5COSC023W - Tutorial 9 Exercises

As part of the tutorial for this week, you should complete **ALL** the tasks described in the following specifications: (**make sure that you ask questions to your tutor for anything that you do not understand or if you are stuck at any point**).

> Tutorial sessions are practical sessions that you need to work towards the exercises set. They will give you the chance to practice the material learned in the lectures and learn new things as well.
>
> **You should not use these sessions to work towards the assessed assignments!** If you decide to work towards your assessed work instead, then you are not considered as part of the tutorial session. You will not get any help on the code of the assessed work by your tutor but you can ask your tutor ONLY about any clarifications you might need regarding the specification of the coursework.

Like all other modules, you are expected to study towards you module outside the lecture and dedicated tutorial slots for a number of hours. If you do not finish all of the exercises in the tutorial session, make sure that you finish them on your own time and by the end of the week. This is a normal process and part of your university learning.

For all the tasks you should use Jetpack Compose and NOT Views!

1 Database Example using the Room library

Implement the database example (keeping track of users data) that we have seen in the lecture using the Room library.

Make sure that you include the dependencies and the plugins in the `build.gradle` files after you create the Android Studio project. Your tutor will help you to add these or with any questions you might have.

2 Extending the Database Example

1. Extend the program developed in the previous exercise by adding another button which deletes the last record of the user added.

   Hint: What kind of SQL query/queries would you need to identify the last user added? Can you utilise the primary key value?

2. Extend the program by ordering the displayed users according to ascending order of their last name. You can use an SQL query to do this.

3. Modify the code so that the sorting of the users is done programmatically, i.e. without using the `order` clause of an SQL query.

4. Extend the app to include a `TextField` which can be used to add new users.

5. Extend the app to include a `TextField` to search for a user in the DB whose name matches the pattern typed in the textfield (a similar approach with the implementation of the function in the lecture slides with the `LIKE` operator can be used.)

   Hint: You can find more details on how to use `LIKE` at the folllowing link:

   https://www.w3schools.com/sql/sql_like.asp

6. Extend the app so that the user is able to delete all records of users whose names contain a specific substring using a case insesitive manner (e.g. typing "Mit" in a textfield will delete users such as "John Smith", "Paul EdmITonus", etc).

   Hint: Use a `TextField` a button and the `LIKE` SQL operator.

7. Implement the last part without using the `LIKE` SQL operator but based on a programmatic approach, i.e. the match of the substring with user names should just use Kotlin code and `DELETE` SQL statements with `WHERE` clauses but without `LIKE`.