НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет
Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:
Завідувач кафедри
_____Наталія АУШЕВА
«___» _____ 2022 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Комп'ютерний моніторинг та геометричне моделювання процесів і систем»

спеціальності 122 «Комп'ютерні науки»

на тему: «Програмна система дескриптивного аналізу медико-біологічних даних»

Виконав:
Студент IV курсу, групи ТР-81
Снитко Олександр Дмитрович _____

Керівник:
доцент, кандидат технічних наук
Крячок Олександр Степанович _____

Рецензент:
доцент, кандидат технічних наук
Реуцький Миколай Олександрович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2022

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет: теплоенергетичний

Кафедра: автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти: перший, бакалаврський

Напрям підготовки: 122 – Комп'ютерні науки

Спеціалізація: «Комп'ютерний моніторинг та геометричне моделювання процесів і систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Наталія АУШЕВА
(підпис)
«___» _____ 2022 р.

ЗАВДАННЯ
на дипломну роботу студенту
_____Снитко Олександр Дмитрович_____
(прізвище, ім'я, по батькові)

1. Тема роботи: Програмна система дескриптивного аналізу медико-біологічних даних

керівник роботи: Крячок Олександр Степанович, кандидат технічних наук, доцент
(прізвище, ім'я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від «8» червня 2022 р. № 965-с

2. Строк подання студентом роботи: 10 червня 2022 р.
3. Вихідні дані до роботи: мова програмування PHP, середовище розробки PHPStorm
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): розробити програмний продукт дескриптивного аналізу медико-біологічних даних; зрозуміти і ознайомитися з програмними ресурсами, які схожі до тих, що розробляються в дипломній роботі. Проконсультуватися з фахівцями та отримати необхідну документацію та літературу.
5. Перелік ілюстративного матеріалу: «Завдання розробки web-сервісу дескриптивного аналізу медико-біологічних даних», «Аналіз існуючих підходів описового аналізу медико-біологічних даних», «Засоби розробки», «Опис програмної реалізації», «Робота користувача з програмною системою», «Висновки».

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання «10» жовтня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	8 червня 2022 р.	
2.	Вивчення та аналіз задачі	2 травня 2022 р.	
3.	Розробка архітектури та загальної структури системи	10 травня 2022 р.	
4.	Розробка структур окремих підсистем	15 травня 2022 р.	
5.	Програмна реалізація системи	22 травня 2022 р.	
6.	Оформлення пояснювальної записки	28 травня 2022 р.	
7.	Захист програмного продукту	2 червня 2022 р.	
8.	Передзахист	9 червня 2022 р.	
9.	Захист	20 червня 2022 р.	

Студент _____ Снитко О.Д.
(підпис) (прізвище та ініціали,)

Керівник роботи _____ Крячок О.С.
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

Метою дипломної роботи є дослідження і розробка програмного забезпечення для дескриптивного аналізу медико-біологічних даних. Ознайомлення з необхідними для описового аналізу методами і їх реалізація. В результаті роботи системи отримаємо звіт, який може бути завантажений в різних форматах та використаний користувачем для наступних аналізів.

В результаті виконання роботи було створено 19 методів статистичного аналізу: середнє гармонійне значення, дисперсія, коефіцієнт осциляції, лінійний коефіцієнт варіації, середнє лінійне відхилення, середнє квадратичне та інші значення.

Дипломна робота має обсяг 44 аркушів, містить 1 додаток і 20 посилань. Також наведено 20 рисунків і 14 формул.

Ключові слова: дескриптивний аналіз, статистичний аналіз, експорт результатів аналізу, безпека даних, методи дослідження.

ABSTRACT

The aim of the thesis is to research and develop software for descriptive analysis of biomedical data. Acquaintance with the methods necessary for descriptive analysis and their implementation. As a result of the system we will receive a report that can be downloaded in various formats and used by the user for further analysis.

As a result of the work 19 methods of statistical analysis were created: mean harmonic value, variance, oscillation coefficient, linear coefficient of variation, linear deviation, standard deviation and other values.

Thesis has a volume of 44 sheets, contains 1 appendix and 20 references. There are also 20 figures and 14 formulas.

Key words: descriptive analysis, statistical analysis, export of analysis results, data security, research methods.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SQL — мова структурованих запитів (англ. Structured query language)

UI — користувальницький інтерфейс (англ. User Interface)

PHP — PHP: гіпертекстовий препроцесор (англ. PHP: Hypertext Preprocessor)

CPU — центральний процесор (англ. Central processing unit)

MSE — середньоквадратична помилка (англ. Minimum square error)

ООП — Об’єктно-орієнтоване програмування (англ. Object-oriented programming)

SPSS — статистичний пакет для суспільних наук (англ. Statistical Package for the Social Sciences)

HTML — мова гіпертекстової розмітки (англ. HyperText Markup Language)

CSS — каскадні таблиці стилів (англ. Cascading Style Sheets)

DOM — об'єктна модель документа (англ. Document Object Model)

БД — база даних

ВСТУП

Важко уявити життя без автоматизації повсякденних справ. Тому все спрямовано зараз на те, щоб полегшити виконання рутинних справ, але одночасно з тим, існує і те, на що автоматизація вплинути не може – здоров'я. Здавалось, а чим автоматизація може допомогти саме нашому організму, організму кожної людини? Відповідь дуже проста: обстеження, різні аналізи, які ми здаємо регулярно, і саме ці результати необхідно розрахувати. Як правило, не можна отримати точний результат, всі результати - непрості алгоритми, які працюють з великими масивами інформації. І тому необхідно створювати і вдосконалювати існуючі системи, які працюють і допомагають мільярдам людей.

Метою даної роботи є аналіз та програмна розробка методів отримання найбільш точних результатів дескриптивного дослідження методи-біологічний даних. Також розробити зручний та зрозумілий інтерфейс взаємодії користувачів. Не менш важливим є питання безпеки даних, тому всі результати мають бути або публічними, або приватними, в залежності від бажання користувача.

Завданнями роботи є ознайомлення та використання методів аналізу медико-біологічний даних для вирішення проблем та вдосконалення існуючих підходів та систем.

В інтернеті існує багато інформації про методи аналізу, але, разом з тим, існує і чимало суперечливих поглядів. Тому необхідно докладно перевіряти кожен метод та звертатися до спеціальної літератури. Як правило, всі схожі сервіси не є публічними, тому важко знайти аналог, щоб виокремити всі плюси і працювати з негативними сторонами. Саме тому система має бути: максимально точною в розрахунках, зрозумілою для користувачів та з можливістю тривалий час зберігати результати аналізу.

Система буде створена мовою програмування PHP версії 8.1 з використанням фреймворків Laravel для серверної частини та HTML, CSS і Bootstrap для клієнтської. Для збереження інформації використовуємо базу даних PostgreSQL 14.3 версії. Для розгортання проекту створено docker image. Для зручного збереження

результатів аналізу використовуємо розширення PhpSpreadsheet та PdfDOM. У підсумку, створено не тільки програмну систему, а й зручний спосіб встановлення на налаштування застосунку, яке буде відбуватися виконанням лише однієї команди до командного рядка.

Необхідно створити сервіс з можливістю дескриптивного аналізу медико-біологічних даних.

1 ЗАДАЧА ДЕСКРИПТИВНОГО АНАЛІЗУ МЕДИКО-БІОЛОГІЧНИХ ДАНИХ

Під час переддипломної практики необхідно створити систему дескриптивного аналізу медико-біологічних даних. Перевірити на прикладах результати роботи проекту та впевнитися в їх коректності.

1.1 Призначення та застосування

Найголовнішим призначенням програмної системи є аналіз великих масивів вхідної інформації медико-біологічного направлення [1]. Коректної валідації масиву, повернення та відображення результатів.

Систему можна використовувати як посередника, тобто для перенаправлення вихідних даних системи іншим сервісам для подальшої обробки.

На вхід будемо отримувати Excel файл з розширенням .xlsx або .xls, розмір документа обмежений 128МБ. Після отримання перевірає файл на цілісність, розмір, розширення та наявність в ньому необхідних для аналізу результатів, якщо валідація проходить успішно, то у відповідь отримуємо перелік стовпців з файлу, які дозволені для опрацювання. Користувачу необхідно буде виділити потрібні стовпці та натиснути на відповідну кнопку. І після обробки в зручному вигляді отримати результати, які можна переглядати за посиланням, або завантажити їх в файл з розширенням .xlsx або .csv. Існуватиме декілька режимів роботи системи: для неавторизованих користувачів та авторизованих. Неавторизовані клієнти зможуть завантажувати дані для аналізу, отримувати вихідну інформацію, але не зможуть переглядати історію завантажень та робити результати приватними, на відміну від авторизованих, які матимуть ці можливості.

1.2 Опис підсистем

Кожна підсистема, або сервіс, мають чітко визначену структуру та вимагають конкретні типи даних на вхід [2], для своєї коректної роботи. Кожен сервіс виконує лише ті процеси, які має виконувати і не може впливати на роботу інших підсистем напряму [3], це дозволяє отримати більш точні результати і бути впевненими, що результат попереднього процесу не впливає на наступний.

Всі підсистеми розроблено зі зручним інтерфейсом використання: легкість і безпечне їх розширення, в зв'язку зі змінами бізнес-логіки. Кожен метод аналізу протестований декількома наборами вхідних даних, що дозволяє не хвилюватися, про коректність результатів і мати впевненість, що наступні системи медико-біологічних даних будуть отримувати валідні вхідні значення.

Інтерфейсом є веб-сайт, який зручним способом передає інформацію від клієнта до методів її обробки, він є посередником між користувачем і сервером. Це гарантує надійність і безпеку передачі даних і отримання вихідних даних. Даний підхід не дає можливості людині працювати неправильно з сервісом, адже у випадку не коректної взаємодії користувач отримає повідомлення з детальною помилкою та рекомендаціями, як потрібно вирішити проблему.

Висновки до розділу 1

В даному розділі поставлена задача розробки програмного забезпечення для аналізу медико-біологічних даних: буде створений додаток, який отримуватиме файл від користувача, аналізуватиме його та в зручному вигляді повертатиме результати своєї роботи. Таким чином, найважливішою частиною програмного продукта - правильність аналізу даних та зручна взаємодія користувача з системою.

2 АНАЛІЗ ПРОБЛЕМИ ДЕСКРИПТИВНОГО АНАЛІЗУ

Дескриптивний аналіз - розділ статистики, що виконує систематизацію і обробку даних, описує через базові статистичні показники і подає результати у вигляді таблиць і графіків. При аналізі інформації отримуємо: показники середнього рівня [1] (середнє значення, мода, медіана), відсотки, показники варіації (розмах варіації, міжквартильний розмах, дисперсія, стандартне відхилення, коефіцієнт варіації та ін.), показників форми розподілу (асиметрія та ексцес). При дослідженні даних важливими є точкові та інтервальні оцінки статистичних показників.

2.1 Ефективність досліджень

Ефективність досліджень визначається багатьма умовами: коректністю постановки завдання, вибором потрібних методик, наявністю необхідної технічної бази та експериментального обладнання, підготовлений дослідницький колектив, організація експерименту, обсяг отриманої інформації, якість математичної обробки даних та, нарешті, рівнем аналізу отриманих результатів. Звичайно, чудово, якщо всі вони на високому рівні. Але на практиці трапляється, що недосконалість деяких ланок компенсується вищим рівнем інших. Так, наявність сучасного обладнання може прикрасити нестачу оригінальності при постановці завдання, а величезний обсяг отриманих даних - використання обмеженого спектра використаних методик.

Для нас особливо важливо те, що високий рівень математичної обробки та аналізу даних може згладити практично всі інші недоліки експериментів. Головне, щоб дані були достовірними, а математика дозволить отримати з них максимум інформації. І встановити зв'язки та закономірності і розрахувати прогноз, тобто заглянути за горизонт, і побудувати математичні моделі [4], що компенсують нестачу методів та засобів вимірювання. І, найважливіше, що це все може бути отримано без

істотної витрати матеріальних ресурсів – грошей і часу. Необхідно лише освоїти можливості математичної статистики та вміти працювати з відповідними програмними засобами.

2.2 Аналіз аналогічних існуючих систем

Проаналізувавши існуючі системи статистичної обробки даних, зрозумів, що майже всі схожі програмні продукти - десктопні додатки. Звичайно, що це дуже потужні програми від відомих компаній, таких як IBM. Наприклад, SPSS Statistics - комп'ютерна програма для статистичної обробки даних, один із лідерів ринку в галузі комерційних статистичних продуктів, призначених для проведення прикладних досліджень у суспільних науках. Має можливості введення та зберігання даних, використання змінних різних типів, частотність ознак, таблиці, графіки, таблиці спряженості, діаграми, можливість первинної описової статистики, виконувати маркетингові дослідження [5] та аналізувати дані маркетингових досліджень. Але, маючи багато можливостей, це десктопна програма, розмір якої більше 800 Мб.

Безкоштовних web-сервісів, схожих на SPSS Statistics не знайшов, тому створення саме web-додатку вважаю кращою ідеєю, адже немає необхідності завантажувати і встановлювати на ПК програми, якщо можна виконати аналіз, перейшовши за посиланням. Найголовніше - легкий і зрозумілий інтерфейс для користувачів, тому що статистичні програмні продукти мають, як правило, не інтуїтивний інтерфейс і клієнту важко зрозуміти як правильно користуватися сервісом. Тому цей аспект відлякує велику кількість користувачів.

Не менш важливою проблемою є звіт про аналіз. Іноді результати потрібно надіслати іншій людині чи завантажити в іншу систему. Рішенням цієї проблеми є можливість завантаження звіту в декількох форматах та можливість поділитися посиланням на результати, щоб не завантажувати файли.

Всі проблеми, які були знайдені під час аналізу аналогічних систем, враховані і будуть вирішені в процесі розробки системи.

Висновки до розділу 2

В цьому розділі було проведено дослідження дескриптивного аналізу, ефективність застосування методів обробки даних, проаналізовано аналоги існуючих систем. Отже, головною метою статистичного аналізу є отримання максимуму інформації із вхідного масиву даних. Також необхідно врахувати слабкі сторони аналогів для усунення їх при розробці програмного продукта.

3 ЗАСОБИ РОЗРОБКИ

Будь-яка система розробляється з використанням різних засобів розробки. Для розробки та розгортання програмного продукту необхідно завантажити та встановити їх.

3.1 Мова програмування PHP

Для написання програмної системи було використано мову програмування PHP версії 8.1. PHP - мова програмування з відкритим вихідним кодом широкого використання. В основному використовується для web розробки, але на ній також можна створювати настільні додатки та CLI виконувані файли. PHP має цікаву історію, адже він був створений в 1994 році програмістом Расмусом Лердорфом як шаблонізатор HTML-документів. Лердорф назвав набір інструментів Personal Home Page Tools. Проте дуже швидко став популярним та тепер розшифровується як PHP: Hypertext Preprocessor. Синтаксис PHP подібний до синтаксису мови C. Деякі елементи, такі як асоціативні масиви та цикл foreach, запозичені з Perl. Код, на відміну від мови програмування JavaScript, користувачі побачити не можуть, тому це добре з точки зору безпеки. Користуватися мовою програмування звучніше з використанням фреймворків, яких у PHP не мало. Основні з них: Symfony, Laravel, Yii, CodeIgniter, CakePHP. Саме Laravel був вибраний мною для написання дипломної роботи.

3.2 Фреймворк Laravel

Laravel - фреймворк з відкритим програмним кодом, за допомогою якого створюється серверна частина web-додатків. Він використовує MVC підхід, але за бажанням дуже просто можна писати API-системи. Фреймворк був створений в 2011 році і до цього часу дуже активно розвивається. Laravel має дуже велике ком'юніті, в 2015 році отримав перші місця в 3-х номінаціяціях в результаті опитування сайту sitepoint.com. Фреймворк використовує Eloquent ORM - реалізація шаблону проектування ActiveRecord. Реалізація дозволяє чітко визначити відносини між об'єктами бази даних, які дуже зручно і легко створити в моделі.

3.3 База даних PostgreSQL

Для зберігання інформації було використано PostgreSQL. PostgreSQL - реляційна СКБД [6], яка має відкрите програмне забезпечення, написана на мові програмування C. Дану базу даних використовують в високонавантажених системах, які потребують надійних механізмів транзакцій та реплікацій. До позитивного можна віднести те, що СКБД дуже легко інтегрується з PHP, має багато типів, що дозволяє працювати з різними типами даних, в тому числі і з JSON. PostgreSQL, як і інші реляційні бази даних, має функції, тригери, індекси, дозволяє успадковуватись та можливість партикування і реплікації.

3.4 Автоматизації розгортання системи

Для автоматизації розгортання системи було використано Docker. Це програмне забезпечення для автоматизації розгортання [7] та управління програмами в середовищах з підтримкою контейнеризації додатків. Docker працює як віртуальна

машина, але кожний завантажений контейнер працює окремо, ізольовано від інших. Контейнер - один процес, наприклад як база даних. Інші процеси не можуть щось змінювати в БД, якщо на це не буде доступно зовнішнього порту з контейнера бази даних. Дуже зручно використовувати саме Docker, адже щоб запустити програмний продукт, яким описаний за допомогою Dockerfile або docker-compose файлу необхідно встановити лише Docker і виконати одну чи декілька команд.

3.5 Додаткові засоби розробки

HTML – стандартизована мова розмітки документів для перегляду веб-сторінок у браузері.

CSS – це код, який використовується для стилізації веб-сторінки.

Розробка програмного коду виконується в IDE PhpStorm. Компанія JetBrains на базі платформи IntelliJ IDEA створює IDE для різним мов програмування, особливістю їх продуктів є зручність і легкість використання, велика кількість розширень і постійні оновлення. Проста інтеграція PhpStorm з PHP, PostgreSQL та Docker.

Висновки до розділу 3

У даному розділі були обрані засоби для розробки програмного продукта, засоби розгортання, база даних для зберігання результатів. Мова програмування - PHP, написання коду відбуватиметься в середовищі PhpStorm. Контеризація сервісу виконуватиметься за допомогою Docker.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Метою дипломною роботи є створення програмної системи, яка за допомогою зрозумілого і простого користувацького інтерфейсу надасть можливість клієнту користуватися всіма її можливостями.

До складу архітектури програмного продукту для дескриптивного аналізу входять такі основні компоненти:

– сервер бази даних PostgreSQL для зберігання інформації доступу до неї;

– сервер веб-додатку;

– веб-клієнт для користувача;

– веб-клієнт для адміністратора.

4.1 Архітектура веб-платформи

Програмний продукт має модель взаємодії “трирівневої архітектуру”. Цей підхід дозволяє розділяти функціонал між клієнтською, серверною частиною і сховищем.

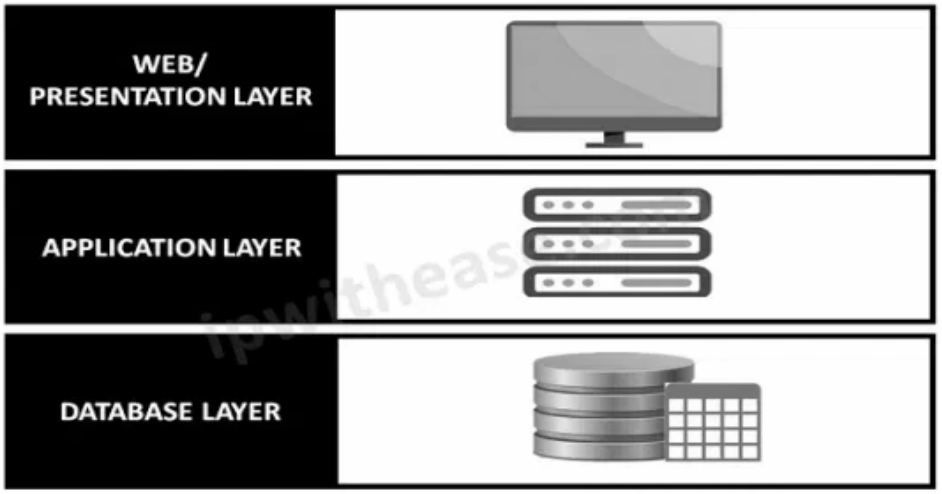


Рисунок 4.1 – Схема взаємодії програмного продукта

Клієнтський шар – інтерфейс користувача. Це може бути веб-браузер, якому надсилаються HTML-сторінки, або графічна програма. Цей шар повинен мати можливість робити запити до шару логіки. В нашому додатку використовуються технології HTML/CSS та Bootstrap 4.

Шар логіки – сервер, на якому відбувається обробка запитів та відповідей. Часто його називають серверним шаром. Також тут відбуваються всі логічні операції: математичні розрахунки, операції з даними, звернення до інших сервісів або сховищ даних. В дипломній роботі за шар логіки відповідає web-сервер Nginx та мова програмування PHP.

Шар даних – сервер баз даних: до нього звертається наш сервер. У цьому шарі зберігається вся необхідна інформація, якою користується програма під час роботи.

Архітектура має багато плюсів. По-перше, розмежування даних, щоб регулювати доступ до них, по-друге - можливість масштабованості та захисну від SQL-ін'єкцій.

4.2 База даних програмної платформи

База даних - організована структура, яку використовують для додавання, зберігання, зміни та вилучення інформації. Для дипломної роботи використано реляційну базу даних PostgreSQL. БД зберігає записи в таблицях, які зв’язані між собою.

Для системи було спроектовано базу даних. Вона містить 9 таблиць, які мають відношення між собою. Структура бази даних зображена на рисунку 4.2.

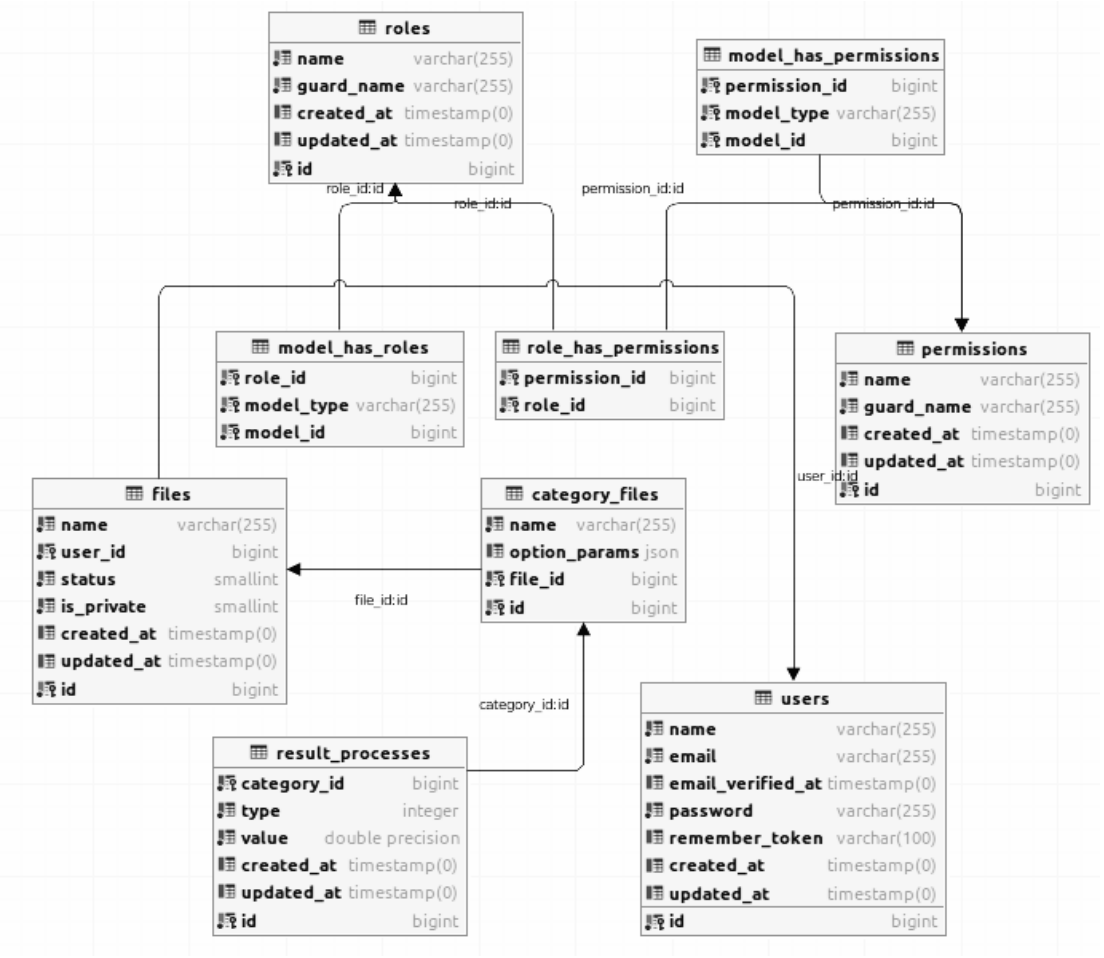


Рисунок 4.2 – Структура бази даних

Розглянемо деякі таблицю окремо, щоб описати існуючі поля, також призначення таблиці. Таблиця **“**users**”** містить інформацію про користувачів системи, активній сесії авторизації та складається з:

– `id` - первинний ключ;

– `name` - ім’я користувача;

– `email` - електронний адрес;

– `password` - пароль клієнта;

– `remember_token` - токен авторизації;

– `created_at` - дата та час реєстрації;

– `updated_at` - дата та час авторизації.

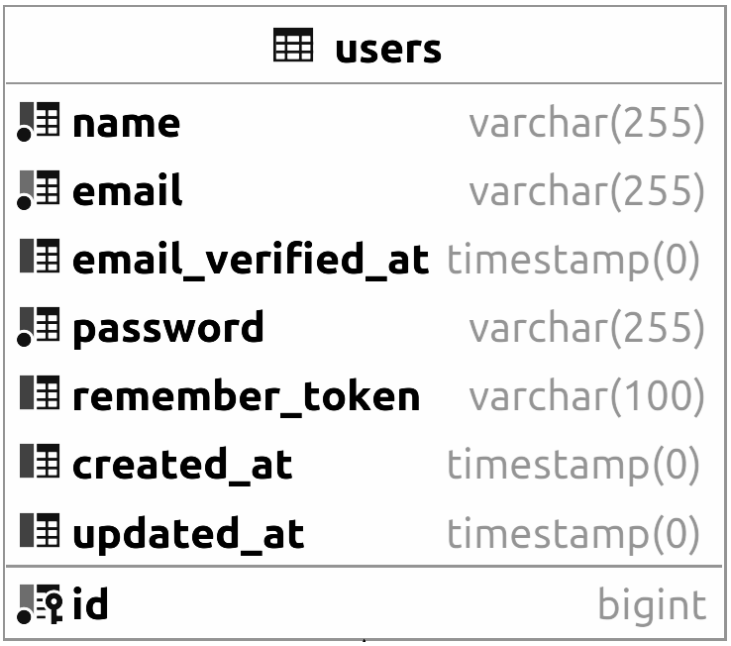


Рисунок 4.3 – Структура таблиці **“**users**”**

Наступна важлива таблиця - **“**files**”**. В ній зберігаються вся інформація про файли, які були завантажені в систему. Структура зображена на рисунку 4.4 і має такі поля:

– `id` - первинний ключ;

– `name` - назва завантаженого файлу;

– `user_id` - ідентифікатор користувача, який завантажив файл;

– `status` - статус файлу(завантажений, опрацьований чи не валідний);

– `is_private` - статус приватності сторінки;

– `created_at` - дата та час завантаження файла;

– `updated_at` - дата та час зміни статуса чи аналізу.

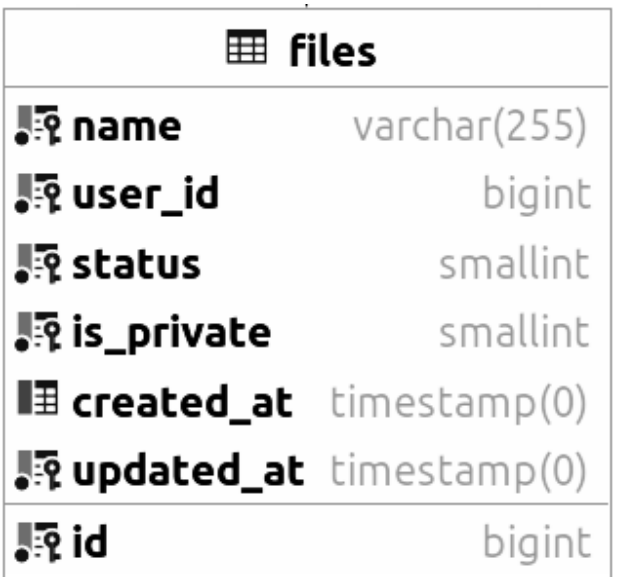


Рисунок 4.4 – Структура таблиці “files”

Таблиця “category_files” містить стовпці завантажених файлів. Необхідна для сортування результатів аналізу та генерування звітів. Структура зображена на рисунку 4.5 і має такі поля:

– `id` - первинний ключ;

– `name` - заголовок стовпця;

– `options_params` - додаткові необхідні параметри для аналізу;

– `file_id` - ідентифікатор завантаженого файлу.

Рисунок 4.5 – Структура таблиці “category_files”

Наступна важлива таблиця - “result_processes”. В ній зберігаються всі результати обробки. Структура зображена на рисунку 4.6 і має такі поля:

– `id` - первинний ключ;

– `type` - тип аналізу;

– `value` - результат аналізу;

– `created_at` - дата та час початку аналізу;

– `updated_at` - дата та час завершення аналізу.

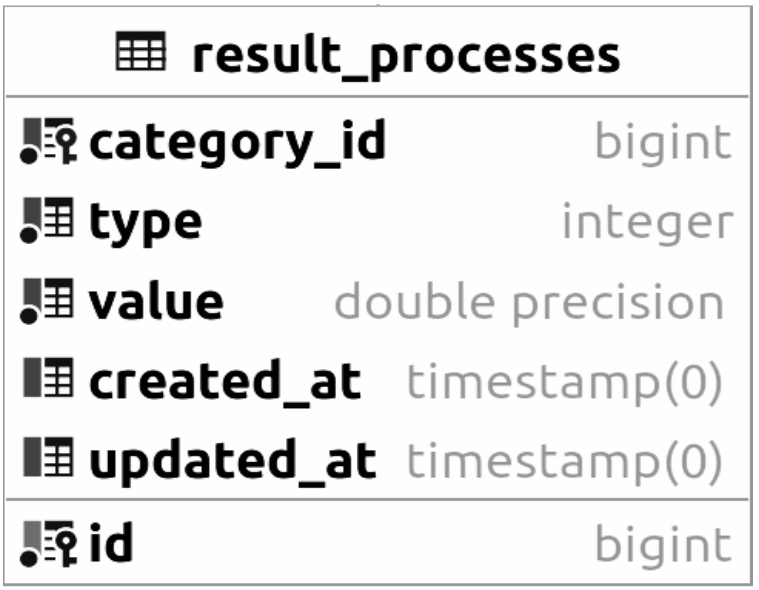


Рисунок 4.6 – Структура таблиці “result_processes”

Таблиця “role” необхідна для ролей системи: адміністратор, користувач. Структура зображена на рисунку 4.7 і має такі поля:

– `id` - первинний ключ;

– `name` - назва ролі;

– `guard_name` - аліас для використання в коді;

– `created_at` - дата та час створення ролі;

– `updated_at` - дата та час оновлення ролі.

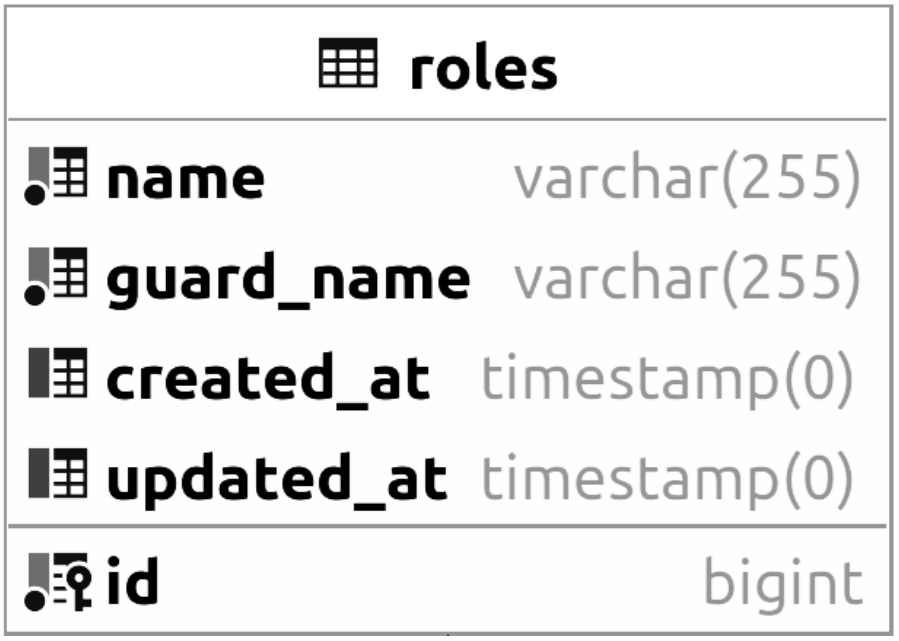


Рисунок 4.7 – Структура таблиці “roles”

Структуру інших таблиць і зв'язки можна подивитися на рисунку 4.2.

В результаті роботи було створено архітектуру бази даних: таблиці, зв’язки, проставлено потрібні індекси для оптимізації роботи системи, перевірено використання індексів через вбудовані можливості PostgreSQL.

4.3 Діаграма прецедентів системи

Програмна система має три категорії користувачів:

– адміністратор системи;

– незареєстрований користувач;

– зареєстрований користувач.

Діаграма прецедентів, яка зображена на рисунку 4.8, відображає сценарії взаємодії між акторами та випадками взаємодії(прецедентами). Діаграма показує можливості кожного актора, в ролі яких є: адміністратор, незареєстрований та зареєстрований користувач.

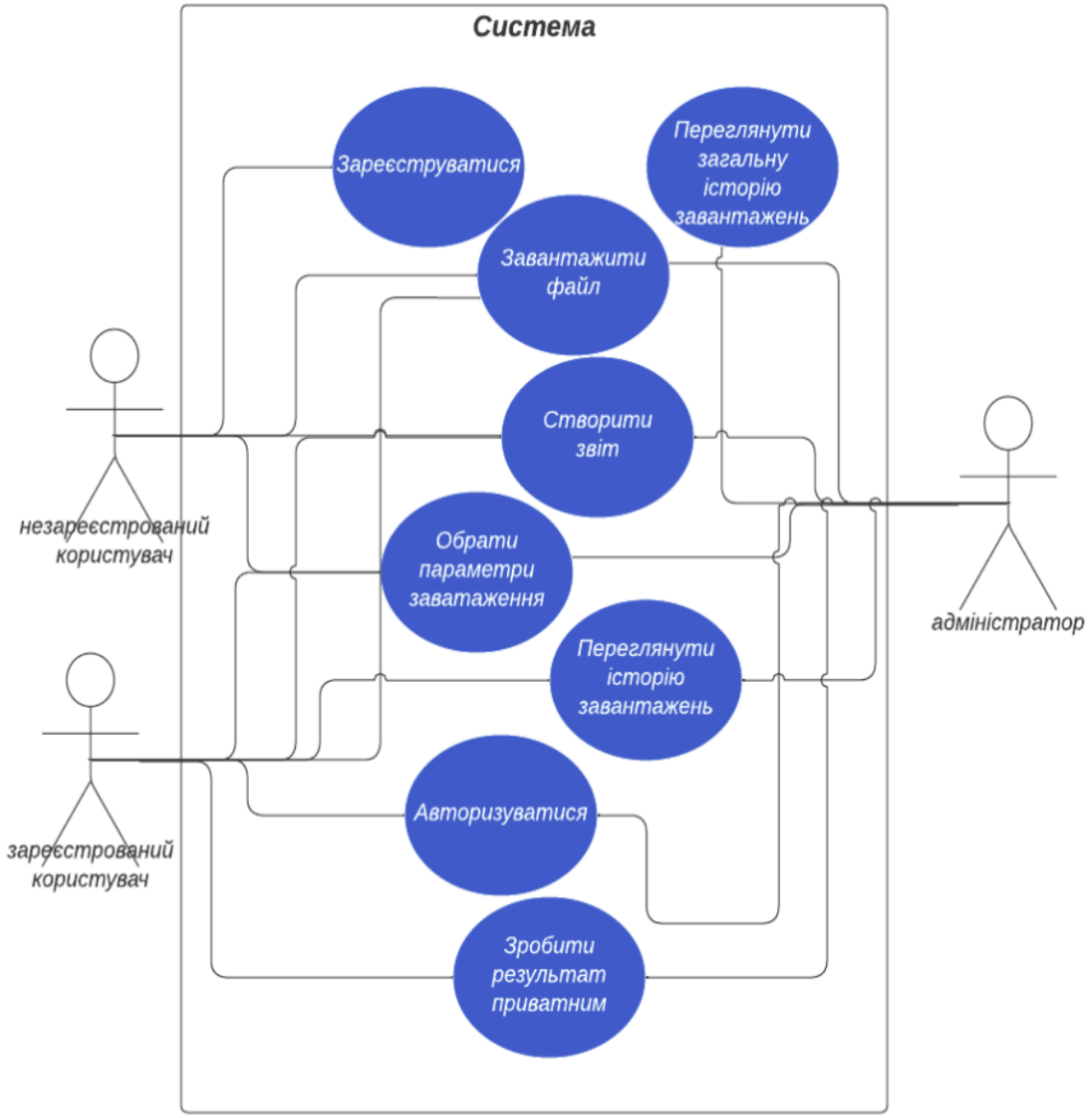


Рисунок 4.8 – Діаграма прецедентів

Опишемо можливість кожної ролі.

Адміністратор може: авторизуватися, завантажити файл для аналізу, вибрати параметри завантаження, створити звіт, зробити результат приватним, переглядати свою та загальну історію завантажень.

Незареєстрований користувач може: зареєструватися, завантажити файл для аналізу, вибрати параметри завантаження, створити звіт.

Зареєстрований користувач може: авторизуватися, завантажити файл для аналізу, вибрати параметри завантаження, створити звіт, зробити результат приватним, переглядати свою історію завантажень.

4.4 Алгоритми аналізу

Для аналізу вхідного масиву даних необхідно розрахувати загальні характеристики. Під масивом даних будемо розуміти сукупність значень, що відносяться до одного і того ж показника, але для різних досліджень.

Основною і важливою характеристикою для будь-якого масива $\{x_i\}$ розміру n є середнє арифметичне значення, яке обчислюється за формулою:

$$\bar{x}_{\text{арифм}} = M = \bar{x} = \left(\sum_{i=1}^{n} x_i\right)/n, \tag{4.1}$$

де x_i – i-тий елемент масива;

n – розмір масива.

Середнє геометричне значення більш точно характеризує ряд динаміки, ніж середнє арифметичне. Його часто використовують для усереднення відносних величин і розраховується за формулою:

$$\bar{x}_{\text{геом}} = \sqrt[n]{x_1 \times \cdots \times x_n} = \sqrt[n]{\prod_{i=1}^{n} x_i}, \tag{4.2}$$

де x_i – i-тий елемент масива;

n – розмір масива.

Середню квадратичну величину (4.3) часто використовують для усереднення за модулем змінних, що мають різний знак, для розрахунку відстаней у багатовимірному просторі, усереднення варіабельності та для багатьох інших цілей.

Це дуже популярна у математичній статистиці характеристика та обчислюється за формулою:

$$\bar{x}_{\text{квадр}} = \sqrt{\frac{x_1^2 + \cdots + x_n^2}{n}} = \sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}, \tag{4.3}$$

де x_i – *i*-тий елемент масива;

n – розмір масива.

Для усереднення об’ємів використовуються середню кубічну величину (4.4)

$$\bar{x}_{\text{куб}} = \sqrt[3]{\frac{\sum_{i=1}^{n} x_i^3}{n}}, \tag{4.4}$$

де x_i – *i*-тий елемент масива;

n – розмір масива.

У більш загальному випадку, для будь-якого ступеня k степенева середня буде розраховуватися за формулою:

$$\bar{x}_{\text{степ}} = \sqrt[k]{\frac{\sum_{i=1}^{n} x_i^k}{n}}, \tag{4.5}$$

де x_i – *i*-тий елемент масива;

n – розмір масива.

Середню гармонійну доцільно використовувати тоді, коли результати спостережень виявляють зворотну залежність, задані оберненими значеннями змінних, наприклад, частками їх знаходження в загальній множині або відсотками. Ця середня часто використовується при розрахунку середньої тривалості життя та середньої швидкості (середнього зростання) та обчислюється за формулою 4.6.

$$\bar{x}_{\text{гарм}} = \frac{n}{\sum_{i=1}^{n}\left(\frac{1}{x_i}\right)}, \tag{4.6}$$

де x_i – i-тий елемент масива;

n – розмір масива.

Найбільш простою параметричною оцінкою варіабельності, яка не залежить від знака відхилення, є середнє лінійне відхилення, формула якого має вигляд:

$$\bar{d} = \frac{1}{n}\sum_{i=1}^{n}\left|x_i - \bar{x}\right|, \tag{4.7}$$

де x_i – i-тий елемент масива;

$\bar{x}$ – середнє арифметичне значення;

n – розмір масива.

Більш популярною є характеристика, яка використовує усереднення не модулів, а квадратів таких відхилень та називається дисперсією:

$$D = \frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2, \tag{4.8}$$

де n – розмір масива;

x_i – i-тий елемент масива;

$\bar{x}$ – середнє арифметичне значення.

Отримаємо з дисперсії квадратний корінь і отримаємо ще одну найважливішу характеристику варіабельності - середньоквадратичне відхилення або просто «сигму», яка обчислюється за формулою 4.8.

$$\sigma = \sqrt{D} = \sqrt{\frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}{(n-1)}}, \quad (4.8)$$

де x_i – i-тий елемент масива;

$\bar{x}$ – середнє арифметичне значення;

n – розмір масива.

Ще однією дуже важливою статистичною характеристикою вибірки, поряд із середньою арифметичною, дисперсією та середньоквадратичним відхиленням, є помилка середньої арифметичної:

$$S_x = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n}}{n(n-1)}}, \quad (4.9)$$

де x_i – i-тий елемент масива;

$\bar{x}$ – середнє арифметичне значення;

n – розмір масива.

Якщо аналізується кілька показників, що мають різні одиниці виміру, проводити зіставлення їх варіабельності неможливо. Для цього доцільно використовувати неіменовані («безрозмірні») характеристики, рівні відношенню показників варіабельності до середньої арифметичної – коефіцієнт осциляції, який можна отримати за допомогою формули.

$$V_R = \frac{R}{\bar{x}} * 100\%, \quad (4.10)$$

де R – розмах вибірки;

$\bar{x}$ – середнє арифметичне значення.

Лінійний коефіцієнт варіації можна розрахувати за формулою:

$$V_d = \frac{\bar{d}}{\bar{x}} * 100\%, \quad (4.11)$$

де $\bar{d}$ – середнє лінійне відхилення;

$\bar{x}$ – середнє арифметичне значення.

Найчастіше використовують коефіцієнт варіації, в який входить середньоквадратичне відхилення:

$$V = \frac{\sigma}{\bar{x}} * 100\%, \quad (4.12)$$

де σ – середньоквадратичне відхилення;

$\bar{x}$ – середнє арифметичне значення.

Іноді в якості варіабельності використовують показник точності оцінки параметрів, який вираховується за формулою:

$$q = \frac{S_{\bar{x}}}{\bar{x}} = \frac{V}{\sqrt{n}}, \quad (4.13)$$

де V – значення варіабельності;

n – розмір масива;

$S_{\bar{x}}$ – помилка середньої арифметичної;

$\bar{x}$ – середнє арифметичне значення.

Нормоване відхилення для кожного елемента можна розрахувати за допомогою формули:

$$g_i = \frac{x_i - \bar{x}}{\sigma}, \quad (4.14)$$

де σ – середньоквадратичне відхилення;

x_i – i-тий елемент масива;

$\bar{x}$ – середнє арифметичне значення.

Описані основні методи обробки вхідних даних. Дескриптивний аналіз складається з великої кількості методів, кожен з яких повертає числове значення, яке характеризує вибірку даних.

Висновки до розділу 4

Отже, в цьому розділі описано архітектуру веб-платформи, структуру бази даних, створено діаграму прецедентів, описано формули методів, які необхідні для статистичного аналізу, і які дають загальну характеристику вхідному масиву інформації.

5 РОБОТА КОРИСТУВАЧА З ПРОГРАМНОЮ СИСТЕМОЮ

Будь-яку програмну систему необхідно завантажити та налаштувати перед тим, як почати нею користуватися. Деякі програми вимагають довгого і важкого розгортування, інші ж можна встановити за декілька хвилин, тому це залежить від типу програмної системи, на якій мові написана, від БД і інших модулів.

5.1 Встановлення та налагодження програмного забезпечення

Для запуску програми необхідно встановити лише Docker. Всі необхідні залежності і частини додатку будуть автоматично завантажені після виконання команди “docker-compose up”: буде створена схема бази даних, веб сервер працюватиме на 8080 порті і доступ до системи буде доступним за посиланням http://localhost:8080. Якщо ж порт занятий іншим процесом, то необхідно замінити його в файлі docker-compose.yml.

5.2 Взаємодія користувача з системою

Після завантаження головної сторінки, користувач бачить основні можливості сервісу: посилання на авторизацію, допомогу та можливість взаємодії анонімно: завантаження файлу. На рисунку 5.1 зображено головну сторінку сервісу.

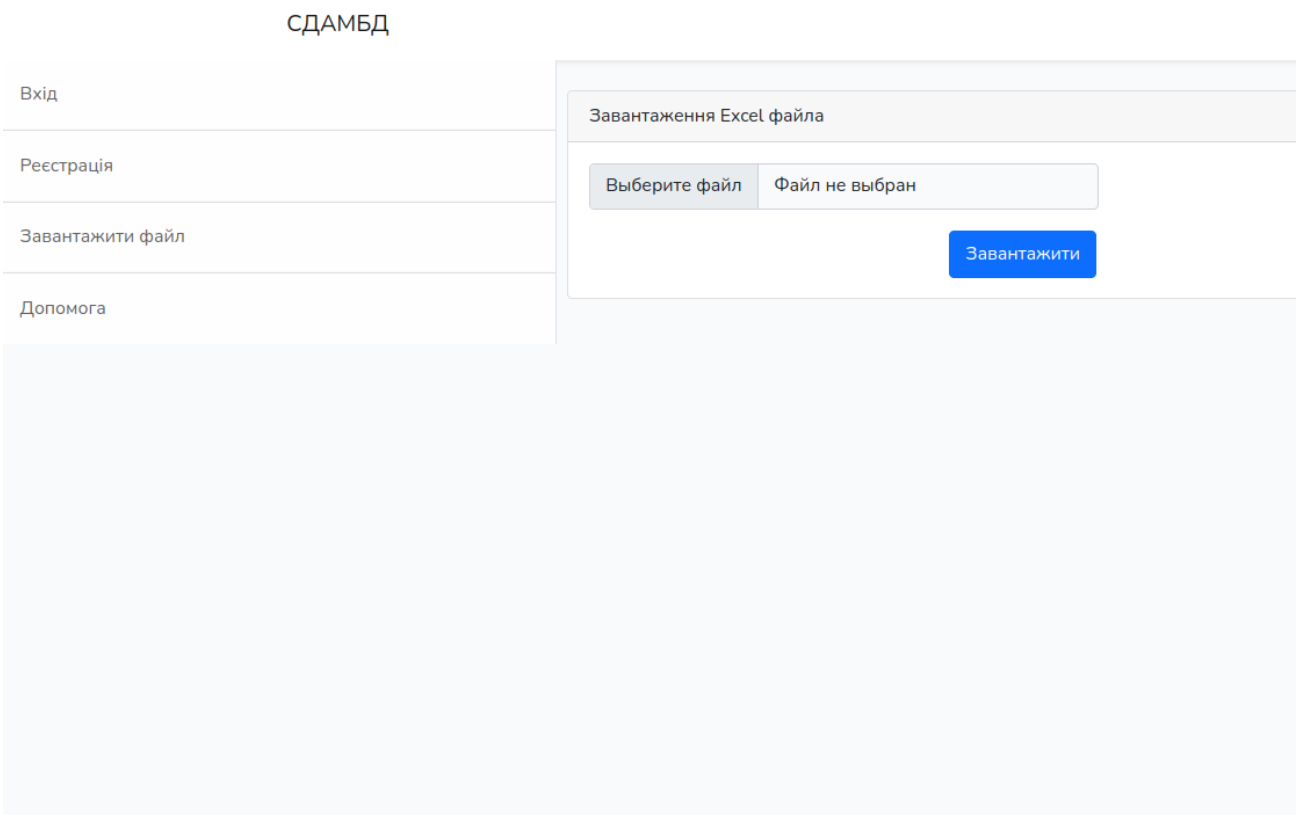


Рисунок 5.1 – Головна сторінка сервісу

Далі користувач може або авторизуватися, або виконувати аналіз без реєстрації. Різниця лише в тому, що якщо користувач виконує обробку файлу авторизованим, то зберігається історія завантажень і результати опрацювання можна зробити приватними, щоб у інших користувачів не було можливості переглядати сторінку. Перейдемо на сторінку реєстрації. Для створення облікового запису необхідно заповнити всі поля, які представлено на рисунку 5.2.

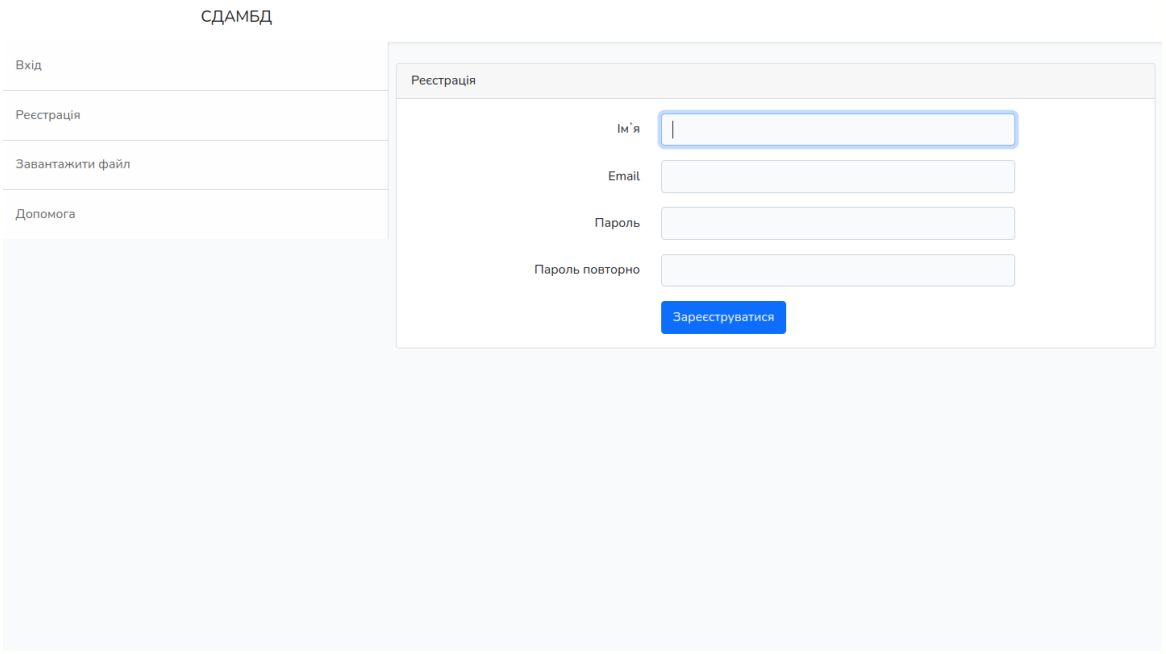


Рисунок 5.2 – Сторінка створення облікового запису

Ми отримаємо обліковий запис, в якому зберігається історія та результати аналізу.

5.3 Огляд результатів

Після авторизації оберемо файл, який хочемо завантажити і тиснемо кнопку “Завантажити”. Програмне забезпечення прочитає файл, знайде назви стовпців з масивами даних і запропонує обрати необхідні стовпці, вказати точність обрахувань, візуалізація на рисунку 5.3. Якщо в файлі не буде потрібної інформації або файл буде іншого розширення, ніж .xlsx чи .xls, то користувач отримає повідомлення з помилкою.

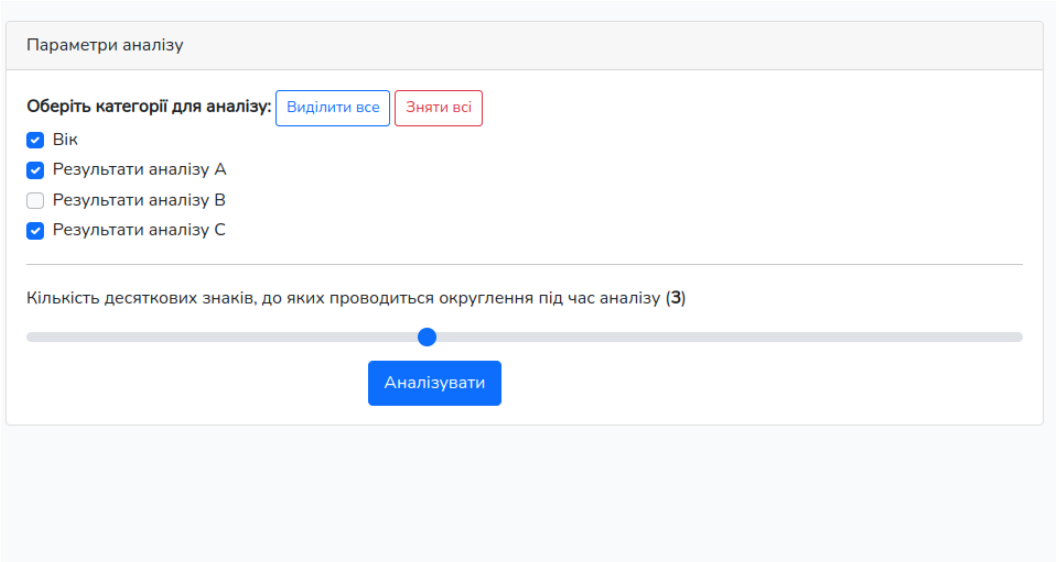


Рисунок 5.3 – Вікно параметрів для початку аналізу

Після конфігурації параметрів натиснемо кнопку “Аналізувати”. Деякий час необхідно серверу, щоб виконати обробку даних. Через деякий час перейдемо на сторінку з результатами аналізу, як продемонстровано на рисунку 5.4.

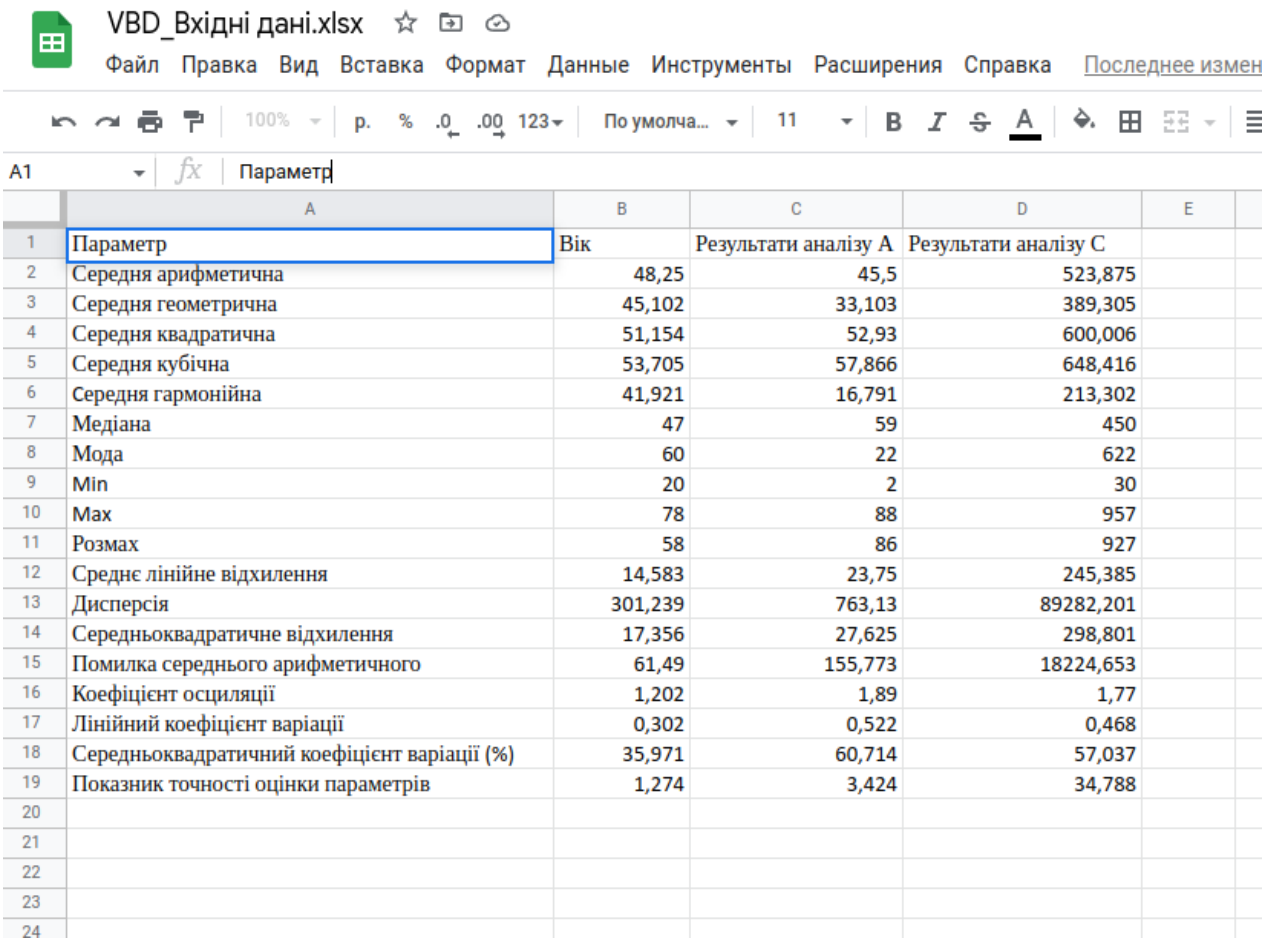
VBD_Вхідні дані.xlsx 06:05 20-05-2022

Приватний.результат http://sntk.tk/result/26 CSV Excel

Параметр	Вік	Результати аналізу A	Результати аналізу C
Середня арифметична	48.25	45.5	523.875
Середня геометрична	45.102	33.103	389.305
Середня квадратична	51.154	52.93	600.006
Середня кубічна	53.705	57.866	648.416
Середня гармонійна	41.921	16.791	213.302
Медіана	47	59	450
Мода	60	22	622
Min	20	2	30
Max	78	88	957
Розмах	58	86	927
Середнє лінійне відхилення	14.583	23.75	245.385
Дисперсія	301.239	763.13	89282.201
Середньоквадратичне відхилення	17.356	27.625	298.801
Помилка середнього арифметичного	61.49	155.773	18224.653
Коефіцієнт осциляції	1.202	1.89	1.77

Рисунок 5.4 – Результати дескриптивного аналізу

На сторінці можна детально ознайомитися з результатами, при необхідності скопіювати посилання, щоб поділитися. Також є можливість зробити сторінку приватною, щоб вихідні дані бачив лише той користувач, що завантажив, для цього потрібно натиснути на чекбокс “Приватний результат”. При необхідності можна завантажити результат з .csv та .xlsx розширеннями. Результат зображено на рисунку 5.5.



	A	B	C	D	E
1	Параметр	Вік	Результати аналізу A	Результати аналізу C	
2	Середня арифметична	48,25	45,5	523,875	
3	Середня геометрична	45,102	33,103	389,305	
4	Середня квадратична	51,154	52,93	600,006	
5	Середня кубічна	53,705	57,866	648,416	
6	Середня гармонійна	41,921	16,791	213,302	
7	Медіана	47	59	450	
8	Мода	60	22	622	
9	Min	20	2	30	
10	Max	78	88	957	
11	Розмах	58	86	927	
12	Середнє лінійне відхилення	14,583	23,75	245,385	
13	Дисперсія	301,239	763,13	89282,201	
14	Середньоквадратичне відхилення	17,356	27,625	298,801	
15	Помилка середнього арифметичного	61,49	155,773	18224,653	
16	Коефіцієнт осциляції	1,202	1,89	1,77	
17	Лінійний коефіцієнт варіації	0,302	0,522	0,468	
18	Середньоквадратичний коефіцієнт варіації (%)	35,971	60,714	57,037	
19	Показник точності оцінки параметрів	1,274	3,424	34,788	
20					
21					
22					
23					
24					

Рисунок 5.5 – Завантажений звіт у форматі .xlsx

Для інших користувачів, крім автора, сторінка має вигляд, як зображено на рисунку 5.6:

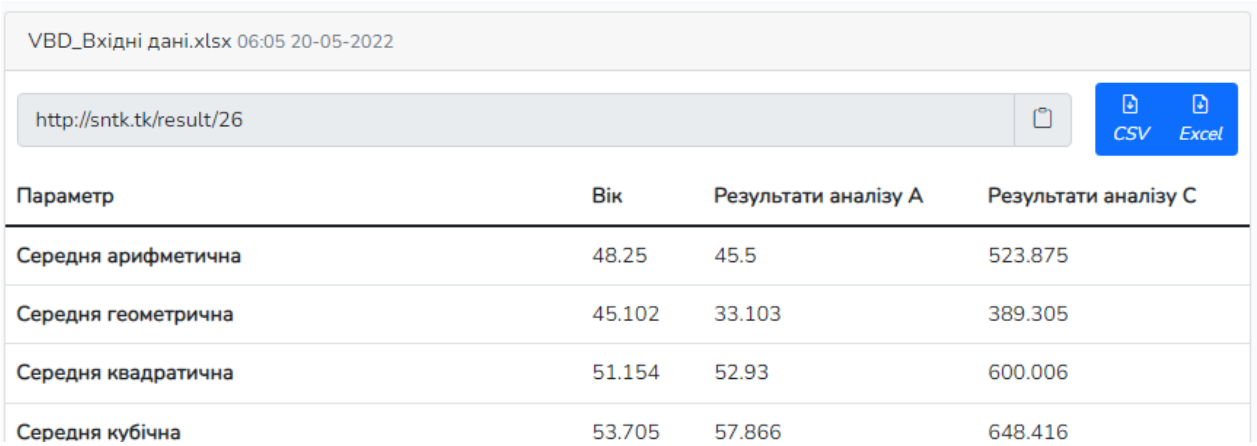


Рисунок 5.6 – Сторінка результатів для публічного перегляду

Користувач, який завантажив файл може зробити результати приватними, для цього необхідно виділити на відповідний чекбокс, який представлений на рисуноку 5.7:

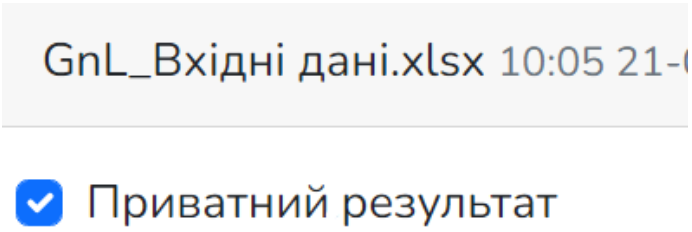


Рисунок 5.7 – Можливість заборонити загальний доступ до результатів

Якщо ж сторонній користувач захоче перейти за посиланням, щоб переглянути приватні результати, то отримає повідомлення з помилкою, яка зображена на рисунку 5.8.

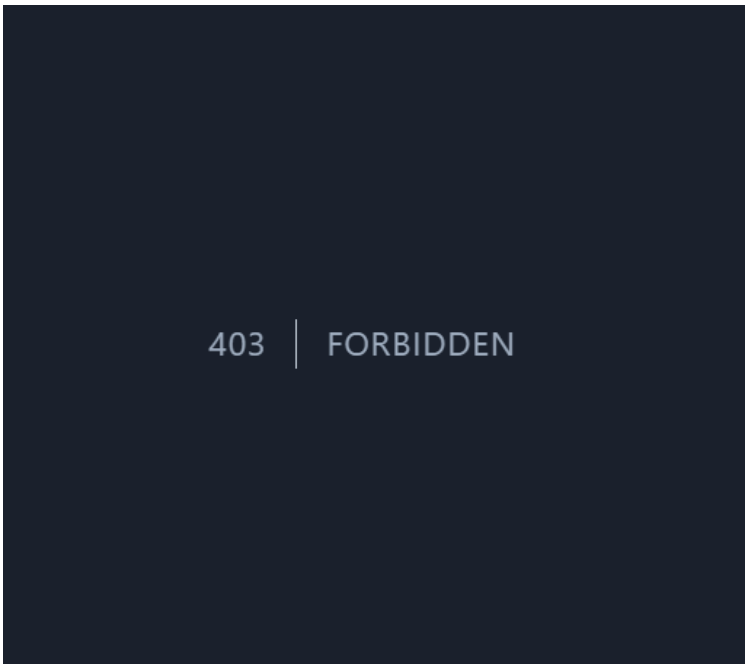


Рисунок 5.8 – Помилка доступу за заборонених результатів

Авторизований користувач може переглядати історію завантажень, для цього необхідно перейти на відповідну сторінку “Історія”. Там можна переглянути доступні результати, дату аналізу, назву файлу. Приклад історії на рисунку 5.9.

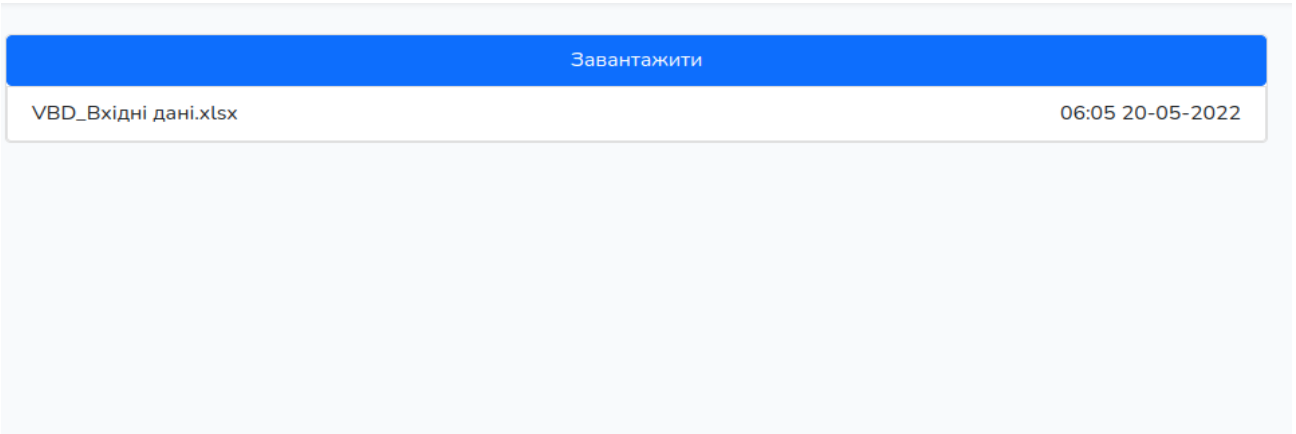


Рисунок 5.9 – Сторінка історії завантажень

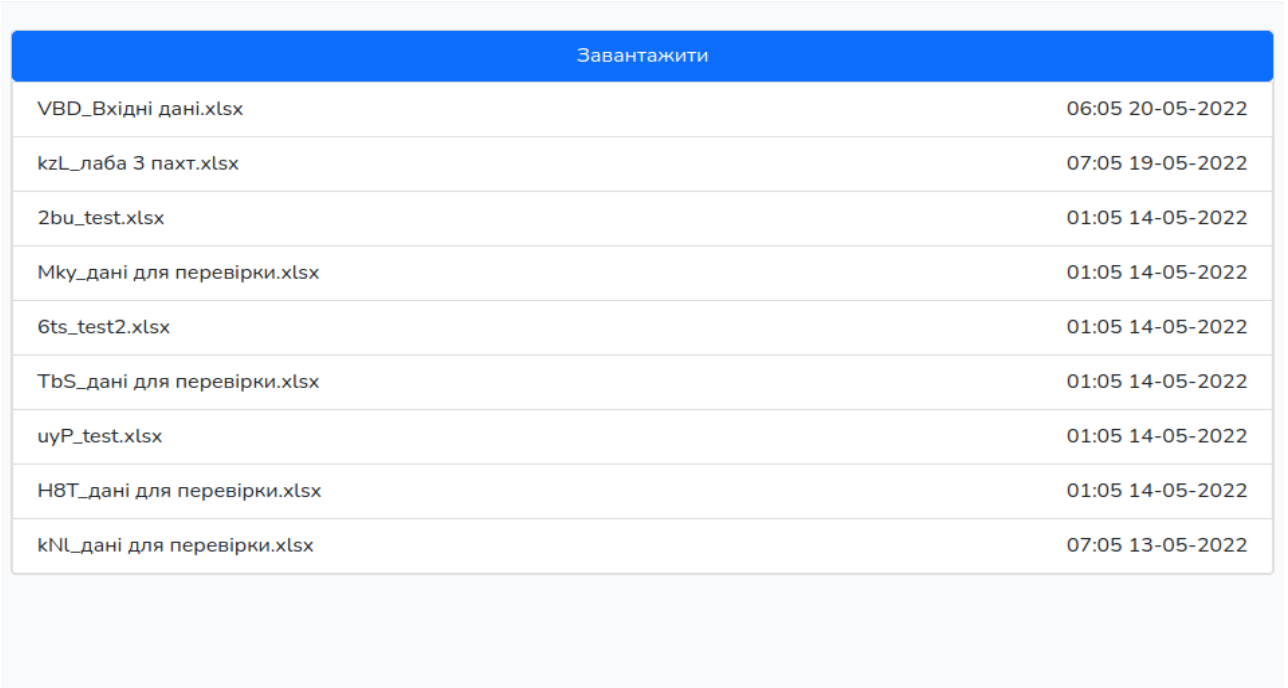


Рисунок 5.10 – Сторінка історії адміністратора

Існує роль адміністратора, який може переглядати всі завантажені файли На рисунку 5.10 представлено історію адміністратора.

Щоб отримати інформацію про формули, які використовуються при аналізі та іншу додаткову інформацію необхідно перейти на сторінку допомоги. На рисунку 5.11 приклад формули середнього квадратичного значення.

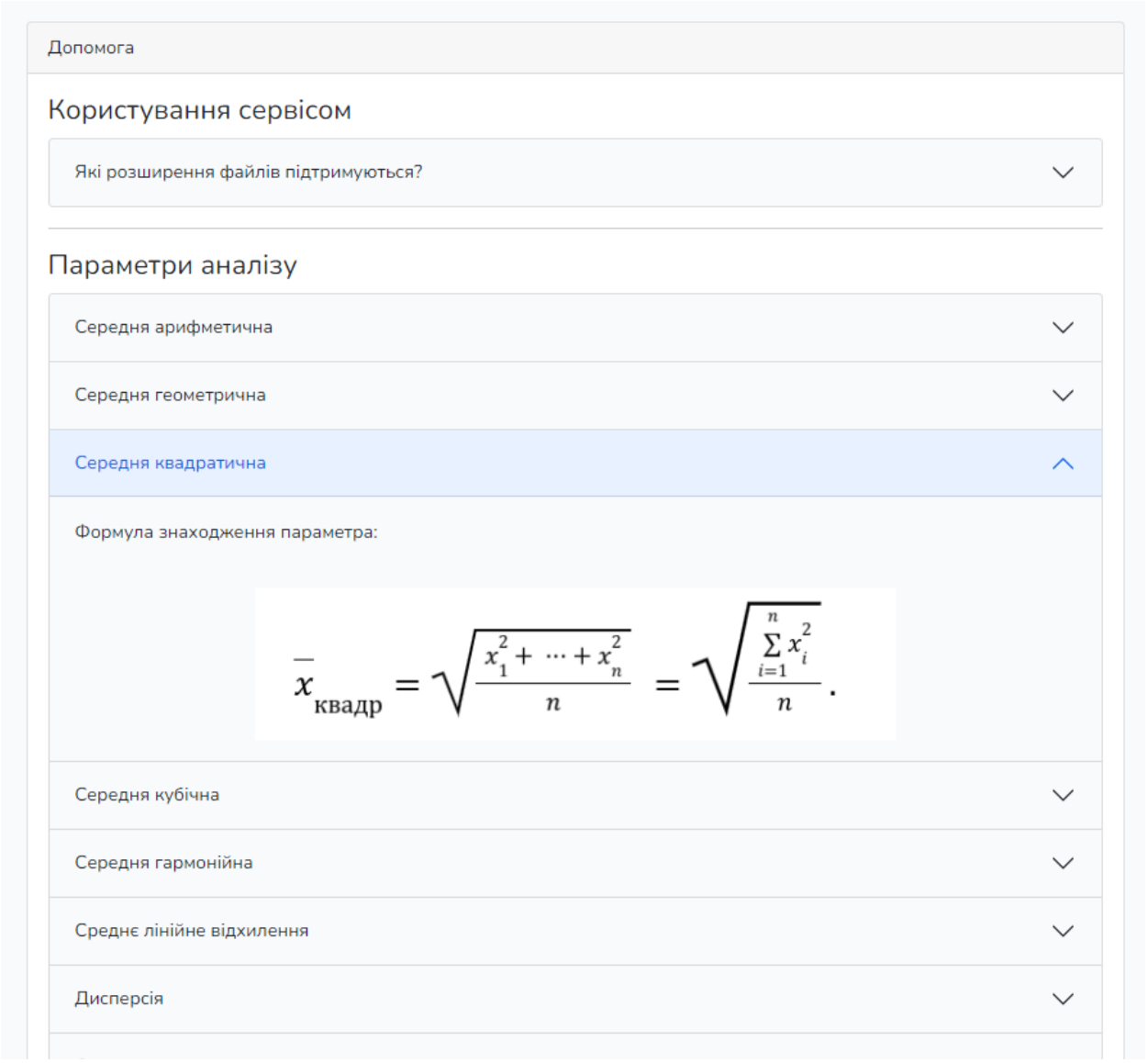


Рисунок 5.11 – Сторінка допомоги

В ситуації, коли користувач завантажив файл з розширенням, яке не підтримується програмною системою, то він отримує повідомлення з помилкою, яке відображено на рисунку 5.12.

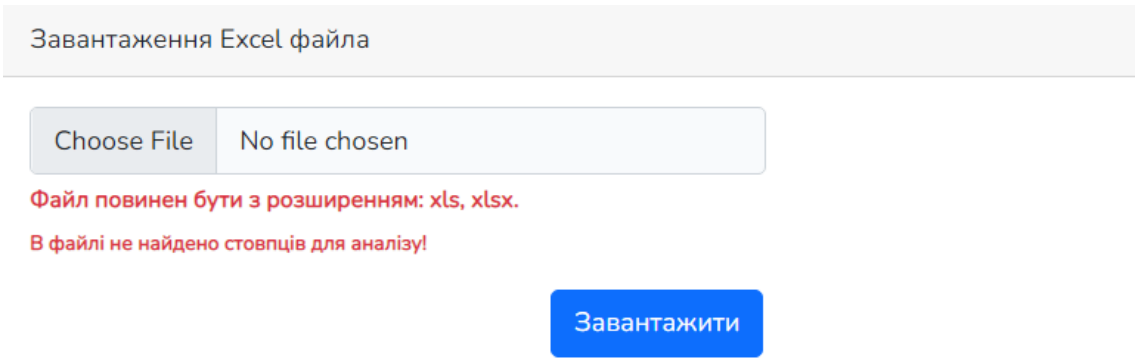


Рисунок 5.12 – Помилка завантаження файла

Якщо завантажено правильний файл і він має числові стовпці, то продовжиться робота програми.

Висновки до розділу 5

Отже, робота користувача з програмною системою є лінійною та зрозумілою. Одразу після завантаження головної сторінки можна обрати та завантажити файл для аналізу, не потрібно обов'язково авторизовуватися чи виконувати інші додаткові дії. Отримавши результати, користувач сам вирішує чи потрібно йому завантажити звіт, або поділитися посиланням на сторінку.

ВИСНОВКИ

У результаті виконання роботи досліджено підходи для обробки медико-біологічний даних, проаналізовано проблеми дескриптивного аналізу. Ознайомився з відповідною літературою, науковими статтями, основними і необхідними для обробки даних характеристик (статистик і параметрів), проаналізував актуальність проблеми, способи її вирішення, обрав потрібні інструменти для розробки програмного коду.

Запропоновано варіанти вирішення цих проблем: програмна система для описового аналізу даних із простим інтерфейсом і можливістю завантаження великих масивів інформації.

Створено систему, яка вхідний масив інформації перетворює в точкову або інтервальну оцінку для подальших аналізів. Вихідні результати програми перевірені великою кількість тестів і мають можливість подальшого застосування.

В ході виконання роботи ширше ознайомився з існуючими проблемами в медико-біологічній сфері. Також вдосконалив навички програмування, створення систем, розробки алгоритмів та їх оптимізації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Антомонов М.Ю. Математическая обработка и анализ медико-биологических данных: Киев, 2-е изд. Киев, 2017. 560 с.
2. Head First Design Patterns: A Brain-Friendly Guide 1st Edition / Eric Freeman, Bert Bates, Kathy Sierra, Elisabeth Robson : Wiley, 2020. 672 с.
3. Robert C. Martin. Clean Code: A Handbook of Agile Software Craftsmanshi : California, 2019. 488 с.
4. Бородкіна І. Книга Інженерія програмного забезпечення. Посібник для студентів вищих навчальних закладів : Київ, 2018. 209 с.
5. Martin Kleppmann Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems: O'Reilly Media, 2017. 614 с.
6. СКБД SQLite: URL: https://uk.wikipedia.org/wiki/SQLite (Дата звернення: 14.03.2022)
7. Godbole S. Discriminative methods formulti—labeled classification, Sarawagi, 2020. — P. 14-21.
8. Read J. Classifier chains for multi-label classification Department of Computer Science, 2019. — P. 2-12.
9. Scott Chacon, Ben Straub, Pro Git : Apress; 2nd ed. edition 2018. 440 p.
10. Brendan Burns Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services 1st Edition, 2018. 188 с.
11. Qureshi M. EVE: explainable vector based embedding technique — Journal of Intelligent Information Systems, 2018. — P. 17–35.
12. Sak H. Long Short-Term Memory recurrent neural network architectures for large scale acoustic modeling / H. Sak, A. Senior, F. — Beaufays, 2018. — P. 14.
13. Малышев К. Построение пользовательских интерфейсов: Минск, 2021. 268 с.
14. CREO VB API documentation: Parametric Technology Corporation, Needham, USA: 2011, 506 p
15. Tolles J. Logistic Regression Relating Patient Characteristics to Outcomes, JAMA, 2016. — с. 353

16. Cooper A. Interface. Fundamentals of interaction design : John Wiley & Sons, 2021. 720 с.
16. Hamming R. Error detecting and error correcting codes — Bell System Technical Journal, 1950. — P. 93-100.
17. Berger J. 2.4.2 Certain Standard Loss Functions. Statistical Decision Theory and Bayesian Analysis (2nd ed.) — New York: Springer-Verlag, 1985. —55 p.
18. Alan Dennis Systems Analysis and Design: An Object-Oriented Approach with UML, 5th Edition : Wiley, 2020. 256 с.
19. ДСТУ 2226-93 Автоматизовані системи. Терміни і визначення. [Чинний від 1994-07-01]. Вид. офіц. Київ: Держспоживстандарт України, 1993. 92 с.
20. Капінос Г.І., Бабій І.В. Операційний менеджмент: навч. посіб. Київ, 2013. 352 с

ДОДАТОК А

Програмна система дескриптивного аналізу медико-біологічних даних

Текст програмного модулю

УКР.НТУУ“КПІ ім. Ігоря Сікорського”.ТР81343_22Б 12-1

Аркушів 10

2022

```
//абстрактний базовий клас для всіх методів аналізу
abstract class BasePrimaryProcessingService
{
    public function __construct(protected array $data = [])
    {
    }
    //геттер вхідних даних
    public function getData(): array
    {
        return $this->data;
    }

    //сеттер вхідних даних
    public function setData(array $data): void
    {
        $this->data = $data;
    }

    // метод аналізу
    public abstract function process():float;
    // метод підрахунку загальної кількості даних
    protected final function count():int
    {
        return count($this->data);
    }
    // метод підрахунку суми
    protected final function sum():float
    {
        $sum = 0;

        foreach ($this->data as $item){

            $sum += $item;
        }

        return $sum;
```

```
    }
}

//Клас помилки середньої арифметичної
class ArithmeticMeanError extends BasePrimaryProcessingService
{
    /**
     * @return float
     */
    public function process(): float
    {
        $dispersion = (new Dispersion($this->data))->process();

        return
ProcessResultFormatter::format($dispersion/sqrt($this->count()));
    }
}

//Клас середнього арифметичного
class ArithmeticalMean extends BasePrimaryProcessingService
{

    public function process(): float
    {
        return
ProcessResultFormatter::format($this->sum()/$this->count());
    }
}

//Клас середньої гармонійної
class AverageHarmonic extends BasePrimaryProcessingService
{
    /**
     *
     * @return float
     */
```

```
    public function process(): float
    {
        $reciprocalAmount = $this->reciprocalAmount();

        if($reciprocalAmount == 0) {
            return 0;
        }
        return
ProcessResultFormatter::format($this->count()/$this->reciprocalAmount(
));
    }

    /**
     * зворотня сума
     */
    public function reciprocalAmount(): float
    {
        $value = 0;

        foreach ($this->data as $item){
            if($item != 0){
                $value += 1/$item;
            }
        }

        return $value;
    }

//Клас середньої степеневої
class DegreeMean extends BasePrimaryProcessingService
{
    /**
     * середня степенева
     *
     * @param array $data Масив даних
     * @param int $degree степінь
```

```
     */
    public function __construct(
        array       $data,
        private int $degree
    )
    {
        parent::__construct($data);
    }

    public function process(): float
    {
        return
ProcessResultFormatter::format(pow($this->multiplyForDegree() /
$this->count(), 1 / $this->degree));
    }

    public function setDegree(int $degree): void
    {
        $this->degree = $degree;
    }

    private function multiplyForDegree(): float
    {
        $value = 0;

        foreach ($this->data as $item) {
            $value += pow($item, $this->degree);
        }

        return $value;
    }
}
class Dispersion extends BasePrimaryProcessingService
{
    /**
     * Дисперсія
```

```
     *
     * @return float
     */
    public function process(): float
    {
        $arithmeticalMean = $this->sum() / $this->count();
        $absoluteSum = $this->absoluteSumSquare($arithmeticalMean);

        $count = $this->count() == 1 ? 1 : $this->count() - 1;

        return ProcessResultFormatter::format($absoluteSum / $count);
    }

    private function absoluteSumSquare(float $arithmeticalMean): float
    {
        $absoluteSum = 0;

        foreach ($this->data as $item) {
            $absoluteSum += pow($item - $arithmeticalMean, 2);
        }

        return $absoluteSum;
    }
}
//Клас середньої геометричної
class GeometricMean extends BasePrimaryProcessingService
{

    public function process(): float
    {
        $result = pow($this->multiply(), 1 / $this->count());

        if(is_nan($result)){
            return 0;
        }
```

```
        return ProcessResultFormatter::format($result);
    }

    private function multiply(): float
    {
        $value = 1;

        foreach ($this->data as $item) {
            if ($item !== 0) {
                $value *= $item;
            }
        }

        return $value;
    }
}
class LinearCoefficientVariation extends BasePrimaryProcessingService
{
    /**
     * Лінійний коефіцієнт варіації
     *
     * @return float
     */
    public function process(): float
    {
        $arith = (new ArithmeticalMean($this->data))->process();
        $linearDeviation = (new
MeanLinearDeviation($this->data))->process();

        if(!$arith){
            return ProcessResultFormatter::format(0);
        }

        return
ProcessResultFormatter::format($linearDeviation/$arith);//todo * 100%
    }
```

```
}
class Max extends BasePrimaryProcessingService
{

    public function process(): float
    {
        return ProcessResultFormatter::format(max($this->data));
    }
}
class MeanLinearDeviation extends BasePrimaryProcessingService
{
    /**
     * Середнє лінійне відхилення
     *
     * @return float
     */
    public function process(): float
    {
        $arithmeticalMean = $this->sum() / $this->count();
        $absoluteSum = $this->absoluteSum($arithmeticalMean);

        return
ProcessResultFormatter::format($absoluteSum/$this->count());
    }

    private function absoluteSum(float $arithmeticalMean): float
    {
        $absoluteSum = 0;

        foreach ($this->data as $item) {
            $absoluteSum += abs($item - $arithmeticalMean);
        }

        return $absoluteSum;
    }
}
```

```
class Median extends BasePrimaryProcessingService
{
    /**
     * Медіана
     *
     * @return float
     */
    public function process(): float
    {
        asort($this->data);
        $length = $this->count();

        return
ProcessResultFormatter::format($this->getMedian($length));
    }

    private function getMedian(int $length)
    {
        if ($length % 2) {
            return $this->getMiddleValue($length);
        } else {
            return $this->getMeanValue($length);
        }
    }

    private function getMiddleValue(int $length)
    {
        return $this->data[floor($length / 2)];
    }

    private function getMeanValue(int $length)
    {
        return ($this->data[$length / 2 - 1] + $this->data[$length /
2]) / 2;
    }
```

```
}
class Min extends BasePrimaryProcessingService
{

    public function process(): float
    {
        return ProcessResultFormatter::format(min($this->data));
    }
}
```