ВІСНИК ПРИАЗОВСЬКОГО ДЕРЖАВНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ
2021р. Серія: Технічні науки Вип. 42
p-ISSN: 2225-6733; e-ISSN: 2519-271X

КОМП'ЮТЕРНІ НАУКИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

УДК 004.421.6 **doi: 10.31498/2225-6733.42.2021.240563**

© Бичко Д.В.[1], Шендрик В.В.[2], Парфененко Ю.В.[3]

РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ ДЛЯ РОБОТИ ЗІ СЛАБОСТРУКТУРОВАНИМИ ДАНИМИ МЕДИЧНИХ ПРОТОКОЛІВ

У даній роботі розглядається реалізація пошуку слабоструктурованих даних медичних протоколів з використанням API, який був створений на базі програмного додатку. Запропоновано процес обробки слабоструктурованого медичного протоколу шляхом формування JSON-файлів, занесення їх до бази даних та представлення на веб-сторінці. Запити користувача оброблюються через розроблений API та відображаються у вигляді результату через створений інтерфейс програмного додатку. Запропоновано структуру зберігання даних медичних протоколів у реляційній базі даних, механізм їх занесення та оновлення бази шляхом виконання автоматизованого скрипту. На прикладі тестових даних представлено градацію параметрів, що отримані шляхом первинної обробки протоколу і чітко характеризують симптоми та дозволяють швидко визначити захворювання. Розроблено програмний інтерфейс, який реалізує автоматичний витяг даних з бази даних, застовуючи PHP, та дозволяє взаємодіяти лікарю з системою шляхом розширення вже існуючої інформаційної системи – довідника медичних протоколів у pdf-форматі, та реалізовано пошук ймовірних хвороб на основі введених лікарем симптомів через API шляхом використання мови програмування PHP. Покроково описані етапи методу обробки вхідних симптомів та представлено у вигляді вихідного запиту з відображенням ілюстрованих прикладів роботи через інструмент тестування та розробки API – Postman. Запити відправляються до бази даних, результат виводиться у форматі JSON. Даний підхід дозволяє динамічно формувати запит та, в залежності від вхідних параметрів, оброблювати існуючі дані у базі даних. Як результат, розроблено програмний додаток, який за введеними симптомами виводить усі можливі хвороби, що мають відповідні симптоми.
Ключові слова: *слабоструктуровані медичні дані, запит, JSON, база даних, API, програмний інтерфейс.*

D.V. Bychko, V.V. Shendryk, Yu.V. Parfenenko. Development of a software application for work with poorly structured data of medical protolocs. *This paper considers the implementation of the search for poorly structured data of medical protocols using the API which was created on the basis of a software application. The processing of a poorly structured medical protocol by creating JSON files, entering them into a database and presenting them on a web page has been proposed. The user's requests are processed through the developed API and are displayed as the result through the created interface of the software application. The structure of data storage of medical protocols in a relational database, the mechanism of their entering and updating the database by executing an automated script*

[1] *аспірант, Сумський державний університет, м. Суми, ORCID: 0000-0002-6854-945X, d.bychko11@gmail.com*
[2] *канд. техн. наук, доцент, Сумський державний університет, м. Суми, ORCID: 0000-0001-8325-3115, v.shendryk@cs.sumdu.edu.ua*
[3] *канд. техн. наук, доцент, Сумський державний університет, м. Суми, ORCID: 0000-0003-4377-5132, yuliya_p@cs.sumdu.edu.ua*

has been proposed. The example of the test data presents the gradation of the parameters obtained by primary processing of the protocol; the parameters clearly characterizing the symptoms and making it possible to quickly identify the particular disease. A software interface that implements automatic retrieval of data from the database using PHP, and makes it possible for the doctor to interact with the system by expanding the existing information system- that is a directory of medical protocols in pdf-format- has been developed, and search for possible diseases based on symptoms entered by the doctor via API use of PHP programming language has been implemented. The steps of the treatment method of the input symptoms have been described step by step and are presented in the form of an output query with the display of illustrated examples of work through the testing and developing API – Postman tool. Queries are sent to the database, the result is displayed in JSON format. This approach makes it possible to dynamically generate a query, and depending on the input parameters, to process the existing data in the database. As a result, a software application has been developed that displays all possible diseases that have the corresponding symptoms based on the entered symptoms.
Keywords: *poorly structured medical data, query, JSON, database, API, software interface.*

Постановка проблеми. У медичній сфері для визначення діагнозу та узгодження лікування використовуються уніфіковані слабоструктуровані медичні протоколи, які, зазвичай, складаються з більш ніж 50 сторінок. Час від часу з розвитком знань стосовно діагностування та лікування тих чи інших хвороб (наприклад, COVID-19) протоколи модифікуються, і, як результат, для надання кваліфікованої медичної допомоги лікарю необхідно актуалізувати свої знання. Але через велику кількість протоколів, відсутність чіткої історії змін версій та збігу симптомів для різних хвороб процес прийняття медичних рішень ускладнюється, і виникає потреба у лікаря більше часу приділяти вивченню всіх протоколів, ніж лікуванню хворих. Тому, для підвищення достовірності встановленого діагнозу з'явилася необхідність у створенні інструменту для витягу даних, які стосуються симптомів хвороби, із затверджених медичних протоколів, зберігання їх у зручній формі та легкої взаємодії лікаря з медичними протоколами під час прийому пацієнтів.

Аналіз останніх досліджень і публікацій. Для збереження усіх даних медичних протоколів в єдиному цифровому форматі було розроблено електронний довідник [1], який зберігає уніфіковані медичні протоколи у pdf-файлах з можливістю пошуку та завантаження. Реалізовано обробку даних з файлів медичних протоколів за допомогою методу, описаного у роботах [2, 3]. Результати обробки скрипту заносяться у JSON-файл. У роботі [4] представлено порівняння двох типів алгоритмів для зберігання у базі даних JSON-файлів та швидкість їх обробки. Цей підхід може використовуватися для зберігання даних медичних протоколів, так як він дозволяє якісно працювати з ієрархією даних у рамках одного файлу. Підхід, представлений у роботі [5], дозволяє підвищити ефективність роботи у сервісно-орієнтованих системах при взаємодії з XML-документами. Через велику кількість документів розмір XML-документів буде більшим, ніж JSON, тому модифікацію даного підходу можна використати у даній роботі з JSON-файлами. Можливість працювати з великими об'ємами даних описано у роботі [6]. Представлено підхід до зберігання та інтеграції даних у різних базах даних. Представлені рішення концептуально можуть бути застосовані у даній роботі, але адаптовані для використання у безкоштовних версіях системи керування базами даних [7].

У результаті огляду останніх досліджень та публікацій встановлено, що необхідно провести модифікацію існуючих рішень з обробки слабоструктурованих даних з урахуванням специфіки предметної області.

Метою даної роботи є розробка програмного додатку для роботи зі слабоструктурованими медичними протоколами на основі запропонованої структури даних для зберігання у базі даних та отримання доступу до даних шляхом обробки вхідних запитів через користувальницький інтерфейс.

Виклад основного матеріалу. Метод обробки слабоструктурованого медичного протоколу (рис. 1) дає змогу структурувати хворобу з її симптомами у зручному вигляді шляхом ділення оброблених даних та представлення у вигляді окремих JSON-файлів, як описано у роботах [2, 3]. Результатом обробки кожного протоколу є: 1) назва хвороби; 2) міжнародний іде-

p-ISSN: 2225-6733; e-ISSN: 2519-271X

нтифікатор МКХ-10; 3) набір симптомів; 4) набір значень для симптомів із кількісними та якісними значеннями.

Рис. 1 – Етапи методу обробки слабоструктурованого медичного протоколу

Розглянемо як приклад отриманий JSON-файл з даними про симптоми однієї хвороби:

```
{
   "disease": [
      {
         "name": "грип",
         "icd10": "J10",
         "нежить": "відсутній",
         "кашель": "середній",
         "температура": "38.5-39.5"
      }
   ]
}
```

Унікальні параметри «name» та «icd10» отримані з неструктурованого медичного протоколу. Параметр «нежить» може приймати наступні можливі значення: «відсутній», «слабкий», «сильний». Параметр «кашель» має наступні можливі значення: «слабкий», «середній», «сильний». Параметр «температура» розділена на наступні можливі діапазони значень: «36.5-37.5», «37.5-38.5», «38.0-39.0», «38.5-39.5».

Одержані JSON-файли зберігаються у реляційній базі даних відповідно до представленої структури, зображеної на рис. 2. У разі необхідності присутня можливість додавання додаткових полів, а саме: параметру та його значення. Поле id створюємо з типом int та робимо його як Primary Key. Всі інші поля – з типом varchar.

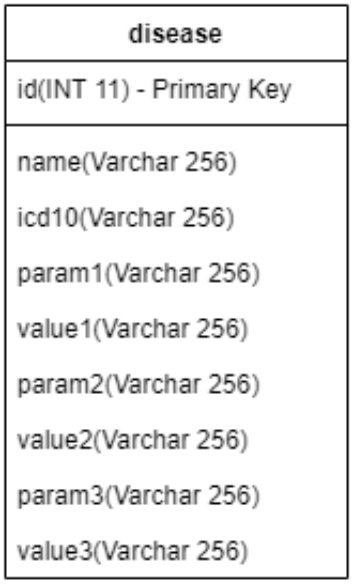

Рис. 2 – Структура даних

p-ISSN: 2225-6733; e-ISSN: 2519-271X

Занесення даних з кожного окремого JSON-файлу до бази даних відбувається шляхом автоматичного виконання скрипту, що розроблено на мові PHP. Результатом роботи скрипту є занесення даних до бази даних MySQL (рис. 3).

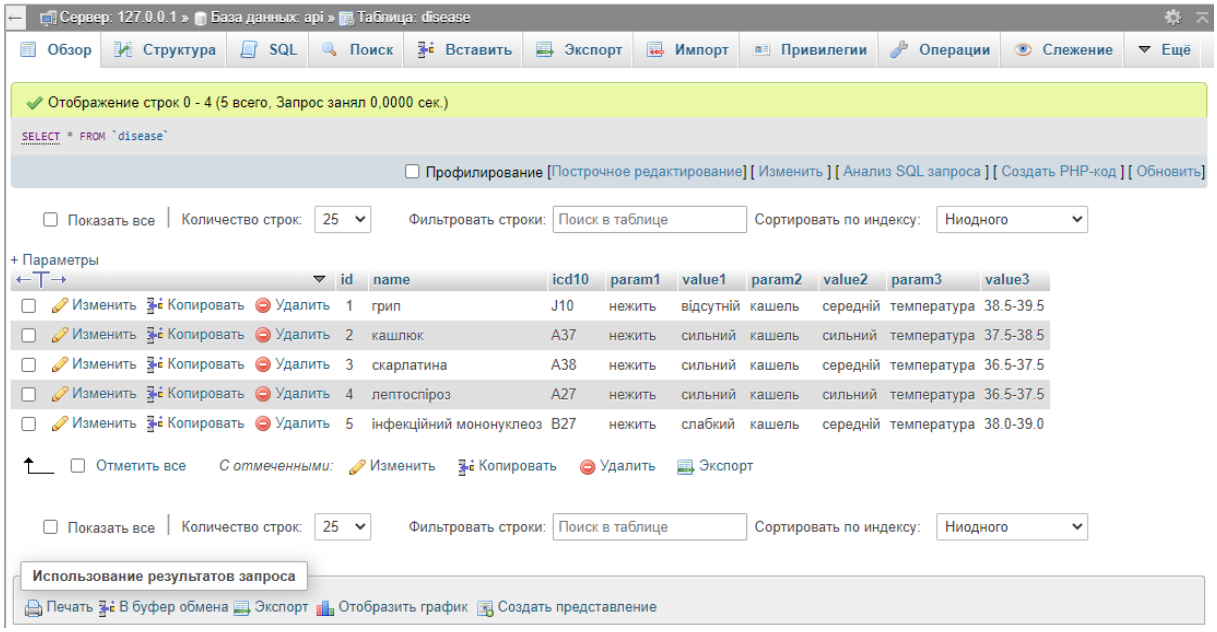

Рис. 3 – Результат заповнення даних з JSON-файлу

Для зручної взаємодії кінцевого користувача з даними медичних протоколів було вирішено інтегрувати раніше створений довідниковий ресурс [1], що складається зі слабоструктурованих медичних протоколів, які представлені у pdf-форматі, та програмний модуль пошуку хвороб за заданими симптомами, й розробити аналогічний дизайн пошукового модуля. Для розробки веб-інтерфейсу програмного додатку відносно макету використано HTML та CSS, а для динамічного додавання/видалення параметрів – JavaScript. Результат розробки представлений на рис. 4.

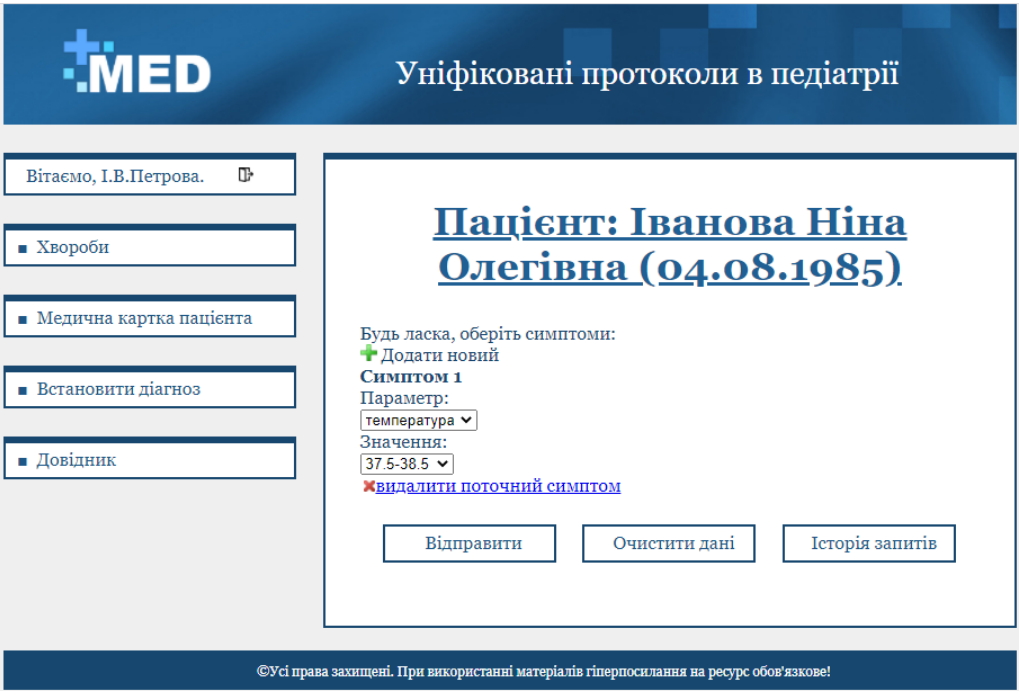

Рис. 4 – Відображення інтерфейсу

p-ISSN: 2225-6733; e-ISSN: 2519-271X

Веб-сторінка складається з 4 основних частин:

- Верхня частина сайту – містить назву ресурсу.
- Меню – знаходиться зліва і складається з 5 блоків:
  - форма з ім'ям та прізвищем лікаря, а також можливість вийти з облікового запису (використання даного ресурсу можливе лише при авторизації, тобто лише при наявності облікового запису лікар має доступ до ресурсу та даних власних пацієнтів);
  - хвороби – посилання на вже існуючий ресурс з медичними протоколами в педіатрії (можна ознайомитися за посиланням [1]);
  - медична картка пацієнта – з історією звернень до лікаря та можливістю відслідковування хронічних хвороб;
  - встановити діагноз – посилання на сторінку, в якій присутній фільтр для отримання даних про хворобу;
  - довідник – короткий довідник для лікаря з детальним описом роботи у системі.
- Контент – місце відображення даних про пацієнта, роботи з симптомами та їх значеннями через динамічне додавання/видалення параметрів, кнопок відправки запиту, очищення списку параметрів та історії запитів.
- Нижня частина сайту – місце, де розміщена інформація про приватність даних для копіювання.

Як видно з рисунку 4, користувач в рамках симптому обирає параметр та його значення, натискає на кнопку «Відправити» і повинен отримати результат у вигляді списку усіх можливих хвороб, які знайдені за заданими симптомами з відсотками збігу та посиланнями на протоколи лікування у довіднику. Для легкості підтримки програмного додатку було вирішено розробити власний API (application programming interface). Дане рішення дозволить відправляти запит напряму, а саме формувати запит у базу даних і передавати його Get-запитом та швидко отримувати необхідний результат.

Для розробки програмного додатку використана мова PHP та програма Postman, яка дозволяє швидко тестувати API. Файл зі скриптом складається з 4 основних функціональних частин, що дозволяють користувачу взаємодіяти з API:

- підключення заголовків для роботи з API;
- підключення до бази даних;
- методу обробки вхідних даних для формування рядку запиту до бази даних;
- функції обробки запитів.

Розглянемо більш детально кожну частину:

1. Для підключення заголовків використаємо представлені нижче рядки коду, які дозволять відправляти та обробляти дані під час запиту до API:

```
header('Access-Control-Allow-Origin: *');
header('Access-Control-Allow-Headers: *');
header('Access-Control-Allow-Methods: *');
header('Access-Control-Allow-Credentials: true');
header('Content-type: json/application').
```

2. Для підключення до бази даних використано PHP-метод mysqli_connect(host, username, password, dbname, port, socket).

3. Метод обробки вхідних даних реалізовано у встановленій конфігурації файлу .htaccess та файлі index.php. Його мета – отриманий запит переробити у рядок для відправки коректного запиту до бази даних. Нехай вхідний запит має наступний вигляд: http://localhost/test.com.ua/diseases&param1=нежить&value1=сильний. Етапи реалізації методу представлені нижче:

1) У файл .htaccess додамо правило RewriteRule ^(.+)$ index.php?q=$1 [L,QSA], який дозволяє після ? отримати параметр q зі значенням =$1.

2) У файлі index.php завдяки $q=$_GET['q']; отримуємо вхідні дані із запиту. За допомогою методу explode('/',$q) розбиваємо вхідний рядок на дві частини: до «/» та після нього. Усі наступні дії робимо для даних після «/».

3) Через цикл перебираємо отримані дані та представляємо у вигляді «$q .= $key.'='.$value.'&';». Результат обробки відображений на рис. 5.

p-ISSN: 2225-6733; e-ISSN: 2519-271X

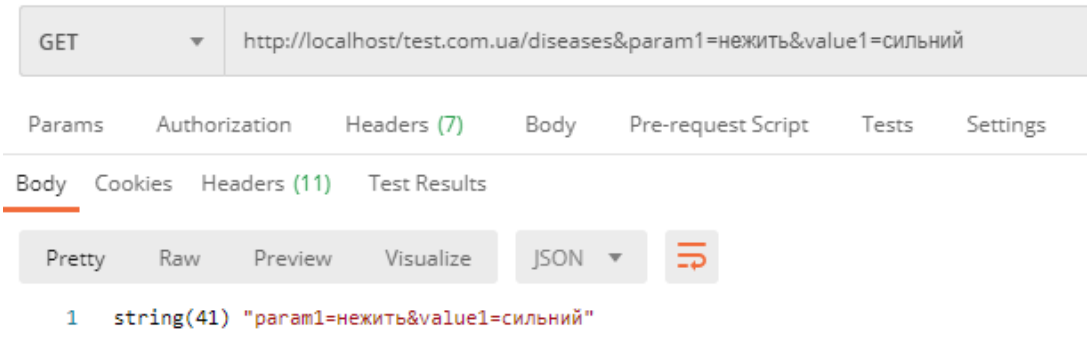

Рис. 5 – Результат обробки даних у циклі для кроку 1

4) За допомогою методу substr_replace() видаляємо останній символ рядку, а саме «&».
5) Методом strstr() видаляємо на початку рядку дані до першого входження «&».
6) За допомогою методу substr() видаляємо перший символ вхідного рядку, а саме «&».
7) Метод str_replace() дає змогу перетворити вхідні дані у вигляді запиту до бази даних. Результат представлений на рис. 6.

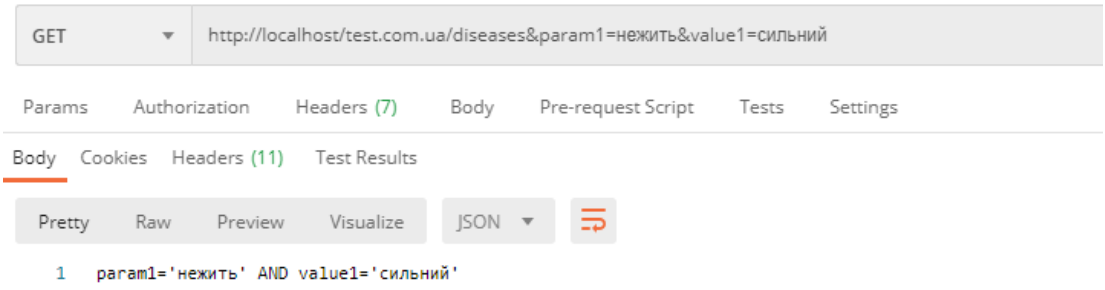

Рис. 6 – Результат роботи str_replace() методу

Отримані дані використовуються у пункті 4.

4) Розроблено дві функції обробки запитів: перша – виводить всі хвороби, у випадку відсутності вхідних параметрів ("SELECT * FROM `disease`"); друга – виводить результат обробки запиту (або інформацію, що нічого не знайдено, або всі поля отриманої хвороби/хвороб) – "SELECT * FROM `disease` WHERE $id". На рис. 7 представлений приклад запиту з трьома параметрами, який повертає результат у вигляді JSON.

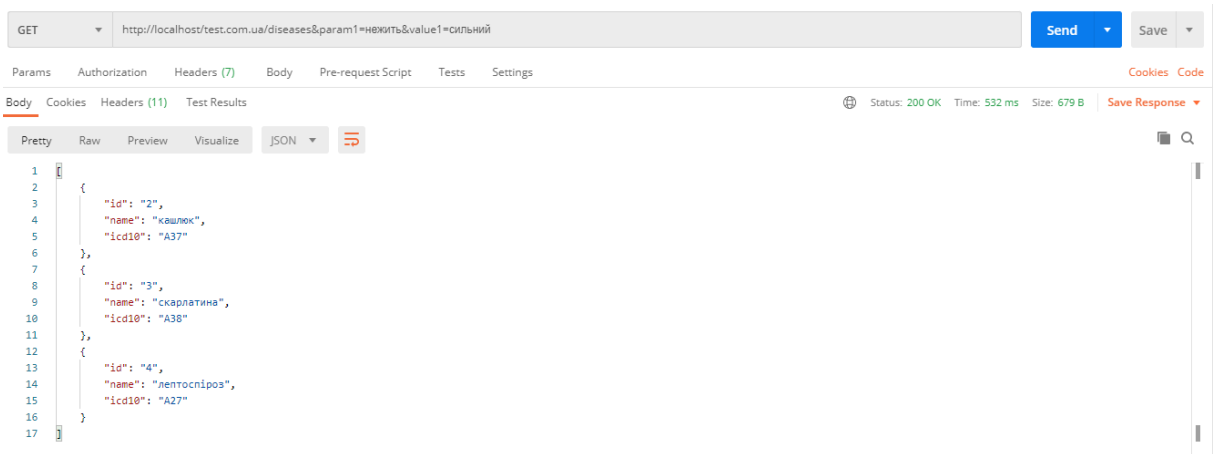

Рис. 7 – Результат роботи запиту до API з одним симптомом

Розроблений метод обробки вхідних даних та API-інтерфейс реалізовано у програмному додатку, який виконує автоматичний пошук усіх можливих хвороб за заданими симптомами. Як результат, лікар через зручний інтерфейс задає симптоми та очікує на виведення списку усіх можливих хвороб, які знайдені за заданими симптомами та отримані шляхом роботи API.

Висновки

У результаті роботи розроблено структуру зберігання даних медичних протоколів, які імпортовані з JSON файлу, у реляційній базі даних. Представлено метод обробки даних для формування запитів через API, який дозволяє швидко отримувати необхідні дані та представляти результати пошуку за заданими симптомами у вигляді списку можливих хвороб користувачу. Розроблено веб-інтерфейс програмного додатку, що реалізує запропонований метод. Створений програмний додаток у подальшому буде інтегровано як складову інформаційної системи підтримки прийняття медичних рішень.

Перелік використаних джерел:

1. Уніфіковані протоколи в педіатрії [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: https://uniprotped.med.sumdu.edu.ua/. – Назва з екрану.
2. Bychko D. Method of Primary Processing Unstructured Medical Data / D. Bychko, V. Shendryk, Y. Parfenenko // 2020 International Conference on e-Health and Bioengineering (EHB). – 2020. – Pp. 1-4. – Mode of access: https://doi.org/10.1109/EHB50910.2020.9280175.
3. Бичко Д. Метод первинної обробки слабоструктурованих медичних даних / Д. Бичко, В. Шендрик, Ю. Парфененко // Information systems and networks. – 2020. – Vol. 8. – С. 1-10. – Режим доступу: https://doi.org/10.23939/sisn2020.08.001.
4. Petkovic D. Shredding JSON Data into Relational Environment / D. Petkovic, A. Piriyaie // International Journal of Computer Applications. – February, 2021. – Vol. 174 (17). – Pp. 25-29. – Mode of access: https://doi.org/10.5120/ijca2021921058.
5. Tiwary G. Compression of XML and JSON API Responses / G. Tiwary, E. Stroulia, A. Srivastava // IEEE Access. – April, 2021. – Pp. 1-15. Mode of access: https://doi.org/10.1109/ACCESS.2021.3073041.
6. Petkovic D. JSON Integration in Relational Database Systems / D. Petkovic // International Journal of Computer Applications. – June, 2017. – Vol. 168 (5). – Pp.14-19. – Mode of access: https://doi.org/10.5120/ijca2017914389.
7. Oracle Database 12c Standard Edition 2 [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Режим доступу: https://softlist.com.ua/catalog/product-oracle-database-12c/. – Назва з екрану.

References:

1. *Unifikovani protokoly v pediatrii* (Unified protocols in pediatrics) Available at: https://uniprotped.med.sumdu.edu.ua/ (accessed 08 March 2021). (Ukr.)
2. D. Bychko, V. Shendryk, Y. Parfenenko. Method of Primary Processing Unstructured Medical Data. *2020 International Conference on e-Health and Bioengineering (EHB)*, 2020, pp. 1-4. **doi: 10.1109/EHB50910.2020.9280175**.
3. Bychko D., Shendryk V., Parfenenko Y. Metod pervinnoї obrobki slabostrukturovanikh medichnikh danikh [The method of primary processing of poorly structured medical data]. *Information systems and networks*, 2020, vol. 8, pp. 1-10. **doi: 10.23939/sisn2020.08.001**. (Ukr.)
4. Petkovic D., Piriyaie A. Shredding JSON Data into Relational Environment. *International Journal of Computer Applications*, February 2021, vol. 174 (17), pp. 25-29. **doi: 10.5120/ijca2021921058**.
5. Tiwary G., Stroulia E., Srivastava A. Compression of XML and JSON API Responses. *IEEE Access*, April 2021, pp. 1-15. **doi: 10.1109/ACCESS.2021.3073041**.
6. Petkovic D. JSON Integration in Relational Database Systems. *International Journal of Computer Applications*, June 2017, vol. 168 (5), pp.14-19. **doi: 10.5120/ijca2017914389**.
7. Oracle Database 12c Standard Edition 2 Available at: https://softlist.com.ua/catalog/product-oracle-database-12c/ (accessed 08 March 2021). (Rus.)

Рецензент: А.С. Довбиш
д-р техн. наук, проф., СумДУ Стаття надійшла 21.03.2021