Київський національний університет імені Тараса Шевченка
Факультет радіофізики, електроніки та комп'ютерних систем
Кафедра комп'ютерної інженерії

Розробка комунікаційних програм для операційної системи Android, що користуються ресурсами веб-служб REST/SOAP

Випускна кваліфікаційна робота бакалавра
студента 4 курсу спеціальності
123 - «Комп'ютерна інженерія»
Корчака Олександра Олександровича

_____
____
(підпис)

Науковий керівник:
доцент кафедри комп'ютерної інженерії
Сергій ЗАГОРОДНЮК

_____
(підпис)

Рецензент
Доцент кафедри геоінформатики ННІ
"Інститут геології"
кандидат фіз.-мат. наук
Всеволод ДЕМИДОВ

_____
(підпис)

До захисту допускаю Зав. кафедри Юрій БОЙКО

Протокол засідання кафедри від
"__"_____2022р. № ____

Київ 2022

РЕФЕРАТ

Робота за об’ємом складає 44 сторінки, містить 26 рисунків, використано 21 інформаційне джерело.

Дана робота містить теоретичні відомості щодо технологій, з допомогою яких можливо виконати поставлену задачу, аналізу ефективності кожної з досліджених технологій та доцільності використання того чи іншого інструменту. Розповідається про цінність вивчення зазначених в поставленій задачі технологій для студентів кафедри комп’ютерної інженерії з точки зору студента 4 курсу цієї спеціальності. Представлений навчальний курс лабораторних робіт з вивчення таких засобів як REST, SOAP, проектування та розробки типових програм для операційної системи Android та прикладів готових лабораторних робіт.

Головною метою роботи є створення курсу лабораторних робіт спрямованих на проектування та розробку програм, що користуються можливостями веб-служб REST та SOAP.

Ключові слова: REST, SOAP, Android, програма, веб-служба, клієнт, сервер, операційна система.

ЗМІСТ

ВСТУП

Станом на листопад 2021 року частка операційної системи Android на ринку всіх операційних систем світу складає 39,26% що на 6.01% більше ніж доля систем Windows та Linux узятих разом - 33.25%:

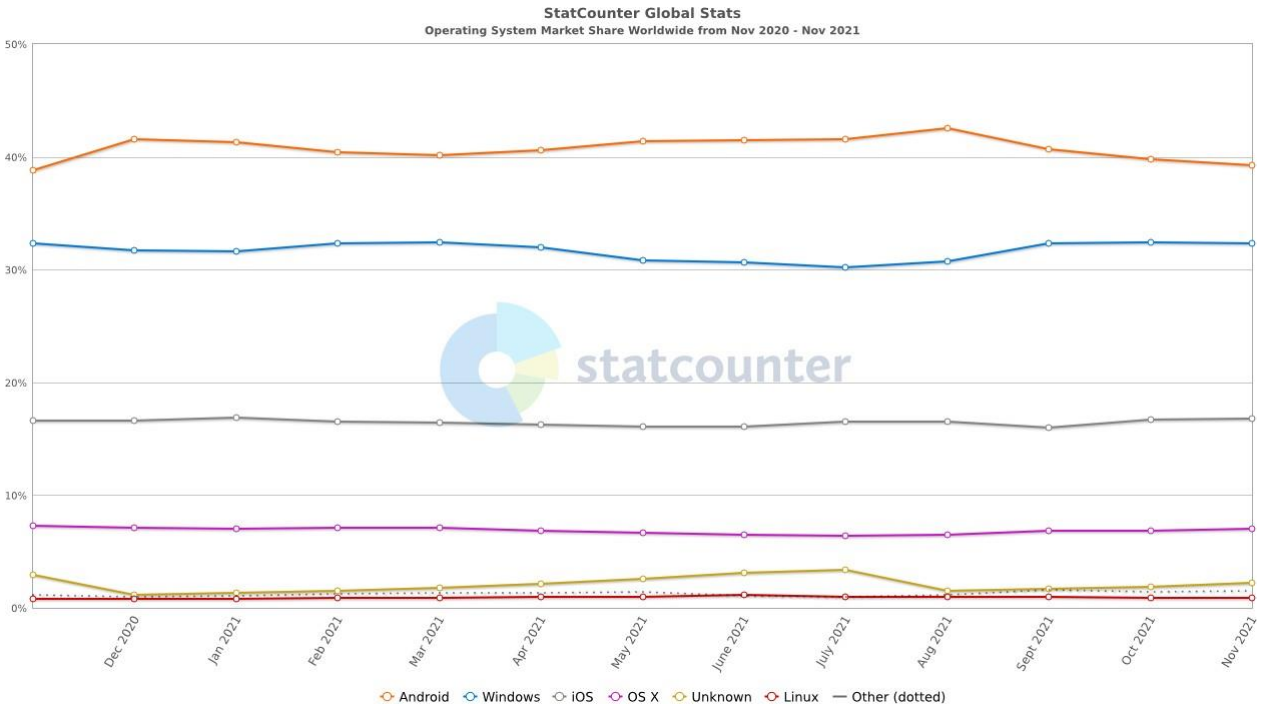

Рисунок 1. діаграма, що ілюструє частку найпопулярніших операційних систем в світі за період листопад 2020 - листопад 2021 [1]

Більшість програм, що були розроблені для операційної системи Android, комунікують як у внутрішніх мережах так і в глобальній мережі з іншими програмними засобами, що можуть бути розроблені для інших операційних систем. Способи передачі даних в мережах ми вивчали в курсі “Комп’ютерні мережі”, але програми можуть будуть мати різну архітектуру в залежності від операційної системи, для якої вони були створені, що створює проблеми при спробах серіалізації даних, що були отримані з мережі . Для вирішення цієї проблеми було створено велику кількість веб-служб, що виступають шаблонами формату для передачі повідомлень, перетворювачами повідомлень з одного типу в інший і т.д. Найбільш популярними з них на сьогодні є архітектурний стиль REST та протокол SOAP. Я пропоную додати до навчального плану кафедри комп’ютерної інженерії окрім вивчення можливостей Windows та Linux курс

лабораторних робіт, що буде спрямований на ознайомлення студентів з можливостями операційної системи та служб, якими вони без перебільшення користуються кожного дня. Очевидними плюсами цього нововведення я вважаю те, що студенти нашої кафедри зможуть ознайомитись з сучасними та потрібними в сфері ІТ технологіями, матимуть практичний досвід роботи з ними та зможуть розширити свій кругозір. Також плюсом можна назвати те, що інформація, отримана здобувачем освіти за час навчання, буде більш пов’язана між собою. Так як зараз, на мою думку, навчальні дисципліни більш фрагментарно подають інформацію і наприклад такі курси як програмування та комп’ютерні мережі досить складно уявити як частини одного цілого, що не сприяє розумінню того, спеціалістом в чому є комп’ютерний інженер. Якщо ж студент буде мати курси, що пов’язують між собою знання на перший погляд різних дисциплін, він отримає глибше розуміння усіх трьох дисциплін.

МЕТА РОБОТИ

Підготувати необхідні теоретичні відомості, необхідні для розробки комунікаційних програм, що працюють під управлінням операційної системи Android і користуються мережевими ресурсами веб-служб архітектурного стилю REST та протоколу SOAP. Обрати мову програмування для розробки прикладних програм для операційної системи Android, що містить найбільший інструментарій для роботи з веб-службами REST/SOAP.

Розробити структуру факультативного курсу лабораторних робіт, який включає в себе ознайомлення студентів кафедри комп'ютерної інженерії з технологіями розробки програм для операційної системи Android, реалізацію мережевого доступу до ресурсів веб-служб, формування практичного досвіду розробки комунікаційної програм, що використовують водночас глобальні мережі TCP/IP та персональні мережі короткої відстані, зокрема Bluetooth.

Створити опис повного курсу лабораторних робіт з прикладами, поясненнями, посиланнями на актуальні документації для самостійного опрацювання студентами, що хочуть виконати додаткові завдання та розширити досвід розробки програм такого типу.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Загальні відомості про архітектурний стиль REST

REST - REpresentational State Transfer (передача репрезентативного стану) - це архітектурний стиль для розподілених інформаційних систем. Цей стиль був описаний і представлений в 2000 році Роєм Філдінгом у своїй дисертації [2]. Як і інші архітектурні стилі, REST має свої принципи та обмеження:

а) uniform interface - інтерфейс повинен дотримуватись принципу загальності. Це дозволяє спростити загальну архітектуру системи та покращити взаємодії в ній. Принципу загальності для інтерфейсу можна досягти використовуючи наступні чотири обмеження:

- identification of resources - інтерфейс повинен одночасно ідентифікувати всі ресурси що беруть участь взаємодії між клієнтом та сервером;
- manipulation of resources through representations - усі ресурси повинні мати уніфіковані представлення у відповіді сервера, щоб клієнт міг, маючи ці представлення, модифікувати або ж видаляти ресурси;
- self-descriptive messages - як зрозуміло з назви, кожне представлення ресурсу повинно мати достатньо інформації для опису обробки повідомлення та додаткових дій над ним, якщо вони потрібні;
- hypermedia as the engine of application state - клієнтська програма повинна динамічно управляти усіма ресурсами та взаємодіями з допомогою гіперпосилань, клієнт ж повинен мати лише початкову адресу програми;

б) client-server - це означає що користувацький інтерфейс або клієнт та сервер повинні бути відокремленими один від одного, щоб наприклад проблеми клієнта не впливали на сервер та навпаки. Цим ми забезпечуємо можливість серверу працювати інтерфейсами користувача на різних платформах. Поки клієнт та сервер еволюціонують, розробник повинен переконатись, що інтерфейс/контракт між клієнтом не порушуються;

в) layered system - поділ системи на шари абстракції шляхом обмеження компонентів дозволяє створити ієрархію шарів компонентів у якій кожен конкретний компонент шару не буде мати доступу за межі цього шару;

г) cacheable - кожна відповідь сервера повинна явно себе позначати як та що кешується або ні щоб дати можливість програмі самій вирішувати які дані є можливість повторно використовувати для еквівалентних запитів і за який період, а які ж ні;

д) stateless - сервер повинен бути розроблений так, щоб не було можливості скористатися будь якою раніше збереженою контекстною інформацією, тому клієнт повинен повністю зберігати стан сеансу та надсилати в запитах всю інформацію, що необхідна для розуміння та виконання запитів сервером;

е) code on demand - необов’язковий насправді принцип що потребує від сервера можливість розширювати функціональність клієнта шляхом завантаження та виконання додаткового коду у вигляді скриптів. З допомогою завантаженого коду можна зменшити кількість функцій, що необхідно попередньо розробити в клієнті.

Якщо прикладний інтерфейс служби/програми або ж по іншому API задовольняє ці принципи, його називають RESTful.

В принципах було згадано про ресурси - ключову абстракцію REST, що включає в себе будь яку інформацію що може бути названа. REST API складається з сукупності взаємопов’язаних ресурсів - моделі ресурсів REST API. Стан ж ресурсу в конкретний момент називається представленням ресурсу, про що також згадувалось вище. Представлення ресурсів складаються з власне даних, метаданих, що описують дані, та гіпермедійних посилань що служать клієнтам для переходу до наступних бажаних станів. Для того щоб відрізняти ресурси, що беруть участь у взаємодії між клієнтськими та серверними компонентами, використовують ідентифікатори ресурсів. Крім того, представлення ресурсів мають бути самоописовими: тобто клієнту не потрібно

знати чим саме є ресурс, він повинен діяти на основі типу медіа пов’язаного з ресурсом.

Ще однією важливою частиною REST є Resource methods - методи, що використовуються для переходу між двома станами ресурсу. Так як REST орієнтований на роботу з HTTP, найлегшим прикладом будуть HTTP методи: POST для додавання інформації, GET для отримання інформації, PUT для оновлення, DELETE відповідно для видалення. Хоча насправді Рой Філдінг ніколи не давав ніяких рекомендацій щодо того, який метод використовувати за яких умов, а лише наголошував що це має бути єдиний інтерфейс, тобто якщо ми вирішимо, що наприклад POST буде видаляти інформацію, наш інтерфейс, все одно буде RESTful.

1.2 Загальні відомості про протокол SOAP

SOAP - Simple Object Access Protocol - протокол призначений для обміну структурованою інформацією в децентралізованому розподіленому середовищі. Основні цілі протоколу SOAP - простота та розширюваність, що досягається з допомогою специфікації повідомлення через XML. SOAP-повідомлення формально визначається як набір інформації, що надає абстрактний опис його вмісту з можливо абсолютно різними серіалізаціями, за рахунок чого досягається універсальність SOAP як протоколу передачі даних для абсолютно будь якої програми в різних операційних системах та через різні протоколи зв’язку, потрібно лише написати обробку повідомлення що приходить.

Так як SOAP є універсальним протоколом, він має досить багато концептів: soap, вузол, роль, прив’язка, функція, модуль, MEP, додаток, повідомлення, конверт, заголовок, блок заголовка, тіло, помилка, відправник, отримувач, шлях повідомлення, початковий відправник, посередник, остаточний отримувач. Декілька слів для загального розуміння кожного концепту:

- soap - формальний набір умов, або ж угоди, що регулюють формат та правила обробки повідомлень SOAP, вони включають взаємодію між

вузлами SOAP, що генерують та приймають повідомлення SOAP з метою обміну інформацією по шляху повідомлення;

- вузол - варіант реалізації логіки, що необхідна для передачі, обробки та/або ретрансляції повідомлень SOAP відповідно до угод цієї спеціалізації. Вузол також відповідає за виконання правил регулювання обміну повідомленнями та отримує доступ до послуг, наданих базовими протоколами, з допомогою прив’язок SOAP;
- роль - очікувана функція або ж функцій приймача SOAP при обробці повідомлення
- прив’язка - формальний набір правил, необхідних для переміщення SOAP-повідомлення всередині або поверх іншого основного протоколу з метою обміну. Для прикладу можна взяти переміщення повідомлення всередині тіла об’єкта HTTP або через TCP;
- функція - розширення платформи обміну повідомленнями, наприклад MEP;
- модуль - специфікація з комбінованим синтаксисом та сематикою блоків заголовків SOAP, може реалізувати вищевказані функції;
- MEP - шаблон обміну повідомленнями між вузлами;
- додаток - суб’єкт, скоріш за все програма, що створює, споживає або іншим чином взаємодіє з повідомленнями SOAP у спосіб, що відповідає моделі обробки SOAP;
- повідомлення - основий об’єкт зв'язку між вузлами SOAP;
- конверт - зовнішня частина повідомлення SOAP;
- заголовок - набір блоків заголовків SOAP, кожен з яких може бути спрямований на будь якого одержувача в межах шляху повідомлення SOAP;
- блок заголовка - елемент, що використовується для розмежування даних та позначається в XML як header;
- тіло - колекція елементів інформації про об’єкт, націлених на кінчевого отримувача у шляху повідомлення;

- помилка - елемент, що містить інформацію про помилку, створену вузлом
- відправник - вузол, що передає повідомлення;
- отримувач - вузол, що власне приймає повідомлення;
- шлях повідомлення - набір вузлів, через які проходить повідомлення, включає в себе початкового відправника, нуль або ж більше посередників та остаточного отримувача SOAP;
- початковий відправник - відправник, що створює повідомлення;
- посередник - одночасно і отримувач, і відправник, і може виступати як ціль повідомлення. Він обробляє блоки заголовка, спрямовані на нього і діє на пересилання повідомлення до кінцевого отримувача;
- остаточний отримувач - отримувач повідомлення, який є кінцевим призначенням повідомлення SOAP, він відповідає за обробку тіла та будь яких блоків заголовка спрямованих на нього

1.3 Загальні відомості про операційну систему Android

Android - операційна система на базі Linux під ліцензією Apache v2 з відкритим кодом, що дозволяє розробляти багато варіацій цієї операційної системи, що вкупі з основним профілюванням на мобільну платформу, таку як телефони та планшети, дозволило завоювати основну долю ринку операційних систем.

Android розроблявся як стратап компанією Android.inc в 2003 році. Спочатку компанія хотіла створити операційну систему для цифрових камер, але досить швидко відмовилась від цієї ідеї на користь більш широкого ринку.

У 2005 році компанія Google придбала Android.inc та її ключових співробітників та представила першу мобільну платформу з подібними гнучкістю та можливістю оновлення. Перша ж загальнодоступна бета для розробників Android 1.0 була випущена в листопаді 2007 року, хоча тоді класичної “десертної” назви вона не мала. Ця традиція з’явилась лише після випуску Android 4.4, коли Google оприлюднила офіційну заяву, щоб пояснити назву: “Оскільки ці пристрої роблять наше життя таким солодким, кожна версія

Android названа на честь десерту”. Однак у 2019 році Google відмовилась від назв десертів в час ребрендингу з випуском Android 10 “Q”.

Операційна система Android дуже добре оптимізована для роботи з усіма функціями, що існують в пристроях під її керуванням. Щодо самих функцій, їх на диво велика кількість, як то управління пам’яттю, вживання CPU, глибокий менеджмент процесів та файлів, гнучне управління протоколами спілкування, наприклад HSDPA, EV-DO, GPS, Bluetooth, т.д.

На даний момент остання стабільна версія Android - Android 11 та доступні для розробників альфа та бета тести Android 12.

2 АНАЛІЗ ІНСТРУМЕНТІВ ДЛЯ ANDROID РОЗРОБКИ

2.1 Огляд існуючих інструментів

Мною було досліджено всі мови програмування, що надають інструментарій для роботи з операційною системою Android і я виділив п’ять мов:

- Python;
- C#;
- C++;
- Kotlin;
- Java.

Кілька слів про кожну з них:

2.1.1 Python - мультиплатформна інтерпретована мова високого рівня з відкритим вихідним кодом, динамічною типізацією та автоматичною збіркою сміття. Python підтримує декілька парадигм програмування, наприклад процедурне, функціональне та об’єктно-орієнтоване програмування. Весь дизайн коду цієї мови пропагандує зрозумілість, логічність, читабельність коду для проектів самого різного масштабу.

2.1.2 C# - мультиплатформна компільована мова високого рівня з закритим вихідним кодом, статичною типізацією та автоматичною збіркою сміття. C# підтримує такі парадигми програмування як об’єктно-орієнтоване, компонентно-орієнтоване, функціональне.

2.1.3 C++ - мультиплатформна компільована мова з закритим вихідним кодом та можливістю маніпулювання пам'яттю низького рівня. C++ було розроблено з орієнтацією на системне програмування та вбудоване програмне забезпечення з обмеженими ресурсами, але з часом C++ розвивалась і тепер має об’єктно-орієнтовані, функціональні функції.

2.1.4 Kotlin - мультиплатформна компільована мова загального призначення із статичною типізацією та можливістю автоматичного визначення типу виразу. Kotlin призначений для взаємодії з Java і версія JVM стандартної бібліотеки Kotlin залежить від Java Class Library. І хоча Kotlin залежить від JVM, вона може компілюватись в JavaScript або нативний код (через LLVM).

2.1.5 Java - мультиплатформна компільована мова високого рівня з статичною типізацією, відкритим вихідним кодом та автоматичною збіркою сміття. Java підтримує декілька популярних парадигм програмування, наприклад об’єктно-орієнтоване, аспектно-орієнтоване, компонентно-орієнтоване, функціональне.

2.2 Вибір найбільш оптимального інструмента

Усі наведені мови є C-подібними, всі мультиплатформні, усі мають підтримку сучасних парадигм програмування та в тій чи іншій мірі комфортний читабельний синтаксис. Для того щоб обрати найбільш оптимальну мову програмування для розробки Android додатків, потрібно дослідити кожен конкретний інструментарій. Після дослідження наведена коротка вижимка з прочитаних матеріалів відносно кожної мови:

- Python має дуже багато технологій для різних сфер розробки програмного забезпечення, тому на перший погляд використовувати

для розробки програм для операційної системи Android доцільно з допомогою наприклад бібліотеки Kivy - бібліотеки з відкритим кодом, що використовується для розробки мобільних додатків і підтримує швидку розробку додатків. Проте недоліком цього варіанту є те, що Kivy не підтримує вбудованих функцій Android, що сильно обмежує можливості розробки додатків за заданою темою.

- C# дуже схожий на Java, тому він досить добре підходить для поставленої задачі. C# має більш простий синтаксис, ніж Java, що робить розробку простішою. До того ж C# тепер з допомогою Xamarin має інструментарій для розробки нативних програм для операційної системи Android. Єдиним недоліком на даний момент я вважаю недостатньо розвинену IDE саме для Android розробки.
- C++ можна використовувати для Android розробки з допомогою Android Native Development Kit або NDK. Однак, нажаль, програму не вийде створити повністю з допомогою C++ тому, що NDK використовується для реалізації частин програми на C++, хоча його бібліотеки і можна використувати наприклад з C# за потреби. Також потрібно згадати набагато складніше налаштування і меншу гнучкість в порівнянні з іншими засобами.
- Kotlin є наразі офіційною мовою розробки для операційної системи Android з 2019 року. До того ж, Kotlin видаляє деякі, невикористовувані функції Java. Тобто на мою думку Kotlin набагато простіший для початківця в порівнянні з Java і його можна використовувати як точку входу для розробки Android програм.
- Java є найбільш підтримуваною мовою мовою Google, має чудову онлайн-спільноту для підтримки у разі будь яких проблем, більшу кількість оптимізованих рідних бібліотек, ніж всі інші інструменти. Також потрібно згадати Android Studio - IDE спеціально для розробки Android додатків, що означає більшу кількість інструментів, поставку в пакеті усіх можливих версій та снапшотів

операційної системи Android. Це є перевагою перед C#, яка не надає такої кількості інструментів, та налаштовується складніше.

Отже, проаналізувавши можливі засоби я вирішив рекомендувати для курсу лабораторних робіт Java та IDE Android Studio.

3 ПЛАН ЛАБОРАТОРНИХ РОБІТ

В цьому розділі наведено складений приблизний план лабораторних робіт навчального курсу “Розробка комунікаційних програм для операційної системи Android, що користуються ресурсами веб-служб REST/SOAP”. Також додано лабораторну роботу №1 в якості прикладу.

3.1 Лабораторна робота №1: Створення стандартної програми для операційної системи Android

1. Теоретичні відомості до виконання лабораторної роботи
   В цьому розділі наводяться теоретичні відомості відносно основних функцій та інструментарію Java для розробки додатків, а також наводяться посилання для більш глибокого вивчення теми.
2. Налаштування робочого місця
   Цей розділ надає інструкцію з установки IDE Android Studio та рекомендації щодо корисних гарячих клавіш.
3. Створення проекту
   Тут розповідається про можливі варіанти проектів, що надає Android Studio та надається інструкція по створенню базового типового проекту програми для Android, з яким потім буде проводитись робота.
4. Розробка та запуск додатку
   В цьому пункті студент отримує завдання по проектуванню, розробці, та тестуванню свого додатку.
5. Контрольні запитання
   Список запитань для підготовки до лабораторної роботи.

3.2 Лабораторна робота №2: Тестування віддаленого RESTful сервісу та налаштування комунікації між додатком та сервісом

1. Теоретичні відомості до виконання лабораторної роботи
   Наводяться відомості щодо розгортання віддалених сервісів, інструментів для спілкування через HTTP, що надає Java, посилання на віддалений сервіс, з яким буде проводитись робота та список методів спілкування.
2. Тестування роботи віддаленого RESTful сервісу
   Інструкція з встановлення засобу відправлення запитів Postman та тестування роботи віддаленого RESTful сервісу.
3. Встановлення з'єднання між віддаленим сервісом та Android додатком
   Створення об'єктів класів, налаштування конфігурацій програми для правильного спілкування з сервісом та створення графічного інтерфейсу для керування з'єднанням.
4. Додатковий пункт: знаходження додаткових функцій сервісу
   Знаходження прихованого функціоналу сервісу, використовуючи знання принципів REST API та HTTP методів.
5. Контрольні запитання

3.3 Лабораторна Робота №3: Створення SOAP з'єднання між пристроями під керуванням операційної системи Android

1. Теоретичні відомості
   Відомості про протокол SOAP та інструменти операційної системи Android, з якими SOAP може працювати.
2. Налаштування конфігурацій для віддаленого спілкування між пристроями з допомогою протоколу SOAP.
   Надається шаблон конфігурацій для обміну повідомленнями між пристроями з допомогою веб-служби SOAP.
3. Запуск спілкування
   Спроби з'єднати пристрій та передати просте повідомлення.

4. Додатковий пункт: адаптування програми для використання на телевізорі або годиннику з системою Android
5. Контрольні питання

4 ПРИКЛАДИ ЛАБОРАТОРНИХ РОБІТ

4.1 Приклад. Лабораторна робота №1: створення стандартної програми для операційної системи Android

Мета: ознайомлення з інструментарієм, що надає Java для розробки програм на операційній системі Android, установка, опанування навичок роботи з IDE Android Studio та створення першої програми для операційної системи Android.

1. Теоретичні відомості до виконання лабораторної роботи

   Java - С-подібна мультиплатформна мова високого рівня з функцією автоматичної збірки сміття, що робить процес розробки більш комфортним. Ця мова підтримує об’єктно орієнтоване програмування, що вивчалось в навчальному курсі “Основи програмування” на першому курсі, а отже основні принципи розробки зрозумілі всім студентам, тож нижче будуть наведені лише посилання для ознайомлення з кореневими бібліотеками [1].

   До того ж Java мова з відкритим вихідним кодом, великою спільнотою та дуже функціональними IDE, тож якщо раптом виникає проблема, Java надає достатню кількість засобів для її вирішення швидко та якісно.

   Java for Android використовує інструменти керування проектами Gradle та подачі структурованої інформації XML. З допомогою Gradle підключаються сторонні інструменти з віддалених сховищ, проходить контроль версій, специфікація збірок проекту і т.д. З допомогою XML програма створює інтерфейс користувача.

Якщо хочете дізнатись більше, ось посилання на документації:[12], [13], [14].

2. Налаштування робочого місця
    a. Перейдіть за посиланням та завантажте останню версію Android Studio: [15]

Рисунок 2. приклад посилання для завантаження IDE Android Studio

    b. Запустіть інсталятор, проведіть установку IDE Android Studio та налаштуйте відповідно до вашого . HINT: переконайтесь, що ваше робоче місце має встановлену java jdk!
    c. Ознайомтесь з комбінаціями клавіш, що будуть корисними для роботи з Android Studio, особливо з розділами Build and run та Navigating and searching within Studio - [16].
3. Створення проекту
    a. Після завершення конфігурації програми відкривається вікно “Ласкаво просимо до Android Studio”, обираємо пункт “Створити новий проект”:

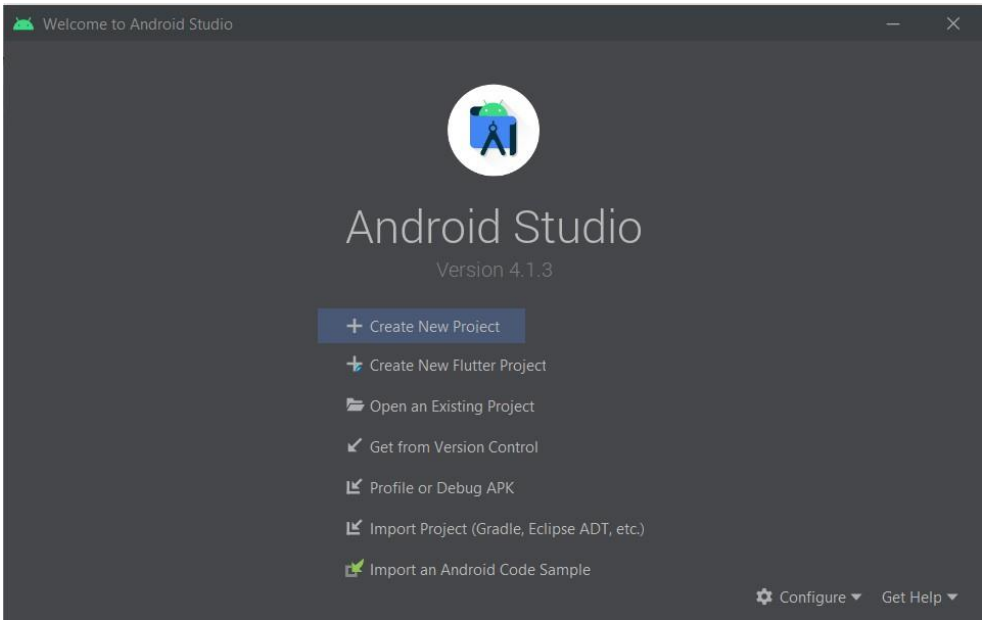

Рисунок 3. початковий екран IDE Android Studio

b. В наступному вікні ви бачите шаблони проектів для різних пристроїв що використовують операційну систему Android, та короткі описи до них. Подивіться та прочитайте можливі варіанти додатків для різних пристроїв та сформулюйте свою думку відносно кількості базових шаблонів, що надає Android Studio розробникам.

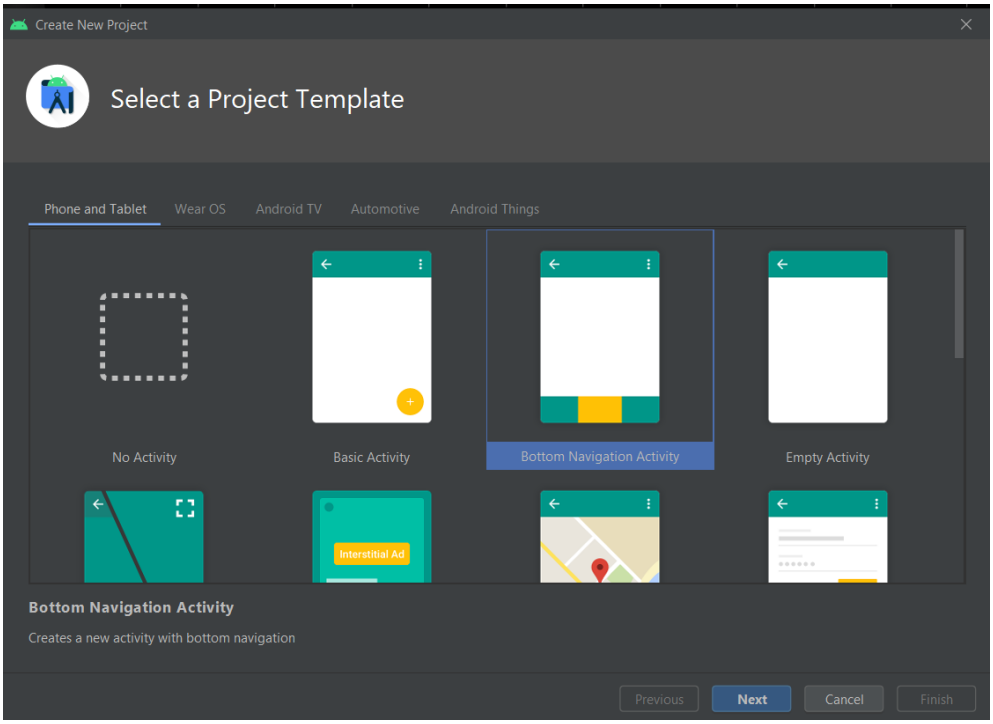

Рисунок 4. обрання шаблону проекту Android Studio

c. Оберіть шаблон “Basic Navigation Activity” та перейдіть далі:

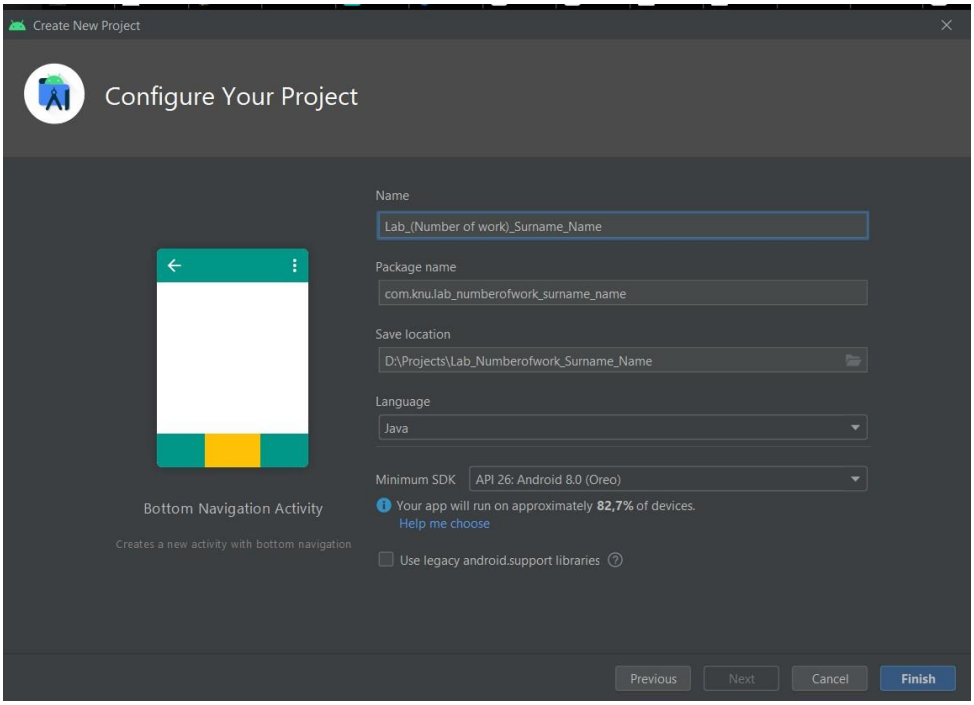

Рисунок 5. конфігурація проекту IDE Android Studio

В цьому вікні введіть назву проекта та пакета відносно шаблону, оберіть мінімальну версію Android, при якій майбутня програма зможе запуститися та натисніть Finish.

HINT: якщо Android Studio вперше відкриває проект, може бути відкрито багато вікон і панелей. Щоб полегшити собі життя та не закинути виконання лабораторно в перші ж секунди, рекомендується натиснути на всіх незрозумілих вікнах кнопку згортання (-) у верхньому правому куті (не плутати з кнопкою згортання IDE).

4. Розробка та запуск додатку

Після відкривання проекту ви побачите приблизно таку картину:

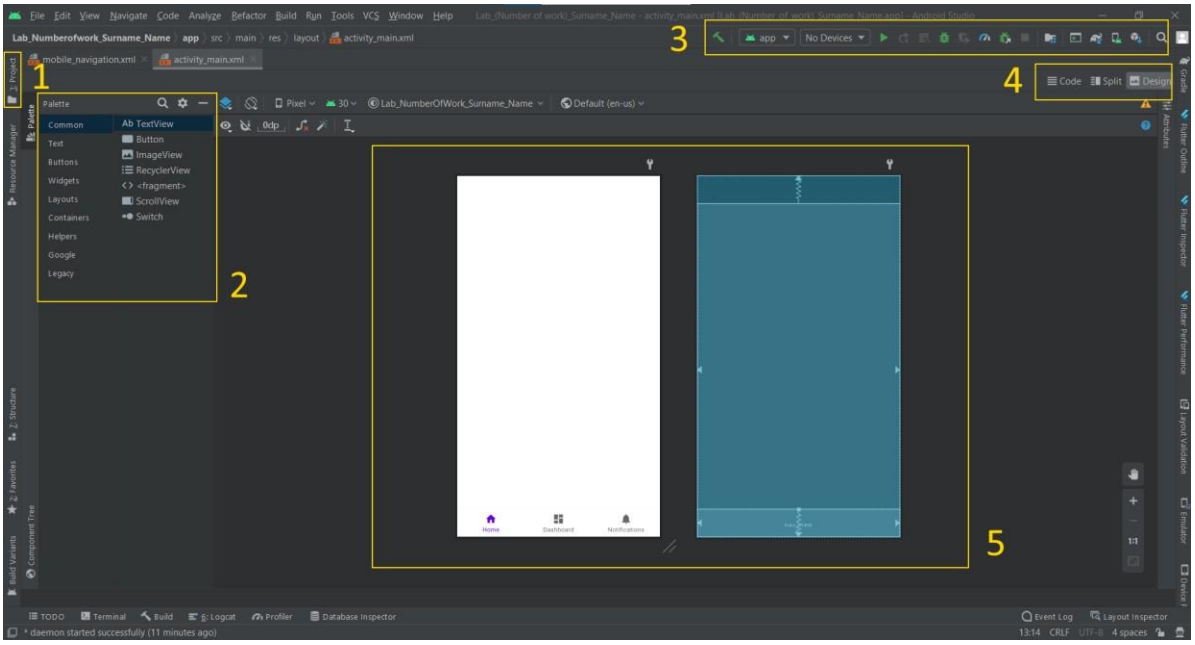

Рисунок 6. головний екран проекту Android Studio з поясненнями

1. Панель проекту - подання ієрархії усіх класів та ресурсів, що є в програмі.

2. Палета - список всіх можливих елементів, для вставки. Зазначте, що елементи можна вкладати один в одного.

3. Панель швидкого управління проектом - будування запуск, відлагодження, вибір емулятора або реального пристрою для запуску/відлагодження програми, створення нових емуляторів і т.д.

4. Панель вигляду - панель що дозволяє вам переключатись між кодом XML та представленням користувача для збільшення зручності розробки та запобіганню багатьох проблем пов’язаних з “фронт-енд розробкою”.

5. Code-space - територія де власне створюється програма.

Завдання кожного студента полягає в тому, щоб дослідити згенерований автоматично код, реалізувати та протестувати на працездатність програму навігації між трьома сторінками. Вміст сторінки достатньо придумати самому на тему варіанту.

Варіанти:

Таблиця 1. Список варіантів лабораторної роботи №1

Номер варіанту	Опис
1	KNU HUB: 1. Home - сторінка зі списком усіх курсів 2. Dashboard - сторінка зі списком усіх курсів до котрих належить конкретний студент 3. Calendar - сторінка з календарем для подій
2	Youtube: 1. Home - сторінка зі списком відео та картинками по бажанню 2. Subscriptions - сторінка зі списком улюблених відео з сайту Youtube 3. Me - сторінка з основною інформацією про користувача
3	Netflix: 1. Home - сторінка зі списком фільмів 2. New - сторінка зі списком останніх фільмів що вийшли на момент роботи над лабораторною 3. Downloads - сторінка зі списком завантажених фільмів
4	Viber: 1. Chats - сторінка зі списком чатів користувача 2. Contacts - сторінка зі списком контактів 3. Account - сторінка з даними про користувача
5	Дзвінки: 1. Останні - сторінка зі списком останніх дзвінків 2. Контакти - сторінка зі списком контактів 3. Обрані - сторінка зі списком контаків що позначаються як вибрані на сторінці “Контакти”
6	Дія: 1. Документи - сторінка зі списком документів користувача 2. Послуги - сторінка зі списком послуг 3. Повідомлення - сторінка з останніми оновленнями та повідомленнями
7	Google Play: 1. Ігри - сторінка зі списком ігор що є на

	телефоні 2. Додатки - сторінка зі списком усіх додатків телефону 3. Фільми та книги - сторінка зі списком останніх прочитаних студентом книг та переглянутих фільмів
8	Карти: 1. Карта - сторінка з картою 2. Збережене - сторінка зі списком збережених місць студента 3. Сповіщення - сторінка зі списком останніх сповіщень за шаблоном оригінального додатку
9	Monobank: 1. Картка - сторінка зі списком карток та історією операцій для кожної з них 2. Кешбек - сторінка зі списком категорій кешбеку та кількістю накопичених грошей в кожній категорії 3. Ще - сторінка зі списком додаткових функцій оригінального додатку
10	Файловий менеджер: 1. Категорії - сторінка зі списком основних категорій файлів на телефоні 2. Сховище - сторінка зі списком файлів та папок з пристрою 3. Відомості - сторінка з відомостями про кількість вільного місця на диску
11	LinkedIn: 1. Home - сторінка зі списком новин з позначенням автора новини 2. My Network - сторінка зі списком пропозицій для додавання людей в друзі 3. Jobs - сторінка зі списком вакансій
12	УкрЗалізниця: 1. Пошук - сторінка з полями для введення початкової та кінцевої точки маршруту та кнопкою додавання квитка по цьому маршруту 2. Квитки - сторінка зі списком доданих квитків

	3. Кабінет - сторінка з інформацією про “юзера”
13	Годинник: 1. Будильник - сторінка зі списком будильників, кнопкою ввімкнення конкретного будильника та виведення часу пристрою 2. Секундомір - сторінка з секундоміром 3. Таймер - сторінка з таймером зворотного відліку
14	Bolt Food: 1. Home - сторінка зі списком ресторанів 2. Search - сторінка з пошуком серед усіх ресторанів по назві 3. History - сторінка зі списком минулих замовлень

Звіт з лабораторної роботи має містити:

- Скріншоти виконання кроків лабораторної роботи;
- власні думки щодо заданих в лабораторній роботі питань;
- посилання на проект на ресурсі github з робочим кодом лабораторної роботи.

5. Контрольні запитання

1. Відмінності Java від раніше вивчених вами мов програмування?

2. Що таке Gradle та навіщо він потрібен?

3. Який функціонал надає IDE Android Studio?

4. Яка комбінація клавіш використовується переходу до останнього місця редагування коду?

5. Назвіть три шаблони проекту, що надає Android Studio розробникам що здаються вам найбільш використовуваними та поясніть чому.

4.2 Приклад оформлення звіту до лабораторної роботи №1

Мета: ознайомитись з інструментами розробки програм для операційної системи Android та створити з їх допомогою тестову програму.

1. Перейдіть за посиланням та завантажте останню версію Android Studio: [15]

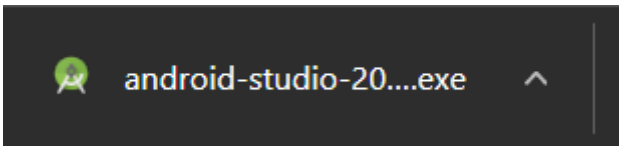

Рисунок 7. приклад завантаженої програми Android Studio

2. Запустіть інсталятор, проведіть установку IDE Android Studio та налаштуйте відповідно до вашого вподобань. У вікні “Import Android Studio settings” натисніть OK.

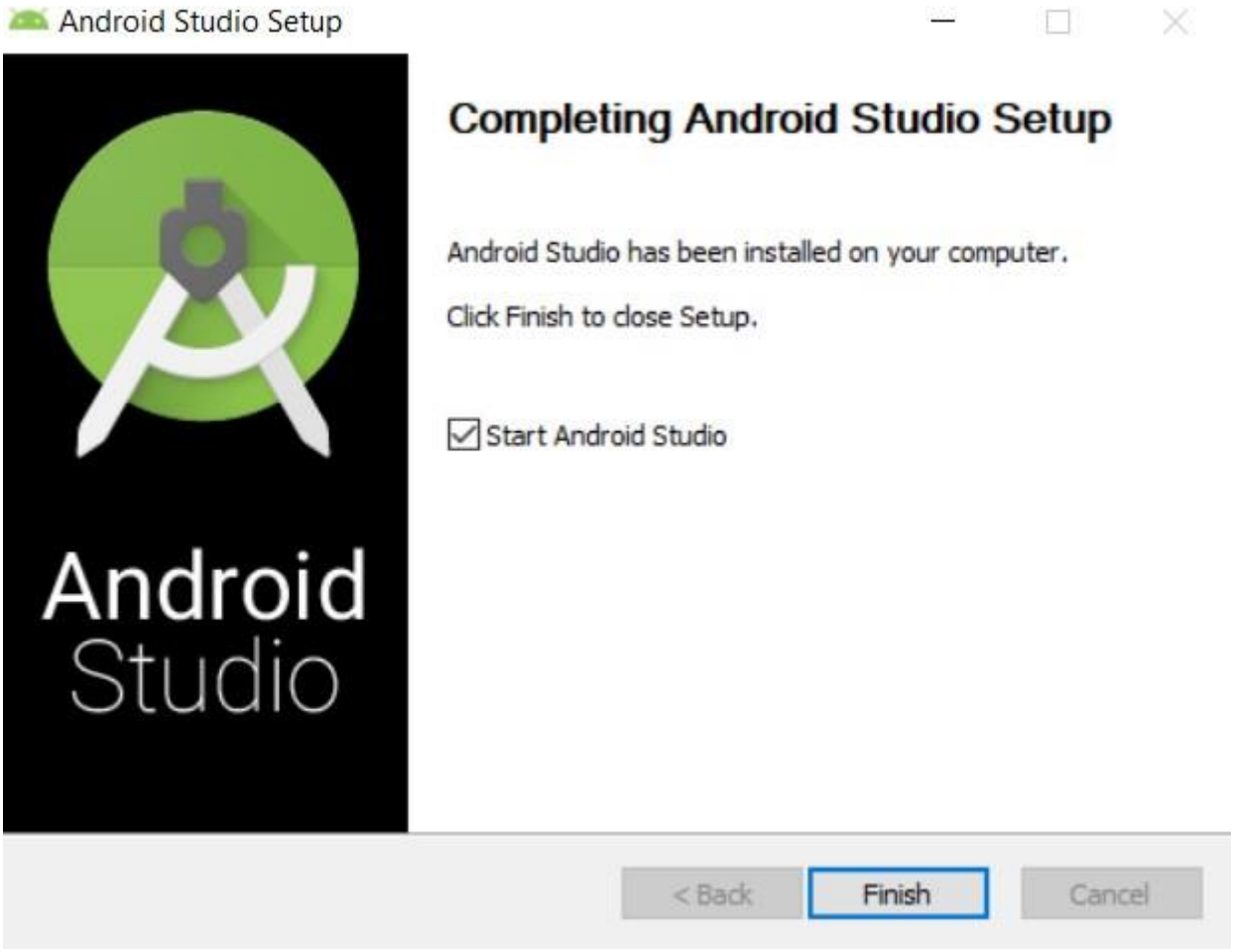

Рисунок 8. установка програми Android Studio

3. Створення проекту
    a. Після завершення конфігурації програми відкривається вікно “Ласкаво просимо до Android Studio”, обираємо пункт “Створити новий проект”:

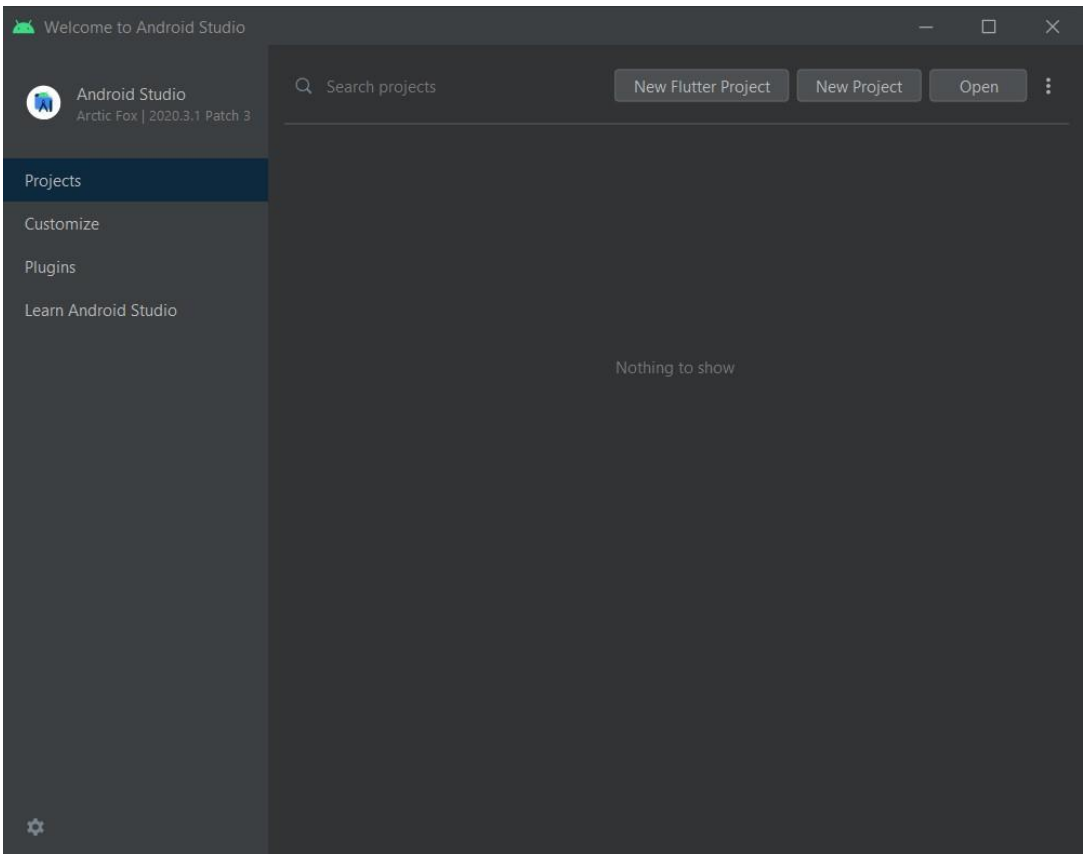

Рисунок 9. стартова сторінка програми Android Studio

b. В наступному вікні ви бачите шаблони проектів для різних пристроїв що використовують операційну систему Android, та короткі описи до них. Подивіться та прочитайте можливі варіанти додатків для різних пристроїв та сформулюйте свою думку відносно кількості базових шаблонів, що надає Android Studio розробникам. Розробники Android Studio надають шаблони для створення практично всіх типів додатків Android додатків, що можуть спасти на думку,до того ж якщо переглянути детальніше вікна з шаблонами, можна зрозуміти, що розробники надають навіть шаблони деяких конкретних функцій як то панель налаштувань, рекламні вставки в додатках та google maps. На мою думку, це говорить про небайдужіть розробників до людей, що будуть працювати з IDE їх виробництва та стеження за основними потребами ринку.

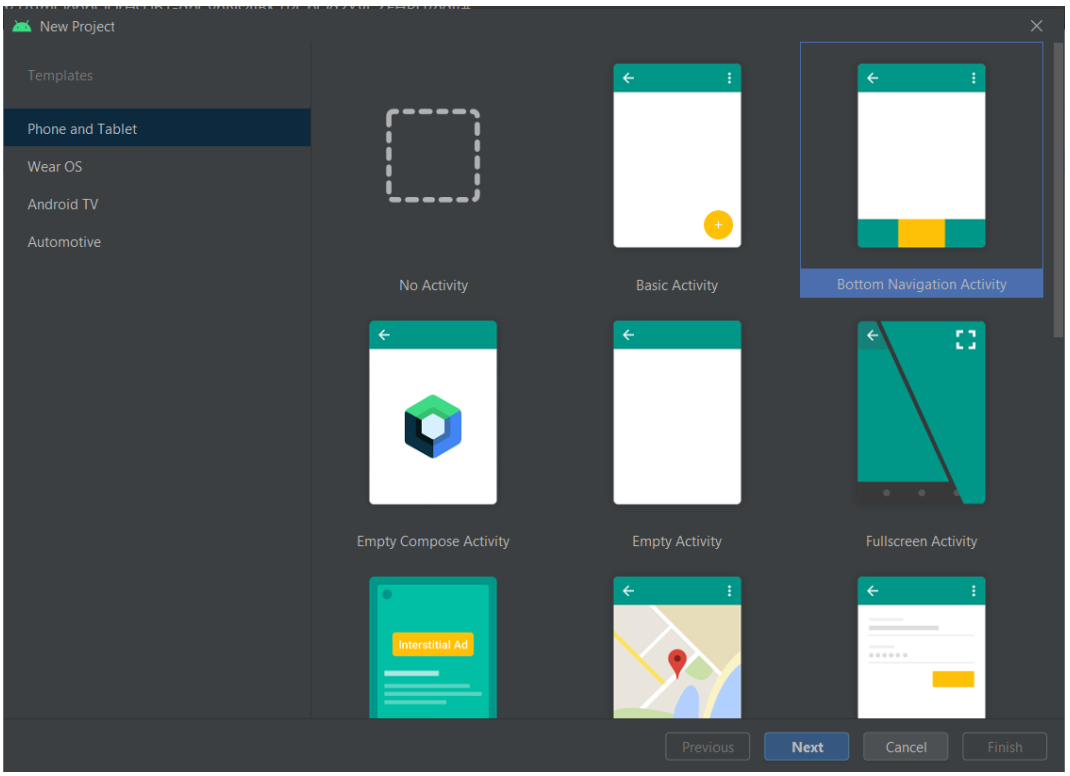

Рисунок 10. обрання шаблону проекту в Android Studio

c. Оберіть шаблон “Bottom Navigation Activity” та перейдіть далі:

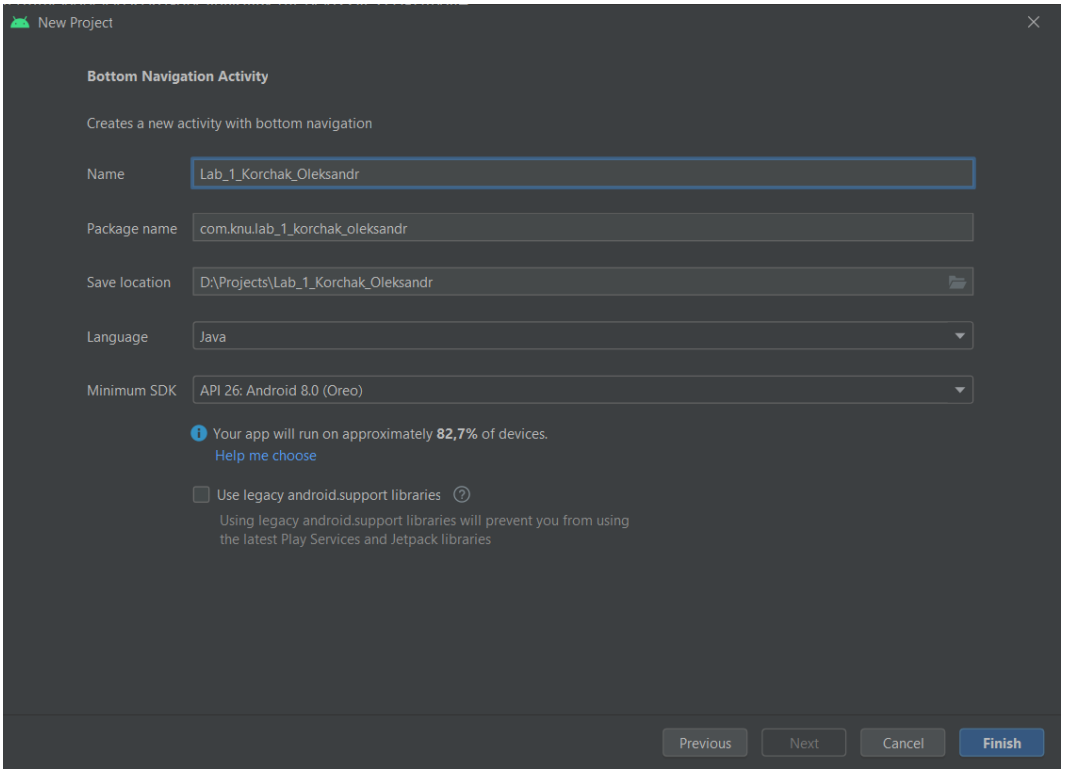

Рисунок 11. конфігурація проекту в програмі Android Studio

4. Розробка та запуск додатку

Варіант 1 - KNU HUB.

Посилання на віддалений репозиторій github: [17].

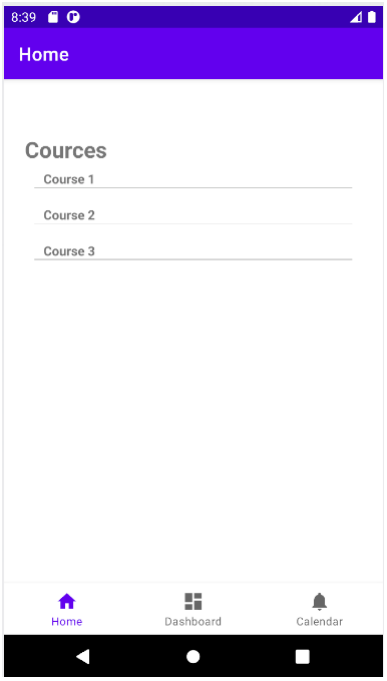

Рисунок 12. вкладка Home створеного додатку

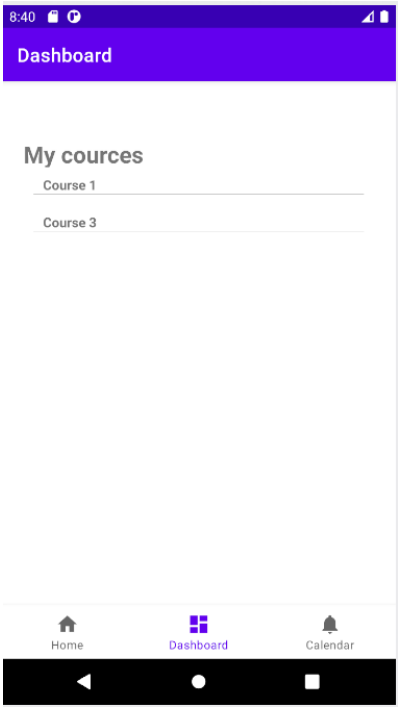

Рисунок 13. вкладка Dashboard створеного додатку

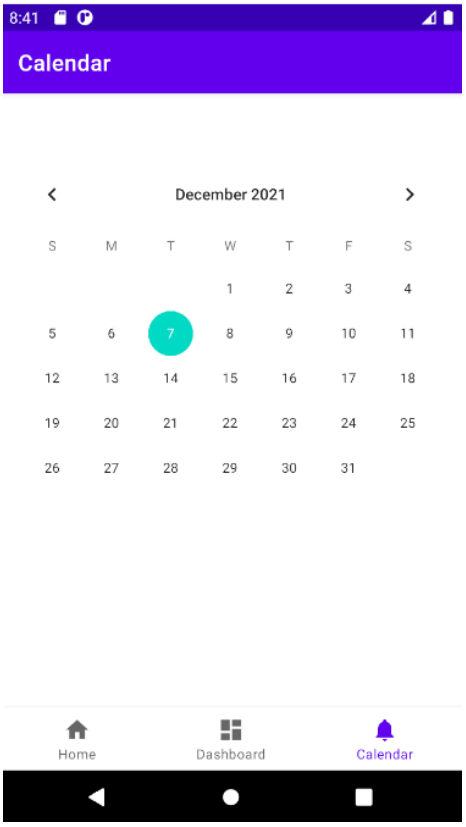

Рисунок 14. вкладка Calendar створеного додатку

Висновок: було досліджено стандартний інструментарій роботи для розробки додатків операційної системи Android з допомогою IDE Android Studio на Java. Розроблено та протестовано мобільний додаток, що емулює три сторінки інтернет-сервісу KNU HUB.

4.3 Приклад. Лабораторна робота №2: Тестування віддаленого RESTful сервісу та налаштування комунікації між додатком та сервісом

Мета: ознайомлення з інструментами розробки додатків для спілкування з віддаленим сервісом через HTTP.

1. Теоретичні відомості до виконання лабораторної роботи

   На сьогоднішній день дуже складно знайти додаток для будь якої операційної системи, що не буде мати методів спілкування з віддаленим сервісом, як то для виводу реклами з Google Play чи надсилання даних до хмари. При цьому не важливо чи написаний сервіс наприклад на Java чи

Python, чи використовується для розробки та запуску сервісу наприклад Apache чи Nginx, спілкування проходить через запити HTTP. Але різні розробники можуть створити абсолютно різні ендпоінти для запитів. Це сильно ускладнює життя їхнім колегам так як доводиться кожного разу дізнаватись заново усі параметри, методи засобів, чи буде конкретно в цьому запиті відповідь і чи потрібне тіло в цьому конкретному POST запиті. Тому, щоб полегшити та уніфікувати написання методів використовуються принципи архітектурного стилю REST. Сервіси ж що повністю притримуються усіх принципів називають RESTful. Ця лабораторна робота створена, щоб познайомити студентів з REST та допомогти зрозуміти усі плюси цього архітектурного стилю та як написати власний запит до віддаленого сервісу.

Так як для написання лабораторних робіт використовується мова програмування Java, список класів, котрі можна застосувати для створення запиту:

a. HttpUrlConnection
b. Volley
c. OkHttp
d. Retrofit

Посилання на віддалений сервіс: [18].

Список методів спілкування:

a. GET - url + GetBooks
b. DELETE - url + {id}
c. POST - url + PostAddOneBook - потрібно записати у тіло запиту JSON об’єкт з такими ж назвами як отримані з допомогою запиту GET.
d. POST - url + PostAddListOfBooks - у тіло запиту вставляється масив JSON з об’єктами котрі ми хочемо додати.

2. Тестування роботи віддаленого RESTful сервісу

Для тестування роботи віддалених сервісів використовуються засоби відправлення запитів такі як Postman [19].

Після встановлення запускаємо засіб:

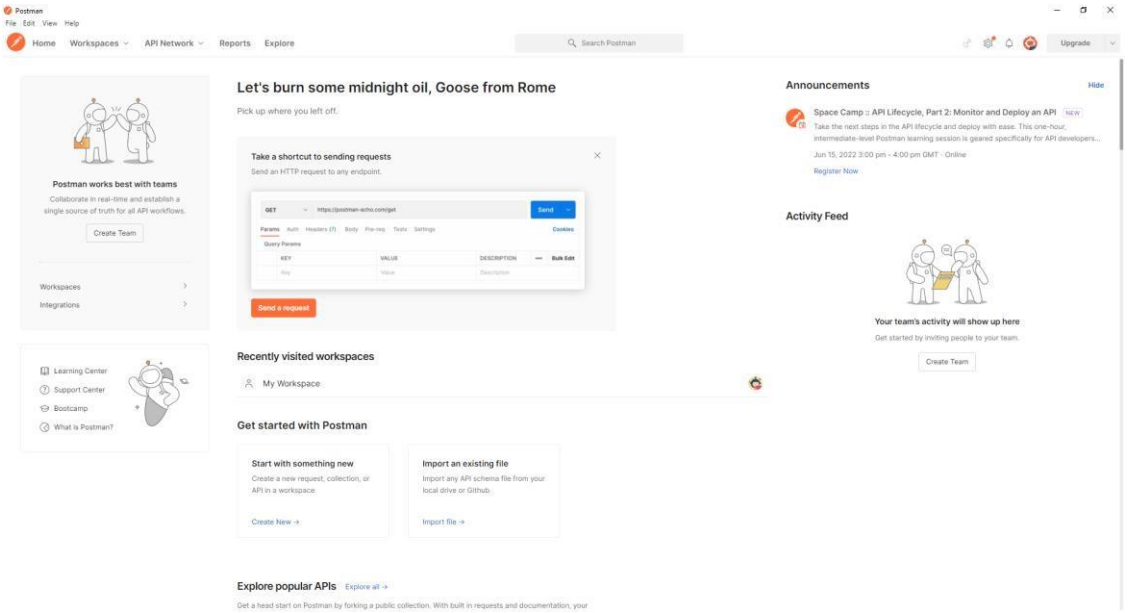

Рисунок 15. головна сторінка засобу Postman

Для того, щоб створити запит натискаємо на відповідну кнопку, вводимо посилання у поле “Enter request URL”, обираємо метод запиту та надсилаємо запит. У вкладці “Response” після обробки запиту на сервері з’являється відповідь, якщо вона є та в правій частині цієї вкладки виводиться код відповіді.

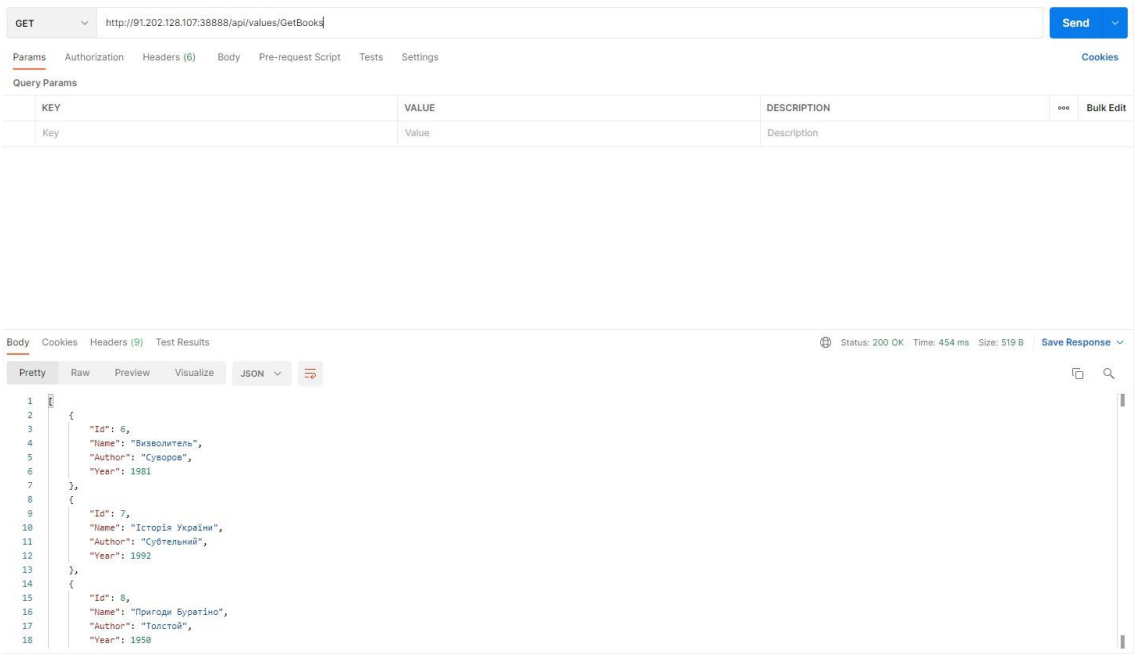

Рисунок 16. приклад надсилання GET запиту та отримання відповіді

В попередньому пункті наведено список методів для роботи з віддаленим сервісом, протестуйте їх та наведіть результати запитів.

3. Встановлення з’єднання між віддаленим сервісом та Android додатком
   Завдання студентів полягає в тому щоб створити додаток з графічним інтерфейсом що коректно надсилає всі вищезадані запити, обробляє та виводить відповіді на екран.

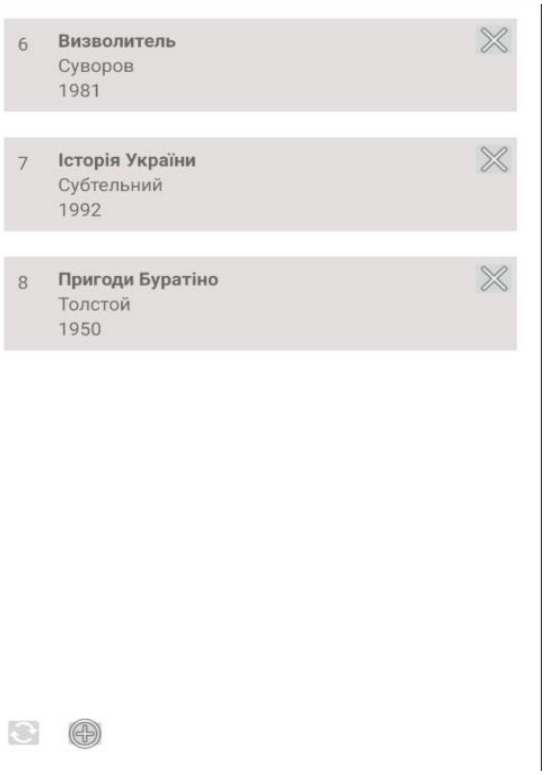

Рисунок 17. приклад кінцевого додатку

Звіт з лабораторної роботи має містити:

- скріншоти виконаних кроків лабораторної роботи
- посилання на проект github

HINT: не забувайте дозволити в маніфесті проекту використовувати інтернет з’єднання та “CleartextTraffic”.

4. Додатковий пункт: знаходження інших функцій сервісу
   Знайти прихований функціонал сервісу, використовуючи знання принципів REST API та HTTP методів.
5. Контрольні запитання
   a. Які основні методи запитів HTTP?
   b. Які класи використовуються для створення запитів мовою Java?
   c. Які типи даних в відповіді підтримує засіб Postman?
   d. За дотримання яких принципів сервіс можна називати RESTful?

4.4 Приклад оформлення звіту до лабораторної роботи №2

Мета: ознайомлення з інструментами розробки додатків для спілкування з віддаленим сервісом через HTTP.

1. Тестування роботи віддаленого RESTful сервісу

a. Запит на отримання усіх книг:

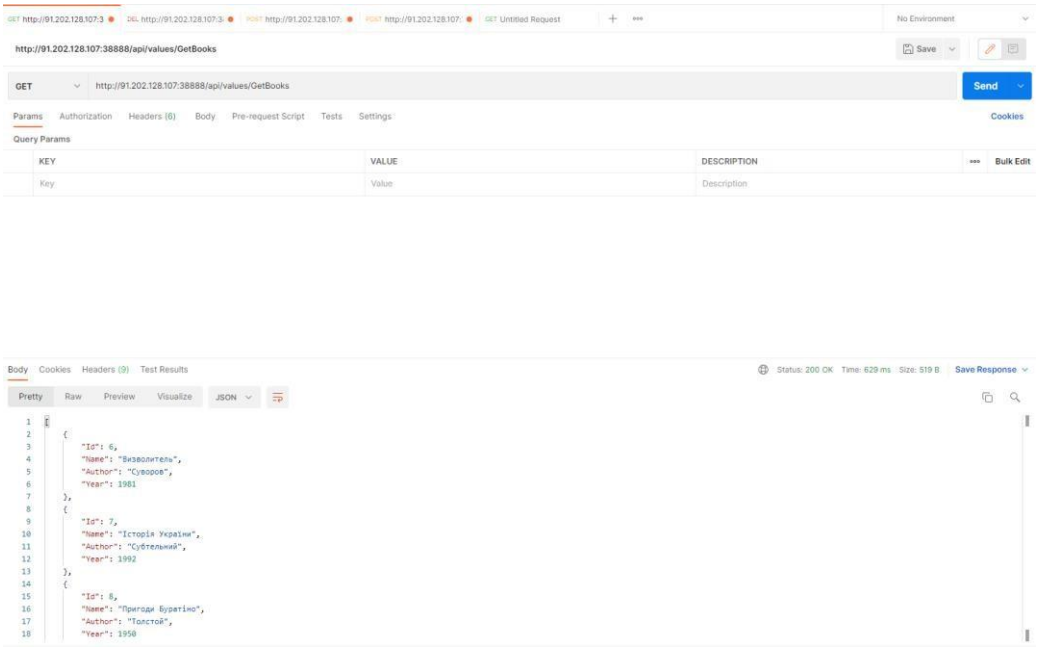

Рисунок 19. GET запит в Postman

Як видно на скріншоті, у відповідь ми отримуємо масив JSON, в якому кожен елемент це окремий об’єкт JSON.

b. Запит на видалення книги по id:

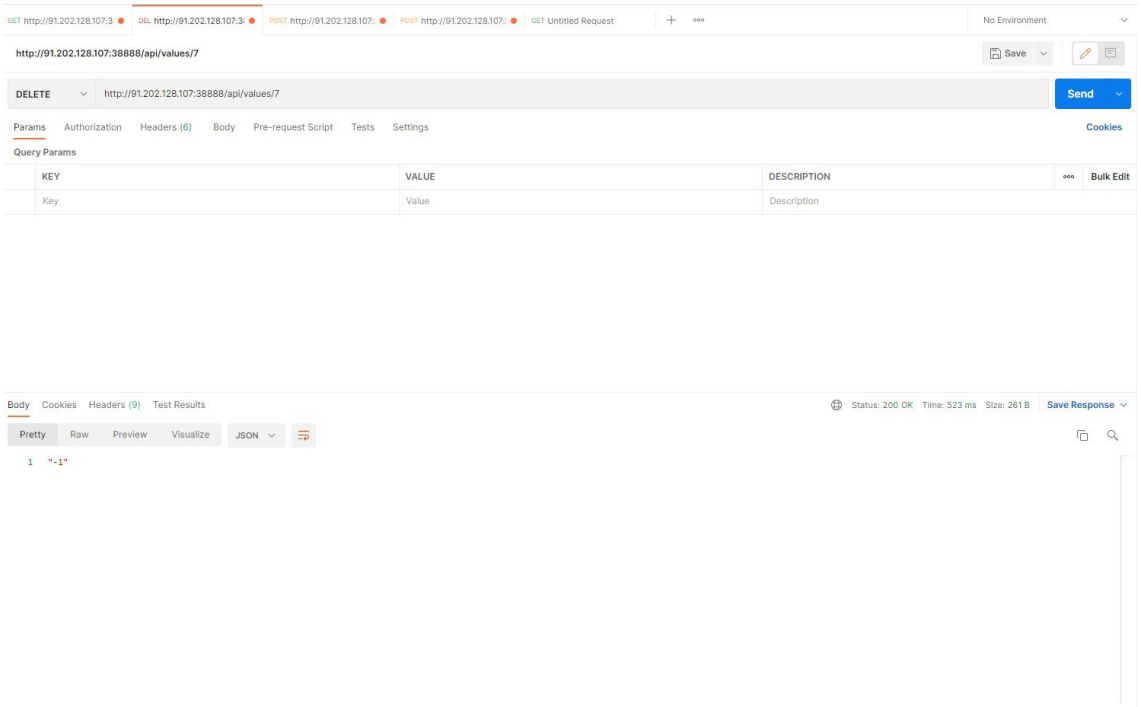

Рисунок 20. DELETE запит в Postman

На цьому скріншоті у відповідь приходить -1 що означає що елемент з таким id був видалений з бази віддаленого сервісу.

с. Запит на додавання однієї книги:

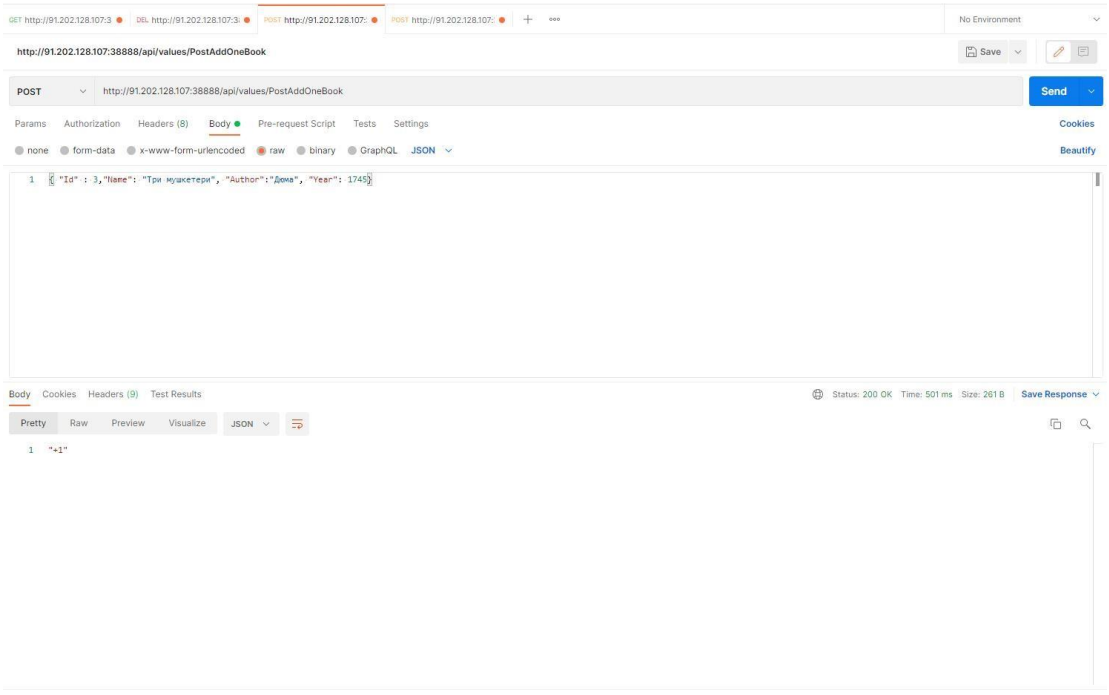

Рисунок 21. POST запит в Postman для додавання однієї книги

Тут ми маємо тіло запиту, що буде записуватись до бази даних сервісу в форматі об’єкту JSON. У відповідь приходить +1, що означає успішне додавання книги.

d. Запит на додавання списку книг:

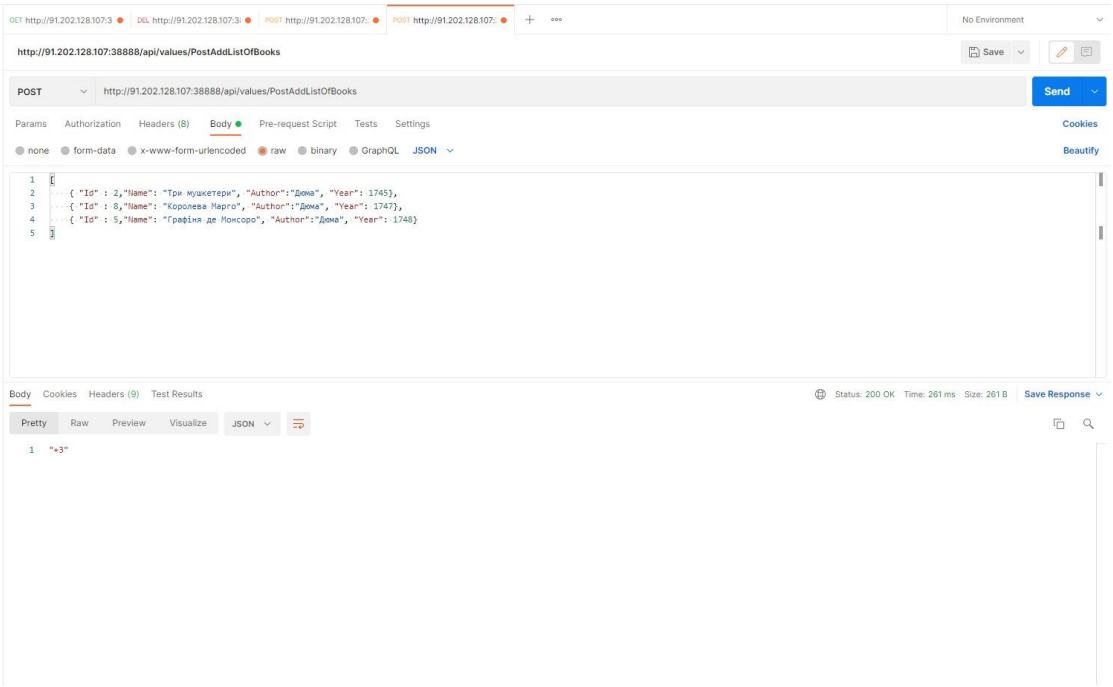

Рисунок 22. POST запит в Postman для додавання списку книг

Так само як і в попередньому запиті ми повинні заповнити тіло запиту, цього разу як масив JSON, у відповідь при успішному додаванні отримуємо скільки елементів додалось до бази даних сервісу. У цьому випадку це 3 книги.

2. Встановлення з’єднання між віддаленим сервісом та Android додатком

Посилання на репозиторій github: [20]

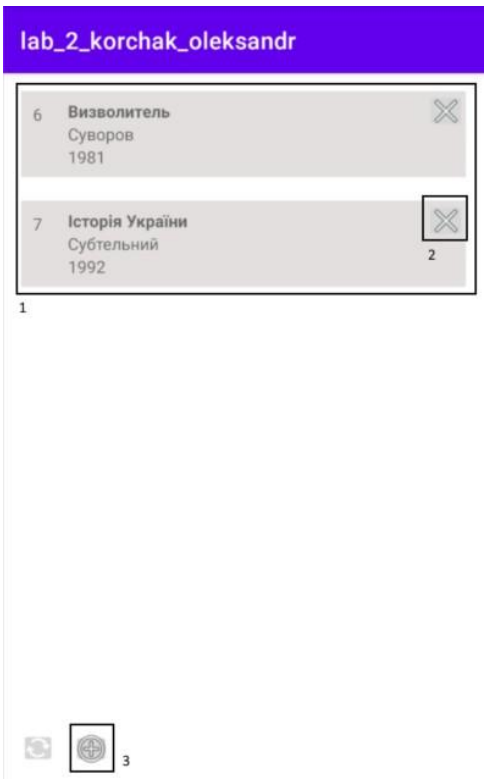

Рисунок 23. Головна сторінка додатку з позначеннями:

1. Список книг
2. Видалення книги
3. Додавання книги або книг

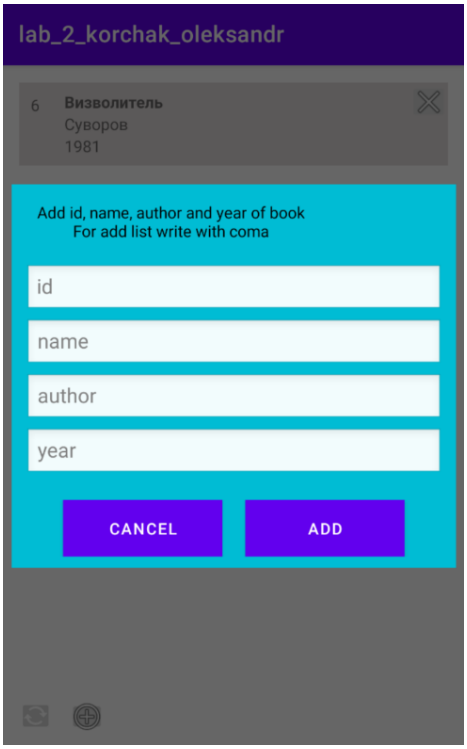

Рисунок 24. Діалог додавання книг

3. Додатковий пункт: знаходження інших функцій сервісу
   Знайдено та додано запит пошуку однієї книги зі списку за Id, це як і запит GetBooks, метод GET запиту.

Висновок: під час виконання цієї лабораторної роботи був досліджений засіб відправки запитів Postman, класи Java для створення HTTP запитів та розроблена програма-”бібліотека”.

4.5 Приклад. Лабораторна Робота №3: Створення SOAP з’єднання між пристроями під керуванням операційної системи Android

1. Теоретичні відомості
   SOAP - Simple Object Access Protocol - протокол призначений для обміну структурованою інформацією в децентралізованому розподіленому середовищі. Основні цілі SOAP - простота та розширюваність, що досягається з допомогою специфікації повідомлення через XML. SOAP-повідомлення формально визначається як набір інформації, що надає абстрактний опис його вмісту з можливо абсолютно різними серіалізаціями, за рахунок чого досягається універсальність SOAP як протоколу передачі даних для абсолютно будь якої програми в системі Android та через різні протоколи зв’язку, як то Bluetooth, потрібно лише написати обробку повідомлення що приходить.
2. Налаштування конфігурацій для віддаленого спілкування між пристроями з допомогою протоколу SOAP.
   Для використання функції Bluetooth потрібно додати декілька

обов’язкових дозволів до файлу проекту AndroidManifest.xml:

```
<uses-permission android:name="android.permission.BLUETOOTH"
    android:maxSdkVersion="30" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"
    android:maxSdkVersion="30" />

<uses-permission android:name="android.permission.BLUETOOTH_SCAN" />

<uses-permission android:name="android.permission.BLUETOOTH_ADVERTISE" />

<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />

<uses-permission android:name="android.permission.ACCESS_BACKGROUND_LOCATION" />

<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
    tools:ignore="CoarseFineLocation" />

<uses-feature android:name="android.hardware.bluetooth"
    android:required="true"/>
<uses-feature android:name="android.software.companion_device_setup"
```

Рисунок 25. Список дозволів для користування функцією Bluetooth

Також рекомендується використовувати бібліотеки для створення SOAP повідомлень такі як ksoap2. Для імплементації цього функціоналу потрібно додати до файлу settings.gradle “maven { url 'https://oss.sonatype.org/content/repositories/ksoap2-android-releases' }“ в пункт “dependencyResolutionManagement()”. Після цього потрібно додати до файлу build.gradle залежність ksoap2. Вищевказана бібліотека надає велику різноманітність можливих шаблонів SOAP повідомлень. Ще обов’язково додати запитування дозволів на використання Bluetooth та обробку підключення пари пристроїв для передачі повідомлень. Використовуючи вищевказані побажання створити програму що підключається до іншого пристрою через Bluetooth та спілкується SOAP повідомленнями з підключеним пристроєм з таким ж додатком.

3. Запуск спілкування

   Спроби з’єднати два пристрої та передати просте повідомлення. HINT: емулятор не підтримує функцію Bluetooth. Додайте свій додаток на

два телефона чи інших пристроїв під керуванням операційної системи Android.

4. Додатковий пункт: адаптування програми для використання на телевізорі або годиннику з системою Android

Звіт з лабораторної роботи має містити:

- посилання на проект на github
- висновки щодо зробленої роботи

5. Контрольні питання
    a. Основні відомості про інтерфейс Bluetooth
    b. Основні відомості про протокол SOAP

4.6 Приклад оформлення звіту до лабораторної роботи №3

Мета: ознайомлення з інструментами розробки додатків для спілкування через протокол SOAP пристроїв під керуванням операційною системою Android. Посилання на віддалений репозиторій: [21].

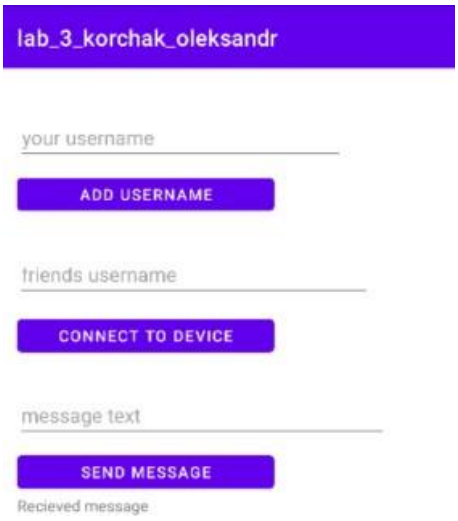

Рисунок 26. Головна сторінка додатку

Висновок: під час виконання цієї лабораторної роботи було досліджено варіанти створення Bluetooth з’єднання та SOAP повідомлень. З допомогою цього був створений додаток “месенджер”.

ВИСНОВКИ

Порівняння засобів розробки програм для операційної системи Android з дозволило зробити висновок, що найбільш універсальними і функціональними засобами є інтегроване середовище програмування Android Studio та мова програмування Java. Для того, щоб студенти легше адаптувались і призвичаїлись до цих засобів, в першій лабораторний роботі запропонована компіляція і тестування мобільного додатку, що емулює три сторінки одного серед відомих серед студентів додатків або інтернет-ресурсів.

В другій лабораторній роботі показано, що розробники можуть використовувати програму Postman для перевірки коректної роботи віддаленого HTTP-сервера, а для формування HTTP-запитів у мові Java передбачені спеціальні бібліотечні класи, зокрема клас HttpUrlConnection, з допомогою яких можна реалізувати HTTP-клієнт у вигляді мобільного додатку або класової бібліотеки з перспективою її використання іншими додатками.

Друга лабораторна робота є повноцінним клієнтом для веб-служби архітектурного стилю REST, в якій продемонстровано виклик усіх веб-функцій, що складають програмний інтерфейс веб-служби. Лабораторна робота віддзеркалює усі етапи еволюції проекту середовища програмування Android Studio: створення порожнього проекту, додавання ресурсів і програмного коду, компіляція проекту, тестування на програмному емулятора, формування синтез інсталятора як APK-файла, передача файла за протоколом FTP на смартфон, встановлення і виконання програми на смартфоні.

В третій лабораторній роботі продемонстровано, що на двох Android-пристроях можна запустити однакову програму і організувати двосторонній обмін повідомленнями між пристроями за допомогою протоколу SOAP, що працює через бездротову мережу Wi-Fi, або за допомогою бездротового інтерфейсу Bluetooth. У такий спосіб доведено, що інтерфейс Bluetooth є єдиним інтерфейсом, через який два сучасні Android-пристрої можуть бути з’єднані напряму, без допомоги точки доступу, необхідної для інтерфейсу Wi-Fi та базової станції, необхідної для мобільних інтерфейсів GSM, UMTS та LTE.

СПИСОК ДЖЕРЕЛ ПОСИЛАННЯ

[1] Operating System Market Share Worldwide [Електронний ресурс]: “statcounter” - Режим доступу до ресурсу: https://gs.statcounter.com/os-market-share/ (Дата звернення: 05.11.2021)

[2] Architectural Styles and the Design of Network-based Software Architectures Dissertation: Chapter 5: Representational State Transfer (REST): Roy Thomas Fielding [Електронний ресурс]: “ics” - Режим доступу до ресурсу: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm (Дата звернення: 05.11.2021)

[3] REST [Електронний ресурс]: “restfulapi” - Режим доступу до ресурсу: https://restfulapi.net/ (Дата звернення: 06.11.2021)

[4] SOAP Version 1.2 Part 0: Primer (Second Edition) [Електронний ресурс]: “w3” - Режим доступу до ресурсу: https://www.w3.org/TR/2007/REC-soap12-part0-20070427/ (Дата звернення: 06.11.2021)

[5] Android Developers [Електронний ресурс]: “developer.android” - Режим доступу до ресурсу: https://developer.android.com/ (Дата звернення: 06.11.2021)

[6] Python [Електронний ресурс]: “python” - Режим доступу до ресурсу: https://www.python.org/ (Дата звернення: 06.11.2021)

[7] C# docs [Електронний ресурс]: “docs.microsoft” - Режим доступу до ресурсу: https://docs.microsoft.com/en-us/dotnet/csharp/ (Дата звернення: 06.11.2021)

[8] C++ Language Reference [Електронний ресурс]: “docs.microsoft” - Режим доступу до ресурсу: https://docs.microsoft.com/en-us/cpp/cpp/cpp-language-reference?view=msvc-170 (Дата звернення: 06.11.2021)

[9] Kotlin Programming Language [Електронний ресурс]: “kotlinlang” - Режим доступу до ресурсу: https://kotlinlang.org/ (Дата звернення: 06.11.2021)

[10] What is Java technology and do I need it [Електронний ресурс]: “java” - Режим доступу до ресурсу: https://www.java.com/en/download/help/whatis_java.html (Дата звернення: 06.11.2021)

[11] Core Libraries [Електронний ресурс]: “docs.oracle” - Режим доступу до ресурсу: https://docs.oracle.com/en/java/javase/17/core/java-core-libraries1.html (Дата звернення: 07.11.2021)

[12] JDK 17 Documentation - Home [Електронний ресурс]: “docs.oracle” - Режим доступу до ресурсу: https://docs.oracle.com/en/java/javase/17/index.html (Дата звернення: 07.11.2021)

[13] Gradle [Електронний ресурс]: “gradle” - Режим доступу до ресурсу: https://gradle.org/ (Дата звернення: 07.11.2021)

[14] XML Introduction - XML: Extensible Markup Language [Електронний ресурс]: “developer.mozila” - Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/XML/XML_introduction (Дата звернення: 07.11.2021)

[15] Download Android Studio and SDK tools [Електронний ресурс]: “developer.android” - Режим доступу до ресурсу: https://developer.android.com/studio (Дата звернення: 07.11.2021)

[16] Keyboard shortcuts | Android Developers [Електронний ресурс]: “developer.android” - Режим доступу до ресурсу: https://developer.android.com/studio/intro/keyboard-shortcuts (Дата звернення: 07.11.2021)

[17] Laboratory work 1 repo [Електронний ресурс]: “github” - Режим доступу до ресурсу: https://github.com/Goosefrom/Lab_1_Korchak_Oleksandr (Дата звернення: 07.11.2021)

[18] Cloud server for laboratory work 2 [Електронний ресурс] - Режим доступу до ресурсу: http://91.202.128.107:38888/api/values/ (Дата звернення: 24.05.2022)

[19] Download Postman [Електронний ресурс]: “postman” - Режим доступу до ресурсу: https://www.postman.com/downloads/ (Дата звернення: 24.05.2022)

[20] Laboratory work 2 repo [Електронний ресурс]: “github” - Режим доступу до ресурсу: https://github.com/Goosefrom/lab_2_korchak_oleksandr (Дата звернення: 24.05.2022)

[21] Laboratory work 3 repo [Електронний ресурс]: “github” - Режим доступу до ресурсу: https://github.com/Goosefrom/lab_3_korchak_oleksandr (Дата звернення: 02.06.2022)