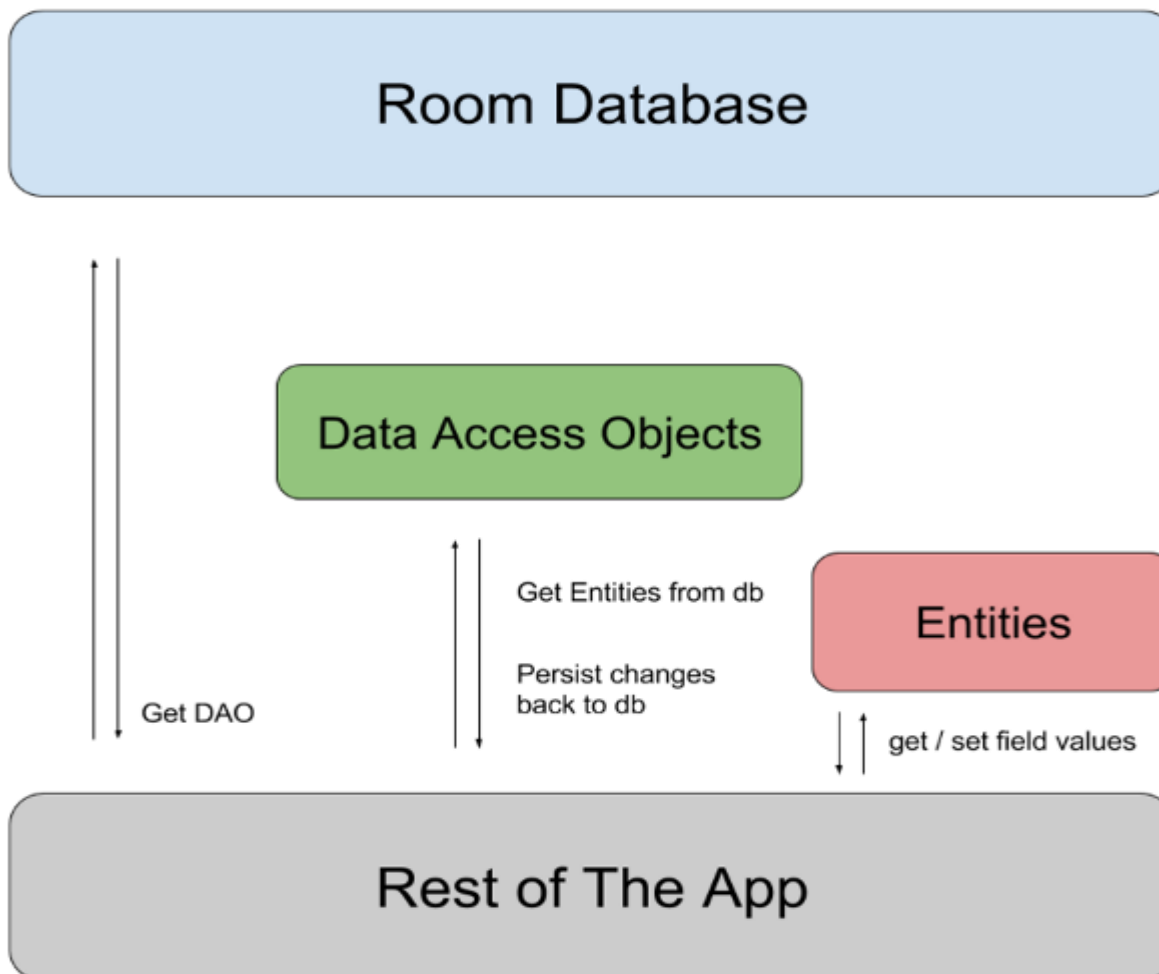




Room Database SQLite In Android



- Room là một cơ sở dữ liệu ORM dựa trên SQLite database. Room là cơ sở dữ liệu được giới thiệu bởi Google, cho phép bạn dễ dàng thao tác các truy vấn SQLite trong Android.
- Có ba thành phần chính trong Room:
 - ▣ Lớp Database chứa cơ sở dữ liệu và đóng vai trò là điểm truy cập chính cho kết nối cơ bản với dữ liệu của ứng dụng.
 - ▣ Entities đại diện cho các bảng trong cơ sở dữ liệu của ứng dụng.
 - ▣ DAO cung cấp các phương thức mà ứng dụng của bạn có thể sử dụng để truy vấn, cập nhật, chèn và xóa dữ liệu trong cơ sở dữ liệu.



- @Entity: định nghĩa lớp Entity
- @PrimaryKey: Trường khóa chính
- @ColumnInfo: định nghĩa trường khác với tên cột trong bảng

```
@Entity
public class User {
    @PrimaryKey
    public int uid;

    @ColumnInfo(name = "first_name")
    public String firstName;

    @ColumnInfo(name = "last_name")
    public String lastName;
}
```

- @Dao: định nghĩa lớp DAO
- @Query: thực hiện truy vấn data
- @Insert: thực hiện thêm dữ liệu
- @Delete: thực hiện xóa dữ liệu
- @Update: thực hiện cập nhật dữ liệu

```
@Update
public void updateUser(User... users);
```

```
@Dao
public interface UserDao {
    @Query("SELECT * FROM user")
    List<User> getAll();

    @Query("SELECT * FROM user WHERE uid IN (:userIds)")
    List<User> loadAllByIds(int[] userIds);

    @Query("SELECT * FROM user WHERE first_name LIKE :first AND " +
        "last_name LIKE :last LIMIT 1")
    User findByName(String first, String last);

    @Insert
    void insertAll(User... users);

    @Delete
    void delete(User user);
}
```

□ Lớp Database là một lớp Abstract

```
@Database(entities = {User.class}, version = 1)
public abstract class AppDatabase extends RoomDatabase {
    public abstract UserDao userDao();
}
```

□ Sử dụng lớp trên để tạo Database

```
AppDatabase db = Room.databaseBuilder(getApplicationContext(),
    AppDatabase.class, "database-name").build();
```

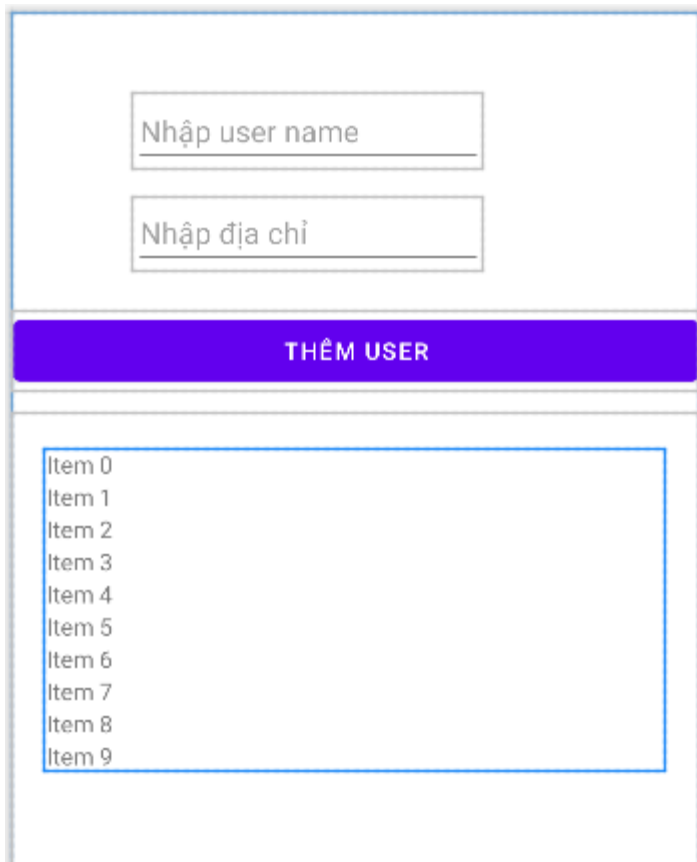
□ Tương tác dữ liệu

```
UserDao userDao = db.userDao();
List<User> users = userDao.getAll();
```

□ Bước 1: Thêm thư viện và thiết kế giao diện

implementation "androidx.room:room-runtime:2.5.1"

annotationProcessor "androidx.room:room-compiler:2.5.1"



UI Design Mockup showing a form with two input fields: "Nhập user name" and "Nhập địa chỉ". Below the inputs is a blue button labeled "THÊM USER". At the bottom, there is a list view containing items from "Item 0" to "Item 9".

- Giao diện activity_main.xml và item_user.xml

TextView

TextView

□ Bước 2: Tạo class Entities

```
@Entity(tableName = "users")
public class Users implements Serializable {
    @PrimaryKey(autoGenerate = true)
    private int id;
    private String username;
    private String address;

    public Users(String username, String address) {
        this.username = username;
        this.address = address;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getUsername() { return username; }

    public void setUsername(String username) { this.username = username; }

    public String getAddress() { return address; }

    public void setAddress(String address) { this.address = address; }
}
```


□ Bước 3: Tạo Adapter

```
public class UserAdapter extends RecyclerView.Adapter<UserAdapter.UserHolder> {
    private List<Users> usersList;

    public void setData(List<Users> list){
        this.usersList = list;
        notifyDataSetChanged();
    }

    @NonNull
    @Override
    public UserHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_user
        ,parent, attachToRoot: false);
        return new UserHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull UserHolder holder, int position) {
        Users users = usersList.get(position);
        if(users == null){
            return;
        }
        holder.username.setText(users.getUsername());
        holder.address.setText(users.getAddress());
    }

    @Override
    public int getItemCount() {
        if(usersList != null){
            return usersList.size();
        }
        return 0;
    }
}
```

```
} public class UserHolder extends RecyclerView.ViewHolder{
    private TextView username;
    private TextView address;

    public UserHolder(@NonNull View itemView) {
        super(itemView);
        username = itemView.findViewById(R.id.textViewUsername);
        address = itemView.findViewById(R.id.textViewAddress);
    }
}
```

□ Bước 3: Tạo Interface DAO

```
@Dao
public interface UsersDao {
    @Query("SELECT * FROM users")
    List<Users> getAll();

    @Query("SELECT * FROM users WHERE id IN (:userIds)")
    List<Users> loadAllByIds(int[] userIds);
```

```
@Insert
void insertAll(Users... users);
```

```
@Update
void update(Users... users);
```

```
@Delete
void delete(Users user);
```

```
}
```

□ Bước 4: Tạo class Abstract Database

```
import android.content.Context;

import androidx.room.Database;
import androidx.room.Room;
import androidx.room.RoomDatabase;

import vn.iotstar.roomdatabase.Users;

@Database(entities = {Users.class}, version = 1)
public abstract class UserDatabase extends RoomDatabase {
    private static final String DATABASE_NAME = "user.db";
    private static UserDatabase instance;

    public static synchronized UserDatabase getInstance(Context context){
        if(instance == null){
            instance = Room.databaseBuilder(context.getApplicationContext()
                ,UserDatabase.class,DATABASE_NAME)
                .allowMainThreadQueries()
                .build();
        }
        return instance;
    }

    public abstract UsersDao usersDao();
}
```

□ Bước 5: Xử lý MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    private EditText editTextUsername;
    private EditText editTextAddress;
    private Button btnThem;
    private RecyclerView rc_list;
    private UserAdapter userAdapter;
    private List<Users> usersList;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        AnhXa();
        userAdapter = new UserAdapter();
        usersList = new ArrayList<>();
        //lấy danh sách user trong Room
        usersList= UserDatabase.getInstance(this).usersDao().getAll();
        userAdapter.setData(usersList);
        rc_list.setHasFixedSize(false);
        LinearLayoutManager layoutManager = new LinearLayoutManager(context: this);
        rc_list.setLayoutManager(layoutManager);
        rc_list.setAdapter(userAdapter);
        btnThem.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { addUser(); }
        });
    }
}
```

□ Bước 5: Xử lý MainActivity.java

```
private void addUser() {
    String username = editTextUsername.getText().toString().trim();
    String address = editTextAddress.getText().toString().trim();
    if(TextUtils.isEmpty(username) || TextUtils.isEmpty(address)){
        return;
    }
    Users users = new Users(username, address);
    //add vào room
    UserDatabase.getInstance(this).usersDao().insertAll(users);
    Toast.makeText(context, this, text: "Thêm user thành công", Toast.LENGTH_SHORT).show();
    //xóa dữ liệu trên edittext
    editTextAddress.setText("");
    editTextUsername.setText("");
    hideSoftKeyboard();

    //lấy danh sách user trong Room
    userList= UserDatabase.getInstance(this).usersDao().getAll();
    userAdapter.setData(usersList);
}

public void hideSoftKeyboard(){
    try {
        InputMethodManager inputMethodManager = (InputMethodManager) getSystemService(Activity.INPUT_METHOD_SERVICE);
        inputMethodManager.hideSoftInputFromWindow(getCurrentFocus().getWindowToken(), flags: 0);
    }catch (NullPointerException ex){
        ex.printStackTrace();
    }
}
```

```
private void AnhXa() {
    editTextUsername = findViewById(R.id.editTextUsername);
    editTextAddress = findViewById(R.id.editTextAddress);
    btnThem = findViewById(R.id.btnThem);
    rc_list = findViewById(R.id.rc_list);
}
```

□ Bước 6: Check user tồn tại

```
@Query("SELECT * FROM users where username= :username")
List<Users> checkUser(String username);
```

Thêm vào interface UserDao

```
private boolean isCheckExist(@NotNull Users user){
    List<Users> list = UserDatabase.getInstance(this).usersDao().checkUser(user.getUsername());
    return list!=null && !list.isEmpty();
}
```

Thêm vào MainActivity.java

```
Users users = new Users(username, address);
if(isCheckExist(users)){
    Toast.makeText(context: this, text: "User đã tồn tại", Toast.LENGTH_SHORT).show();
    return;
}
```

Thêm vào hàm addUser()

□ Bước 7: Cập nhật User

Thiết kế lại
item_user.xml

```

item_user.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     android:layout_width="match_parent"
6     android:layout_height="wrap_content">
7     <LinearLayout
8         android:layout_toStartOf="@id/btnUpdate"
9         android:layout_centerVertical="true"
10        android:layout_width="match_parent"
11        android:layout_height="wrap_content"
12        android:orientation="vertical">
13        <TextView
14            android:id="@+id/textViewUsername"
15            android:layout_width="wrap_content"
16            android:layout_height="wrap_content"
17            android:layout_marginStart="16dp"
18            android:layout_marginTop="16dp"
19            android:text="TextView"
20            android:textColor="#3F51B5"
21            android:textSize="20sp"
22            android:textStyle="bold"/>
23        <TextView
24            android:id="@+id/textViewAddress"
25            android:layout_width="wrap_content"
26            android:layout_height="wrap_content"
27            android:layout_marginStart="16dp"
28            android:layout_marginBottom="16dp"
29            android:text="TextView" />

```

TextView

TextView

CẬP NHẬT

</LinearLayout>

<Button

```

    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:gravity="center"
    android:layout_margin="10dp"
    android:layout_centerVertical="true"
    android:id="@+id/btnUpdate"
    android:text="Cập nhật"/>

```

</RelativeLayout>

□ Bước 7: Cập nhật User

```
public class UserHolder extends RecyclerView.ViewHolder{
    private TextView username;
    private TextView address;
    private Button btnUpdate;
    public UserHolder(@NonNull View itemView) {
        super(itemView);
        username = itemView.findViewById(R.id.textViewUsername);
        address = itemView.findViewById(R.id.textViewAddress);
        btnUpdate = itemView.findViewById(R.id.btnUpdate);
    }
}
```

Cập nhật lại Holder
trong UserAdapter

□ Bước 7: Cập nhật User

Tạo Interface để
nhận dữ liệu trả về

```
public interface iClickListener{
    void updateUser(Users users);
}
```

Tạo Constructor
trong Adapter

```
public class UserAdapter extends RecyclerView.Adapter<UserAdapter.UserHolder> {
    private List<Users> usersList;
    private iClickListener listener;

    public UserAdapter(iClickListener listener) { this.listener = listener; }

    public void setData(List<Users> list){
        this.usersList = list;
        notifyDataSetChanged();
    }
}
```

□ Bước 7: Cập nhật User

```
@Override
public void onBindViewHolder(@NonNull UserHolder holder, int position) {
    Users users = usersList.get(position);
    if(users == null){
        return;
    }
    holder.username.setText(users.getUsername());
    holder.address.setText(users.getAddress());

    //xử lý sự kiện
    holder.btnUpdate.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { listener.updateUser(users); }
    });
}
```

Sử lý sự kiện cho nút Update

□ Bước 7: Cập nhật User

```
private static final int MY_REQUEST_CODE = 10 ;
public static final String TAG = MainActivity.class.getName();

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    AnhXa();

    userAdapter = new UserAdapter(new iClickListener() {
        @Override
        public void updateUser(Users users) { clickUpdateUser(users); }
    });

    usersList = new ArrayList<>();
    loadData();
    rc_list.setHasFixedSize(false);
    LinearLayoutManager layoutManager = new LinearLayoutManager(context: this);
    rc_list.setLayoutManager(layoutManager);
    rc_list.setAdapter(userAdapter);
    btnThem.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { addUser(); }
    });
}
```

Khai báo hằng số trong MainActivity.java

Chỉ lại cách khởi tạo Adapter trong MainActivity.java

□ Bước 7: Cập nhật User

```
private void clickUpdateUser(Users users) {
    Intent intent = new Intent( packageContext MainActivity.this, UpdateUserActivity.class);
    Bundle bundle = new Bundle();
    bundle.putSerializable("object name", users);
    intent.putExtras(bundle);
    mActivityResultLauncher.launch(intent);
}
```

Viết hàm truyền dữ liệu gói qua cho UpdateUserActivity trong MainActivity.java

Khai báo ActivityResultLauncher thay cho startActivityForResult() đã khai tử trong MainActivity.java

```
private ActivityResultLauncher<Intent> mActivityResultLauncher = registerForActivityResult(
    new ActivityResultContracts.StartActivityForResult(),
    new ActivityResultCallback<ActivityResult>() {
        @Override
        public void onActivityResult(ActivityResult result) {
            Log.e(TAG, msg: "onActivityResult");
            if(result.getResultCode()== Activity.RESULT_OK){
                //request code
                Intent data = result.getData();
                if(data==null){
                    return;
                }
                loadData();
            }
        }
    }
);
```

Hàm lấy tất cả user trong Room

```
public void loadData(){
    //lấy danh sách user trong Room
    usersList= UserDatabase.getInstance(this).usersDao().getAll();
    userAdapter.setData(usersList);
}
```

□ Bước 7: Cập nhật User

```
public class UpdateUserActivity extends AppCompatActivity {
    private EditText editTextUsername;
    private EditText editTextAddress;
    private Button btnUpdate;
    private Users users;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_update_user);
        AnhXa();

        users = (Users) getIntent().getExtras().get("object name");
        if(users != null){
            editTextUsername.setText(users.getUsername());
            editTextAddress.setText(users.getAddress());
        }

        btnUpdate.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { updateUsers(); }
        });
    }
}
```

Tạo Empty Activity
và đặt tên
UpdateUserActivity

Nhận gói dữ liệu từ
MainActivity

Bắt sự kiện nút Cập
nhật User trong
UpdateUserActivity

□ Bước 7: Cập nhật User

```
private void updateUsers() {
    String username = editTextUsername.getText().toString().trim();
    String address = editTextAddress.getText().toString().trim();
    if(TextUtils.isEmpty(username) || TextUtils.isEmpty(address)){
        return;
    }
    users.setUsername(username);
    users.setAddress(address);
    UserDatabase.getInstance(this).usersDao().update(users);
    Toast.makeText(context: this, text: "Cập nhật thành công", toast.LENGTH_SHORT).show();
    Intent intentResult = new Intent();
    setResult(Activity.RESULT_OK,intentResult);
    finish();
}
```

Gọi hàm update()
trong UserDao

Trả kết quả về
MainActivity

Hàm ánh xạ các View
trong UpdateUserActivity

```
private void AnhXa() {
    editTextUsername = findViewById(R.id.editTextUsername);
    editTextAddress = findViewById(R.id.editTextAddress);
    btnUpdate = findViewById(R.id.btnUpdate);
}
```

□ Bước 7: Cập nhật User

CẬP NHẬT USER

Giao diện của activity_update_user.xml
Copy từ giao diện của activity_main.xml

Tạo hàm update() trong
Interface UserDao()

```

@Dao
public interface UserDao {
    @Query("SELECT * FROM users")
    List<Users> getAll();

    @Query("SELECT * FROM users WHERE id IN (:userIds)")
    List<Users> loadAllByIds(int[] userIds);

    @Insert
    void insertAll(Users... users);

    @Update
    void update(Users... users);

    @Delete
    void delete(Users user);

    @Query("SELECT * FROM users where username= :username")
    List<Users> checkUser(String username);
}

```

□ Bước 8: Xóa User

Tạo hàm update() trong Interface UserDao()

```
@Dao
public interface UserDao {
    @Query("SELECT * FROM users")
    List<Users> getAll();

    @Query("SELECT * FROM users WHERE id IN (:userIds)")
    List<Users> loadAllByIds(int[] userIds);

    @Insert
    void insertAll(Users... users);

    @Update
    void update(Users... users);

    @Delete
    void delete(Users user);

    @Query("SELECT * FROM users where username= :username")
    List<Users> checkUser(String username);
}
```


□ Bước 8: Xóa User

Bổ sung Button xóa và ánh xạ trong UserAdapter



```
public class UserHolder extends RecyclerView.ViewHolder{
    private TextView username;
    private TextView address;
    private Button btnUpdate, btnDelete;
    public UserHolder(@NonNull View itemView) {
        super(itemView);
        username = itemView.findViewById(R.id.textviewUsername);
        address = itemView.findViewById(R.id.textviewAddress);
        btnUpdate = itemView.findViewById(R.id.btnUpdate);
        btnDelete= itemView.findViewById(R.id.btnDelete);
    }
}
```

□ Bước 8: Xóa User

Khai báo thêm hàm
deleteUser trong
Interface

```
public interface iClickListener{
    void updateUser(Users users);
    void deleteUser(Users users);
}
```

Implement method thêm hàm
deleteUser() trong
MainActivity.java

```
userAdapter = new UserAdapter(new iClickListener() {
    @Override
    public void updateUser(Users users) {
        clickUpdateUser(users);
    }
    @Override
    public void deleteUser(Users users) {
        clickDeleteUser(users);
    }
});
```

□ Bước 8: Xóa User

Viết hàm xử lý xóa 01 dữ liệu trong Room

Xóa bảng users trong Room

```
@Query("DELETE FROM users")
void deleteAll();
```

```
private void clickDeleteUser(Users users) {
    new AlertDialog.Builder( context: this)
        .setTitle("Confirm delete user")
        .setMessage("Are you sure?")
        .setPositiveButton( text: "Yes", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                UserDatabase.getInstance(MainActivity.this).usersDao().delete(users);
                Toast.makeText( context: MainActivity.this, text: "Đã xóa thành công", Toast.LENGTH_SHORT).show();
                loadData();
            }
        })
        .setNegativeButton( text: "No", listener: null)
        .show();
}
```

□ Bước 9: Search User

Thiết kế EditText cho chức năng search

```
<EditText
    android:id="@+id/editTextSearch"
    android:layout_width="match_parent"
    android:layout_height="35dp"
    android:layout_margin="20dp"
    android:layout_marginBottom="16dp"
    android:drawableStart="@drawable/ic_baseline_search_24"
    android:background="@drawable/back_search"
    android:ems="10"
    android:hint="Bạn cần tìm gì?"
    android:imeOptions="actionSearch"
    android:inputType="text"
    android:paddingStart="10dp"
    app:layout_constraintBottom_toTopOf="@+id/scrollView2"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.4"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/tvDeleteAll" />
```

Room Database

Nhập user name

Nhập địa chỉ

THÊM USER

Delete All

🔍 Bạn cần tìm gì?

Trung
hcm

CẬP NHẬT

XÓA

□ Bước 9: Search User

Khai báo hàm
searchName trong
UseDao()

```
@Query("SELECT * FROM users WHERE username LIKE '%' || :name || '%'")
List<Users> searchName(String name);
```

```
editSearch.setOnEditorActionListener(new TextView.OnEditorActionListener() {
    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
        if(actionId == EditorInfo.IME_ACTION_SEARCH){
            searchUser();
        }
        return false;
    }
});
```

Bắt xử kiện nút search
trên bàn phím ảo của
thiết bị

```
private void searchUser() {
    String strKey = editSearch.getText().toString().trim();
    usersList = new ArrayList<>();
    usersList = UserDatabase.getInstance(MainActivity.this).usersDao().searchName(strKey);
    userAdapter.setData(usersList);
    hideSoftKeyboard();
}
```

Gọi hàm searchName()

- Bước 10: Xử lý lỗi Room nếu có thay đổi Entities
- Ví dụ ta thêm 01 trường mới là email vào Entities User thì khi chạy chương trình sẽ báo lỗi như sau

```
@Entity(tableName = "users")
public class Users implements Serializable {
    @PrimaryKey(autoGenerate = true)
    private int id;
    private String username;
    private String address;
    private String email;

    public Users(String username, String address) {
        this.username = username;
        this.address = address;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }
}
```

Tạo getter và setter cho trường email

IllegalStateException: A migration from 1 to 2 was required but not found. Please provide the necessary Migration path via RoomDatabase.Builder.addMigration(Migration ...)

- ❑ Bước 10: Xử lý lỗi Room nếu có thay đổi Entities
- ❑ Ta thay đổi lại FirebaseDatabase.java như sau

```
@Database(entities = {Users.class}, version = 2)
public abstract class FirebaseDatabase extends RoomDatabase {
    //migration database version 1 to version 2 khi entities thay đổi
    static Migration migration_1_to_2 = new Migration(1,2) {
        @Override
        public void migrate(@NonNull SupportSQLiteDatabase database) {
            database.execSQL("ALTER TABLE users ADD COLUMN email TEXT");
        }
    };
};
```

Sửa lại version = 1
thành version = 2

Tạo static Migration để
migrate từ ver 1 sang
2 dùng ALTER TABLE....

```
private static final String DATABASE_NAME = "user.db";
private static FirebaseDatabase instance;

public static synchronized FirebaseDatabase getInstance(Context context){
    if(instance == null){
        instance = Room.databaseBuilder(context.getApplicationContext()
            ,FDatabase.class,DATABASE_NAME)
            .allowMainThreadQueries()
            .addMigrations(migration_1_to_2)
            .build();
    }
    return instance;
}

public abstract UsersDao usersDao();
}
```

Gọi migrate

- Nguyễn Hữu Trung
- 0908617108
- trungnh@hcmute.edu.vn
- utex.hcmute.edu.vn