УДК 004.41, 004.05
DOI https://doi.org/10.32782/IT/2023-2-13

Оксана КИРИЧЕНКО
асистент кафедри математичних проблем управління і кібернетики, Чернівецький національний університет імені Юрія Федьковича, вул. Коцюбинського 2, м. Чернівці, Україна, 58002
***ORCID:** 0000-0003-0282-9958*

Бібліографічний опис статті: Кириченко, О. (2023). Особливості архітектури програмного забезпечення для збору та аналізу статистичної інформаціїї в глобальній мережі. *Information Technology: Computer Science, Software Engineering and Cyber Security*, 2, 107–112, doi: https://doi.org/10.32782/IT/2023-2-13

ОСОБЛИВОСТІ АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ЗБОРУ ТА АНАЛІЗУ СТАТИСТИЧНОЇ ІНФОРМАЦІЇЇ В ГЛОБАЛЬНІЙ МЕРЕЖІ

Для вивчення структури та характеристики web-простору необхідно використовувати спеціалізоване програмне забезпечення. В даній роботі розглянуто особливості архітектури подібного програмного забезпечення та приділена особлива увага важливим аспектам його роботи, таким як: механізми навігації, стратегії сканування та обробка отриманих даних. У статті також розглянуто проблеми ефективності та масштабованості подібних рішень при обробці великої кількості веб-ресурсів. Розроблено програмне забезпечення, яке складається з кроулера та аналітичного модуля. Функціональним призначенням аналітичного модуля є проведення кластерного та статистичного аналізу великих об'ємів даних за допомогою різноманітних статистичних методів. Архітектура розробленого додатку відповідає останнім тенденціям у сфері програмного забезпечення, враховуючи сучасні вимоги та стандарти. За допомогою даного програмного продукту з використанням розробленого аналітичного модуля досліджені статистичні та кластерні характеристики різних сегментів веб-простору (українського – edu.ua, net.ua; польського сегменту – edu.pl та ізраїльського – ac.il).

Ключові слова: *програмне забезпечення, тестування програмного забезпечення, рівні тестування програмного забезпечення, вимоги до програмного забезпечення: функціональні характеристики (вимоги), нефункціональні характеристики (вимоги), специфікація, статистичні методи.*

Oksana KYRYCHENKO
Assistant Professor, Department of Mathematical Problems of Control and Cybernetics, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine, 58002, o.kyrychenko@chnu.edu.ua
***ORCID:** 0000-0003-0282-9958*

To cite this article: Kyrychenko O. (2023). Osoblyvosti arkhitektury prohramnoho zabezpechennia dlia zboru ta analizu statystychnoi informatsiii v hlobalnii merezhi. [Features of software architecture for collecting and analyzing statistical information on the global network]. *Information Technology: Computer Science, Software Engineering and Cyber Security*, 2, 107–112, doi: https://doi.org/10.32782/IT/2023-2-13

FEATURES OF SOFTWARE ARCHITECTURE FOR COLLECTING AND ANALYZING STATISTICAL INFORMATION ON THE GLOBAL NETWORK

To study the structure and characteristics of World Wide Web, specialized software is essential. This paper examines the peculiarities of the such software architecture and pays high attention to important aspects of its performing, such as navigation mechanisms, scanning strategies, and the processing of acquired data. The article also addresses the challenges of efficiency and scalability in similar solutions when handling a large number of web resources. The developed software consists of a crawler and an analytical module. The functional purpose of the analytical module is to conduct cluster and statistical analysis of large volumes of data by means of statistical methods. The architecture of the developed application aligns with the latest trends in software development, considering modern requirements and standards. Using this software with integrated analytical module, statistical and cluster characteristics of various segments of WWW have been researched, including the Ukrainian segments (edu.ua, net.ua), the Polish segment (edu.pl), and the Israeli segment (ac.il).

***Key words:** software, software testing, levels of software testing, software requirements: functional requirements, non-functional requirements, requirements, statistical methods.*

Основною задачею програмного забезпечення для збору, аналізу та використання інформації з глобальної мережі є навігація по всесвітній павутині та завантаження контенту веб-сторінок. Подібне програмне забезпечення повинно взаємодіяти з веб-ресурсами для знаходження та збору інформації, використовуючи ефективні алгоритми навігації та сканування, видобувати та аналізувати дані з різних джерел, включаючи структуровані та неструктуровані дані.

Важливим компонентом будь-яких аналітичних програм, які обробляють велику кількість веб-сторінок є веб-кроулер.

Веб-кроулер – це програма, яка за допомогою однієї або кількох початкових URL-адрес завантажує веб-сторінки, пов'язані з цими URL-адресами, витягує будь-які гіперпосилання, що містяться в них, і рекурсивно продовжує завантажувати веб-сторінки за цими гіперпосиланнями (Najork, Heydon, 2002; Najork, 2017).

Загалом розробка високопродуктивних веб-кроулерів є нетривіальною задачею та включає ряд складнощів і викликів. Зокрема, веб-кроулерам доводиться стикатися з обмеженнями, накладеними веб-серверами. Деякі сервери можуть обмежувати швидкість доступу до сторінок для одного IP, що ускладнює підтримку високої швидкості сканування. Також, сканування великих обсягів даних може призводити до проблем із зберіганням, обробкою та передачею цих даних мережею. Кроулери повинні ефективно масштабуватися та використовуввати паралельну обробку даних. Робота із великою кількістю задач одночасно потребує правильної архітектури та керування ресурсами.

Крім того, однією з найскладніших задач є обробка динамічного контенту сайтів, згенерованого з використанням технологій, таких як AJAX або JavaScript.

Для розробки якісного програмного забезпечення потрібно визначити функціональні та нефункціональні вимоги, тобто визначити характеристики, якими повинен володіти програмний продукт, щоб відповідати потребам поставленої задачі.

Функціональні вимоги представляють основні функціональні можливості або особливості, якими повинна володіти система програмного забезпечення, щоб відповідати призначеній меті. Створюючи функціональні вимоги, важливо мати на увазі, що вони повинні бути конкретними, вимірними, досяжними, релевантними та обмеженими у часі.

Нефункціональні вимоги доповнюють функціональні вимоги, вказуючи, як програмна система повинна виконувати певні функції. Вони задають атрибути якості системи, а не її специфічні особливості, встановлюють стандарти продуктивності, безпеки та зручності використання системи, пояснюють обмеження системи, що проектується. До характеристик, якими визначаються нефункціональні вимоги відносять, продуктивність, масштабованість, портативність, сумісність, надійність, доступність, безпека, локалізація, зручність використання тощо.

Саме тому, визначення вимог є ключовим етапом у розробці подібного програмного забезпечення (Говорущенко, Боднар, Кушнір, 2019; Hovorushchenko et al., 2019).

Опишемо функціональні та нефункціональні вимоги до системи, яка розроблена (Paech et al., 2002; Alashqar et al., 2015).

Функціональні вимоги:

− задавати вхідні URL-адреси та глибину сканування;

− генерувати список посилань для завантаження;

− фільтрувати на основі URL-адрес, у якому шаблони визначають імена хостів та/або шляхи, що цікавлять;

− паралельно завантажувати веб-сторінки за гіперпосиланнями;

− парсити завантажені веб-сторінки та видобувати гіперпосилання;

− перетворювати отримані посилання відповідно до правил нормалізації посилань;

− забезпечувати збереження відповідної структури посилань;

− здійснювати проведення статистичного аналізу з використанням статистичних методів;

− здійснювати проведення кластерного аналізу різних зон веб-простору.

Нефункціональні вимоги:

- планування (визначає порядок завантаження URL-адрес);

- політика відвідування сторінок (обмежує навантаження на веб-сервер при скануванні веб-простору);

- обробка веб-сторінок (видобуток гіперпосилань);

− визначення дублікатів (розпізнавати під час сканування дублікатів);

- інтерактивність (підтримувати файли cookie та прості схеми паролів);

- стабільність (бути стійким до можливих перерв у роботі, забезпечити ведення логування для моніторингу своєї поведінки та діагностики будь-яких проблем);

- протоколи (додавати нові протоколи завантаження до кроулеру);

- висока пропускна здатність;

- портативність (здатний працювати на Linux, виконувати невеликі та тестові запуски на Windows).

Розроблений кроулер складається з декількох модулів: Downloader, Content parser, Link extractor, Url resolver, Data access module, Url manager, Logging, Analytical module, Url queue, Download queue, Data store (Кириченко, Kanovsky, Остапов, 2013, с. 102). Модифіковану архітектуру розробленого кроулера зображено на рис. 1.

Гнучкість конфігурування та стабільна робота під навантаженням забезпечується модульною архітектурою кроулера. Наведемо короткий опис призначення модулів розробленого програмного забезпечення (Кириченко, Kanovsky, Остапов, 2013, с. 102).

Download queue – забезпечує збереження гіперпосилань для подальшого завантаження.

Downloader – завантажує контент сторінки за вказаним URL, обробляє заголовок запиту за сторінкою, дозволяє здійснювати завантаження контенту у багато поточному режимі.

Content parser – розбирає контент веб-сторінки, виділяючи структуру html-тегів та інформацію, яка знаходиться всередині тегів.

Link extractor – знаходить та добуває з контенту завантаженої сторінки посилання, враховуючи зовнішні та внутрішні а також посилання Flash-об'єктів.

Url resolver – перетворює отримані посилання відповідно до правил нормалізації посилань. Такі перетворення запобігають завантаженню однакових ресурсів більш, ніж один раз.

Data access module – забезпечує обробку та збереження відповідної структури посилань в базі даних.

Url manager – генерує список посилань, для завантаження. Цей модуль реалізує політику планування щодо порядку завантаження URL-адрес, відповідає за усунення дублікатів та визначає пріоритети при завантаженні сторінок.

Data store – використовується для збереження веб-графу, тобто сторінок, їх структури посилань, різноманітної статистики.

Analytical module – надає інтерфейс для проведення статистичного та кластерного аналізу веб-графу.

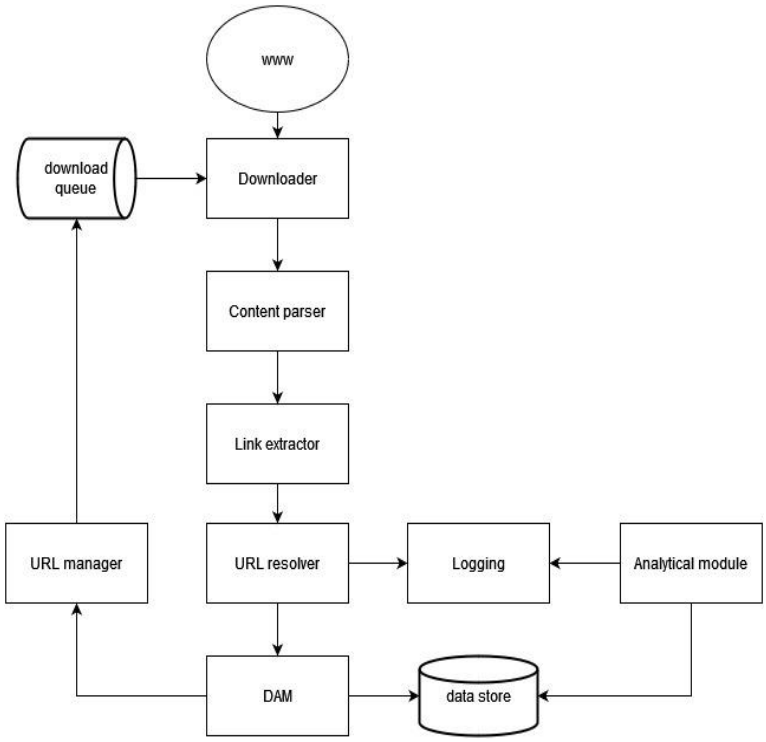

Рис. 1. Архітектура розробленого кроулера

Logging module – агрегує всі лог журнали, створені різними компонентами, щоб уніфікувати всі події в системі.

Головною особливістю розробленого нами програмного забезпечення є наявність аналітичного модуля, детальний опис якого наведено нижче.

Аналітичний модуль надає інтерфейс для проведення статистичного та кластерного аналізу веб-графу. А саме:

− побудови розподілу ймовірності вузлів за ступенями по вхідних зв'язках (in degree) та розподілу ймовірності вузлів за ступенями по вихідних зв'язках (out degree);

− обчислення коефіцієнтів кластерності веб-графу;

− визначення середніх значень ступеня вузла для неорієнтованих графів;

− визначення оптимальної кількості кластерів методом k-Core decomposition;

− знаходження центрів кластерів;

− розбиття досліджуваних мереж на кластери за допомогою алгоритму PIC (Power iteration clustering).

Архітектуру аналітичного модуля зображено на рис. 2.

Наведемо короткий опис компонентів аналітичного модуля.

Service interface – надає API для користувача, яке дозволяє ініціалізувати процес проведення статистичного та кластерного аналізу,

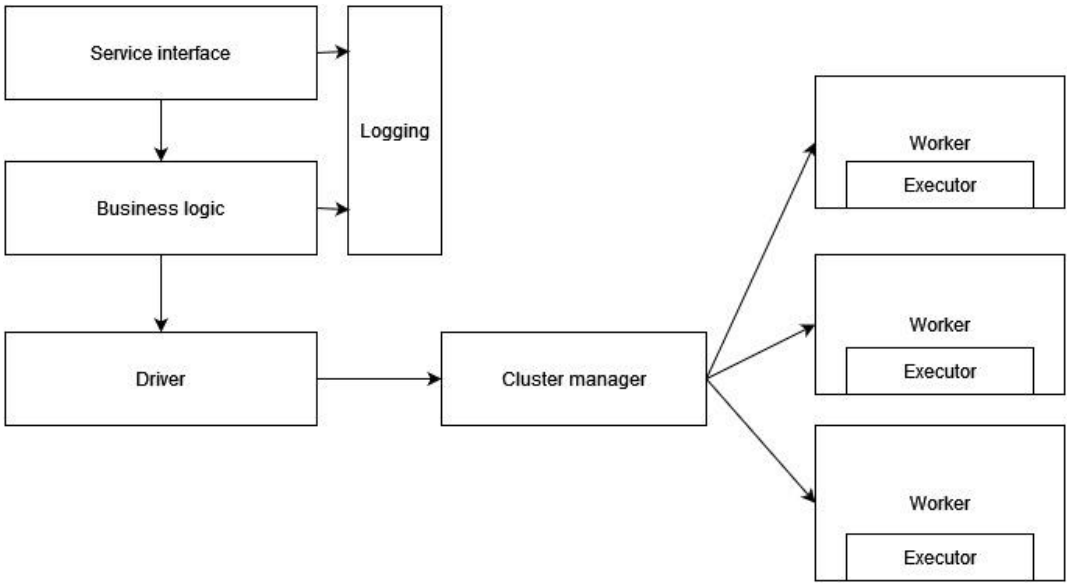

Рис. 2. Архітектура аналітичного модуля

отримує вхідні параметри та перенаправляє їх на рівень бізнес-логіки.

Business logic – відповідно до вхідних параметрів витягує необхідні дані зі сховища даних, здійснює їх перетворення та передає на вхід драйверу для здійснення обчислень.

Driver – координує воркери і контролює виконання завдань. Він складається з коду для виконання і відповідного контексту. Контекст отримує загальну задачу та ділить її на невеликі завдання, які обробляються виконавцями.

Workers – отримують від драйвера завдання та виконують їх. Виконавці утворюють кластер.

Cluster manager – відповідає за взаємодію з драйвером та виконавцями, здійснюючи такі завдання:

- керування виділенням ресурсів;
- керування поділом програми;
- керування виконанням програми.

Logging – це не фактичний леєр, а наскрізний компонент (тобто – він доступний для всіх леєрів). Цей компонент містить бібліотеку логування, яка використовується компонентом, і генерує події, які обробляються модулем логування.

Для проведення статистичного та кластерного аналізу веб-графу модуль надає REST API, що дозволяє користувачу вказати початкові URL-адреси.

Таким чином, використання багаторівневої архітектури дозволяє забезпечувати незалежну розробку і розвиток окремих частин додатку, ще більше збалансувати навантаження на різні вузли і мережу. Додаток складається з логічних та функціональних компонент, які можна використовувати повторно. Такий підхід значно спрощує розробку та підтримку розподілених систем (Jaiswal, 2019, p. 2454).

З іншого боку, подібна архітектура дозволяє ізолювати додаток, повністю інкапсулюючи середовище виконання, і запускати його на віртуальній машині замість фізичного сервера. З метою оптимізації використання ресурсів віртуального або фізичного сервера була запроваджена технологія контейнеризації.

Контейнери – це абстракція на рівні програми, яка пакує код і залежності разом. Кілька контейнерів можуть працювати на одній машині та спільно використовувати ядро операційної системи з іншими контейнерами, кожен із яких працює як ізольований процес у просторі користувача. Контейнери займають менше місця, ніж віртуальні машини, можуть працювати з більшою кількістю програм і потребують менше віртуальних машин і операційних систем (Cook, 2017, p. 31). Найпопулярнішим інструментом для контейнеризації є Docker. Docker – це відкрита платформа для розробки та експлуатації додатків.

Для розробки веб-кроулера, який би працював під операційною системою Windows та операційними системами сімейства Unix, було обрано мову програмування Java, яка є потужним інструментом для розробки розподілених та багатопоточних систем. Для створення додатків на Java використали Java Platform, Enterprise Edition (JEE), яка підтримує:

- багаторівневу архітектуру;
- незалежність від платформи;
- роботу в багатопоточному режимі;
- масштабованість;
- гнучкість;

- високу здатність до розширення;
- контейнеризацію;
- швидке розгортання.

Було проведено тестування розробленого програмного забезпечення. Основною метою тестування програмного забезпечення є оцінка відповідності програмного забезпечення зазначеним потребам.

Тестування програмного забезпечення зазвичай здійснюється з метою виявлення помилок для гарантування якості програмного забезпечення та виконання остаточної перевірки відповідності специфікаціям, дизайну. Тестування програмного забезпечення є необхідним етапом, який зазвичай виконується відповідно до установлених вимог (специфікацій) (Hovorushchenko et al., 2019). Тестування поділяється на різні рівні, такі як Unit (тестування окремих модулів або компонентів програми), Integration (тестування взаємодії між модулями або компонентами програми), System (тестування всієї системи як єдиного цілого) та Acceptance testing (тестування для підтвердження готовності системи до використання та відповідності вимогам) (Umar, 2019). Для здійснення ефективного тестування необхідно ретельно обирати засоби тестування.

Для оцінки якості розробленого програмного забезпечення використовувалися конкретні тестові сценарії та метрики, а саме коректність обробки структури веб-сайту, швидкості сканування, масштабованості. Функціонал компонентів програмного продукту покривався Unit тестами та інтеграційними тестами. Окрема увага приділялася приймальному тестуванню, яке засвідчило, що розроблене програмне забезпечення повністю відповідає зазначеним вище (функціональним та нефункціональним) вимогам.

За допомогою розробленого програмного забезпечення методом зондування досліджувались зони Web-простору: українська (edu.ua та net.ua), ізраїльська (ac.il) та польська (edu.pl). Для кожного сегменту було просканово більше 400 тисяч веб-сторінок. За допомогою аналітичного модуля проведена статистична обробка зібраних даних, розраховано: ступінь вузла (вхідний та вихідний), розподіл ступенів вузлів по вхідних зв'язках (indegree) та вихідних зв'язках (outdegree), коефіцієнт клстеризації та ін. Дослідження кластерної структури зон веб-простору проводилось за допомогою алгоритму спектральної кластеризації PIC (Power iteration clustering), обчислено оптимальну кількість кластерів методами «ліктя» та k-Core decomposition.

Висновки. За результатами проведених досліджень можна зробити наступні висновки:

1. Розроблено програмне забезпечення (кроулер та аналітичний модуль) для збору інформації з мережі WWW і проведення статистичного та кластерного аналізу.

2. Обґрунтовано вибір архітектурного стилю, елементів та обмежень при розробці програмного забезпечення.

3. Розроблений додаток підтримує багаторівневу архітектуру, забезпечує незалежність від платформи, роботу в багатопоточному режимі, масштабованість, гнучкість, високу здатність до розширення, контейнеризацію, швидке розгортання. Такий підхід відповідає сучасним тенденціям у сфері розробки програмного забезпечення.

ЛІТЕРАТУРА:

1. Najork M., Heydon A. High-Performance Web Crawling. In: (eds) *Handbook of Massive Data Sets. Massive Computing* / J. Abello, P.M. Pardalos, M.G.C. Resende, Boston : Springer, 2002. Vol. 4. P. 25-45.

2. Najork M. Web Crawler Architecture. *Encyclopedia of Database Systems*, 2017. P. 1–4.

3. Говорущенко Т.О., Боднар М.А., Кушнір В.О. Сучасні проблеми формування та аналізу вимог до програмного забезпечення. *Вимірювальна та обчислювальна техніка в технологічних процесах*. 2019. № 1. С. 45–53.

4. T. Hovorushchenko, O. Pavlova, M. Bodnar. Development of an Intelligent Agent for Analysis of Nonfunctional Characteristics in Specifications of Software Requirements. *Eastern-European Journal of Enterprise Technologies*. 2019. Vol. 1. No. 2 (97). P. 6–17.

5. Paech, Barbara & Dutoit, Allen & Kerkow, Daniel & Knethen, Antje. (2002). Functional requirements, non-functional requirements, and architecture should not be separated. A position paper. 2002.

6. Alashqar A., Elfetouh A., El-Bakry H. Requirement Engineering for Non-Functional Requirements. *International Journal of Information and Communication Technology Research.* 2015. № 5. P. 21–27.

7. Кириченко О.Л., Kanovsky I., Остапов С.Е. Програмне забезпечення для дослідження статистичних характеристик глобальної мережі WWW. *Системи обробки інформації.* 2013. Вип. 3 (110). Т. 2. С. 99–104.

8. Jaiswal M. Software Architecture and Software Design. *International Research Journal of Engineering and Technology*. 2019. Vol. 6. P. 2452–2454.

9. Cook J. Docker for Data Science: Building Scalable and Extensible Data Infrastructure Around the Jupyter Notebook Server. Berkeley : Apress, 2017. 257 p.

10. Umar M. Comprehensive study of software testing: Categories, levels, techniques, and types. *International Journal of Advance Research, Ideas and Innovations in Technology.* 2019. № 5. P. 32–40.

REFERENCES:

1. Najork, M., Heydon, A. (2002). High-Performance Web Crawling. In: Abello, J., Pardalos, P.M., Resende, M.G.C. (eds) Handbook of Massive Data Sets. Massive Computing, vol. 4. Springer, Boston, MA. DOI: 10.1007/978-1-4615-0005-6_2

2. Najork, Marc. (2017). Web Crawler Architecture. *Encyclopedia of Database Systems*. DOI: 10.1007/978-1-4899-7993-3_457-3

3. Hovorushchenko T.O., Bodnar M.A., Kushnir V.O. (2019). Suchasni problemy formuvannia ta analizu vymoh do prohramnoho zabezpechennia. [Modern problems of the formation and analysis of software requirements. Measuring and computing equipment in technological processes]. *Vymiriuvalna ta obchysliuvalna tekhnika v tekhnolohichnykh protsesakh – Measuring and computing equipment in technological processes, 1*, 45–53 [in Ukrainian].

4. T. Hovorushchenko, O. Pavlova, M. Bodnar (2019). Development of an Intelligent Agent for Analysis of Nonfunctional Characteristics in Specifications of Software Requirements. *Eastern-European Journal of Enterprise Technologies*, *1, 2 (97)*, 6–17.

5. Paech, Barbara & Dutoit, Allen & Kerkow, Daniel & Knethen, Antje. (2002). Functional requirements, non-functional requirements, and architecture should not be separated. A position paper.

6. Alashqar, Abdelkareem & Elfetouh, Ahmad & El-Bakry, Hazem. (2015). Requirement Engineering for Non-Functional Requirements. *International Journal of Information and Communication Technology Research, 5*, 2127.

7. Kyrychenko O.L., Kanovsky I., Ostapov S.E. (2013). Prohramne zabezpechennia dlia doslidzhennia statystychnykh kharakterystyk hlobalnoi merezhi WWW [Software for researching the statistical characteristics of the WWW global network]. *Systemy obrobky informatsii*. *– Information processing systems, 2*: *3 (110)*, 99–104 [in Ukrainian].

8. Jaiswal, Manishaben, Software Architecture and Software Design (November 5, 2019). *International Research Journal of Engineering and Technology* (IRJET) e-ISSN: 2395-0056, p-ISSN: 2395-0072, Vol. 06, Is. 11, s. no-303, pp. 2452–2454. Available at: https://www.irjet.net/archives/V6/i11/IRJET-V6I11303.pdf, Available at SSRN: https://ssrn.com/abstract=3772387 or http://dx.doi.org/10.2139/ssrn.3772387

9. Cook, Joshua, (2017). Docker for Data Science: Building Scalable and Extensible Data Infrastructure Around the Jupyter Notebook Server. Apress, Berkeley, 257 p.

10. Umar, Mubarak Albarka. (2020). Comprehensive study of software testing: Categories, levels, techniques, and types. DOI: 10.36227/techrxiv.12578714.v2.