Одеський національний університет імені І.І.Мечникова
(повне найменування вищого навчального закладу)
Факультет математики, фізики та інформаційних технологій
(повне найменування інституту/факультету)
Кафедра теоретичної механіки
(повна назва кафедри)

Дипломна робота

на здобуття ступеня вищої освіти «бакалавр»
на тему: «Розробка клієнт-серверного додатку для інформації про банківський курс валют»
«Development Of a Client-Server Application For Information On the Bank's Exchange Rate»

Виконав: студент денної форми навчання
спеціальності 113 Прикладна математика (механіка)
Соколов Микола Володимирович

Керівник д.т.н., проф. Волков В.Е.
(науковий ступінь, вчене звання, прізвище та ініціали, підпис)

Рецензент к.т.н, доцент Косой М.Б.
(науковий ступінь, вчене звання, прізвище та ініціали, підпис)

Рекомендовано до захисту:
Протокол засідання кафедри
№ ____ від _____ 2020 р.

Завідувач кафедри
_____ Волков В.Е.
(підпис)

Захищено на засіданні ЕК №
протокол №____ від _____ 2020 р.
Оцінка _____ / _____ / _____
(за національною шкалою, шкалою ECTS, бали)

Голова ЕК
_____ Волков В.Е.
(підпис)

Одеса – 2020

Зміст

Вступ…….....…3

Постановка задачі…….....4

Розділ 1. Розробка Back-end……..…….……..6

1.1. Підключення технологій та використання *Ktor*…….....……....6

1.2. СУБД *SQLite*.…….…….…….....10

1.3. Алгоритм HTML parser.…….……....….....12

1.4. Робота API……..…….….....16

1.5. Робота з клієнтом…….…….....17

Розділ 2. Розробка Android-додатку.....….…….…….....19

2.1. Дизайн програми та технології……...….....19

2.2. Екран показу інформації про основні три валюти……...……..22

2.3. Екран показу інформації про банковські курси валют…….23

Висновки…….....……..…….....26

Список літератури…….....27

ДОДАТОК 1. Підключення бібліотек……...…….....28

ДОДАТОК 2. Класс CurrencyService …….……...…….31

ДОДАТОК 3. HTML Parser …….……...33

ДОДАТОК 4. Application.....…….38

ДОДАТОК 5. MainActivity…….…….……41

ДОДАТОК 6. CurrenciesView…….…….…..42

ДОДАТОК 7. RetrofitManager……..……..47

ДОДАТОК 8. BanksView……...48

Вступ

В сучасному світі існує багато мобільних сервісів, які полегшують людям життя. Спочатку це були тільки дзвінки, SMS, вихід в Internet, а далі, з розвитком комп'ютерних технологій з'явилися такі сервіси, як виклик таксі, перевірка балансу в банках, доставка їжі, перевірка документів онлайн, оплата комунальних послуг, музикальні платформи. Це все зараз існує як мобільні додатки та користуються попитом у користувачів. Має сенс розробити мобільний додаток, що допоможе користувачу знайти оптимальний курс банківських валют (с точки зору як обміну, так і розташування обмінного пункту).

Мета роботи полягає в тому, щоб розробити клієнт-серверний додаток на мові програмування *Kotlin* для зручності пошуку інформації щодо поточного курсу валют. В результаті роботи додатку користувачу буде не потрібно шукати найближчий банк чи пункт обміну валют, він зможе зайти до мобільного додатку та знайти найвигідніший курс обміну та банк, що пропонує такий курс.

Kotlin – це сучасна мова програмування розроблена компанією JetBrains. Ця мова є статично-типізованою, підтримує як об'єктно-орієнтоване, так і процедурне програмування [3,4].

Основні можливості та переваги Kotlin:

- компілюється в байткод JVM або в JavaScript;
- програма може використовувати всі існуючі Java-фреймворки та бібліотеки;
- Kotlin можна інтегрувати з Maven, Gradle та іншими системами збірки;
- мова є неважкою для вивчення та має простий синтаксис;
- гарантує null-безпеку;
- мову заточено під функціональне програмування.

Постановка задачі

Для розробки клієнт-серверного додатку потрібно розв'язати наступні задачі:

1. Розробити серверний додаток(API), який буде працювати з даними.
2. Розробити клієнт, в даному випадку Android додаток, який буде запитувати потрібні користувачу дані з серверу.

Серверний додаток(API) має виконувати такі задачі:

- Мати зв'язок з базою даних та змогу брати актуальні дані з бази
- Віддавати клієнту актуальні дані по запиту
- Кожні 5-10 хвилин оновлювати актуальні дані в базі даних

По суті, серверна частина являється мозком усього клієнт-серверного додатку. Для зручності розуміння, роботу усієї програми можна схематично зобразити на Рис. 1:

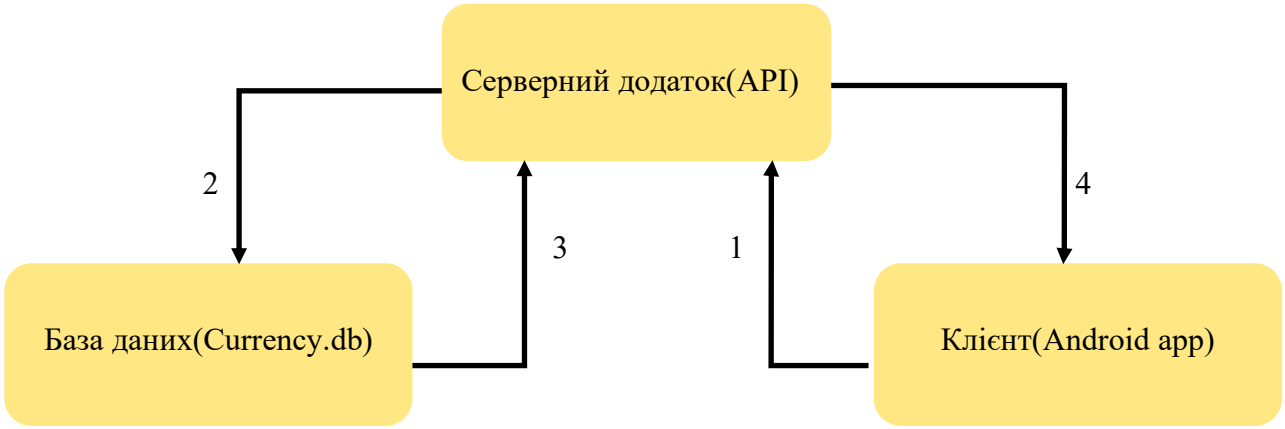

Рис. 1. Робота серверного додатку

Тобто, алгоритм роботи з клієнтом такий:

1. Клієнт запитує деякі дані на сервері (1)
2. Серверний додаток оброблює запит, та якщо він валідний, виконує запит до бази даних (2)
3. База даних повертає потрібні дані серверу (3)
4. Серверний додаток віддає актуальні дані клієнту (4)

Алгоритм роботи с оновленням даних такий:

1. Кожні 5-10 хвилин(в випадковому порядку) запускається функція, яка ходить на сайт з актуальним курсом валют, бере HTML-документ з цього сайту та за html-тегами, назвами класів, знаходить потрібну інформацію
2. Збирає отриману інформацію в зручну модель для роботи з базою даних
3. Оновлює інформацію в базі даних.

Тобто клієнт не може знати, що буде зберігатися в базі даних, він має змогу працювати лише через API.

Схема роботи клієнта з сервером схематично зображено на Рис.2

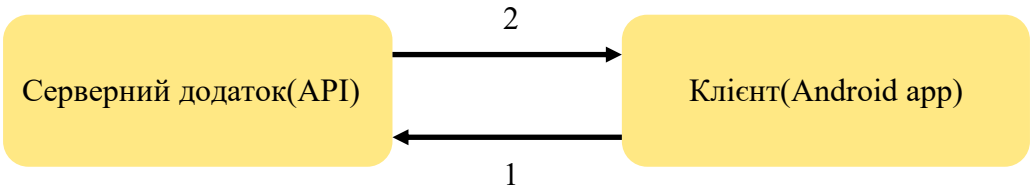

Рис. 2. Робота мобільного додатку

Алгоритм роботи зі сторони клієнту простий:

1. Клієнт запитує потрібні дані в API (1)
2. API віддає потрібні дані, якщо запит був валідний (2)

Розробка Back-end частини(API) на *Kotlin*

Для того, щоб створити серверний додаток на мові програмування *Kotlin,* будемо використовувати платформу *Ktor* та базу даних *SQLite.* Для зручної роботи з базою даних будемо *ORM фреймворк*, написаний на *Kotlin*, *Exposed*.

Алгоритм роботи серверного додатку

1. Зв’язати серверний додаток з базою даних за допомогою ORM фреймворку.

2. Кожні 5 - 10 хвилин забирати нову інформацію по банківським курсам валют та оновлювати інформацію в базі даних.
3. Надсилати оновлені дані до клієнту.

Розробка мобільного додатку на *Kotlin* для ОС Android

Будемо використовувати архітектуру розробки **MVVM(Model-View-ViewModel)**. Бібліотеки для розробки мобільного додатку:

1. Data Binding
2. Lifecycle, LiveData
3. RxJava2, RxKotlin
4. Retrofit

Що робить кожна з бібліотек та чому було вирішено розробляти за допомогою MVVM нижче.

Алгоритм роботи мобільного додатку

1. Під час запуску кожного екрану запросити дані з серверного додатку та показати їх.

1. Розробка Back-end

1.1. Підключення технологій та використання *Ktor*

Розробляти API будемо в середовищі розробки IntelliJ IDEA. Створимо новий проект, в основі якого буде сервер **Netty.** Взагалі, **Ktor** може працювати на основі таких серверів як:

- Netty
- Jetty
- TomCat
- інші

Код буде знаходиться в вихідній папці src/main.

Перш за все, потрібно записати порт, по якому можна буде забрати дані с сервера, та модуль додатку, в якому буде основний код. Для цього в **Ktor** існує файл **application.conf,** в якому буде основна конфігурація API.

Висновки

1. Завдання було виконано на сучасній мові програмування Kotlin. Були створені Back-end та мобільний додаток на Android.

2. Для розробки Back-end був створений алгоритм, що знаходить, зберігає, віддає потрібну клієнту інформацію. За допомогою таких бібліотек як Ktor, Exposed, бази даних SQLite, була написана Back-end частина додатку. Back-end навчився розбирати HTML сайтів для пошуку потрібної клієнту інформації. База даних має 4 таблиці, в кожної з них є своя мета та логічне пояснення. Був налагоджений зв’язок між Back-end та базою даних.

3. Був розроблений мобільний додаток на Android для відображення інформації користувачу. Мобільний додаток має два екрани, які переключаються за допомогою сучасної технології, яка часто використовується в мобільних додатках як BottomNavigationView(елемент навігації в Android). Був розроблений дизайн двох екранів та його відображення в XML. При розробці використовувалася така архітектура розробки, як MVVM, що дозволило зробити код програми гарним і лаконічним. View завжди слухає зміни даних в ViewModel, та якщо вони змінилися, View їх оновить. Були використані такі компоненти відображення ViewPager, RecyclerView.

Були протестовані обидві частини задачі та виправлені помилки.

Список літератури

1. https://ktor.io/
2. kotlinlang.org/
3. Kotlin в действии. Исакова С. Жемеров Д.
4. Android Programming with Kotlin for Beginners: Build Android apps, John Horton
5. https://www.codeflow.site/ru/article/kotlin-ktor
6. https://github.com/JetBrains/Exposed/wiki
7. https://habr.com/ru/post/432310/
8. https://jsoup.org/
9. https://www.spigotmc.org/threads/connecting-hikaricp-to-sqlite-database.116558/
10. https://developer.android.com/jetpack/docs/guide?gclid=CjwKCAjw4871BRAjEiwAbxXi2yl9VP_wBLheYLA3gCoQhrYSFumYE0XPw93qUMB6-LmDCRu8ObWFfxoCmGAQAvD_BwE&gclsrc=aw.ds
11. https://developer.android.com/topic/libraries/architecture/viewmodel
12. http://developer.alexanderklimov.ru/android/views/recyclerview.php
13. https://stfalcon.com/ru/blog/post/android-mvvm
14. https://devcolibri.com/unit/%D1%83%D1%80%D0%BE%D0%BA-10-%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0-%D1%81-recyclerview-%D0%BD%D0%B0-%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80%D0%B5-tweetsrecyclerview-2/
15. https://developer.android.com/training/animation/screen-slide
16. https://github.com/square/retrofit
17. https://github.com/ReactiveX/RxJava
18. https://github.com/ReactiveX/RxAndroid
19. https://stackoverflow.com/questions/13120397/limit-the-length-of-textview/13120431