Android lab

1 Instructions

1. During this exercise, document your progress using screenshots (of the entire screen) to demonstrate that you have indeed completed all the required tasks.

2. Save all the screenshots in the GitHub project, in a dedicated folder called "proof", and present them chronologically in the README file. See the forum to understand what present in README means.

3. There's no need to make hundreds of screenshots, but please take enough to convince the person checking your submission that you completed the exercise as instructed.

4. Your README file must also include a link to the repository.

5. When you are ready to submit, download the repository from GitHub (the entire repository) and submit it. Your submission MUST NOT contain links to other locations where the files are (like Google Drive or a link to the GitHub repository instead of the actual files). You are allowed to submit a zip file.

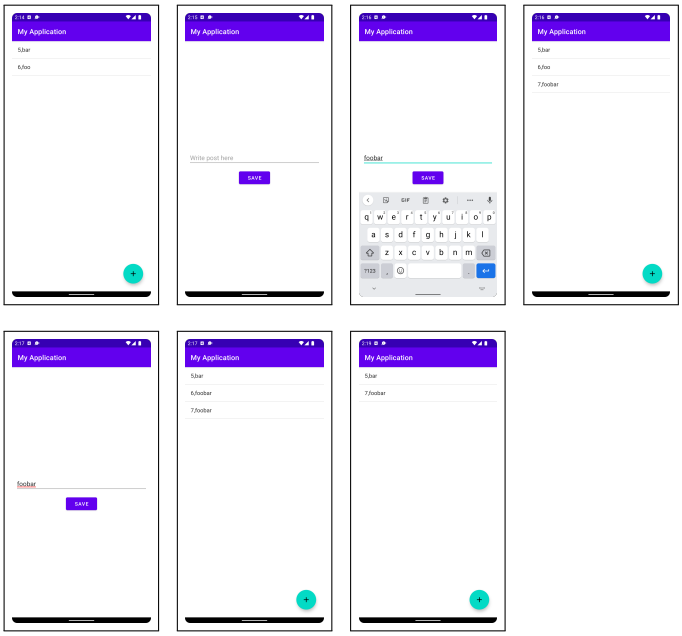
6. The Moodle has a hard limit on the upload size, and will not allow you to upload files beyond 5MB. There is nothing I can do about it.

2 Room

We want to implement an application that:

- Displays a list of items, **which are stored in a local database**.
- We can add new items to the list.
- Edit existing items.
- Delete existing items.

The following screenshots demonstrate the "flow" of the application, performing the CRUD operations.



I will give you the code, based on the code we saw in the lecture. **Your goal is to run the code, and demonstrate all the application features (creating item, editing item, deleting item), but must importantly, make sure you understand the code and the use of Room.**

To that end, you will need to create the following layouts files:

```xml
<?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=".MainActivity">

    <ListView
        android:id="@+id/lvPosts"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/btnAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_margin="10pt"
        android:src="@drawable/ic_baseline_add_24"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

```xml
<?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=".FormActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <EditText
            android:id="@+id/etContent"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_margin="10dp"
            android:ems="10"
            android:hint="Write post here" />

        <Button
            android:id="@+id/btnSave"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:text="Save" />
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

And the following activities: (especially, notice lines: 2, 5, 14-17, 23, 33, 37).

```
public class FormActivity extends AppCompatActivity {
    private AppDB db;
    private ActivityFormBinding binding;
    private Post post;
    PostDao postDao;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityFormBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        db = Room.databaseBuilder(getApplicationContext(),
                                  AppDB.class, "FooDB")
                 .allowMainThreadQueries().build();
        postDao = db.postDao();

        handleSave();

        if (getIntent().getExtras() != null){
            int id = getIntent().getExtras().getInt("id");
            post = postDao.get(id);

            binding.etContent.setText(post.getContent());
        }
    }

    private void handleSave() {
        binding.btnSave.setOnClickListener(view -> {
            if (post == null) {
                post = new Post(binding.etContent.getText().toString());
                postDao.insert(post);
            }
            else {
                post.setContent(binding.etContent.getText().toString());
                postDao.update(post);
            }

            finish();
        });
    }

}

public class MainActivity extends AppCompatActivity {
    private ActivityMainBinding binding;
    private AppDB db;
    private ListView lvPosts;
    private List<String> posts;
    private List<Post> dbPosts;
    private ArrayAdapter<String> adapter;
    private PostDao postDao;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        db = Room.databaseBuilder(getApplicationContext(), AppDB.class, "FooDB")
                .allowMainThreadQueries().build();

        postDao = db.postDao();
        handlePosts();
```

```
        binding.btnAdd.setOnClickListener(view -> {
            Intent intent = new Intent(this, FormActivity.class);
            startActivity(intent);
        });
    }
    @Override
    protected void onResume() {
        super.onResume();
        loadPosts();
    }

    private void handlePosts() {
        lvPosts = binding.lvPosts;
        posts = new ArrayList<>();
        adapter = new ArrayAdapter<>(this,android.R.layout.simple_list_item_1,posts);

        loadPosts();
        lvPosts.setAdapter(adapter);
        lvPosts.setOnItemClickListener((adapterView, view, i, l) -> {
            Intent intent = new Intent(this, FormActivity.class);
            intent.putExtra("id", dbPosts.get(i).getId());
            startActivity(intent);
        });

        lvPosts.setOnItemLongClickListener((adapterView, view, i, l) -> {
            posts.remove(i);
            Post post = dbPosts.remove(i);
            postDao.delete(post);
            adapter.notifyDataSetChanged();
            return true;
        });
    }

    private void loadPosts() {
        posts.clear();
        dbPosts = postDao.index();
        for (Post post : dbPosts){
            posts.add(post.getId() + "," + post.getContent());
        }

        adapter.notifyDataSetChanged();
    }
}
```

3 ViewModel

1. Implement the code example from the lecture slides regarding view model (slides 7-8).

2. Note that if your themes.xml resource file has a 'NoActionBar' theme, the code will not work. In such case, please change the theme to another theme that does have an action bar.