НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

**Кафедра системного програмування та спеціалізованих комп’ютерних систем**

**Лабораторна робота №3**

з дисципліни **Програмування інтерфейсів користувача**

Виконав:

студент V курсу

групи КВ-32мп

Неживий Богдан Миколайович

Перевірив: Петрашенко А. В.

Київ – 2024

**Загальна інформація**

**Завдання на лабораторну роботу***:* розробити клієнтську частину Web-додатку.

[Посилання на Github-репозиторій](https://github.com/n0tark/user-interfaces-3);

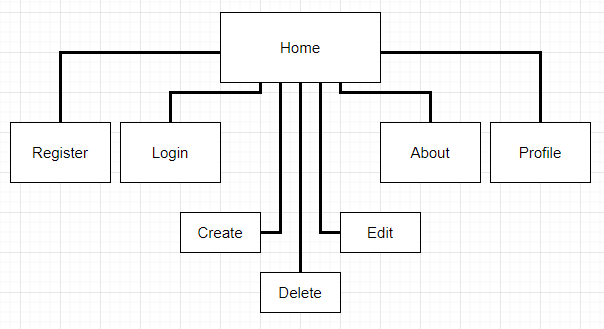
*Посилання студента в* [Telegram](https://t.me/unalive1);

**Назва додатку***:* ContactSphere

**Короткий опис функціональності***:* Цей додаток призначений для зберігання контактів, опису, даних для входу користувачів та інформації про їхні профілі. Він дозволяє зберігати дані про користувачів і контакти, а також забезпечує можливості автентифікації та управління профілями. Користувачі можуть створювати, переглядати та видаляти контакти за допомогою зручного інтерфейсу.

**Технології, використані в розробці***:* Android Studio, Android-платформа, Django

**Карта сайту**

****

**Перелік REST API запитів**

**RetrofitClient**

describe('The About Page', () => {

  beforeEach(() => {

    cy.visit('http://localhost:3000/about')

  })

  it('should display the correct heading', () => {

    cy.get('h1').should('contain', 'About ContactMe')

  })

  it('should display the correct paragraphs', () => {

    cy.get('p').eq(0).should('contain', 'Contact Manager is a web application designed for storing, deleting, and editing contacts.')

    cy.get('p').eq(1).should('contain', 'Contact Manager is a web application designed to connect people through a database of contacts.')

  })

  it('should have a mobile-friendly design', () => {

    cy.viewport('iphone-x')

  })

})

1. RetrofitClient: Це клас, який відповідає за створення і налаштування клієнта Retrofit.
2. getRetrofitInstance(): Цей метод створює один екземпляр клієнта Retrofit для взаємодії з веб-службами. Якщо екземпляр вже існує, він повертає його, інакше створює новий. Він використовує базовий URL "<http://192.168.0.105:8000/>" для всіх запитів і конвертує відповіді у форматі JSON за допомогою GsonConverterFactory.
3. Methods: Це інтерфейс, який визначає HTTP методи (GET, POST, DELETE) та шляхи до веб-служб.
4. getAllData(): Цей метод виконує HTTP GET-запит до адреси "phone-book/" і отримує всі дані з телефонного довідника. Він повертає об'єкт типу Call<Model>, де Model - це об'єкт, що відповідає структурі даних телефонного довідника.
5. deleteContact(int id): Цей метод виконує HTTP DELETE-запит для видалення контакту з телефонного довідника за його ідентифікатором (id). Він приймає id контакту в якості параметра.
6. addContact(Model.Contact contact): Цей метод виконує HTTP POST-запит для додавання нового контакту до телефонного довідника. Він приймає об'єкт контакту Model.Contact в якості параметра.

**Methods**

package com.example.phonebook;

import retrofit2.Call;

import retrofit2.http.Body;

import retrofit2.http.DELETE;

import retrofit2.http.GET;

import retrofit2.http.POST;

import retrofit2.http.Path;

public interface Methods {

    @GET("phone-book/")

    Call<Model> getAllData();

    @DELETE("phone-book/{id}/")

    Call<Void> deleteContact(@Path("id") int id);

    @POST("phone-book/")

    Call<Void> addContact(@Body Model.Contact contact);

}

Цей код є частиною інтерфейсу Methods, що використовується для взаємодії з веб-службою телефонного довідника за допомогою бібліотеки Retrofit. Він визначає методи для отримання всіх даних з телефонного довідника, видалення контакту за його ідентифікатором і додавання нового контакту до телефонного довідника через веб-службу.

1. \*\*@GET("phone-book/")\*\*: Ця анотація позначає метод як HTTP GET і вказує шлях "phone-book/", за яким буде виконуватися запит. Цей метод використовується для отримання всіх даних з телефонного довідника. Він повертає об'єкт типу Call<Model>, де Model представляє структуру телефонного довідника.

2. \*\*@DELETE("phone-book/{id}/")\*\*: Ця анотація позначає метод як HTTP DELETE і вказує шлях "phone-book/{id}/", де {id} - це параметр, що вказує ідентифікатор контакту, який потрібно видалити. Цей метод використовується для видалення контакту з телефонного довідника за його ідентифікатором. Він повертає об'єкт типу Call<Void>, оскільки відповідь не містить даних.

3. \*\*@POST("phone-book/")\*\*: Ця анотація позначає метод як HTTP POST і вказує шлях "phone-book/", за яким буде виконуватися запит. Цей метод використовується для додавання нового контакту до телефонного довідника. Він приймає об'єкт контакту Model.Contact у тілі запиту за допомогою анотації @Body і повертає об'єкт типу Call<Void>, оскільки відповідь не містить даних.

Функції запитів на сервер

MainActivity**Methods methods = RetrofitClient.*getRetrofitInstance*().create(Methods.class);**

Тут створюється екземпляр інтерфейсу Methods через клас RetrofitClient, який використовується для взаємодії з веб-службою.

**Call<Model> call = methods.getAllData();**

Цей метод викликається для отримання всіх даних з телефонного довідника.

call.enqueue(new Callback<Model>() {

            @Override

            public void onResponse(Call<Model> call, Response<Model> response) {

                if (response.isSuccessful() && response.body() != null) {

                    contacts.clear();

                    contacts.addAll(response.body().getContacts());

                    adapter.notifyDataSetChanged();

                    // Перевірка на відсутність контактів

                    if (contacts.isEmpty()) {

                        // Показати повідомлення про відсутність контактів

                        runOnUiThread(new Runnable() {

                            @Override

                            public void run() {

                                findViewById(R.id.text\_view\_empty).setVisibility(View.VISIBLE);

                            }

                        });

                    } else {

                        // Приховати повідомлення про відсутність контактів

                        runOnUiThread(new Runnable() {

                            @Override

                            public void run() {

                                findViewById(R.id.text\_view\_empty).setVisibility(View.GONE);

                            }

                        });

                    }

                } else {

                    Log.e(TAG, "Unsuccessful response or body is null");

                }

            }

Метод enqueue() виконує асинхронний запит до сервера і передає йому екземпляр Callback, який містить методи onResponse() і onFailure(). onResponse() викликається при успішній відповіді від сервера, а onFailure() - при невдачі.

@Override

            public void onFailure(Call<Model> call, Throwable t) {

                Log.e(TAG, "onFailure: " + t.getMessage());

            }

        });

У методі onResponse() дані, отримані від сервера, додаються до відповідного адаптера для відображення у відповідному списку (або іншому елементі інтерфейсу користувача). Також перевіряється, чи немає отриманих контактів, і відповідно до цього відображається відповідне повідомлення.

private void addContact(Model.Contact contact) {

        Methods methods = RetrofitClient.getRetrofitInstance().create(Methods.class);

        Call<Void> call = methods.addContact(contact);

        Log.d(TAG, "Adding contact: " + contact.getName() + ", " + contact.getPhone\_number()); // Додати цей рядок

        call.enqueue(new Callback<Void>() {

            @Override

            public void onResponse(Call<Void> call, Response<Void> response) {

                if (response.isSuccessful()) {

                    contacts.add(contact);

                    adapter.notifyItemInserted(contacts.size() - 1);

                } else {

                    Log.e(TAG, "Unsuccessful add contact request");

                }

            }

            @Override

            public void onFailure(Call<Void> call, Throwable t) {

                Log.e(TAG, "Failed to add contact: " + t.getMessage());

            }

        });

    }

Метод `addContact(Model.Contact contact)` використовується для додавання нового контакту до телефонного довідника через веб-службу. Він викликається при натисканні кнопки або іншій події у користувацькому інтерфейсі додатка. Коли метод викликається, він спочатку створює екземпляр інтерфейсу Methods за допомогою RetrofitClient. Після цього він викликає метод `addContact(contact)`, передаючи об'єкт контакту `contact`. Запит на додавання контакту відправляється до сервера через веб-службу.

Через асинхронний виклик `enqueue()`, метод очікує відповідь від сервера. Після отримання відповіді викликаються два обробники: `onResponse()` і `onFailure()`. У `onResponse()` перевіряється, чи була відповідь успішною. Якщо так, контакт додається до локального списку контактів, і адаптер оновлюється для відображення нового контакту. Якщо відповідь невдала, виводиться відповідне повідомлення про помилку. У `onFailure()` обробляється випадок помилки під час виконання запиту до сервера, і виводиться повідомлення з деталями помилки.

Видалення елемента

public void deleteContact(int position) {

    Model.Contact contact = contacts.get(position);

    Methods methods = RetrofitClient.getRetrofitInstance().create(Methods.class);

    Call<Void> call = methods.deleteContact(contact.getId());

    call.enqueue(new Callback<Void>() {

        @Override

        public void onResponse(Call<Void> call, Response<Void> response) {

            if (response.isSuccessful()) {

                contacts.remove(position);

                notifyItemRemoved(position);

            } else {

                Log.e(TAG, "Unsuccessful delete request");

            }

        }

        @Override

        public void onFailure(Call<Void> call, Throwable t) {

            Log.e(TAG, "Failed to delete contact: " + t.getMessage());

        }

    });

}

Метод `deleteContact(int position)` призначений для видалення контакту з телефонного довідника через веб-службу. Він викликається при натисканні на кнопку або подібній події у користувальницькому інтерфейсі додатка, передаючи позицію контакту, який потрібно видалити.

1. Спочатку метод отримує контакт, який потрібно видалити, за його позицією у списку контактів.

2. Потім створюється екземпляр інтерфейсу Methods за допомогою RetrofitClient для взаємодії з веб-службою.

3. Викликається метод `deleteContact(contact.getId())`, передаючи ідентифікатор контакту для видалення.

4. Запит на видалення контакту відправляється до сервера через веб-службу за допомогою асинхронного виклику `enqueue()`.

5. Після отримання відповіді від сервера викликаються обробники `onResponse()` і `onFailure()`. У `onResponse()` перевіряється, чи була відповідь успішною. Якщо так, контакт видаляється з локального списку контактів, і адаптер оновлюється для відображення змін. У разі невдалої відповіді виводиться відповідне повідомлення про помилку.

6. У випадку помилки під час виконання запиту до сервера, метод `onFailure()` виводить повідомлення з деталями помилки.

Структура даних, які отримуємо

package com.example.phonebook;

public class Contact {

    public String id;

    public String name;

    public String phones;

    public String getId() {

        return id;

    }

    public String getName() {

        return name;

    }

    public String getPhones() {

        return phones;

    }

}

Поля класу:

* id: Ідентифікатор контакту.
* name: Ім'я контакту.
* phones: Рядок, що містить один або декілька телефонних номерів, розділених комами або іншими розділювачами.

Методи доступу (getters):

1. getId(): Повертає ідентифікатор контакту.
2. getName(): Повертає ім'я контакту.
3. getPhones(): Повертає рядок з телефонними номерами контакту.

Цей клас слугує для зберігання інформації про контакти телефонного довідника. Кожен об'єкт класу Contact містить ідентифікатор, ім'я та телефонний номер контакту. Він використовується, наприклад, для передачі даних про контакти між клієнтською і серверною сторонами додатка або для відображення інформації про контакти у користувацькому інтерфейсі.

Для відображення списку контактів створено:

package com.example.phonebook;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.TextView;

import androidx.annotation.NonNull;

import androidx.recyclerview.widget.RecyclerView;

import java.util.List;

public class ContactAdapter extends RecyclerView.Adapter<ContactAdapter.ContactViewHolder> {

    private List<Model.Contact> contacts;

    private OnContactClickListener listener;

    public ContactAdapter(List<Model.Contact> contacts, OnContactClickListener listener) {

        this.contacts = contacts;

        this.listener = listener;

    }

    @NonNull

    @Override

    public ContactViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {

        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.contact\_item, parent, false);

        return new ContactViewHolder(view);

    }

    @Override

    public void onBindViewHolder(@NonNull ContactViewHolder holder, int position) {

        Model.Contact contact = contacts.get(position);

        holder.textViewName.setText(contact.getName());

        holder.textViewPhoneNumber.setText(contact.getPhone\_number());

        holder.buttonDelete.setOnClickListener(v -> {

            if (listener != null) {

                listener.onDeleteClick(position);

            }

        });

    }

    @Override

    public int getItemCount() {

        return contacts.size();

    }

    public interface OnContactClickListener {

        void onDeleteClick(int position);

    }

    public static class ContactViewHolder extends RecyclerView.ViewHolder {

        public TextView textViewName;

        public TextView textViewPhoneNumber;

        public Button buttonDelete;

        public ContactViewHolder(@NonNull View itemView) {

            super(itemView);

            textViewName = itemView.findViewById(R.id.text\_view\_name);

            textViewPhoneNumber = itemView.findViewById(R.id.text\_view\_phone\_number);

            buttonDelete = itemView.findViewById(R.id.button\_delete);

        }

    }

}

Цей код визначає адаптер `ContactAdapter`, який використовується для відображення списку контактів у `RecyclerView`. Розглянемо кожну частину класу:

1. \*\*Конструктор\*\*:

- Приймає список контактів `contacts` та об'єкт типу `OnContactClickListener` для обробки подій кліків.

2. \*\*Метод `onCreateViewHolder()`\*\*:

- Створює новий екземпляр `ContactViewHolder`, який буде використовуватися для відображення елемента списку.

3. \*\*Метод `onBindViewHolder()`\*\*:

- Прив'язує дані контакту до відповідного `ContactViewHolder`.

- Встановлює текст для імені контакту та його номера телефону.

- Налаштовує обробник події для кнопки видалення, який викликає метод `onDeleteClick(position)` у `listener`, якщо такий існує.

4. \*\*Метод `getItemCount()`\*\*:

- Повертає кількість елементів у списку контактів.

5. \*\*Інтерфейс `OnContactClickListener`\*\*:

- Визначає метод `onDeleteClick()`, який буде викликаний, коли користувач натисне на кнопку видалення контакту.

6. \*\*Внутрішній клас `ContactViewHolder`\*\*:

- Представляє елемент списку.

- Ініціалізує елементи `View`, такі як `TextView` для імені та номера телефону контакту, і кнопку для видалення.

`ContactAdapter` використовується для відображення списку контактів у `RecyclerView`, а також для встановлення обробників подій для кнопок видалення кожного контакту.

Шаблони інтерфейсів:

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/main"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context=".MainActivity">

    <include

        android:id="@+id/navbar"

        layout="@layout/navbar"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        app:layout\_constraintTop\_toTopOf="parent"

        />

    <androidx.recyclerview.widget.RecyclerView

        android:id="@+id/recycler\_view\_contacts"

        android:layout\_width="match\_parent"

        android:layout\_height="0dp"

        app:layout\_constraintTop\_toBottomOf="@+id/navbar"

    app:layout\_constraintStart\_toStartOf="parent"

    app:layout\_constraintEnd\_toEndOf="parent"/>

    <Button

        android:id="@+id/button\_add\_contact"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="@string/add\_contact"

        app:layout\_constraintTop\_toBottomOf="@+id/recycler\_view\_contacts"

        app:layout\_constraintStart\_toStartOf="parent"

        android:background="@color/light\_blue"

        android:paddingStart="16dp"

        android:paddingTop="8dp"

        android:paddingEnd="16dp"

        android:paddingBottom="8dp"

    app:layout\_constraintEnd\_toEndOf="parent"/>

    <TextView

        android:id="@+id/text\_view\_empty"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="@string/no\_contact"

        android:visibility="gone"

        android:textSize="24sp"

        android:textColor="@color/black"

        app:layout\_constraintTop\_toBottomOf="@+id/button\_add\_contact"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"

        android:layout\_marginTop="16dp"/>

    <RelativeLayout

        android:layout\_width="match\_parent"

        android:layout\_height="match\_parent"

        tools:layout\_editor\_absoluteX="0dp"

        tools:layout\_editor\_absoluteY="294dp">

        <include

            android:id="@+id/footer"

            layout="@layout/footer"

            android:layout\_width="match\_parent"

            android:layout\_height="wrap\_content"

            android:layout\_alignParentBottom="true" />

    </RelativeLayout>

</androidx.constraintlayout.widget.ConstraintLayout>

**contact\_item**

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/main"

    android:layout\_width="match\_parent"

    android:layout\_height="wrap\_content"

    android:paddingTop="16dp"

    android:layout\_marginBottom="16dp">

    <TextView

        android:id="@+id/text\_view\_name"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:textSize="18sp"

        android:textStyle="bold"

        app:layout\_constraintTop\_toTopOf="parent"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"/>

    <TextView

        android:id="@+id/text\_view\_phone\_number"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:textSize="14sp"

        app:layout\_constraintTop\_toBottomOf="@+id/text\_view\_name"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"/>

    <Button

        android:id="@+id/button\_delete"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="@string/delete"

        android:textSize="10sp"

        app:layout\_constraintTop\_toTopOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"

        app:layout\_constraintBottom\_toBottomOf="parent"/>

</androidx.constraintlayout.widget.ConstraintLayout>

**activity\_login**

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context=".LoginActivity">

    <!-- Навбар -->

    <include

        android:id="@+id/navbar"

        layout="@layout/navbar"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        app:layout\_constraintTop\_toTopOf="parent"

        />

    <EditText

        android:id="@+id/edit\_text\_email"

        android:layout\_width="0dp"

        android:layout\_height="wrap\_content"

        android:hint="Email"

        app:layout\_constraintTop\_toBottomOf="@id/navbar"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"

        android:layout\_marginTop="16dp"

        android:layout\_marginStart="16dp"

        android:layout\_marginEnd="16dp"

        />

    <EditText

        android:id="@+id/edit\_text\_password"

        android:layout\_width="0dp"

        android:layout\_height="wrap\_content"

        android:hint="Password"

        app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_email"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"

        android:layout\_marginTop="16dp"

        android:layout\_marginStart="16dp"

        android:layout\_marginEnd="16dp"

        />

    <Button

        android:id="@+id/button\_register"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_marginStart="16dp"

        android:layout\_marginTop="140dp"

        android:layout\_marginEnd="16dp"

        android:text="Log in"

        app:layout\_constraintEnd\_toEndOf="parent"

        app:layout\_constraintHorizontal\_bias="0.524"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_password" />

    <!-- Футер -->

    <RelativeLayout

        android:layout\_width="match\_parent"

        android:layout\_height="match\_parent">

        <include

            android:id="@+id/footer"

            layout="@layout/footer"

            android:layout\_width="match\_parent"

            android:layout\_height="wrap\_content"

            android:layout\_alignParentBottom="true" />

    </RelativeLayout>

</androidx.constraintlayout.widget.ConstraintLayout>

**activity\_register**

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context=".LoginActivity">

    <include

        android:id="@+id/navbar"

        layout="@layout/navbar"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        app:layout\_constraintTop\_toTopOf="parent"

        />

    <EditText

        android:id="@+id/edit\_text\_email"

        android:layout\_width="0dp"

        android:layout\_height="wrap\_content"

        android:hint="Email"

        app:layout\_constraintTop\_toBottomOf="@id/navbar"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"

        android:layout\_marginTop="16dp"

        android:layout\_marginStart="16dp"

        android:layout\_marginEnd="16dp"

        />

    <EditText

        android:id="@+id/edit\_text\_password"

        android:layout\_width="0dp"

        android:layout\_height="wrap\_content"

        android:hint="Password"

        app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_email"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintEnd\_toEndOf="parent"

        android:layout\_marginTop="16dp"

        android:layout\_marginStart="16dp"

        android:layout\_marginEnd="16dp"

        />

    <Button

        android:id="@+id/button\_register"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_marginStart="16dp"

        android:layout\_marginTop="140dp"

        android:layout\_marginEnd="16dp"

        android:text="Log in"

        app:layout\_constraintEnd\_toEndOf="parent"

        app:layout\_constraintHorizontal\_bias="0.524"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_password" />

    <RelativeLayout

        android:layout\_width="match\_parent"

        android:layout\_height="match\_parent">

        <include

            android:id="@+id/footer"

            layout="@layout/footer"

            android:layout\_width="match\_parent"

            android:layout\_height="wrap\_content"

            android:layout\_alignParentBottom="true" />

    </RelativeLayout>

</androidx.constraintlayout.widget.ConstraintLayout>

**dialog\_add\_contact**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="match\_parent"

    android:layout\_height="wrap\_content"

    android:orientation="vertical"

    android:padding="25dp">

    <EditText

        android:id="@+id/edit\_text\_name"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:hint="Name"/>

    <EditText

        android:id="@+id/edit\_text\_phone\_number"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:hint="Phone number"/>

    <Button

        android:id="@+id/button\_add"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="Add"/>

</LinearLayout>

**footer**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="match\_parent"

    android:layout\_height="wrap\_content"

    android:orientation="horizontal"

    android:padding="16dp"

    android:background="@android:color/holo\_blue\_light"

    android:gravity="center\_vertical|center\_horizontal"

    android:layout\_gravity="bottom">

    <TextView

        android:id="@+id/text\_view\_footer"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="@string/\_2024\_phonebook"

        android:textSize="14sp"

        android:textColor="@android:color/white"/>

</LinearLayout>

**navbar**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="match\_parent"

    android:layout\_height="wrap\_content"

    android:orientation="horizontal"

    android:padding="16dp"

    android:background="@android:color/holo\_blue\_light"

    android:gravity="center">

    <TextView

        android:id="@+id/text\_view\_title"

        android:layout\_width="0dp"

        android:layout\_height="wrap\_content"

        android:layout\_weight="1"

        android:text="@string/phonebook"

        android:textSize="24sp"

        android:textColor="@android:color/white"

        android:textStyle="bold"/>

    <TextView

        android:id="@+id/text\_view\_login"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="@string/log\_in"

        android:textSize="18sp"

        android:textColor="@android:color/white"

        android:paddingStart="16dp"

        android:paddingEnd="16dp"/>

    <TextView

        android:id="@+id/text\_view\_register"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="@string/registration"

        android:textSize="18sp"

        android:textColor="@android:color/white"

        android:paddingStart="16dp"

        android:paddingEnd="16dp"/>

</LinearLayout>

**Копії екрану мобільного додатка**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

У разі відсутності або не з’єднанні з сервером

*Изображение выглядит как текст, программное обеспечение, веб-страница, Веб-сайт

Автоматически созданное описание*

Основна сторінка

*Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание*

Приклад видалення контакту - Tom

*Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание*

Вікно для додавання елемента

*Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание*

Заповнення вікна

*Изображение выглядит как текст, программное обеспечение, веб-страница, Веб-сайт

Автоматически созданное описание*

Отриманий результат

*Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание*

Сторінка Log in

*Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание*

Сторінка регістрації

**Висновки**

У результаті виконання лабораторної роботи було створено мобільний додаток "PhoneBook", призначений для управління контактами компанії. Додаток надає користувачам можливість додавати, редагувати та видаляти контакти, пов’язані з певною компанією чи організацією. Основні функції додатка включають зберігання інформації про користувачів і їхні контакти, а також можливість автентифікації та ведення профілю користувача.

Для взаємодії з сервером використовувався Retrofit для виконання HTTP-запитів. Додаток застосовує REST API з першої лабораторної роботи "Технологія розробки Web-додатків" для отримання, додавання та видалення контактів із сервера.

Основний інтерфейс додатка реалізовано за допомогою RecyclerView, що дозволяє зручно відображати список контактів. Користувачі можуть авторизуватися або зареєструватися в додатку. Також було реалізовано діалогове вікно для додавання нового контакту з необхідною інформацією.

Таким чином, лабораторна робота була успішно виконана, і мобільний додаток "PhoneBook" готовий до використання, забезпечуючи зручне управління корпоративними контактами.