
*рис. 2.4 вікно «Вибір нотаріуса – Реєстр нотаріальних бланків»*

**

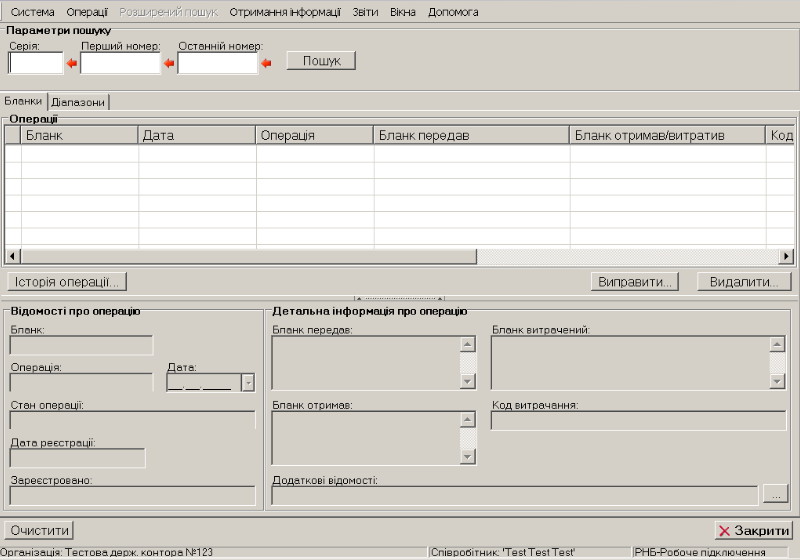
*рис 2.5 вікно «Результат пошуку інформації про нотаріуса»*

****

*рис 2.8 вікно «Внесення звітів зі знайденими бланками»*

**

*рис 2.7 вікно «Звіт про витрачання – Реєстр нотаріальних бланків»*

**

*рис 2.9 вікно «Операції з бланками – Реєстр нотаріальних бланків»*

ЗАСОБИ РОЗРОБКИ І СКЛАД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

Розроблюване програмне забезпечення складатиметься з:

* Серверної частини (server-side) – складається з переліку модулів(наведених у описі структури системи), які в сукупності виконують основну роботу системи.
* Клієнтської частини (client-side) – включає в себе обмін даними з серверною частиною та візуалізацію отриманих від серверу даних.
* Тести серверної та клієнтської частини – автоматизовані тести основного функціоналу системи.

Розроблювана система буде веб-орієнтованою. Як засоби розробки буде використано мову програмування Python 3.6 для серверної частини, та мова програмування JavaScript із фреймворком React. В якості СКБД виступатиме PostgreSQL.

ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

Технічне забезпечення системи Реєстру складається із серверного обладнання для розгортання розробленого програмного забезпечення.

Даний сервер має забезпечувати швидку роботу системи Реєстру, тому він має відповідати наступним вимогам:

* Статична IP адреса в мережі Інтернет.
* Пропускна здатність інтернет-з'єднання: не менше 100 Mbps.
* ОС: Linux Debian 10.3 або Windows Server 2019.
* CPU: 4 ядра, тактова частота, не менше: 3.5 GH.
* RAM: не менше 32 GB.
* HDD space: не менше 256 GB.
* Предстановлене ПЗ: Docker, Docker-compose, Kubernetes.

ЧИСЕЛЬНІСТЬ, КВАЛІФІКАЦІЯ І ФУНКЦІЇ ПЕРСОНАЛУ

Персонал: Кравчук Аркадій, Песчанський Даниїл.

Чисельність персоналу: 2 людини.

Кваліфікації персоналу: інженери програмного забезпечення комп’ютерних систем, які є студентами третього курсу ФПМ КПІ ім. Ігоря Сікорського.

Функції персоналу: вчасне і якісне завершення процесу створення та тестування ПЗ відповідно до вимог, які містяться у ТЗ системи, розгортання системи на виділеному сервері.

Розподіл робіт між учасниками:

* Кравчук Аркадій: розробка користувацького інтерфейсу та архітектури БД, тестування серверної частини.
* Песчанський Даниїл: розробка серверної частини та архітектури БД, тестування користувацького інтерфейсу.