Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: «Розробка Android - застосунку мовою Java для пошуку та збору інформації в середовищі Eclipse»

Виконав(ла): студент(ка) VI курсу, групи СПм-61
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва спеціальності)

(підпис) Дарнобит Н.Я. (прізвище та ініціали)

Керівник (підпис) Цуприк Г.Б. (прізвище та ініціали)

Нормоконтроль (підпис) Стоянов Ю.М. (прізвище та ініціали)

Завідувач кафедри (підпис) Петрик М.Р. (прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль 2023

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. М.Р. Петрик, Д.М. Михалик, О.Ю. Петрик, Г.Б. Цуприк. Методичні вказівки до виконання атестаційної роботи магістра за спеціальністю 121 – “Інженерія програмного забезпечення” для усіх форм навчання [Текст] – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя – 2020 – 27 с.

2. Нікольський Ю.В. Дискретна математика / Ю.В.Нікольський, В.В.Пасічник, Ю.М.Щербина – Львів: Магнолія Плюс, 2007. – 608 с.

3. Інформаційні технології видобутку даних (Data mining, високопродуктивні обчислення у складних системах): навчальний посібник ІВ Бойко, МР Петрик, Г Цуприк - 2020

4. Дискретні структури (Алгебраїчні та числові системи, комбінаторний аналіз) : навчально-методичний посібник для студентів спеціальності 121 «Інженерія програмного забезпечення», аспірантів та викладачів вищих навчальних закладів / Укладач : Бойко І.В., Петрик М.Р., Цуприк Г.Б. – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя, 2017 – 64 с.

5. Моделювання та видобуток даних (висопродуктивні обчислення у великих алгебраїчних та числових системх, комбінаторному аналізі): навчальний посібник. Тернопіль: : ТНТУ 2019 – 62 с.

6. Сутність_—_зв'язок [Електронний ресурс] – Режим доступу : URL : https://uk.wikipedia.org/wiki/Модель_«сутність_—_зв'язок»

7. James Rumbaugh, Ivar Jacobson, Grady Booch (1999). The unified modeling language reference manual (англ.). Addison Wesley Longman Inc. ISBN 0-201-30998-X.

8. База даних [Електронний ресурс] – Режим доступу : URL : https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85.

9. Реляційна база даних [Електронний ресурс] – Режим доступу: URL: https://uk.wikipedia.org/wiki/Реляційна_база_даних.

10. Oracle [Електронний ресурс] – Режим доступу: URL: www.omega.ru/oracleinfo.html.

11. MSDN [Електронний ресурс] – Режим доступу: URL: https://msdn.microsoft.com/.

12. MySQL [Електронний ресурс] – Режим доступу: URL: https://ru.wikipedia.org/wiki/MySQL.

13. Переваги MySQL [Електронний ресурс] – Режим доступу: URL: http://www.znannya.org/?view=concept:304.

14. Основні-відомості-про-запити [Електронний ресурс] – Режим доступу: URL: https://support.microsoft.com/uk-ua/office/основні-відомості-про-запити-a9739a09-d3ff-4f36-8ac3-5760249fb65c

15. «Сутність_—_зв'язок» [Електронний ресурс] – Режим доступу : URL: https://uk.wikipedia.org/wiki/Модель_«сутність_—_зв'язок»

16. Основні_поняття_моделі_«сутність_–_зв’язок» [Електронний ресурс] – Режим доступу: URL: https://wiki.cuspu.edu.ua/index.php/Модель_«сутність – зв’язок». Основні поняття моделі «сутність – зв’язок»: сутності, зв’язки, атрибути та їх класифікація

17. Життєвий_цикл_програмного_забезпечення [Електронний ресурс] – Режим доступу: URL: https://uk.wikipedia.org/wiki/ Життєвий_цикл_програмного_забезпечення

18. Unified_Modeling_Language [Електронний ресурс] – Режим доступу: URL: https://uk.wikipedia.org/wiki/Unified_Modeling_Language

19. Триярусна архітектура [Електронний ресурс] – Режим доступу : URL : https://uk.wikipedia.org/wiki/Триярусна_архітектура

20. Android [Електронний ресурс] – Режим доступу : URL : https://developers.google.com/android/.

21. Android-development-tools-eclipse Електронний ресурс] – Режим доступу : URL : https://marketplace.eclipse.org/content/android-development-tools-eclipse

22. Список_версій_Android [Електронний ресурс] – Режим доступу : URL : https://uk.wikipedia.org/wiki/Список_версій_Android

23. Step-by-step-guide-to-Android-development-with-Eclipse [Електронний ресурс] – Режим доступу: URL: https://www.theserverside.com/tutorial/Step-by-step-guide-to-Android-development-with-Eclipse

24. Android SDK [Електронний ресурс] – Режим доступу: URL: http://androidfanclub.ru/programming/%D0%BE%D0%B1%D0%B7%D0%BE%D1%80-android-sdk .

25. Компоненти Android [Електронний ресурс] – Режим доступу: URL: http://developer.alexanderklimov.ru/android/android1.php.

26. Developer.android [Електронний ресурс] – Режим доступу: URL: https://developer.android.com/about/versions/11/setup-sdk

27. Developer.android [Електронний ресурс] – Режим доступу: URL: https://developer.android.com/about/versions/11/setup-sdk

28. Developer.android [Електронний ресурс] – Режим доступу: URL: https://developer.android.com/about/versions/11/setup-sdk

29. instrumenti-dlja-profesijnoi-rozrobki-pid- android [Електронний ресурс] – Режим доступу: URL:https://www.zapptreinamentos.com.br/novo/2020/07/17/instrumenti-dlja-profesijnoi-rozrobki-pid-android/

30. Хостинг [Електронний ресурс] – Режим доступу : URL : https://ru.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3.

31. [Електронний ресурс] – Режим доступу: URL: https://pricework.org/uk/news/devops/data-transfer-protocols-what-are-they-what-are-they-and-what-are-the-differences/

32. File Transfer Protocol [Електронний ресурс] – Режим доступу: URL: https://tools.ietf.org/html/rfc959.

33. PHP [Електронний ресурс] – Режим доступу: URL: http://php.net/manual/ua/.

34. Тестування програмного забезпечення [Електронний ресурс] – Режим доступу : URL :http://lib.mdpu.org.ua/e-book/vstup/L11.htm

35. Дистанційний курс «Основи охорони праці» сайту дистанційного навчання ТНТУ [Електронний ресурс]. – Режим доступу: URL: http://dl.tntu.edu.ua/index.php

36. Про затвердження Правил охорони праці під час експлуатації ЕОМ [Електронний ресурс]. – Режим доступу: URL: http://zakon4.rada.gov.ua/laws/show/z0382-99

37. Методичний посібник для здобувачів освітнього ступеня «магістр» всіх спеціальностей денної та заочної (дистанційної) форм навчання «БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ» / В.С. Стручок –Тернопіль: ФОП Паляниця В. А., –156 с. Отримано з https://elartu.tntu.edu.ua/handle/lib/39196.

38. Інформацію при написанні зазначеного підрозділу можна отримати з навчального посібника: Навчальний посібник «ТЕХНОЕКОЛОГІЯ ТА ЦИВІЛЬНА БЕЗПЕКА. ЧАСТИНА «ЦИВІЛЬНА БЕЗПЕКА»» / автор-укладач В.С. Стручок– Тернопіль: ФОП Паляниця В. А., – 156 с. Отримано з http://elartu.tntu.edu.ua/handle/lib/39424

39. Н.Я.Дарнобит, Г.Б Цуприк Розробка Android-застосунку для пошуку та збору інформації мовою Java. Матеріали XI науково-технічної конфції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя, (Тернопіль, 13-14 грудня 2023 р.). – Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2023. – 257 с.

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить загальним обсягом:

стр., рис., табл., літ. дж., додатків.

КВАЛІФІКАЦІЙНА РОБОТА, ОБ'ЄКТ, ПРЕДМЕТ, МЕТА, АРХІТЕКТУРА, БАЗИ ДАНИХ, ІНТЕРНЕТ, СЕРВЕР, ЗАСТОСУНОК, ANDROID, MYSQL, JAVA.

Об'єктом дослідження є вдосконалення можливості доступу та пошуку необхідних даних різних категорій, а вже предметом дослідження є вже конкретно розробка застосунку призначеного для збору та пошуку інформації через бібліотеки даних з використанням можливостей сучасних IT технологій та алгоритмів.

Метою роботи – розробити універсальний застосунок призначенням якого буде пошук та збір інформації через бібліотеки даних .

Для розробки використано сучасні інформаційні технології, зокрема трирівнева модель архітектури, Android SDK, середовище Eclipse, мова програмування Java, база даних MySQL, СКБД — phpMyAdmin.

The object of the research is the improvement of the ability to access and search for the necessary data of various categories, and the subject of the research is specifically the development of an application designed to collect and search for information through data libraries using the capabilities of modern IT technologies and algorithms.

The goal of the work is to develop a universal application, the purpose of which will be the search and collection of information through data libraries.

For development, modern information technologies were used, in particular, three-level architecture model, Android SDK, Eclipse environment, Java programming language, MySQL database, DBMS — phpMyAdmin.

ЗМІСТ

ВСТУП

Безпечний, вільний, швидкий доступ в умовах військового стану – це завжди актуально, ніколи не буває лишнім і завжди це щось нове, оскільки технології розвиваються, підходи вдосконалюються, коло питань розширюється, потреби залежать від вимог. Зрозуміло, що інформація є різна, і призначення її різне, проте на суть питання це ніяким чином не впливає, тому я її і обрав для себе як основну для моєї кваліфікаційної роботи. Також на моє рішення вплинуло і те, що під час мого навчання у мене часто виникала проблема із тим, щоби визначитись з новизною моєї роботи, чи це вже хтось робив, що робив, які отримав результати, переваги чи недоліки. Також бувало, що мені не вдавалось одразу знайти потрібну інформацію, із-за чого мав проблеми.

Програмна система, що розробляється, дозволить вирішити дану проблему. За її допомогою користувач розробленого додатка отримає доступ до будь-якого сервера із базою даних віддалено, таким чином отримає доступ та можливість в будь-який момент часу при наявності навіть приватного пристрою та доступу в мережу інтернет знайти потрібну інформацію з бази з високою релевантністю запиту, обирати, виділяти, скачувати, та опрацьовувати значно спрощуючи та полегшуючи при цьому пошук. Такий пошук можна здійснювати і по бібліотекам.

На сьогоднішній день темпи розвитку мобільних платформ так само як і поява все новішого програмного забезпечення під них є одними із найінтенсивніших і найвищих посеред інших інструментів та засобів, а кількість пристроїв, які працюють під операційну систему Android чи iOS вже точно і значно перевищує кількість персональних комп’ютерів.

В представленій на розгляд екзаменаційної комісії дипломній роботі акцентовано увагу на бібліотеку, як предметну область. Враховуючи той

факт, що фактично в кожного є пристрій, і часто навіть не один, який використовує Android, наявний постійний доступ до мережі Інтернет, є можливість надати змогу їх користувачам віддалений доступ до серверів чи бібліотек з метою пошуку та опрацювання даних.

Таким чином вимальовується об’єкт дослідження - вдосконалення можливості доступу та пошуку необхідних даних різних категорій, напрямку та об’ємів, а вже предметом дослідження є вже конкретно використання сучасних інформаційних технологій, зокрема трирівнева моделі архітектури, додаток — клієнтська частина продукту, розробка базується на технології Android SDK, проводитиметься розробка в середовищі Eclipse, мовою програмування Java, .а якості бази даних обрано базу MySQL, СКБД — phpMyAdmin.

Основне завдання і перевага додатку — можливість миттєво, віддалено дізнатись про потрібну інформацію, якщо це стосується паперових оцифрованих джерел чи фоліантів взнати, чи наявні в бібліотеці потрібні користувачу примірники, при чому без необхідності очного відвідування. Важливо, що це дасть змогу користувачеві безпечно навчатись та розвиватись коли є можливість, а сьогодні, в час хоча тимчасових, але певних обмежень, це вкрай важливо та актуально.

Важливим елементом програми також є постійна актуальність даних, оскільки оновлення бази даних відбувається при кожному запуску додатку.

Окрім вище переліченого, під час виконання кваліфікаційної роботи потрібно детально проаналізувати предметну область та виявити наявні проблеми та додатки, які вже є на ринку. Також повинна бути спроєктована і розроблена база даних, а також серверна та клієнтська частина, яка і виступає в якості додатку, програмного продукту.

1 РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ

1.1 Аналіз вимог до програмного додатку

1.1.1 Аналіз предметної області

Предметна область, яку я обрав для вивчення, аналізу та визначення об’єкту дослідження в представленій записці до кваліфікаційної роботи – бібліотека даних, при чому йдеться про універсальність додатку, проте за приклад обрано бібліотеку та всі паперові джерела, які вже оцифровано чи частково оцифровано, а таких станом на сьогодні є більшість.

В результаті аналізу було виявлено основну проблему характерну для даної предметній області — відсутність реального віддаленого доступу до інформації. Тобто на словах він є, але при дійсній необхідності часто я як користувач стикався із помилками, які виникали після активаціє подібного толку додатків. Також проблемою є те що при необхідності отримати джерело, потрібно, і подекуди, і це не окремі випадки, особисто з’явитись та укласти якийсь документ. Також буває так, що запит надіслано, підтвердження отримано, а в результаті отримати інформацію не можливо. Тобто ніби маємо підтвердження, а скачати нічого неможливо, паперовий варіант також не знаходиться із якихось, найрізноманітніших причин. Буває навіть і так, що потрібно знову і знову звертатись, і це може призвести у зневіри та розуміння, що не варто витрачати час. А програмна система, що розробляється, дозволить вирішити дану проблему. За її допомогою користувач розробленого додатка отримає доступ до віддаленого сервера, на якому і знаходиться база даних з необхідною інформацією. Таким чином, користувач матиме змогу в будь-який момент часу дізнаватися про наявність у бібліотеці тих чи інших книжок, а також відмічати їх як такі, що потрібні йому, щоб пізніше звернутися до бібліотекаря із проханням видати йому книжки. Також багато часу займає власне пошук потрібних книжок серед

усієї маси, так що дана програмна система дасть змогу значно спростити його.

Проаналізувавши проблеми, наявні в предметній області, визначено, що із системою повинен працювати єдиний користувач – власник додатка, якому і потрібно отримати інформацію чи дані.

1.1.2 Постановка завдання

Попередньо я вже визначився із предметною областю, зробив огляд основних функцій і проблем і на даному етапі прийшов час оцінки завдання яке потрібно виконати для даного проєкту [1].

Отже, потрібно вирішити такі завдання:

1. Проаналізувати актуальність обраної області.

2. По можливості зробити статистичне опитування користувачів, які вже здійснюють діяльність в обраній мною предметній області [2-5].

3. Проаналізувати на предмет проблемних питань даної області.

4. Проаналізувати на предмет коректності розстановки у розробленій ER-діаграмі пріоритетності зв’язків між сутностями.

5. Розстановка коректних атрибутів до кожної із сутностей за відповідними типами даних.

6. Аналіз та перевірка на предмет повноти наведення всіх акторів присутніх в реальних бібліотеках.

7. Процес моделювання зручного інтерфейсу програми для встановленої цільової аудиторії.

8. Визначення потреби використання серверної частини програмування.

9. Передбачення можливості реєстрації нових користувачів як самостійно так і через адміністратора.

Вирішивши всі ці питання, потрібно буде розглянути реальних дійових осіб і сутності з відповідною інформацією більш детально і вже з врахуванням конкретно обраної мною теми. Отже, після аналізу предметної області, я визначився що із системою буде працювати єдиний користувач. Таким чином, система повинна забезпечувати виконання наступних функцій [6]:

1. Реєстрація нового користувача.

2. Вхід користувача в систему з обов'язковую реєстрацією та введенням паролю.

3. Виведення списку наявних джерел.

4. Виведення списку відмічених джерел для кожного окремого користувача.

5. Пошук інформації про джерела (назва, рік, автор).

6. Можливість позначення джерел.

7. Відображення усіх та їх здійснення можливостіми графічного інтерфейсу, в основному через натискання відповідних кнопок.

1.1.3 Пошук актантів, варіантів використання

Розглянувши у попередніх розділах користувачів системи та основні функції і можливості системи, перейдемо до виявлення безпосередньо актантів, а також варіантів використання [7].

Отож, у програмній системі буде єдиний користувач, який повинен мати змогу здійснювати наступні дії:

- Зареєструватися

- Увійти у систему
- Переглянути доступні джерела
- Переглянути відмічені джерела
- Відмітити потрібні джерела
- Здійснити пошук джерел
- Вийти із системи

Таким чином, діаграма варіантів використання набуває наступного вигляду (як представлено рис.1):

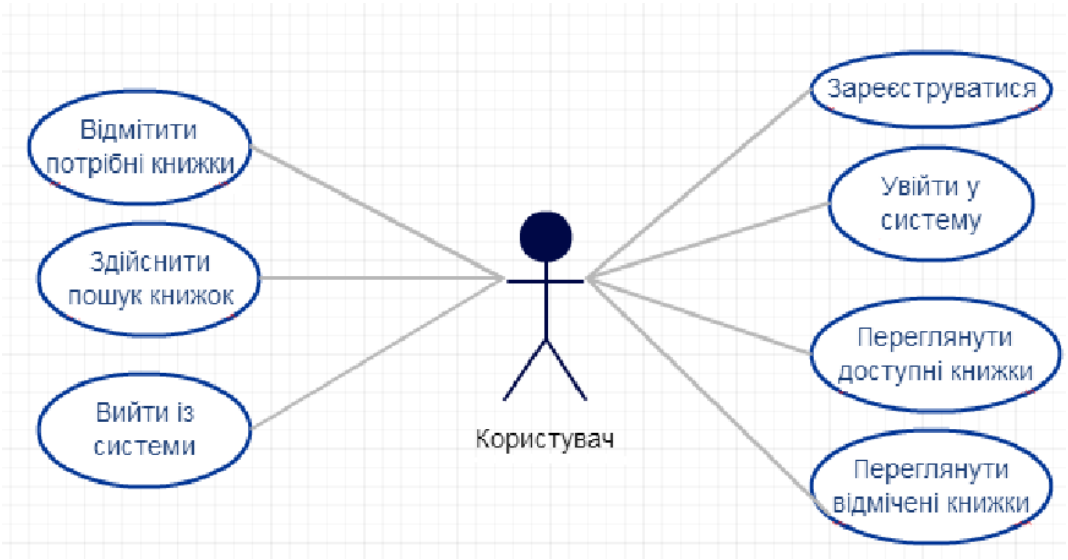

Рисунок 1 – Варіанти використання

1.2 Проєктування програмного додатку

1.2.1 Побудова схеми бази даних

Традиційно базою даних називають впорядкований набір логічно пов’язаних між собою даних, які використовуються разом та призначенням яких є задоволення інформаційних потреб користувача [7-9].

Головним завданням бази даних є ґарантоване зберігання певних часто значних обсягів інформації -записів даних, а також можливість отримати до неї доступу як для користувачів так і для прикладної програми. Отже я можу говорити про те, що база зазвичай складається із 2-ох частин: 1-безпосередньо збереженої інформації; 2-системи керування збереженою інформацією.

Також не є секретом і те, що бази даних вже є невід’ємною частиною нашого повсякдення. Їх успішно використовують у багатьох областях та галузях людської діяльності. В сучасних же умовах одною із найбільш поширених задач, яка може виникнути при користуванні комп’ютерною технікою є використання систем, які засновані на базах даних.

Сьогодні розрізняють два типи баз даних:

1-структуровані – ті, в яких використовуються структури даних, а саме вдаються до структурованого опису типу фактів при допомозі схеми даних - моделі даних. В свою чергу модель даних слугує як для опису об’єктів так і для взаємовідношень між ними;

2-не структуровані, до яких відносяться повнотекстові бази даних, в яких містяться не структуровані дані, у нашому випадку це, наприклад, статті чи книги у формі, що дозволить здійснити швидкий та адекватний запитам пошук потрібної інформації.

Ще, при вивченні літературних джерел мені натрапилась характеристика баз даних, яка ґрунтується основі певних параметрів, чи ж

необхідних вимог, для прикладу наведу значну кількість даних; їх незалежність; можливість відкритого доступу; підтримки транзакцій із гарантуванням дотриматись відповідних властивостей; забезпечення відсутності збоїв; можливість одночасної роботи із великою кількістю користувачів без втрати якості. Важливо, на мою думку, також розуміти і те, що бази даних достатньо динамічно розвиваються, тому природно що до вказаних вимог можуть і будуть додаватися нові, і саме тому про повноту цієї характеристики говорити напевно не дуже коректно [8].

В результаті аналізу, я прийняв, що в якості баз даних в у моїй роботі, я надам перевагу саме реляційному типу, як найоптимальнішому для моєго предмету розробки, оскільки в моєму випаду – швидкість пошуку не є пріоритетним.

Таким чином оскільки реляційна база даних є такою, яка базується на реляційній моделі даних, то й для роботи із нею застосовується і система керування реляційного типу. Іншими словами я матиму справу із базою, суть якої зі сторони сприйняття користувачем полягає у наборі різноступеневих нормалізованих відношень. Крім цього я матиму перевагу - швидкість та гнучкість за рахунок, того що для реляційних базі даних характерно, що збереження відбувається окремими таблицями. Зв’язок таблицями відбувається через відношення, в результаті я отримую можливість при виконанні запиту об’єднувати дані із кількох таблиць. Для запитів пропоную використати мову структурованих запитів SQL, яка є однією із найпоширеніших із тих, які застосовуються з метою отримати доступ до баз даних [9].

В результаті аналізу я зробив висновок, що Microsoft SQL, так само як і Oracle та MySQL є найпоширенішими та найпопулярнішими сьогодні [10-13], http://db-engines.com/en/ranking. Саме тому я вирішив детальніше проаналізувати їх, щоб зробити свій вибір на користь однієї їз них відповідно до вимог.

OracleDatabase чи OracleRDBMS є об’єктно-реляційною системою керівання базами даних від компанії Oracle, є потужним програмним комплексом за допомогою якого можна створювати додатки з різним степенем складності. . В ядрі цього комплексу знаходиться база даних, у якій і зберігається інформація. Важливо, що хоча інформації є великі об’єми, високоефективно працювати із нею причому одночасно може практично не обмежена кількість користувачів (однак при наявності достатньої кількості апаратних ресурсів). Цікаво, що навіть при різкому збільшенні кількості користувачів тенденції до зниження продуктивності системи не виявлено.

До переваг варта також віднести і те, що просто при додаванні нових вузлів кластера в механізмах масштабування в системі керування БД типу Oracle (остання версія) є можливість необмежено збільшувати потужність і швидкість роботи серверу Oracle та своїх додатків. Для цього не потрібно зупиняти працюючі додатки, також немає необхідності переписувати «старі» додатки, які було розроблено під звичайну одно-машинну архітектуру. Також слід зазначити, що якщо вийдуть з ладу окремі вузли кластера, це не призведе до програмної зупинки.

В цілому ж Oracle на сьогодні можна вважати де-факто за стандарт системи керування БД для Internet-у. Це стало можливим завдяки вбудовуванню до системи керунку БД Oracle JavaVM, повномасштабної підтримки серверних технологій (JavaServerPages, Java-сервлетів, модулів Enterprise Java Beans, інтерфейсів прикладного програмування типу CORBA).

Не можу не згадати і про багатоплатформенність, як ще одну із переваг СКБД Oracle, оскільки її поставляють фактично до усіх операційних систем існуючих на сьогодні.

Також плюсом є і те, що останні версії Oracle значно простіші для установки і початкового налаштування. Також покращилась можливість стосовно спеціалізованого під конкретну задачу налаштування роботи СКБД. Саме тому і під час роботи з OLTP-системою, і при роботі із сховищем для

даних, при використанні цих можливостей із налаштування СКБД Oracle, можна досягти справді хороших результатів [14].

Система керування типу Oracle зазвичай поставляється у 4-х варіантах Oracle Database Enterp.Edit., Oracle Database Stand.Edit., Oracle Database Personal Edit. та зовсім полегшений мобільний варіант, який призначено перш за все для laptop-пів. Таким чином усі варіанти серверу Oracle володіють у своїй основі одним і тим самим кодом та є функціонально ідентичними, за вийнятком деяких опцій, які для прикладу, можуть бути доступнішими лише під Oracle Database Enterp.Edit. та не поставлятися з другими варіантами систем керування [10] .

MS SQL Server вважається системою управління базами даних, яку розроблено в корпорації Microsoft. Основною використовуваною мовою для запитів є мова Transact-SQL, яку було створено спільно із компанією Microsoft та компанією Sybase. Transact-SQL вважається реалізацією для стандартів типу ANSI/ISO по структурованій мові запитів (SQL) із розширенням. Зазвичай її використовують при роботі із базами даних розмірністю від персональних і аж до великих баз даних.

Вихідний код MS SQL Ser. ґрунтувався на базі коду SybaseSQL Server, що й дозволило для Microsoft вихід на ринок БД для організацій, на якому конкурували такі компанії як Oracle,IBM,Sybase.

Найважливішими характеристиками даної системи керування базами можна вважати:

1- простоту адміністрування,

2- наявність можливості для підключення до Web,

3- швидкодія та функціональні можливості механізма серверу системи керування базами,

4- наявні засоби для віддаленого доступу.

До комплекту засобів для адміністративного керування вище названої системи керування для баз даних відносять цілий набор спеціальних

інструментів та засоби автоматичного налаштування параметрів конфігурації. Крім цього представлена база володіє засобами тиражування, за допомогою яких можлива синхронізація даних персонального комп'ютера з інформацією наявною в базі даних та навпаки. Наявний в комплектації при поставці сервер типу OLAP надає можливість зберігання та аналізу усіх наявних в користувача даних. В цілому ж дана система керування для баз представляє собою сучасну цільнофункціональну базу даних, яка, на мою думку, найкраще підходить як для малих так і для середніх організацій. Варта також зазначити і те що SQL Server дещо поступається іншим системам керування по принаймні двом важливим показникам: 1-програмованість, 2-засоби роботи. Під час розробки баз даних додатків клієнтського типу на основі таких мов програмування як Java та HTML досить часто може виникати проблема в недостатності програмних засобів SQL Server-у, в результаті чого користуватися цією системою керування БД буде значно важче, аніж системами типу DB2, Informix,Oracle чи, наприклад,Sybase. У 21 столітті загальної світової тенденції набула практично масова міграція на платформу LINUX, при цьому SQL Server може функціонувати лише у середовищі Windows. Саме тому, я вважаю, що використовувати SQL Server доцільно, лише у випадку якщо з метою доступу до вмісту БД застосовується виключно стандарт ODBC, а в іншому варіанті варта використовувати інші системи керування для баз даних [11].

MySQL вважається вільною реляційною системою керування для баз даних,розробку та супровід із підтримкою якої забезпечується корпорацією Oracle. Цей продукт поширюють як для GNU General Public License, так і для власної комерційної ліцензії. Крім того, розробниками створюється функціональність на замовлення ліцензованих користувачів, і саме завдячуючи йому у перших ранніх версіях і з'явився механізм, який називають механізмом реплікації.

Якщо говорити про гнучкість системи керувння MySQL, то вона підтримується великою кількістю типів таблиць, зокрема користувач може обирати як таблиці типу MyISAM, якими підтримується повнотекстовий пошук, так і обрати таблиці типу InnoDB, якими, на рівні окремих записів, підтримуються транзакції. Більше цього, MySQL надається разом з EXAMPLE - спеціальний тип таблиць, яким демонструються принципи створення таблиць за новими типами. Слід зазначити, що весь час з'являються нові типи таблиць, і це є можливим саме завдячуючи відкритій архітектурі та GPL -ліцензуванню.

MySQL має API для мов Delphi, C, C++, Ейфель, Java, Lisp, Perl, PHP, Python, Ruby, Smalltalk, Компонентний Паскаль і Tcl, бібліотеки для мов платформи .NET, а також забезпечує підтримку для ODBC за допомогою ODBC-драйвера MyODBC.

Максимальним розміром таблиць у MySQL є від 3.22 і до 4 ГБ, але вже для наступних версій максимальний розмір обмежується лише максимальним розміром для файлу в операційній системі.

Однак, розмір таблиці обмежується її типом. В загальному ж випадку тип MyISAM обмежується лише граничним розміром файла в файловій системі для операційної системи.

Також хочу зазначити, що на відміну від MyISAM, у InnoDB існує суттєве обмеження щодо кількості стовпців, яку можна додати до однієї таблиці. Важливо, що розмір для сторінки пам’яті по замовчуванню складає 16 кілобайт, із них під дані відведено 8123 байта. Розмір покажчика для динамічних полів становить 20 байтів. Отже, при використанні динамічного формата рядку(ROW_FORMAT=DYNAMIC), в одну таблицю можна помістити максимально 409 стовпці типу blob чи text [12].

Таким чином проаналізувавши найпопулярніші бази, я зробив висновки, що доцільно обрати для роботи MySQL так як вони володіють наступними перевагами:

- простотою при встановленні, а також під час використання;
- можливістю підтримки будь-якої кількості користувачів, які можуть одночасно працювати з базою данних;
- кількістю рядків у таблицях, яка може складати до 50 млн.;
- високою швидкістю щодо виконання команд;
- наявністю простої і разом з тим ефективної системи безпеки [12].

Варта зазначити, що щоби мати можливість управління базою даних, потрібно мати систему керування для баз даних, яка є сукупністю як програмних так і лінгвістичних засобів загального чи спеціального призначення, якими забезпечуються керування для створення і використання баз даних. Так як в кваліфікаційній роботі передбаченно, що база даних розміщатиметься віддалено на сервері, тому в якості інструменту адміністрування СКБД MySQL буде використано phpMyAdmin.

PHPMyAdmin вважається web-додатком із відкритим кодом, який написано мовою PHP і який представляє собою web-інтерфейс призначенням якого є адміністрування СКБД MySQL. За допомогою цього web-додатку можна через браузер здійснити адміністрування серверу MySQL, здійснювати запуск команди SQL і перегляд вмісту таблиць та безпосередньо баз даних. У web-розробників цей додаток є досить популярним завдяки тому, що за його допомогою є можливість керувати СКБД MySQL без необхідності безпосередньо вводити SQL-команд, за допомогою дружнього інтерфейсу.

Широке використання на практиці сьогодні PHPMyAdmin пов’язане із тим, що серед розробників існує тенденція інтенсивного просування свого продукту і з врахуванням усіх нововведень СКБД MySQL. Переважною більшістю провайдерів цей додаток використовується як панель керування з метою надавання своїм клієнтам можливості адміністрування виділених для них баз даних [13].

Модель вигляду ‘’сутність – зв’язок’’ чи якщо іншими словами ER-модель (англійською повністю Entity-relationship model) є моделлю даних, за допомогою якої можна писати схеми концептуального толку при допомозі узагальнюючих конструкцій блоків. Модель вигляду ‘’сутність -зв'язок’’ вважається мета-моделлю даних, тобто засобом для описування даних за допомогою різних моделей.

ER-модель є зручною при проєктуванні систем інформаційного типу, а також баз даних, архітектури для комп’ютерних застосунків так само як і систем(моделей). При допомозі моделі такого типу виділяються найсуттєвіші, найважливіші елементи, такі як вузли та блоки, моделі і встановлюються взаємозв’язки поміж ними.

Також існують і цілий ряд моделей призначенням яких є представлення знань. Одним із найзручніших інструментів призначенням якого є уніфіковане представлення даних, не залежно від програмного забезпечення, за допомогою якого його можна реалізувати, є ER- модель.

Модель по типу "сутність-взаємозв’язок" базується на важливій інформації семантичного типу, яка стосується реального світу призначенням якої є логічне представлення даних. За допомогою неї визначаються значення даних в контексті та їх взаємозв’язок з іншими даними. Важливо також і те, що ER- модель може породжувати усі існуючі моделі даних (ієрархічну, мережеву, реляційну та об'єктну), і саме тому вона і є найбільш загальною. В загальному ж будь-який фрагмент реальної області можна представляти у вигляді безлічі сутностей, між якими і існує певна безліч взаємозв’язків [15,16].

На даному етапі кваліфікаційної роботи прийшов час розглянути сутності, які будуть потрібні для побудови таблиць бази даних.

1. Джерело інформації

   Id – для пошуку

   Name – назва джерела інформації

Author – автор

Year – рік видання, розміщення

LibraryId – унікальний номер в бібліотеці/базі

Description – короткий опис/анотація

Required – логічне значення, доступне джерело інформації в даний момент чи ні.

2. Користувач

Id – для пошуку

Name – ім’я користувача

Surname – прізвище

Patronym – по-батькові

Group – група

Також додам сутність, яка відповідатиме таблиці із логінами та паролями кожного користувача

Логін/пароль

Id – унікальний ідентифікатор

Username – логін, унікальне значення

Password – пароль

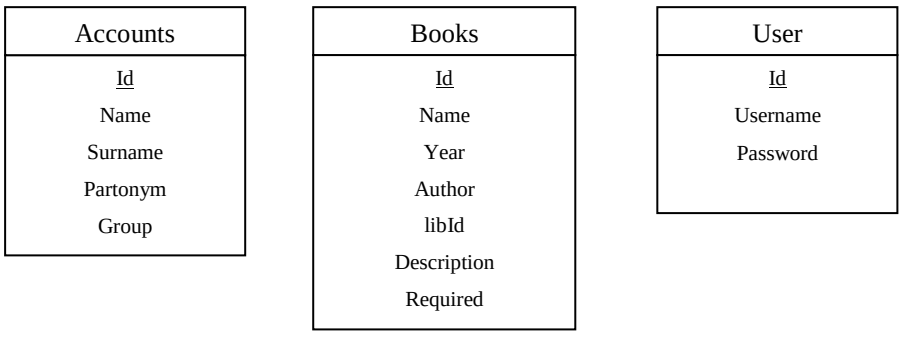

Рисунок 2 – Представлення сутностей

Далі я виявив присутні взаємозв’язки між цими сутностями.

Спочатку розгляну зв’язок між користувачами і джерелами інформації. Оскільки один користувач може мати багато джерел інформації, то тут присутній зв’язок по типу один до багатьох.

Зв’язок між користувачами і таблицею логінів/паролів — один до одного, оскільки єдиному користувачу належить єдине значення логіна і пароля і навпаки.

Між джерелами інформації і таблицею логінів ніяких зв’язків виявлено не було.

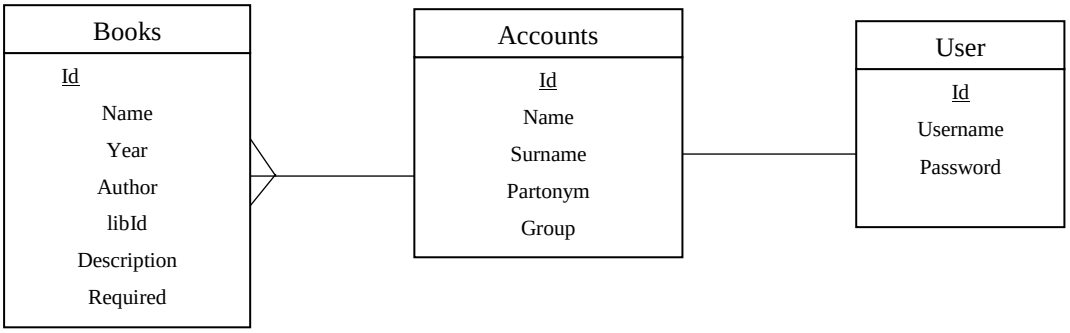

Рисунок 3 – Представлення існуючих взаємозв’язків поміж сутностями

Отже, мною отримано усі наявні сутності і взаємозв’язки поміж ними, а це значить, що я можу переходити до наступного етапу яким є власне побудова бази даних.

Базу даних було створено за допомогою phpMyAdmin версії 3.5.2.2. Система збереження даних — MyISAM. MyISAM — одна з основних (поряд з InnoDB) систем зберігання даних в СУБД MySQL. Вона ґрунтується на принципах ISAM і володіє в порівнянні з ним поруч корисних доповнень. Підтримується з версій MySQL 3.x, до версій MySQL 5.5 була системою зберігання за замовчуванням. Таблиці MyISAM прекрасно підходять для використання в невеликих інтернет-проектах (WWW) та інших середовищах, де переважають запити на читання і немає жорстких вимог до надійності.

Відповідно до схеми сутностей та зв’язків поміж ними я створив наступні таблиці:

User:

- Id – primary key, унікальне значення, автоінкремент.
- Username – стрічка змінної довжини, унікальне значення для збереження логінів користувачів.
- Password – стрічка змінної довжини, використовується для збереження паролів.

Accounts:

- Id – primary key, унікальне значення, автоінкремент.
- Name – ім’я користувача
- Surname – прізвище
- Patronym – по-батькові
- Group – група

Books:

- Id – primary key, унікальне значення, автоінкремент
- Name – назва джерела
- Author – автор джерела
- libId – унікальне значення, бібліотечний номер джерела інформації
- Year – значення типу YEAR, рік видання/розміщення у доступі
- Description – значення типу TEXT, короткий опис інформації
- Required – логічне значення, представлене у вигляді TINYINT(1), приймає значення 0 або не 0, визначає, чи доступне на даний момент джерело інформації.

Також вирішено додати проміжну таблицю user_has_books, яка міститиме значення id користувачів і відповідні унікальні значення id книжок, що взяті даними користувачами.

Таким чином, остаточна схема таблиць та зв’язків поміж ними набуває такого вигляду.

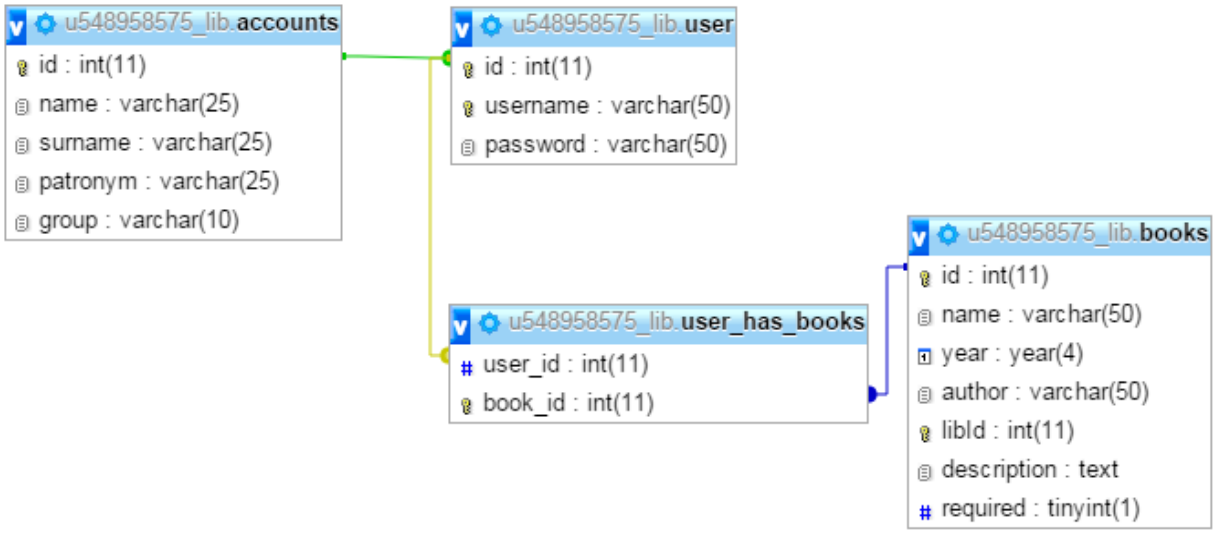

Рисунок 4 – Схема бази даних

1.2.2 Побудова діаграми класів

Діаграмою класів або UML діаграмою називається представлення у вигляді діаграми, якою демонструються класи в системі, їх атрибути, методи та взаємозв’язок поміж ними [16,17].

Розроблювана програмна система міститиме наступні класи — це класи, що відповідають за поведінку кожного з екранів додатка та їх елементів, а також клас, за допомогою якого відбуватиметься отримання даних із сервера.

Класи, що керують елементами екрана, в якості базового мають клас Activity, а від класу AsyncTask наслідуватиметься клас взаємодії з сервером.

Кожен клас також матиме методи, за допомогою яких обробляється взаємодія із елементами екрана і методи, що реалізують основний функціонал програми.

Таким чином, діаграма класів матиме наступний вигляд:

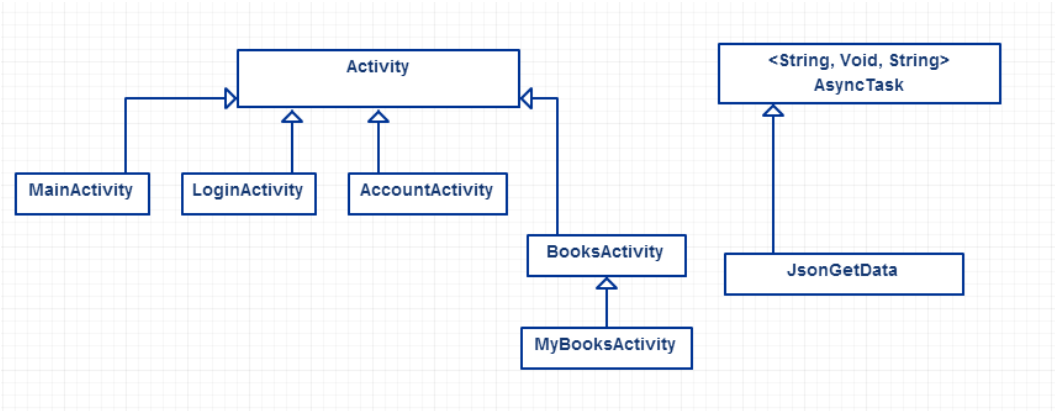

Рисунок 5 – UML-діаграма класів

1.2.3 Моделювання архітектури застосунку

Для написання програмного продукту я обрав архітектуру трьохрівневого типу, яка передбачає певні програмні компоненти, зокрема: 1-клієнт (безпосередньо продукт який я розробляю) та 2-сервер, на якому виконуються PHP-скрипти. Він підключений до безпосередньо серверу бази даних.

На першому рівні знаходиться розроблюваний додаток, в якому не передбачено напряму зв’язку із базою даних, та яким передбачено лише найпростішу бізнесову логіку. Передбачено авторизацію (перевірку даних), які вносяться, на предмет допустимості та операції із вже залитими даними.

Сервер, як і більша частина бізнесової логіки, розташовані на наступному 2-му рівні.

Збереження даних на сервері баз даних підтримується та виноситься вже до останнього 3-го рівня. Як правило це традиційна реляційна чи може бути і об’єктно-орієнтована система керування для баз данних.

В своїй роботі я обрав трьохрівневу архітектуру, оскільки вона володіє рядом переваг в порівнянні із архітектурами типу клієнт-сервер або файл-сервер, зокрема наявність:

- масштабованості;
- конфігурованості -ізольованості рівнів один від одного, що, у випадку правильного розгортання архітектури, дає змогу достатньо швидко та при використанні простих засобів здійснити переконфігурацію системи у випадку виникнення збоїв або при необхідності планового обслуговування на одному із рівнів
- високого рівня безпеки;
- високого рівня надійності;
- низьких вимог до швидкості мережі поміж терміналами та сервером для застосунків;
- низьких вимог щодо продуктивності а також відносно технічних характеристик терміналів, в результаті зниження їх вартості. В якості терміналу може слугувати не лише комп’ютер, але й, для наприкладу, мобільний телефон [19].

1.3 Конструювання програмного додатку

1.3.1 Вибір засобів, мови та середовища для розробки

За своїм призначенням Android є операційною системою призначеною для смартфонів, комп’ютерів планшетного типу, електронних книг, розумих годинників, ігрових додатків, сучасних телевізорів та інших сучасних пристроїв. Додатками призначеними під операційну систему Android є програми з нестандартним байт-кодом. При порівнянні зі звичайними додатками під Linux додаткам Android характерні додаткові правила,

зокрема: 1-Content Providers; 2-Resource Manager; 3-Notification Manager; 4-Activity Manager.

Google у вільному доступу надається інструментарій Software Development Kit призначений для розробки який підходить x86-машин під ОС типу Linux,Mac OS X (10.4.8 і вище), WindowsXP, WindowsVista і Windows 7 та вище. Також для розробки необхідний JDK.

Отже, відомо, що для розробки додатків під Android можна використати мову програмування Java (версія не нижче 1.5). Наявним плагіном для Eclipse є Android Development Tools (ADT), основним призначенням якого є Eclipse версій 3.3-3.7. Так само існує і плагін під IntelliJ IDEA, за допомогою якого полегшується розробка Android-додатків. Крім цього ще існує Motodev Studio під Android є комплексним середовищем на базі Eclipse для розробки, яке робить можливим роботу безпосередньо із Google SDK.

Ще в 2013 році Google-ом було представлено нове середовище для розробки і це Android Studio, засноване на IntelliJ IDEA від JetBrains.

В тому ж 2013 році також відбувся і реліз Embarcadero RAD Studio-XE5. Надає можливість для розробки нативних додатків призначених для платформи Android. Безпосередньо сам процес створення Android-програм не потребує якихось додаткових пристроїв, окрім Android пристроїв (достатньо емулятора).

В Android-а версії 4.4 появилась можливість для зміни віртуальної машини Dalvik на AndroidRuntime, перевагою якого є підвищена швидкість для завантаження програми. В Android-і версії 5.0 вибір машини зневілювався, оскільки замість Dalvik почала використовуватись AndroidRuntime [20].

Для написання додатка насамперед потрібно вибрати середовище розробки. Аналізуючи наявні, вважаю, що основними середовищами розробки на сьогодні вважаються Android Studio від Google, та Eclipse, для

якого встановлють плагін Android Development Tools(ADT). Таким чином, проаналізувавши, для розробки мого програмного продукту я обрав середовище Eclipse Juno, версії 4.2.1. Сам додаток буде написано мовою програмування Java [21-23].

Отже, для моєї розробки необхідно встановити набір для розробки програмного забезпечення AndroidSDK, до якого входять бібліотеки, документація та інструменти, при допомозі яких розробляються мобільні додатки призначені під платформу Android.

API Android SDK вважається API бібліотеками Android, прийначенням яких є розробка додатків. Рівень API є цілочисельним числом, яким однозначно визначається версія API платформи під Android. Для розробки додатку за цільову платформу взято API 21-Android 5.0, а в якості мінімальної платформи, на якій можливо реалізувати запуск є API8(Android). До документації SDK включається велика довідкова інформація, за допомогою якої деталізується, що саме включено до кожного пакету і класу і як це можна використати при розробці додатків.

AVD (Android Virtual Device) — інтерактивний емулятор мобільного пристрою Android. Використовуючи емулятор, можна запускати і тестувати програми без використання реального пристрою Android.

Development Tools — SDK включає кілька інструментальних засобів для розробки, які дозволяють компілювати і налагоджувати створювані додатки.

Sample Code — Android SDK надає типові додатки, які демонструють деякі з можливостей Android, і прості програми, які показують, як використовувати індивідуальні особливості API в вашому коді.

Для того, щоб мати можливість запускати додатки, знадобиться емулятор пристрою. SDK включає в себе менеджер віртуальних пристроїв AVD, З його допомогою можна вказати версію SDK, дозвіл екрана, ємність SD-карти, апаратні можливості (сенсорний екран, GPS). Але на даний момент

цей емулятор має доволі серйозний недолік — через те, що він повністю емулює середовище Android, час його запуску є доволі великим. Також існують такі емулятори, як Genymotion, що представляє собою фактично віртуальну машину на VirtualBox, або Android x86 — також віртуальна машина, але з незручним доступом до adb [24].

Врешті, вибір зупинився на BlueStacks — так званому «плеєрі додатків» для Windows чи Mac. Головна його особливість — він не емулює повноцінний пристрій, а лише надає потрібний функціонал для запуску Android-додатків, через що є доволі швидким у використанні.

Для керування версіями SDK можна використовувати менеджер SDK через панель, доступну в меню Window → Android SDK Manager. Ви можете переглядати встановлені версії SDK, а також оновлювати їх у міру потреби. При виборі пункту Availiable Packages ви можете перевіряти репозиторій на наявність доступних, але ще не невстановлених пакетів та архівів. Помітьте прапорцем необхідні файли для завантаження і скачайте потрібні пакети.

ADB є клієнт-серверним додатком, за допомогою якого отримується доступ до працюючого емулятору чи пристрою. З ним отримується можливість копіювання файлу, встановлення скомпільованих програмних пакетів і запуску консольних команд. Використовуючи консоль є можливість змінити налаштування журналу а також взаємодіяти із базами даних SQLite, збереження яких забезпечується пристроєм [24].

Основними компонентами Android є:

– Activity – являє собою екран для користувача інтерфейсу, який представляється через клас Activity, а також розмітка у вигляді XML-файлу. Варта зауважити, що розмітка можна створюватися і програмно, проте сьогодні це не вважається сучасним. Android-додаток може складатися з кількох форм-Activity, між якими він може переключатися поміж ними під час використання додатку;

– Intents представляється асинхронними повідомленнями, за допомогою яких є можливість подати запит на функції від інших служб чи дій. Додатоком можуть робитися прямі запити на службу чм дію (т.з. явний намір) чи запити через Android до зареєстрованих служб та додатків (т.з. неявний намір).
– Views – користувальницький інтерфейс, який створюється за допомогою віджетів класів, до якого входять такі елементи керування як: кнопки, текстові поля, прапорці, перемикачі та інші.
– Services – для виконання фонових завдань без надання користувацького інтерфейсу (наприклад, як варіант це програвання музики). Ними можуть повідомлятися користувачі за допомогою системи повідомлень Android.
– ContentProvider – надаються дані для додатків при допомозі контент-провайдеру. Додаток має можливість обмінюватися з другими додатками даними.
– BroadcastReceiver – беруться системні повідомлення і неявні, запити через Android до зареєстрованих служб та додатків, наміри. Можуть використовуватись для реакції на зміну стану самої системи. Додаток може реєструватись в якості приймача певних деяких подій і може запускатись, якщо така подія все таки настане [25-29].

1.3.2 Серверна частина

Хо́стингом або hosting називається послуга, за допомогою якої надається дисковий простір призначений для розташування фізичної інформації на сервері, яка весь час знаходиться у в мережі такій як Internet.

Як правило, під поняттям послуги «хостинг» говорять про по мінімуму про таку послугу яка передбачає розташування на сервері файлів сайта, на якому запущено програмне забезпечення, яке є необхідним при обробці запитів до цих файлів (web-сервер). Зазвичай, до послуг хостинга уже входить отримання місця під поштову кореспонденцію, бази даних, DNS файлового сховища та інше, так само як і підтримка функціонування відповідних сервісів, проте їх можна надавати і окремо. А в загальному хостинги бувають безкоштовні та платні [30].

Як і описано в додатку А, база даних, що була створена і описана в попередньому розділі, повинна розміщуватися на віддаленому сервері. Для того, щоб здійснити це, використовуватимемо безплатний хостинг, щоб розмістити на сервері базу даних, а також скрипти на мові PHP, за допомогою яких відбуватиметься обмін із базою даних.

На даний час діє велика кількість хостингових компаній, які пропонують свої послуги. Завданням будь-якої хостинг-компанії є надання доступу до обладнання чи до даних клієнта через Internet-мережу. За основні характеристики, якими визначається технічна якість хостингової послуги варта віднети:

– Постійний та безперебійний доступ до Uptime і або іншими словами до вузлів мереж;

– пов’язаність через глобальну мережу (доступ з других сегментів Internet-у).

– забезпечення та надання як фізичної так і інформаційної безпеки для даних а також працюючого обладнання;

– доступність достатньої кількості серверних ресурсів для забезпечення безперебійного функціонування сайту. Стосується випадків віртуального хостинга та VDS);

– забезпечення та підтримка безпечної експлуатації та дотримання необхідних для цього умов [30].

В якості хостингу у своїй роботі я використав serversfree.com. Серед усіх варіантів його виділяють такі основні переваги:

- безплатність;
- відсутність реклами;
- до 10 ГБ вільного місця;
- FTP;
- cPanel;
- установник PHP-скриптів.

FileTransfer Protocol (загальноприйняте скорочення FTP) або в перекладі звучить як «протокол призначенням якого є передача файлів» вважається за стандартний протокол призначеннням якого є передача файлів черех TCP-мережі, такі як для прикладу Internet. В FileTransfer Protocol-і використовується 21-й порт. Протоколом передачі даний досить часто користуються для завантаження web-сторінок та другий документів [31].

Отже, протокол побудовано в використанням «клієнт-серверної» архітектури з використанням різних мережевих з'єднань призначенням яких є передача команд та даних поміж клієнтом та сервером. Користувачі FTP-протоколу призначенням якого є передача файлів можуть проходити аутентифікацію шляхом передачі логіну і паролю відкритим текстом, чи, якщо це дозволяється сервером, можуть підєднуватися без ідентифікації, тобто анонімно. Також є можливість використати протокол SSH, з метою забезпечення безпечної передачі, за допомогою якого можна приховати (шифруванням) логін та пароль, тобто через шифрування вмісту [32].

Після реєстрації і отримання адреси хоста(в нашому випадку це http://library.bugs3.com) можна завантажувати на сервер файли. Для цього використовуватимемо FTP-клієнт FileZilla, який підключається до сервера за іменем хоста, користувача, пароля і порта через FTP.

Тепер же розглянемо власне скрипти, за допомогою яких і обмінюватимемося даними із БД.

Скрипти для керування БД будуть написані мовою PHP.

PHP (англійською скорочено PHP:HypertextPreprocessor-«PHP: препроцесор гіпертексту) вважається скриптовою мовою із загальним призначенням, яку досить інтенсивно використовують з метою розробки web-додатків. Станом на сьогодні скриптова мова підтримується більшістю хостингових провайдерів та займає лідируючу позицію посеред мов, призначенням яких та які застосовують з метою створити динамічні web-сайти. В напрямку web-програмування, а саме коли йдеться про серверну частину, PHP є однією із найпопулярніших сценарних мов (заразом із JSP,Perl та мовами, які використовуються у ASP.NET) завдячуючи тому, що вона є простою, володіє швидким виконанням, багатою функціональністю, багатоплатформеністю та розповсюдженням початкових кодів на базі ліцензії від PHP.

Варта означити, що визначальним в популярності напрямку побудови web-сайтів є наявність достатньо значного набора вбудованих засобів основним призначенням яких є розробка web-додатків. Зокрема, до таких засобів варта віднести:

- автоматизоване вилучення POST та GET параметрів, так само як і змінні оточення web-сервера до зумовлених масивів;
- взаємодію із з великою кількістю різноманітних систем керування для баз даних (My SQL,My SQLi, SQLite, Postgre SQL, Oracle (OCI8),Oracle, MicrosoftSQL Server, Sybase, ODBC,mSQL, IBM DB2, Cloudscape таApache Derby, Informix, OvrimosSQL, Lotus Notes , DB++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird/InterBase, ParadoxFileAccess, Max DB, ІнтерфейсPDO);
- автоматизовану відправку заголовків типу HTTP;
- роботу із авторизацією типу HTTP;
- роботу з cookies-ами та сесіями;

– роботу із локальними та віддаленими файлами,сокетами;
– обробку файлів, які «заливаються» на сервер;
– роботу ізXForms[33].

Тепер настав час перейти безпосередньо до етапу написання даних скриптів.

В першу чергу я створив скрипт, в якому знаходитимуться значення, необхідні для при’єднання до бази даних.

Лістинг 1 – Файл config.php

```
<?php
define("DB_HOST", "mysql.serversfree.com");
define("DB_USER", "u548958575_root");
define("DB_PASSWORD", "123456");
define("DB_DATABASE", "u548958575_lib");
?>
```

Тепер реалізовуватимемо скрипти, які виконуватимуть SQL-запити до бази даних з метою реалізації функцій додатка.

Лістинг 2 – Вибірка всіх користувачів із бази даних

```
<?php
require_once('./config.php');
$con=mysql_connect(DB_HOST,DB_USER,DB_PASSWORD)or
die("cannot connect");
mysql_select_db(DB_DATABASE)or die("cannot select DB");
$sql = "select * from user";
$result = mysql_query($sql);
$json = array();
if(mysql_num_rows($result)){
while($row=mysql_fetch_assoc($result)){
$json['user'][]=$row;
}
}
mysql_close($con);
echo json_encode($json);
?>
```

Тут я підключаюсь до бази даних, вказуючи значення із попереднього скрипта, присвоюю змінній sql значення запиту, виконую його функцією mysql_query, після чого створюю масив json і в циклі записую в нього всі

рядки вибірки з бази даних. My sql _ close замикає під'єднання до БД, а тоді командою echo виводиться масив, кодований у JSON. Даний скрипт отримує логіни і паролі користувачів, а вже у додатку шукаються співпадіння між введеними користувачем даними і вибраними із БД.

Якщо ж користувач реєструється в системі, то потрібно навпаки, вставити дані в таблицю. Це виконується у наступному скрипті:

Лістинг 3 – Реєстрація користувача

```
$username = $_POST['username'];
$password = $_POST['password'];
$sql = "INSERT INTO `user`(username, password) VALUES
('$username','$password')";
$result = mysql query($sql);
```

У дві змінних username і password методом POST записуються дані, які відправляються із клієнтського додатка, після чого відбувається вставка цих даних у таблицю.

Реєстрація також передбачає вказування своїх персональних даних, для чого створено схожий скрипт.

Лістинг 4 – Вставка персональних даних в таблицю користувачів

```
$name = $_POST['name'];
$surname = $_POST['surname'];
$patronym = $_POST['patronym'];
$group = $_POST['group'];
$sql = "insert into `accounts` (name, surname, patronym,
`group`) values ('$name', '$surname', '$patronym', '$group')";
$result = mysql_query($sql) or die (mysql_error());
```

Слово group в запиті узяте в лапки, щоб відрізняти назву поля у таблиці від ключового слова SQL.

Для зручності користувача у додатку я відображатиму його ім'я та прізвище, які братиму із таблиці його персональних даних за допомогою наступного коду:

Лістинг 5 – Вибірка із таблиці користувачів

```
$sql = "select id, name, surname from accounts";
$result = mysql_query($sql);
$json = array();
if(mysql_num_rows($result)){
while($row=mysql_fetch_assoc($result)){
$json['accounts'][]=$row;
}
}
echo json_encode($json);
```

Аналогічно вибірці з таблиці логінів, вибираю ідентифікатор, ім’я та прізвище і вивожу у JSON.

Далі, користувач повинен мати змогу переглядати усі джерела, наявні в бібліотеці, тому наступний скрипт вибиратиме дані про джерела із таблиці і виводитиме їх у JSON.

Лістинг 6 – Вибірка джерел

```
mysql_query("set names utf8") or die(mysql_error());
$sql = "select * from `books` where required = 0";
$result = mysql_query($sql);
$json = array();
if(mysql_num_rows($result)){
while($row=mysql_fetch_assoc($result)){
$json['books'][]=$row;
}
}
echo json_encode($json);
```

Щоб була можливість розпізнавання символів кирилиці, виконую set names utf8, а у самому запиті вибірки вказую умову, що джерело повинно бути не зарезервованим (required = 0, в SQL логічне значення представляється типом tinyint, що приймає значення 0 або не 0).

Основна можливість користувача — зарезервовувати для себе джерела, буде реалізована наступним кодом:

Лістинг 7 – Резервування джерел

```
$userId = $_POST['userId'];
$id = $_POST['id'];
$sql = "INSERT INTO `user_has_books`(`user_id`, `book_id`)
VALUES ('$userId', '$id')";
$out = "update books set required = 1 where id = '$id'";
mysql_query($sql) or die (mysql_error());
mysql_query($out);
```

Тут із додатка передаються дві змінних — id користувача та id вибраних джерел, а запит вставляє ці значення у проміжну таблицю, що служить для відображення того, який користувач які джерела для себе відмітив. Тут же ще одним запитом змінюється значення поля required на 1, а це значить, що запит вибірки джерел уже не відображатиме її як доступну.

І, нарешті, користувач повинен мати змогу переглянути відмічені джерела.

Лістинг 8 – Перегляд відмічених джерел

```
mysql_query("set names utf8") or die(mysql_error());
$id = $_GET['userId'];
$sql = "select books.id, books.name, books.author,
books.libId, books.year, books.description from books join
user_has_books on user_has_books.book_id = books.id
join user on user_has_books.user_id = user.id where user.id
= '$id'";
$result = mysql_query($sql) or die (mysql_error());
$json = array();
if(mysql_num_rows($result)){
while($row=mysql_fetch_assoc($result)){
$json['books'][]=$row;
}
}
echo json_encode($json);
```

Оскільки тут в одному запиті потрібно одночасно і отримати дані від клієнта, і вивести їх, то використаю метод GET для отримання id користувача і надалі виберу із таблиці джерел ті, чиї id співпадають із id в проміжній таблиці user_has_books , а id користувача в цій же таблиці співпадає із отриманим від клієнта значенням.

Таким чином, дані скрипти реалізовують усі потрібні дії із базою даних. У наступному розділі пропоную розглянути розробку самого додатка, і як додаток взаємодіє із сервером.

1.3.3 Клієнтська частина, розробка додатка

Для зчитування даних із сервера пропоную використати JSON. Також, щоб передавати дані, я використав AsyncTask. За допомогою цього класу я отримав змогу здійснювати так звані фонові операції та публікувати результати в UI потоці. Щоб використовувати AsyncTask, потрібно створити власний клас, який наслідуватиметься від AsyncTask і в ньому реалізувати потрібні методи. Яи реалізую два методи цього класу, doInBackground, який виконується в новому потоці, і в якому виконуватиметься основна робота, та OnPostExecute, що виконується після завершення роботи doInBackground. Щоб запустити таск, створюю об’єкт власного класу і викликаю метод execute.

В doInBackground я створюю HTTP клієнт, HTTP post, в який при виклику таску передаватиметься URL скрипта, response, що інкапсулює відповідь HTTP, а також записую в оголошену раніше стрічку за допомогою написаномо методу inputStreamToString вміст відповіді.

OnPostExecute виконається після виконання doInBackgroung, у ньому викликатимуться методи, що використовують JSON, оскільки це потрібно робити лише після виконання таску, так як напряму методи цього класу викликати не рекомендується.

Лістинг 9 – Клас для роботи з AsyncTask

```
private class JsonGetData extends AsyncTask<String, Void,
String> {
     @Override
     protected String doInBackground(String... params) {
     HttpClient httpclient = new DefaultHttpClient();
     HttpPost httppost = new HttpPost(params[0]);
             try {
     HttpResponse response = httpclient.execute(httppost);
jsonResult = inputStreamToString(
     response.getEntity().getContent()).toString();
             }
     catch (ClientProtocolException e) {
              e.printStackTrace();
             } catch (IOException e) {
              e.printStackTrace();
             }
             return null;
            }
private StringBuilder inputStreamToString(InputStream is) {
          String rLine = "";
          StringBuilder answer = new StringBuilder();
          BufferedReader rd = new BufferedReader(new
          InputStreamReader(is));
             try {
              while ((rLine = rd.readLine()) != null) {
               answer.append(rLine);
              }
             }
             catch (IOException e) {
               e.printStackTrace();             }
             return answer;          }
            @Override
            protected void onPostExecute(String result) {
     try {
          getUser();
     } catch (JSONException e) {
          e.printStackTrace();
     }
            }
                }
```

Запускатиметься виконання таску зразу ж при початку роботи з додатком, для цього метод execute треба викликати у onCreate, який виконується зразу при створенні активіті.

Зразу ж при запуску додатка першим екраном, що показуватиметься користувачу, є екран входу в систему. На екрані відображаються поля для вводу, значення з яких після натиснення кнопки входу порівнюватимуться із тими, що ми отримуємо із сервера, і при співпадінні відбуватиметься вхід.

Лістинг 10 – Отримання і перевірка логіна/пароля

```
private void getUser() throws JSONException{
     JSONObject jsonResponse = new JSONObject(jsonResult);
     JSONArray jsonMainNode = jsonResponse.optJSONArray("user");
for (int i = 0; i < jsonMainNode.length(); i++) {
     JSONObject jsonChildNode = jsonMainNode.getJSONObject(i);
     int id =  jsonChildNode.optInt("id");
                    ids.add(String.valueOf(id));
     String name = jsonChildNode.optString("username");
                    usernames.add(name);
     String pass = jsonChildNode.optString("password");
                    passwords.add(pass);
     }
     }
private void checkUser(){
     for(int i = 0; i < usernames.size(); i++){
     if(usernames.get(i).equals(login.getText().toString()))
                   {
     if(passwords.get(i).equals(password.getText().toString())){
                     userId = ids.get(i);
Intent intent = new Intent(this, MainActivity.class);
                      intent.putExtra("id", userId);
                      startActivity(intent);
                      }
                      break;
                  }
else
     Toast.makeText(getApplicationContext(), "Невірний логін або
пароль",Toast.LENGTH_LONG).show();
     }
          }
```

Створюється JSON об’єкт, за допомогою якого я отримую дані в JSON масив за допомогою методу optJSONArray, передаючи параметром ім’я масиву в JSON на сервері. Після цього ми проходимо через цей масив, і за допомогою optInt чи optString отримуємо значення, що відповідають аналогічним полям таблиці у базі даних на сервері і додаємо ці значення у відповідні списки ArrayList.

Метод checkUser викликається при кліку на кнопку входу і перевіряє, чи відповідають кожен по порядку логін та пароль тим, що введені у полях вводу. При успішній перевірці я отримую значення id користувача, яке знадобиться при подальшій роботі для ідентифікації і запускаю наступне активіті за допомогою механізму намірів(Intent), одночасно передаючи в нього значення id.

Якщо ж користувач не зареєстрований у системі, то існує механізм реєстрації, який відправлятиме введені користувачем дані на сервер, де раніше описаний скрипт отримуватиме їх і вставлятиме у базу даних.

Лістинг 11 – Реєстрація

```
private void registerUser() throws ClientProtocolException,
IOException{
     if(validateLoginPass()){
     new Thread()
    {
        @Override
        public void run()
        {
      HttpClient regclient = new DefaultHttpClient();
        HttpPost regpost = new
HttpPost("http://library.bugs3.com/library/register.php");
        ArrayList<NameValuePair> nameValuePairsList = new
ArrayList<NameValuePair>();
     nameValuePairsList.add(new BasicNameValuePair("username",
regLogin.getText().toString()));
     nameValuePairsList.add(new BasicNameValuePair("password",
regPass.getText().toString()));
     try {
          regpost.setEntity(new
UrlEncodedFormEntity(nameValuePairsList));
     } catch (UnsupportedEncodingException e1) {
          e1.printStackTrace();
     }
         try {
               regresponse = regclient.execute(regpost);
          } catch (ClientProtocolException e) {
               e.printStackTrace();
          } catch (IOException e) {
               e.printStackTrace();
          }
        }
}.start();
Intent i = new Intent(this, AccountActivity.class);
startActivity(i);
}
     }
```

Даний метод відправляє на сервер значення, що знаходяться в списку із об’єктів типу ім’я-значення, де іменем є назва поля в базі даних, а значенням — текст із полів вводу, який потрібно у цю базу даних вставити. Для відправки я, після перевірки значень на правильність, запускаю новий

потік, у якому і звертаюсь до сервера за адресою скрипта, що і виконує дії над базою даних. Після цього відбувається перехід на нове активіті, де користувач вказуватиме свої персональні дані і аналогічно наведеного вище способу ці дані відправлятимуться на сервер.

Насамперед при переході у головне активіті потібно отримати значення id користувача. Це робиться наступним чином:

```
Intent i = getIntent();
userId = i.getStringExtra("userId");
```

Оскільки перехід в дане активіті відбувається тільки із попереднього, і там прямо вказується MainActivity.class, то методом getIntent отримуємо інтент і за значенням ключа отримуємо стрічку із значенням.

Для зручності в полі Title екрана відображатимемо ім’я та прізвище користувача. Для цього мені потрібно отримати ці значення із бази даних.

Лістинг 12 – Відображення імені та прізвища

```
JSONArray jsonMainNode = jsonResponse.optJSONArray("accounts");
... ...
     String id = jsonChildNode.optString("id");
      String name = jsonChildNode.optString("name");
      String surname = jsonChildNode.optString("surname");
      String result = name + " " + surname;
      if(id.equals(userId)){
      setTitle(result);
     break;
}
```

Тут використовується метод setTitle, що встановлює заголовок для активіті, в якому викликається. Може викликатися так, як вказано у лістингу, або ж через ключове слово this.

Активіті містить кнопки що відповідають за перехід в інші активіті, де реалізовано основний функціонал додатка. Для обробки натискання кнопок імплементується інтерфейс OnClickListener і реалізується метод onClick.

Лістинг 13 – Перехід в інші активіті

```
allBooksButton = (Button) findViewById(R.id.allBooksButton);
allBooksButton.setOnClickListener(this);
myBooksButton = (Button) findViewById(R.id.myBooksButton);
myBooksButton.setOnClickListener(this);
exitButton = (Button) findViewById(R.id.exitButton);
exitButton.setOnClickListener(this);
.....................
@Override
public void onClick(View v) {
     switch(v.getId()){
     case R.id.allBooksButton:
          Intent i = new Intent(this, BooksActivity.class);
          i.putExtra("userId", userId);
          startActivity(i);
          break;
     case R.id.myBooksButton:
     Intent in = new Intent(this, MyBooksActivity.class);
          in.putExtra("userId", userId);
          startActivity(in);
          break;
     case R.id.exitButton:
          System.exit(0);}}
```

Параметром методу, як фактично і у всіх методів, що реалізовують лістенери, виступає View — в Android це всі елементи управління, такі як кнопки, перемикачі, радіокнопки, поля для вводу чи відображення тексту, кожному з яких відповідає власний клас. Іншим способом обробляти клік є створення нового лістенера в методі setOnClickListener і там реалізовувати onClick. Тут же при кліку отримується id натиснутого елемента і, якщо в класі R є таке значення виконується код. Клас R в Android — статичний клас, що містить такі ж статичні класи, як id, layout, drawable, у яких знаходяться статичні константні значення унікального id кожного елемента.

Тепер розглянемо інші екрани, де містяться методи, що реалізують основний функціонал додатка.

Насамперед, при переході в активіті, що відображає наявні джерела ці самі джерела я відображатиму у listView. Раніше описаний скрипт робить вибірку усіх джерел із бази даних, а у додатку ми отримаємо ці дані у JSON. Для того, щоб дані відобразити у listView, потрібно використати адаптер, що

одним із параметрів приймає список відображень, у якому і містяться дані про джерела. Тепер створю метод, що створюватиме потрібне відображення, яке потім можна додавати у список.

Лістинг 14 – Створення HashMap

```
public HashMap<String, String> createBook(String key, String
value) {
HashMap<String, String> map = new HashMap<String, String>();
               map.put(key, value);
          return map;                                                    }
```

Також у мене буде власне список для даних, зокрема

```
List<Map<String, String>> allBooks = new ArrayList<Map<String,
String>>();
```

і надалі я працюватиму саме із цим списком.

Після отримання даних, аналогічно з вищеописаним способом отримання логінів/паролів, додаю всі джерела у список:

Лістинг 15 – Створення списку джерел і відображення їх у listView

```
     String output = " \" " + name + " \" " + " \n " + author +
", " + year;
.............
     allBooks.add(getData.createBook("books", output));
     SimpleAdapter adapter = new SimpleAdapter(this, allBooks,
android.R.layout.simple_list_item_multiple_choice,
     new String[]{"books"}, new int[]{android.R.id.text1});
          listview.setAdapter(adapter);
```

Для того, щоб listView міг відобразити наші дані, присвоюю йому адаптер, в якому вказую контекст(this), власне дані, а також layout, що відповідає за тип списку. В даному випадку було вирішено вибрати простий список із можливістю множинного вибору, так як це знадобиться у подальшому для відмічання бажаних книжок.

Для роботи з пунктами списку імплементую 2 інтерфейси, що відповідатимуть за коротке і довге натиснення — OnItemClickListener та OnItemLongClickListener відповідно.

При натисненні на пункт додаватиму значення, що міститься в ньому в окремий список, myList. Потім, при підтвердженні свого вибору, я отримаю id кожного значення — це буде id джерела, яке потрібно буде вставити у базу даних.

Лістинг 16 – Відмічання джерел

```
private void giveBooks(){
final List<Integer> idsList = new ArrayList<Integer>();
     for(int i = 0; i < myList.size(); i++){
          for (int j = 0; j < ids.size(); j++) {
          if(myList.get(i).containsValue(ids.get(j).getName())){
          idsList.add(Integer.parseInt(ids.get(j).getValue()));
               }}}
.................
for (int i = 0; i < myList.size(); i++) {
nameValuePairsList.add(new BasicNameValuePair("userId",userId));
nameValuePairsList.add(new                BasicNameValuePair("id",
idsList.get(i).toString()));
          try {
post.setEntity(new UrlEncodedFormEntity(nameValuePairsList));
          } catch (UnsupportedEncodingException e1) {
               e1.printStackTrace();
          }
           try {
          HttpResponse response = client.execute(post);}
.................
}
```

Проходимо в циклі список відмічених джерел та список id усіх доступних джерел, і при виявлення співпадіння це означає, що дане джерело відмічена користувачем, а унікальність id виключає можливість помилки у виборі. Дані значення додаються в новий список. Нижче додаю id користувача і id кожного джерела в список ім’я-значення, який і відправляється на сервер.

Щоб підтвердити всій вибір, на екрані знаходиться меню, у якому є пункт «Відмітити». Меню можна створити програмно, але набагато кращим варіантом є попереднє створення у XML-файлі, а вже при створенні активіті разом із ним створюється і меню, беручи дані із вказаного файлу. Для

обробки вибору пунктів меню перевизначається метод onOptionsItemSelected базового класу Activity.

Лістинг 17 – Створення і обробка меню

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
     getMenuInflater().inflate(R.menu.main, menu);
    return (super.onCreateOptionsMenu(menu));
};

@Override
public boolean onOptionsItemSelected(MenuItem item) {
     switch(item.getItemId()){
     case R.id.getBooks:
          giveBooks();
          break;
}
     return super.onOptionsItemSelected(item);
}
```

Однозначно, що при наявності у базі даних великої кількості джерел, пошук потрібного простим перелистуванням списку є довгою справою, тому організуємо живий пошук по списку, що оновлятиме його вміст після кожного введеного символу. Для вводу даних використовуватиметься EditText, оскільки, на відміну від SearchView, він дозволяє використовувати TextChangedListener, щоб зразу реагувати на зміну тексту в полі вводу. В лістенері реалізується інтерфейс TextWacher який має методи beforeTextChanged, afterTextChanged та onTextChanged, який мені і потрібно реалізувати.

Лістинг 18 – Пошук джерел

```
     private void searchBook(){
     search.addTextChangedListener(new TextWatcher(){
     ........................
               @Override
               public  void  onTextChanged(CharSequence  s,  int
start, int before,int count) {
List<Map<String,       String>>       findedBooks       =       new
ArrayList<Map<String, String>>();
for(int i = 0; i < allBooks.size(); i++){
     if(allBooks.get(i).toString().toLowerCase(Locale.getDefault(
)).contains(s)){
findedBooks.add(getData.createBook("books", getBook(i)));
SimpleAdapter   sa   =   new   SimpleAdapter(BooksActivity.this,
findedBooks, android.R.layout.simple_list_item_multiple_choice,
new String[]{"books"}, new int[]{android.R.id.text1});
listview.setAdapter(sa);
                         }
                    }}});}
```

При кожній зміні тексту в полі вводу викликається onTextChanged. Як я попередньо і говорив, я працюю із списком allBooks, в даному разі перевіряю кожен елемент списку на вміст тексту в полі вводу, і якщо є співпадіння, додаю цей елемент в список знайдених джерел, після чого, за допомогою адаптера, встановлюю у listView значення із даного списку.

Також у користувача є змога переглянути короткий опис книжки, для цього і використовується обробка довгого натиснення на елемент списку. Інформація про книжку виводитиметься у вигляді повідомлення, об’єкта класу AlertDialog.Builder.

Лістинг 19 – Демонстрація інформації про джерело

```
     AlertDialog.Builder           ab           =           new
AlertDialog.Builder(BooksActivity.this);
     ........................
private void getDescription(int id){
String desc = id_desc.get(id).get(String.valueOf(id));
ab.setMessage(desc).setCancelable(true).setNeutralButton("OK",
new OnClickListener(){
     @Override
public void onClick(DialogInterface arg0, int arg1) {}}).show();
     }
```

Стрічка опису отримується із списку, що містить значення id-опис, і за значенням id, що передається у метод, вибирається відповідний опис і демонструється методом setMessage.

Користувач, врешті-решт, повинен мати змогу переглянути джерела, які він відмітив. Для цього створено ще одне активіті, яке має схожу на попереднє структуру і працює зі скриптом 2.8. Так як скрипт використовує метод GET, то щоб передати значення id користувача, достатньо після адреси скрипта, за якою і відбувається звернення, додати наступну конструкцію:

```
    String url = "http://library.bugs3.com/library/mybooks.php";
.....................
    Intent i = getIntent();
    userId = i.getStringExtra("userId");
    url += "?userId=" + userId; // get user id
```

Отже, вищеописані функції реалізують увесь функціонал, який і повинен мати даний додаток.

1.4 Розгортання програмної системи та системні вимоги

Розгортанням програмного забезпечення називають сукупність дій, які здійснюються для того, щоби програмна система була приведена до стану готовності для використання. Ця процедура відноситься до частини життєвого циклу будь-якого програмного забезпечення.

В загальному процес розгортки для будь-якого програмного забезпечення складається із декількох взаємнопов’язаних між собою дій з допустимими переходами (їх варіантами) поміж ними. Така активність може здійснюватись і виробником програми і споживачем. Так як кожну програмну систему можна вважати унікальною, тому все від процесів і до процедур під час процесу розгортання досить важко передбачати. Таким

чином, розгортка повинна трактуватися в цілому як загальний процес відповідний певним вимогам та характеристикам. Сама розгортка програмного продукту зазвичай здійснюється розробником під час процесу розробки.

Проєктований програмний продукт розгортається саме в процесі розробки, тому що з метою щоби система була повністю готова для використання, потрібно щоби працював сервер із базою даних, а він також буде працювати і під час етапу розробки. Звичайно зрозуміло, що для використання додатку потрібен пристрій, який працює під керуванням операційної системи Android.

Отже, настав час навести вимоги до програмного продукту:

- ОС Android із API версії не нижче8(Android Froyo).
- Оперативна пам’ять розміром не менш ніж256 мб.
- Можливість підключення до мережі інтернет.

2 ТЕСТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

2.1 План тестування

Щодо тестувального етапу створення проґрамного забезпечення, що є важливим процесом пов’язаним із технічним дослідженням, суть якого полягає у виявленні інформації стосовно якості продукту відносно контексту, в якому він має використовуватись. Техніка тестування передбачає безпосередньо проходження процесу знаходження помилок чи будь-яких інших дефектів, оцінка випробуванням програмних складових.

Планом тестування, як документом, описується повний обсяг робіт по тестуванню, а саме: опис об’єкту, стратегія, розклад, критерії початку і завершення тестування, до рівня достатнього і необхідного для безперебійного функціонування обладнання, спеціальні знання, оцінка ризиків та варіанти їх вирішення [34].

Зазвичай за об’єкт тестування виступає розроблювана мною програмна система, а саме додаток для операційної системи Android.

Під час проходження процедури тестування потрібно детально розглянути кожну функцію, з тих, якими наділено додаток і порівняти на відповідність результати виконання цих функцій з очікуваними результатами.

Тестування проводиться різними способами, зокрема, тестувальна діяльність, яка пов’язана із аналізуванням результату розробки проґрамного забезпечення є так зване статичне тестуванням за допомогою процедури якого здійснюється перевірка програмного коду, контроль, перевірка програми без запускання на комп’ютері. Зокрема, на вище згаданому етапі здійснюється перевірка усієї документації, отриманої в результаті життєвого циклу програмного продукту. Сюди відноситься процедура технічного завдання, специфікації, вихідного тексту програми мовою програмування.

Всю документацію аналізується предметно щодо дотримування вимог стандартів програмування. Таким чином результатом статичної перевірки реально визначити чи відповідає та наскільки програмний продукт відповідає до заданих критеріїв та вимог безпосередньо його замовника.

Такого виду тестування, якою передбачена експлуатація програмного продукту, вже відноситься до динамічного тестування, методи та підходи якого застосовні під час проходження процесу безпосередньо виконання програми. А вже перевірка програми на предмет коректності здійснюється велькою кількістю тестів чи набору підготованих вхідних даних. При «проганянні» кожного з таких тестів здійснюється збір та аналіз даних стосовно відмов та збоїв у роботі програми.

Ці два види, є взаємодоповнюючими тестуваннями програмного продукту [15].

2.2 Розробка тестів

Розглянемо спроби виконання кожної функції, що має виконувати додаток, і за результатами виконання судитимемо, наскільки правильно працює додаток.

Отже, першою функцією, яка необхідна для роботи з програмою є авторизація, а при першому вході — реєстрація. Вікно входу зображено на рисунку 6.

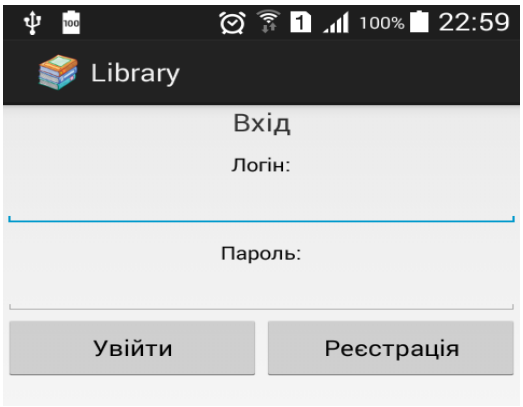

Рисунок 6 – Екран входу

Користувач вводить свої дані, після чого відбувається перевірка логіна і пароля на правильність і, якщо є неточності, користувачу виводиться відповідне повідомлення. Якщо ж користувач ще не працював із додатком, він може натиснути кнопку «Реєстрація», після чого видимим стане інший layout з елементами, що відповідають за реєстрацію. Користувач повинен ввести бажаний логін, пароль, а також повторити пароль, лише після цього він перейде на наступний екран, де треба вказати персональні дані. Якщо ж поля вводу не заповнені, бажаний логін уже зайнятий чи не співпадають паролі, користувачу має бути продемонстроване відповідне повідомлення.

Як видно з рисунку 7, дані перевірки працюють правильно.

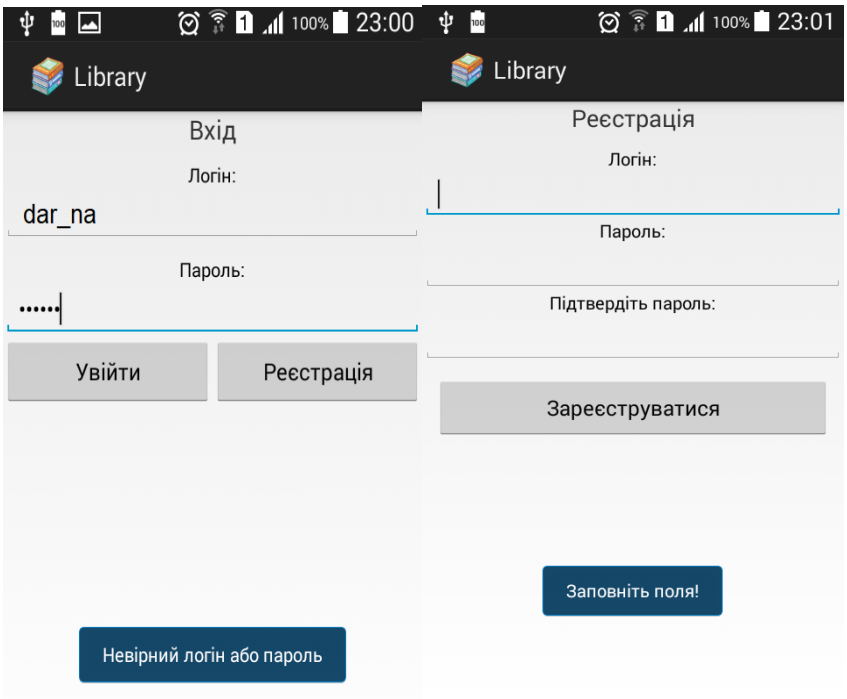

Рисунок 7 – Перевірки на правильність даних

На екрані введення персональних даних також повинна відбуватися перевірка на заповненість полів вводу, внаслідок якої користувач зможе остаточно зареєструватися лише, якщо заповнено усі поля. В протилежному випадку буде виведено повідомлення, аналогічне зображеному на рисунку 7. Якщо ж поля заповнено, як представлено на рисунку 8, то користувач повинен повернутися на екран входу і ввести свої логін та пароль.

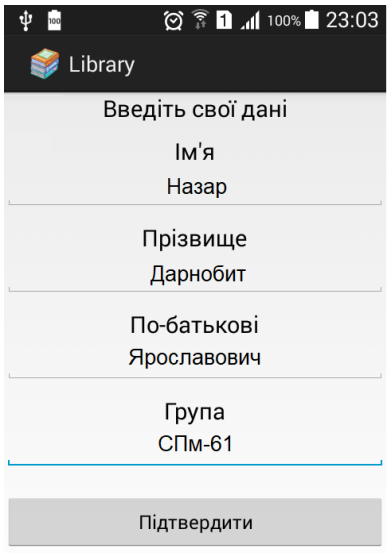

Рисунок 8 – Правильно вказані дані

Далі користувач має змогу залогінитися і переходить у власне меню (рис. 9), із якого він повинен мати змогу переглянути загальний список книжок, або ж свої, відмічені книжки.

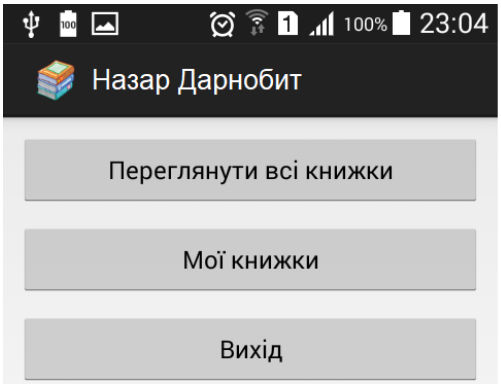

Рисунок 9 – Меню користувача

При натисненні кнопки перегляду усіх книжок повинен відкриватися новий екран, де і відображаються отримані з сервера дані про книжки. Це продемонстровано на рисунку 10.

Рисунок 10 – Перегляд книжок

Як було сказано раніше, користувач повинен мати змогу здійснювати пошук по списку, а також відмічати для себе бажані книжки. Для цього, як видно на рисунку, було створено поле вводу, при вводі символів у яке список має динамічно оновлюватися, а також сам список створений із можливістю множинного вибору, щоб потім за допомогою меню була можливість відмітити дані книжки. Також однією із функцій, доступних користувачу,

мав бути реалізований перегляд короткої інформації, опису книжки, що має відображатися при довгому натисненні на відповідний пункт списку. На наступних рисунках (рис. 11, 12, 13) відображено, як працюють дані функції.

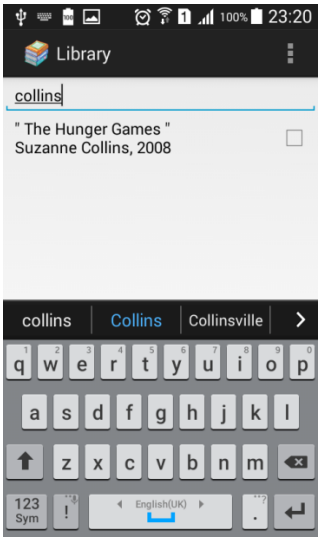

Рисунок 11 – Пошук по списку

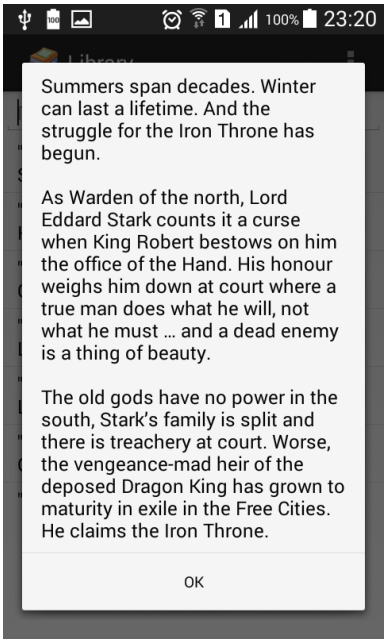

Рисунок 12 – Опис книжки

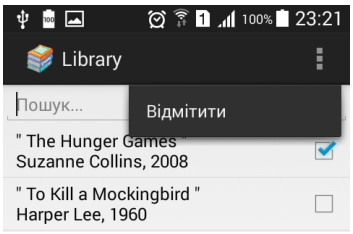

Рисунок 13 – Відмічення книжок

Для того, щоби перевірити, чи дійсно працює остання функція, а також чи відображаються користувачу відмічені ним книжки, треба перейти у другий пункт меню (див. рис. 9), і переглянути екран, що відкривається при натисканні кнопки «Мої книжки».

Отже, під час тестування додатка було виконано усі функції, які реалізовані даним програмним продуктом. При тестуванні помилок виявлено не було, усі функції працюють саме так, як і повинно бути.

3 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

3.1 Охорона праці [35,36]

Темою кваліфікаційної роботи є Розробка Android - застосунку мовою Java для пошуку та збору інформації в середовищі Eclipse. Розробка проводиться з урахуванням всіх етапів життєвого циклу ПЗ. При використанні ПЗ, яке є результатом даної розробки, як і при використанні будь-якого іншого ПЗ, необхідно дотримуватися вимог з охорони праці при роботі з ПК. Розглянемо основні нормативні документи, в яких зазначені вимоги до робочих місць та приміщень при використанні ПК.

Робочі місця працівників, обладнані персональними комп'ютерами, повинні відповідати вимогам «Правил охорони праці під час експлуатації електронно-обчислювальних машин», затверджених Наказом Державного комітету України з промислової безпеки, охорони праці та гірничого нагляду (Правила), та «Державних санітарних правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин», затверджених постановою Головного державного санітарного лікаря України (ДСанПіН). Правила поширюються на всіх суб'єктів господарювання незалежно від форм власності, які у своїй діяльності здійснюють роботу, пов'язану з персональними комп'ютерами, у тому числі на тих, які мають робочі місця, обладнані персональними комп'ютерами і периферійними пристроями. Зазначені нормативно-правові акти встановлюють санітарно-гігієнічні вимоги до приміщення, в якому розташоване робоче місце, власне до робочого місця, освітлення, рівнів вібрації і шуму, мікроклімату в приміщенні тощо.

Будівлі та приміщення, де розміщені робочі місця, повинні відповідати вимогам нормативно-технічної та експлуатаційної документації виробника персональних комп'ютерів ДСанПіН та Правил. Будівлі та приміщення, де розміщені робочі місця операторів, мають бути не нижче другого ступеня вогнестійкості. Для всіх будівель і приміщень, де знаходяться робочі місця, повинно бути визначено клас зони. Відповідне позначення повинно бути нанесено на вхідних дверях кожного приміщення. Не дозволяється розташування приміщень з робочими місцями у підвалах і цокольних поверхах. Неприпустимим є розташування приміщень категорій А і Б, а також виробництв з мокрими технологічними процесами поряд з приміщеннями, де розташовуються робочі місця, а також над ними чи під ними. При цьому площа приміщення має бути не менше 6,0 кв. м. із розрахунку на одне робоче місце, а об'єм – не менше 20,0 куб. м.

Віконні прорізи приміщень для роботи з персональними комп'ютерами мають бути обладнані регульованими пристроями (жалюзі, завіски, зовнішні козирки. Для внутрішнього оздоблення приміщень з персональними комп'ютерами слід використовувати дифузно-відбивні матеріали з коефіцієнтами відбиття для стелі 0,7-0,8, для стін 0,5-0,6. Покриття підлоги повинне бути матовим з коефіцієнтом відбиття 0,3-0,5. Поверхня підлоги має бути рівною, неслизькою, з антистатичними властивостями. Забороняється для оздоблення інтер'єру приміщень з персональними комп'ютерами застосовувати полімерні матеріали (деревинно-стружкові плити, шпалери, що миються, рулонні синтетичні матеріали, шаруватий паперовий пластик тощо), що виділяють у повітря шкідливі хімічні речовини.

У приміщеннях з джерелами шкідливих виробничих факторів робочі місця операторів мають розміщуватися в ізольованих кабінах, які обладнані повітрообміном.

Заземлені конструкції, що знаходяться в приміщеннях, де розміщені робочі місця (батареї опалення, водопровідні труби, кабелі із заземленим

відкритим екраном), мають бути надійно захищені діелектричними щитками або сітками з метою недопущення потрапляння працівника під напругу. Приміщення, де розміщені робочі місця, мають бути оснащені системою автоматичної пожежної сигналізації і вогнегасниками відповідно до вимог чинного законодавства України. Проходи до засобів пожежогасіння мають бути вільними.

У приміщеннях, в яких розташовані робочі місця, слід щоденно робити вологе прибирання. Крім того, ці приміщення мають бути оснащені аптечками першої медичної допомоги, а при них мають бути обладнані побутові приміщення для відпочинку під час роботи, кімната психологічного розвантаження.

При розміщенні робочих столів з персональними комп'ютерами слід дотримувати:

– відстань між бічними поверхнями персональних комп'ютерів 1,2 м.;

– відстань від тильної поверхні одного персонального комп'ютера до екрана іншого – 2,5 м.

За потреби особливої концентрації уваги під час виконання робіт суміжні робочі місця операторів необхідно відділяти одне від одного перегородками висотою 1,5 – 2м.

Конструкція робочого місця користувача персонального комп'ютера має забезпечити підтримання оптимальної робочої пози офісного працівника. Конструкція робочого столу має відповідати сучасним вимогам ергономіки і забезпечувати оптимальне розміщення на робочій поверхні використовуваного обладнання (дисплея, клавіатури, принтера) і документів. Висота робочої поверхні робочого столу має регулюватися в межах 680-800 мм, а ширина і глибина – забезпечувати можливість виконання операцій у зоні досяжності моторного поля (рекомендовані розміри: 600-1400мм, глибина – 800-1000мм).

Для забезпечення захисту і досягнення нормованих рівнів комп'ютерних випромінювань необхідно застосування приекранних фільтрів, локальних світлофільтрів (засобів індивідуального захисту очей) та інших засобів захисту, що пройшли випробування в акредитованих лабораторіях і мають щорічний гігієнічний сертифікат.

Приміщення для роботи з персональними комп'ютерами мають бути обладнані системами опалення, кондиціонування повітря, або припливно-витяжною вентиляцією. У приміщеннях на робочих місцях мають забезпечуватись оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря у відповідності до СН.

Рівні позитивних і негативних іонів у повітрі мають відповідати санітарно-гігієнічним нормам СН.

Для підтримки допустимих значень мікроклімату та концентрації позитивних та негативних іонів необхідно передбачати установки або прилади зволоження та/або штучної іонізації, кондиціювання повітря.

Приміщення, в яких встановлені персональні комп'ютери, повинні мати природне та штучне освітлення Природне освітлення має здійснюватись через світлові прорізи, орієнтовані переважно на північ чи північний схід. Штучне освітлення в приміщеннях з робочими місцями має здійснюватись системою загального рівномірного освітлення. У разі переважної роботи з документами, допускається застосування системи комбінованого освітлення (крім системи загального освітлення додатково встановлюються світильники місцевого освітлення). Зазначення освітленості на поверхні робочого столу в зоні розміщення документів має становити 300-500лк. Якщо ці значення освітленості неможливо забезпечити системою загального освітлення, допускається використовувати місцеве освітлення. При цьому світильники місцевого освітлення слід встановлювати таким чином, щоб не створювати відблисків на поверхні екрана, а освітленість екрана має не перевищувати 300лк. Як джерела світла в разі штучного освітлення мають застосовуватись

переважно люмінесцентні лампи типу ЛБ. У разі влаштування відбитого освітлення у приміщеннях, де переважним чином працюють з документами, допускається застосування металогалогенних ламп потужністю 250Вт. Допускається застосування ламп розжарювання у світильниках місцевого освітлення. Система загального освітлення має становити суцільні або переривчасті лінії світильників, розташовані збоку від робочих місць (переважно ліворуч), паралельно лінії зору працюючих.

Застосування світильників без розсіювачів та екрануючих грат заборонено. Яскравість світильників загального освітлення в зоні кутів випромінювання від 50 до 90 градусів з вертикаллю в повздовжній та поперечній площинах має становити не більше ніж 200 кд/м^2, захисний кут світильників – не менше ніж 40 градусів. Світильники місцевого освітлення повинні мати відбивач, що просвічує, із захисним кутом, не меншим ніж 40 градусів.

Слід передбачити обмеження прямої блискості від джерел природного та штучного освітлення. При цьому яскравість світлих поверхонь (вікна, джерела штучного освітлення), що розташовані в полі зору повинна бути не більше ніж 200 кд/м^2. Необхідно обмежувати відбиту блискість на робочих поверхнях відносно джерел природного і штучного освітлення. При цьому яскравість відблисків на екрані ВДТ має не перевищувати 40 кд/м^2, а яскравість стелі в разі застосування системи відбитого освітлення – 200 кд/м^2.

Показник осліпленості у разі використання джерел загального штучного освітлення у виробничих приміщеннях має не перевищувати 20, а показник дискомфорту в адміністративно-громадських приміщеннях має бути не більше за 40. Необхідно обмежувати нерівномірність розподілу яскравості в полі зору працюючих з ВДТ. При цьому співвідношення яскравостей робочих поверхонь має бути не більшим ніж 3:1, а співвідношення яскравостей робочих поверхонь та поверхонь стін,

обладнання тощо – 5:1. Коефіцієнт запасу для освітлювальних установок загального освітлення має дорівнювати 1,4. Коефіцієнт пульсації має не перевищувати 5%, що забезпечується застосуванням газорозрядних ламп у світильниках загального та місцевого освітлення з ВЧ ПРА для світильників будь-яких типів. Якщо не має світильників з ВЧ ПРА, то лампи багатолампових світильників або світильники загального освітлення, розташовані поруч, слід вмикати на різні фази трьохфазної мережі

Устаткування, що становить джерело шуму (АЦП, принтери тощо), слід розташовувати поза приміщеннями, де знаходяться робочі місця. Для забезпечення допустимих рівнів шуму на робочих місцях слід застосовувати засоби звукопоглинання, вибір яких має обґрунтовуватись спеціальними інженерно-акустичними розрахунками.

Інтенсивність потоків ультрафіолетового випромінювання має не перевищувати допустимих значень відповідно до СН.

Персональні комп’ютери, периферійні пристрої, інше устаткування (апарати управління, контрольно-вимірювальні прилади, світильники), електропроводи та кабелі за виконанням і ступенем захисту мають відповідати класу зони, мати апаратуру захисту від струму короткого замикання та інших аварійних режимів. Усі провідники мають відповідати номінальним параметрам мережі та навантаження, умовам навколишнього середовища, умовам розподілу провідників, температурному режиму та типам апаратури захисту.

У приміщенні, де одночасно експлуатуються понад п’ять персональних комп’ютерів і периферійних пристроїв, на помітному та доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім освітлення.

Персональні комп’ютери і периферійні пристрої повинні підключатися до електромережі тільки за допомогою справних штепсельних з’єднань і електророзеток заводського виготовлення. У штепсельних з’єднаннях та

електророзетках, крім контактів фазового та нульового робочого провідників, мають бути спеціальні контакти для підключення нульового захисного провідника. Їхня конструкція має бути такою, щоб приєднання нульового захисного провідника відбувалося раніше, ніж приєднання фазового та нульового робочого провідників. Порядок роз'єднання при відключенні має бути зворотним. Не допускається підключати персональні комп'ютери та периферійні пристрої до звичайної двопровідної електромережі, в тому числі з використанням перехідних пристроїв.

При організації робочих місць операторів електромережу штепсельних розеток для живлення персональних комп'ютерів, периферійних пристроїв і у центрі приміщення прокладають у каналах або під знімною підлогою в металевих трубах або гнучких металевих рукавах. При цьому не допускається застосовувати провід і кабель в ізоляції з вулканізованої гуми та інші матеріали, які містять сірку.

При організації праці, що пов'язана з використанням персональних комп'ютерів, для збереження здоров'я працюючих, запобігання професійним захворювання і підтримки працездатності слід передбачити внутрішньозмінні регламентовані перерви для відпочинку. Внутрішньозмінні режими праці і відпочинку мають передбачати додаткові нетривалі перерви в періоди, що передують появі об'єктивних і суб'єктивних ознак стомлення і зниження працездатності.

В дипломній роботі виконано розробку централізованої системи управління електронного документообігу для органів місцевого самоврядування в Україні. Розробка проводилась з урахуванням всіх етапів життєвого циклу ПЗ, тому важливо було розглянути основні вимоги до приміщень та робочих місць, де використовують ПК, що й було зроблено в цьому підрозділі. Також були наведені правила електробезпеки під час роботи з ПК та вимоги до пожежної безпеки в приміщенні. Щоб гарантувати

безпечні умови праці під час розробки програмного забезпечення, необхідно дотримуватися наведених вимог.

3.2 Безпека в надзвичайних ситуаціях

3.2.1 Визначення, причини виникнення та класифікації надзвичайних ситуацій [37,38]

Щодня в світі фіксуються тисячі подій, при яких відбувається порушення нормальних умов життя і діяльності людей і які можуть призвести або призводять до загибелі людей та/або до значних матеріальних втрат. Такі події називаються надзвичайними ситуаціями.

Надзвичайна ситуація (НС) – порушення нормальних умов життя і діяльності людей на об’єкті або території, спричинене аварією, катастрофою, стихійним лихом, епідемією, епізоотією, епіфітотією, великою пожежею, застосуванням засобів ураження, що призвели або можуть призвести до людських і матеріальних втрат.

Небезпека – це негативна властивість матерії, яка проявляється у здатності її завдавати шкоди певним елементам Всесвіту, потенційне джерело шкоди. Якщо мова йде про небезпеку для людини, то це явища, процеси, об'єкти, властивості, здатні за певних умов завдавати шкоди здоров'ю чи життю людини або системам, що забезпечують життєдіяльність людей.

Фактори, які призводять до надзвичайних ситуацій можуть бути:

- прямі – несуть загрозу для людей, навколишнього середовища та економічних об'єктів (удар, вибух тощо);
- непрямі – діють опосередковано (ожеледиця, злива), викликаючи інші небезпечні фактори. Наприклад, обледеніння, яке само по собі не

несе небезпеки людині, викликало руйнування електросистеми у кількох областях України, що призвело до припинення господарської діяльності, значних матеріальних збитків, пов'язаних з відновленням ліній електропередач, не випуском продукції підприємствами, а також створило загрозу для життя та здоров'я людей через порушення теплозабезпечення будинків.

Будь-яка з надзвичайних ситуацій може стати причиною виникнення іншої та викликати небезпечні екологічні наслідки: соціальні, природні, техногенні, небезпечні екологічні наслідки.

Наприклад, землетрус – природна НС – призводить до руйнування споруд, пожеж, що характерно для техногенної надзвичайної ситуації, крім того, під час землетрусу гине багато людей, руйнуються житлові будинки, інфраструктура життєзабезпечення, що викликає соціальну НС. Дані свідчать, що в сучасних умовах практично будь-яка надзвичайна ситуація є комплексною. Визначення причин та закономірностей розвитку таких надзвичайних ситуацій є складним завданням.

У кожному конкретному випадку надзвичайні ситуації виникають через ряд причин, які можна узагальнити. Природні надзвичайні ситуації в більшості є наслідком закономірного розвитку природних метеорологічних, космічних, гідрологічних чи тектонічних процесів. Це урагани, землетруси, обвали, падіння космічних тіл тощо.

Причини виникнення природних надзвичайних ситуацій та небезпечних явищ: Закономірні природні процеси, негативний антропогенний вплив на розвиток природних процесів, Випадковість у розвитку природних процесів

Але все частіше причинами природних надзвичайних ситуацій виступає людська діяльність. Техногенний розвиток досяг такого рівня, що можна штучно викликати великі природні надзвичайні ситуації будь-якого характеру, наприклад, землетруси, цунамі, засухи, епідемії тощо.

Група дослідників страхового товариства Munchener Ruck (Німеччина) понад 25 років збирала й аналізувала дані про природні катаклізми (землетруси, шторми, повені, виверження вулканів, тайфуни), що відбулися в світі з часів Римської імперії. "Немає жодних сумнівів у тому, що частота і сила руйнування природних катастроф значно зростають", - роблять висновок автори. Причини для цього, на їх думку, створює сама людина. Перш за все, йдеться про урбаністичний розвиток, використання потенційно небезпечних територій, зміни клімату та стану довкілля.

Те, що за таке дослідження взялися співробітники страхового товариства, пояснюється тим, що кожна катастрофа приносить не лише економічні збитки, а й зменшує доходи у страховому бізнесі. З кінця шістдесятих років виплати страхових компаній на покриття збитків від природних катаклізмів у всьому світі зросли від одного до більше десяти мільярдів доларів на рік.

Виникнення соціальних надзвичайних ситуацій, перш за все, пов'язують з поширенням ідей, що часто носять антисоціальний та відверто людиноненависницький характер. До соціальних конфліктів також призводять національні, економічні, псевдо релігійні та політичні причини. Війна в Чечні, Афганістані, Іраку, Палестино-Ізраїльський конфлікт, військові дії в Іраку є наочною ілюстрацією цього з тією різницею, що для кожного конфлікту вагомість причин різна. В ряді випадків причинами соціальних НС, а саме, страйків, забастовок є економіка та політика.

Причини виникнення соціальних надзвичайних ситуацій: політичні, економічні, національні, релігійні ідеологічні.

Внаслідок природних та техногенних катастроф теж виникають соціальні НС через порушення у функціонуванні систем життєзабезпечення, величезних матеріальних збитків, значного травматизму населення, психологічних факторів.

Соціальною надзвичайною ситуацією, масштаби проявлення якої останнім часом збільшуються, є тероризм. Причини тероризму криються у жадобі - влади, слави, багатства, використовуючи будь-які методи. Тероризм може експлуатувати будь-яку ідею, для формування якої використовує різноманітні расистські, нацистські, спотворені релігійні, національні, економічні, політичні, соціальні погляди, в т. ч. ідеї національного визволення, соціалізму, комунізму. рівності людей, вищої раси, формування "нового порядку", відвернення планетарної катастрофи, антиглобалізму тощо.

Збільшення кількості підприємств, ускладнення технічних систем та зростання їх потужності, розширення транспортних мереж, урбанізація на тлі недостатньої уваги до питань безпеки закономірно призводять до зростання кількості техногенних надзвичайних ситуацій, аварій та катастроф. Збільшення кількості населення на планеті та його концентрація у промислово розвинених регіонах створює умови для зростання кількості жертв під час надзвичайних ситуацій техногенного характер.

Усі причини, через які виникають надзвичайні ситуації техногенного характеру можна розділити на три групи:

технічні - недосконалість, застарілість конструкцій;

природні - специфічні метеорологічні, гідрологічні чи тектонічні умови, природні надзвичайні ситуації, випадковості (наприклад, однією з причин катастроф авіалайнерів є потрапляння птахів у двигун літака);

антропогенні ("людський фактор") – недотримання правил безпеки, помилки, необережність, халатність.

Причини виникнення техногенних надзвичайних ситуацій: недодержання правил безпеки та необережність, недосконалість у проектуванні, кримінальні елементи та тероризм, воєнні дії, природні явища.

ВИСНОВКИ

Отже, під час виконання кваліфікаційної роботи було реалізовано програмний продукт, що вирішує виявлені в предметній області недоліки та виконує весь запланований функціонал.

В розділі «Розробка програмної системи» проведено аналіз вимог до програмного продукту, спроєктовано програмну систему та описано реалізацію серверної та клієнтської частин програми.

В розділі «Тестування програмної системи» було розроблено план тестування та проведено тестування на виявлення недоліків у реалізованому функціоналі програмної системи.

В розділі «Охорона праці» розглянуто нормативні акти з охорони праці на окремих об’єктах та питання безпеки.

Таким чином, в дипломній роботі було виконано поставлені завдання щодо реалізації віддаленого доступу до бази даних бібліотеки за допомогою додатка для ОС Android. Програмний продукт значно полегшує отримання інформації про книжки у бібліотеці та дозволяє зразу ж після появи такої потреби забронювати для себе потрібну книжку. Разом із тим, програма має деякі недоліки, які планується виправити в майбутньому:

- Реалізація офлайн-режиму з використанням вбудованої в Android СУБД SQLite та синхронізації даних при підключенні до інтернету.
- Веб-сайт для адміністрування бібліотеки.
- Розробка додатків для iOS та Windows Phone.

В представленій на розгляд ЕК29 дипломній роботі акцентовано увагу на бібліотеку, як предметну область. Враховуючи той факт, що фактично в кожного є пристрої, які використовують Android, наявний постійний доступ до мережі Інтернет, є можливість надати змогу їх користувачам можливість віддаленого доступу до серверів чи бібліотек з метою пошуку та опрацювання даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

ДОДАТКИ

Додаток А Слайди презентації

Додаток Б Апробація результатів роботи

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

XI НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

13-14 грудня 2023 року

ТЕРНОПІЛЬ
2023

УДК 001
М34

ПРОГРАМНИЙ КОМІТЕТ

Голова: Приймак Микола – професор кафедри комп'ютерних систем та мереж, д.т.н., професор.

Співголови: Марущак Павло – проректор з наукової роботи, докт. техн. наук, професор.
Баран Ігор – канд. техн. наук, доцент, декан факультету ФІС.

Науковий секретар: Семенишин Галина – старший викладач.

Члени: Василь Кривень - завідувач кафедри математичних методів в інженерії д.ф.-м.н., професор; Галина Осухівська – завідувач кафедри комп'ютерних систем та мереж, к.т.н., доцент; Микола Карпінський - професор кафедри кібербезпеки, д.т.н., професор; Жанна Баб'як - завідувач кафедри української та іноземних мов, к.пед. н., доцент; Ярослав Литвиненко – професор кафедри комп'ютерних наук, д.т.н., професор; Михайло Петрик - завідувач кафедри програмної інженерії, д.ф.-м.н., професор; Наталія Загородна – завідувач кафедри кібербезпеки, к.т.н., доцент.

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

Голова: Скоренький Юрій Любомирович – канд. техн. наук, доцент кафедри фізики.

Члени: доцент кафедри комп'ютерних наук, к.т.н. В. Никитюк; доцент кафедри програмної інженерії, к.т.н. Д. Михалик; доцент кафедри кібербезпеки, к.т.н. М. Стадник; асистент Н. Шаблій; ст. викладач Л. Джиджора.

Матеріали XI науково-технічної конфції «Інформаційні моделі, системи та
М34 технології» Тернопільського національного технічного університету імені Івана Пулюя, (Тернопіль, 13-14 грудня 2023 р.). – Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2023. – 257 с.

Адреса оргкомітету: ТНТУ ім. І. Пулюя, м. Тернопіль, вул. Руська, 56, 46001,
тел. (0352) 52-41-33, факс (0352) 254983.
E-mail: confis2023@gmail.com

Редагування, оформлення, верстка: Семенишин Г.М.

СЕКЦІЇ КОНФЕРЕНЦІЇ, ЯКІ ПРЕДСТВЛЕНІ В ЗБІРНИКУ

- Математичне моделювання;
- Інформаційні системи та технології;
- Комп'ютерні системи та мережі;
- Програмна інженерія та моделювання складних розподілених систем;
- Новітні фізико-технічні та освітні технології.

В збірнику надруковано тези доповідей XI науково-технічної конференції «Інформаційні моделі, системи та технології» (Тернопіль, 13-14 грудня 2023 р.) за такими науковими напрямками: математичне моделювання; інформаційні системи та технології; комп'ютерні системи та мережі; програмна інженерія та моделювання складних розподілених систем; новітні фізико-технічні та освітні технології.

Розрахований на науковців, викладачів та студентів вузів.

За зміст тез та дотримання норм академічної доброчесності відповідальність несе автор.

Додаток Б Апробація результатів роботи

УДК 004.41

Н.Я.Дарнобит, Г.Б Цуприк, канд. техн. наук, доц.
Тернопільський національний технічний університет імені Івана Пулюя, Україна

РОЗРОБКА ANDROID-ЗАСТОСУНКУ ДЛЯ ПОШУКУ ТА ЗБОРУ ІНФОРМАЦІЇ МОВОЮ JAVA

N.Ya.Darnobyt, H.B. Tsupryk, PhD, Assoc.Prof.

DEVELOPMENT OF ANDROID APPLICATION FOR INFORMATION SEARCHING AND COLLECTION ON JAVA LANGUAGE

Безпечний, вільний, швидкий доступ в умовах військового стану – це завжди актуально, ніколи не буває лишнім і завжди це щось нове та потрібне, оскільки технології розвиваються, підходи вдосконалюються, коло питань розширюється, а потреби залежать від попиту, вимог та можливостей. Зрозуміло, що інформація є різна, і призначення її різне, проте на суть та важливість питання це ніяким чином не впливає, тому я і обрав це за основний напрямок. Також на моє рішення вплинуло і те, що під час мого навчання у мене часто виникала проблема із тим, щоби визначитись з новизною моєї роботи (чи це вже хтось робив, що робив, які отримав результати), перевагами та недоліками. Також бувало, що мені не вдавалось одразу знайти потрібну інформацію.

На сьогоднішній день темпи розвитку мобільних платформ, так само як і поява все новішого програмного забезпечення під них, є одними із найінтенсивніших і найвищих посеред інших інструментів та засобів, а кількість пристроїв, які працюють під операційну систему Android чи iOS вже точно і значно перевищує кількість персональних комп'ютерів.

В роботі акцентовано увагу на бібліотеку, як предметну область. Враховуючи той факт, що фактично в кожного є пристрій, і часто навіть не один, який працює під Android, наявний постійний доступ до мережі Інтернет, є можливість надати змогу користувачам віддалений доступ до серверів чи бібліотек з метою пошуку та опрацювання даних.

Таким чином вимальовується об'єкт дослідження – вдосконалення, через врахування наявних проблем, можливості доступу та пошуку необхідних даних різних категорій, напрямку та об'ємів, а вже предметом дослідження є конкретно використання сучасних інформаційних технологій для розробки Android-застосунку, зокрема трирівневої моделі архітектури, , яка передбачає певні програмні компоненти, зокрема: 1-клієнт (безпосередньо продукт який я розробляю) та 2-сервер, на якому виконуються PHP-скрипти. Він підключений до безпосередньо серверу бази даних, додаток – клієнтська частина продукту, розробка базується на технології Android SDK, проводитиметься розробка в середовищі Eclipse, мовою програмування Java, .а якості бази даних обрано базу MySQL, СКБД — phpMyAdmin.

Заплановано також реалізацію off-line режиму з використанням вбудованої в Android СУБД SQLite та синхронізації даних при підключенні до інтернету, проєктування веб-сайту для адміністрування бібліотеки та додатків для iOS та Windows Phone.

Література

1. Моделювання та видобуток даних (високопродуктивні обчислення у великих алгебраїчних та числових системах, комбінаторному аналізі): навчальний посібник. Тернопіль: : ТНТУ 2019 – 62 с.

2. Бойко І.В., М.Р. Петрик, Г.Б. Цуприк. Інформаційні технології видобутку даних (Data mining, високопродуктивні обчислення у складних системах): навчальний посібник. Тернопіль: ТНТУ 2020 – 62 с.

3. М.Р. Петрик, Д.М. Михалик, О.Ю. Петрик, Г.Б. Цуприк. Методичні вказівки до виконання атестаційної роботи магістра за спеціальністю 121 – "Інженерія програмного забезпечення" для усіх форм навчання [Текст] – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя – 2020 – 27 с.3.

Додаток В Відгук та рецензія

ЗМІСТ