****

**UNIVERSITI MALAYSIA TERENGGANU**

**FACULTY OF COMPUTER SCIENCE AND MATHEMATICS**

**­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**CSM3123 - NATIVE MOBILE PROGRAMMING**

**BANCHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING) WITH HONORS**

**LAB 4**

**SEMESTER 5 2024/2025**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**PREPARED FOR:**

**DR RABIEI B MAMAT**

**PREPARED BY:**

**NUR EZREENA SHUHADA BT EMRAN**

**S66467**

**Link Github:** [**https://github.com/NurEzreena/CSM3123\_LAB-NATIVE-PROGRAMMING.git**](https://github.com/NurEzreena/CSM3123_LAB-NATIVE-PROGRAMMING.git)

**Activity: To understand and implement a Room database for local data storage in Android applications.**

MainActivity.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.os.Bundle; import android.text.TextUtils; import android.view.View; import android.widget.EditText; import android.widget.Toast;  import androidx.appcompat.app.AppCompatActivity; import androidx.lifecycle.ViewModelProvider; import androidx.recyclerview.widget.LinearLayoutManager; import androidx.recyclerview.widget.RecyclerView;  import java.util.List;  public class MainActivity extends AppCompatActivity {   private UserViewModel userViewModel;  private EditText editTextName, editTextAge;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);   // Initialize UI components  editTextName = findViewById(R.id.*edit\_text\_name*);  editTextAge = findViewById(R.id.*edit\_text\_age*);  RecyclerView recyclerView = findViewById(R.id.*recycler\_view*);   // Setup RecyclerView  recyclerView.setLayoutManager(new LinearLayoutManager(this));  recyclerView.setHasFixedSize(true);   // Initialize the RecyclerView Adapter  UserAdapter adapter = new UserAdapter();  recyclerView.setAdapter(adapter);   // Initialize ViewModel  userViewModel = new ViewModelProvider(this).get(UserViewModel.class);   // Observe the LiveData from the ViewModel  userViewModel.getAllUsers().observe(this, adapter::setUsers);   // Set the Add button click listener  findViewById(R.id.*button\_add*).setOnClickListener(v -> addUser());  }   // Method to add a user to the database  private void addUser() {  String name = editTextName.getText().toString().trim();  String ageText = editTextAge.getText().toString().trim();   // Validate input  if (TextUtils.*isEmpty*(name) || TextUtils.*isEmpty*(ageText)) {  Toast.*makeText*(this, "Please enter both name and age!", Toast.*LENGTH\_SHORT*).show();  return;  }   int age;  try {  age = Integer.*parseInt*(ageText);  } catch (NumberFormatException e) {  Toast.*makeText*(this, "Age must be a number!", Toast.*LENGTH\_SHORT*).show();  return;  }   // Add user to the database  User user = new User(name, age);  userViewModel.insert(user);   // Clear the input fields  editTextName.setText("");  editTextAge.setText("");   Toast.*makeText*(this, "User added successfully!", Toast.*LENGTH\_SHORT*).show();  } } |

User.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import androidx.room.Entity; import androidx.room.PrimaryKey;  @Entity(tableName = "user\_table") public class User {  @PrimaryKey(autoGenerate = true)  private int id;  private String name;  private int age;   public User(String name, int age) {  this.name = name;  this.age = age;  }   public int getId() {  return id;  }   public void setId(int id) {  this.id = id;  }   public String getName() {  return name;  }   public int getAge() {  return age;  } } |

UserAdapter.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.TextView;  import androidx.annotation.NonNull; import androidx.recyclerview.widget.RecyclerView;  import java.util.ArrayList; import java.util.List;  public class UserAdapter extends RecyclerView.Adapter<UserAdapter.UserHolder> {  private List<User> users = new ArrayList<>();   @NonNull    @Override  public UserHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  View itemView = LayoutInflater.*from*(parent.getContext())  .inflate(R.layout.*user\_item*, parent, false); // Ensure this matches user\_item.xml  return new UserHolder(itemView);  }    @Override  public void onBindViewHolder(@NonNull UserHolder holder, int position) {  User currentUser = users.get(position);  holder.textViewName.setText(currentUser.getName());  holder.textViewAge.setText(String.*valueOf*(currentUser.getAge()));  }   @Override  public int getItemCount() {  return users.size();  }   public void setUsers(List<User> users) {  this.users = users;  notifyDataSetChanged();  }   static class UserHolder extends RecyclerView.ViewHolder {  private TextView textViewName;  private TextView textViewAge;   public UserHolder(@NonNull View itemView) {  super(itemView);  textViewName = itemView.findViewById(R.id.*text\_view\_name*);  textViewAge = itemView.findViewById(R.id.*text\_view\_age*);  }  } } |

UserDao.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import androidx.lifecycle.LiveData; import androidx.room.Dao; import androidx.room.Delete; import androidx.room.Insert; import androidx.room.Query; import androidx.room.Update;  import java.util.List;  @Dao public interface UserDao {  @Insert  void insert(User user);   @Update  void update(User user);   @Delete  void delete(User user);   @Query("SELECT \* FROM user\_table ORDER BY id ASC")  LiveData<List<User>> getAllUsers();   @Query("SELECT \* FROM user\_table WHERE name LIKE :searchQuery")  LiveData<List<User>> searchUsers(String searchQuery); } |

UserDatabase.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.content.Context;  import androidx.room.Database; import androidx.room.Room; import androidx.room.RoomDatabase;  // Annotate the class as a Room Database and declare its entities and version @Database(entities = {User.class}, version = 1, exportSchema = false) public abstract class UserDatabase extends RoomDatabase {   // Singleton instance of the database  private static UserDatabase *instance*;   // Abstract method for accessing DAO  public abstract UserDao userDao();   // Synchronized method to get the instance of the database  public static synchronized UserDatabase getInstance(Context context) {  if (*instance* == null) {  // Create the database if it does not exist  *instance* = Room.*databaseBuilder*(context.getApplicationContext(),  UserDatabase.class, "user\_database")  .fallbackToDestructiveMigration() // Handle migrations destructively  .build();  }  return *instance*;  } } |

UserRepository.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.app.Application;  import androidx.lifecycle.LiveData;  import java.util.List;  public class UserRepository {  private UserDao userDao;  private LiveData<List<User>> allUsers;   public UserRepository(Application application) {  UserDatabase database = UserDatabase.*getInstance*(application);  userDao = database.userDao();  allUsers = userDao.getAllUsers();  }   public void insert(User user) {  new Thread(() -> userDao.insert(user)).start();  }   public void update(User user) {  new Thread(() -> userDao.update(user)).start();  }   public void delete(User user) {  new Thread(() -> userDao.delete(user)).start();  }   public LiveData<List<User>> getAllUsers() {  return allUsers;  }   public LiveData<List<User>> searchUsers(String query) {  return userDao.searchUsers(query);  } } |

UserViewModel.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.app.Application;  import androidx.annotation.NonNull; import androidx.lifecycle.AndroidViewModel; import androidx.lifecycle.LiveData;  import java.util.List;  public class UserViewModel extends AndroidViewModel {  private UserRepository repository;  private LiveData<List<User>> allUsers;   public UserViewModel(@NonNull Application application) {  super(application);  repository = new UserRepository(application);  allUsers = repository.getAllUsers();  }   public void insert(User user) {  repository.insert(user);  }   public void update(User user) {  repository.update(user);  }   public void delete(User user) {  repository.delete(user);  }   public LiveData<List<User>> getAllUsers() {  return allUsers;  }   public LiveData<List<User>> searchUsers(String query) {  return repository.searchUsers(query);  } } |

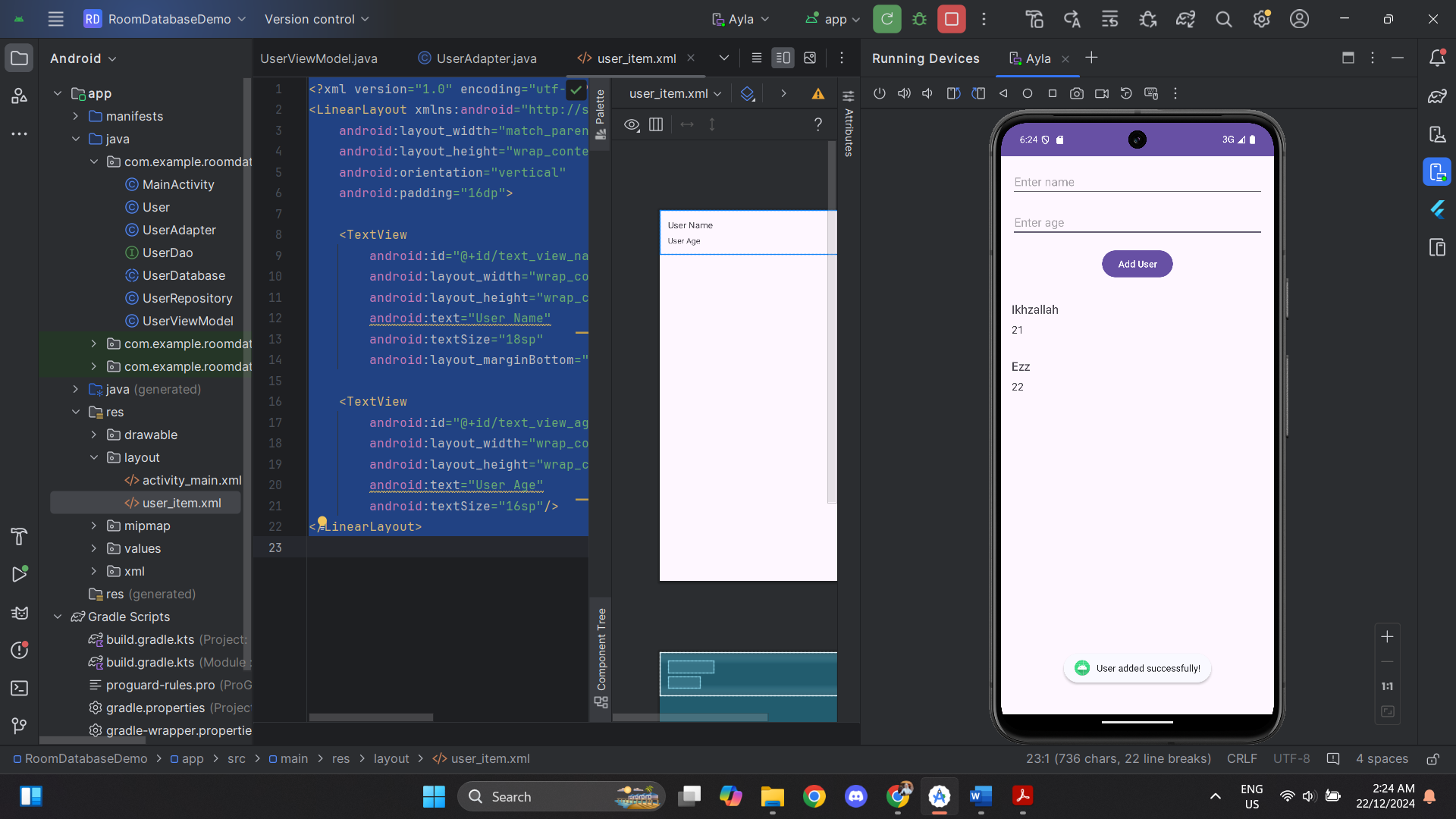
Activity\_main.xml

|  |
| --- |
| <?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">   <!-- Input Fields -->  <EditText  android:id="@+id/edit\_text\_name"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:hint="Enter name"  android:layout\_margin="16dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />   <EditText  android:id="@+id/edit\_text\_age"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:hint="Enter age"  android:inputType="number"  android:layout\_margin="16dp"  app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_name"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />   <!-- Buttons -->  <Button  android:id="@+id/button\_add"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Add User"  android:layout\_margin="16dp"  app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_age"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />   <!-- RecyclerView -->  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/recycler\_view"  android:layout\_width="0dp"  android:layout\_height="0dp"  android:layout\_marginTop="16dp"  app:layout\_constraintTop\_toBottomOf="@id/button\_add"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout> |

User\_item.xml

|  |
| --- |
| <?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="16dp">   <TextView  android:id="@+id/text\_view\_name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="User Name"  android:textSize="18sp"  android:layout\_marginBottom="8dp"/>   <TextView  android:id="@+id/text\_view\_age"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="User Age"  android:textSize="16sp"/> </LinearLayout> |

Output:



**Activity 1: Add Update and Delete Operations**

MainActivity.xml

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.os.Bundle; import android.text.TextUtils; import android.view.LayoutInflater; import android.view.View; import android.widget.Button; import android.widget.EditText; import android.widget.LinearLayout; import android.widget.Toast;  import androidx.appcompat.app.AlertDialog; import androidx.appcompat.app.AppCompatActivity; import androidx.lifecycle.ViewModelProvider; import androidx.recyclerview.widget.LinearLayoutManager; import androidx.recyclerview.widget.RecyclerView;  public class MainActivity extends AppCompatActivity {   private UserViewModel userViewModel;  private EditText editTextName, editTextAge;  public Button buttonAddUser;  private User selectedUserForUpdate; // Tracks the user selected for update   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);   // Initialize UI components for adding users  editTextName = findViewById(R.id.*edit\_text\_name*);  editTextAge = findViewById(R.id.*edit\_text\_age*);  buttonAddUser = findViewById(R.id.*button\_add*);   RecyclerView recyclerView = findViewById(R.id.*recycler\_view*);  recyclerView.setLayoutManager(new LinearLayoutManager(this));  recyclerView.setHasFixedSize(true);   // Set up Adapter  UserAdapter adapter = new UserAdapter();  recyclerView.setAdapter(adapter);   // Initialize ViewModel  userViewModel = new ViewModelProvider(this).get(UserViewModel.class);  userViewModel.getAllUsers().observe(this, adapter::setUsers);   // Add User button functionality  buttonAddUser.setOnClickListener(v -> addUser());   // Handle RecyclerView item click listeners  adapter.setOnItemClickListener(new UserAdapter.OnItemClickListener() {  @Override  public void onUpdateClick(User user) {  selectedUserForUpdate = user;   // Show Update dialog  showUpdateDialog(user);  }   @Override  public void onDeleteClick(User user) {  userViewModel.delete(user);  Toast.*makeText*(MainActivity.this, "User deleted successfully!", Toast.*LENGTH\_SHORT*).show();  }  });  }   // Method to add a new user  private void addUser() {  String name = editTextName.getText().toString().trim();  String ageText = editTextAge.getText().toString().trim();   if (TextUtils.*isEmpty*(name) || TextUtils.*isEmpty*(ageText)) {  Toast.*makeText*(this, "Please enter both name and age!", Toast.*LENGTH\_SHORT*).show();  return;  }   int age;  try {  age = Integer.*parseInt*(ageText);  } catch (NumberFormatException e) {  Toast.*makeText*(this, "Age must be a valid number!", Toast.*LENGTH\_SHORT*).show();  return;  }   User user = new User(name, age);  userViewModel.insert(user);   // Clear input fields  editTextName.setText("");  editTextAge.setText("");   Toast.*makeText*(this, "User added successfully!", Toast.*LENGTH\_SHORT*).show();  }   // Method to show the Update dialog  private void showUpdateDialog(User user) {  // Create an AlertDialog to update user info  AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);  builder.setTitle("Update User");   // Set up the input fields (name and age)  final EditText inputName = new EditText(MainActivity.this);  inputName.setText(user.getName()); // Pre-fill with the current name   final EditText inputAge = new EditText(MainActivity.this);  inputAge.setText(String.*valueOf*(user.getAge())); // Pre-fill with the current age  inputAge.setInputType(android.text.InputType.*TYPE\_CLASS\_NUMBER*); // Set input type to number   // Create a layout to hold the inputs  LinearLayout layout = new LinearLayout(MainActivity.this);  layout.setOrientation(LinearLayout.*VERTICAL*);  layout.addView(inputName);  layout.addView(inputAge);  builder.setView(layout);   // Set up the positive button for updating  builder.setPositiveButton("Update", (dialog, which) -> {  String updatedName = inputName.getText().toString().trim();  String updatedAgeText = inputAge.getText().toString().trim();   if (TextUtils.*isEmpty*(updatedName) || TextUtils.*isEmpty*(updatedAgeText)) {  Toast.*makeText*(MainActivity.this, "Please enter both name and age!", Toast.*LENGTH\_SHORT*).show();  return;  }   int updatedAge;  try {  updatedAge = Integer.*parseInt*(updatedAgeText);  } catch (NumberFormatException e) {  Toast.*makeText*(MainActivity.this, "Age must be a valid number!", Toast.*LENGTH\_SHORT*).show();  return;  }   // Update the user object  user.setName(updatedName);  user.setAge(updatedAge);   // Call ViewModel to update the user in the database  userViewModel.update(user);   // Notify user of success  Toast.*makeText*(MainActivity.this, "User updated successfully!", Toast.*LENGTH\_SHORT*).show();  });   // Set up the negative button (cancel)  builder.setNegativeButton("Cancel", (dialog, which) -> {  dialog.dismiss();  });   // Show the dialog  builder.show();  } } |

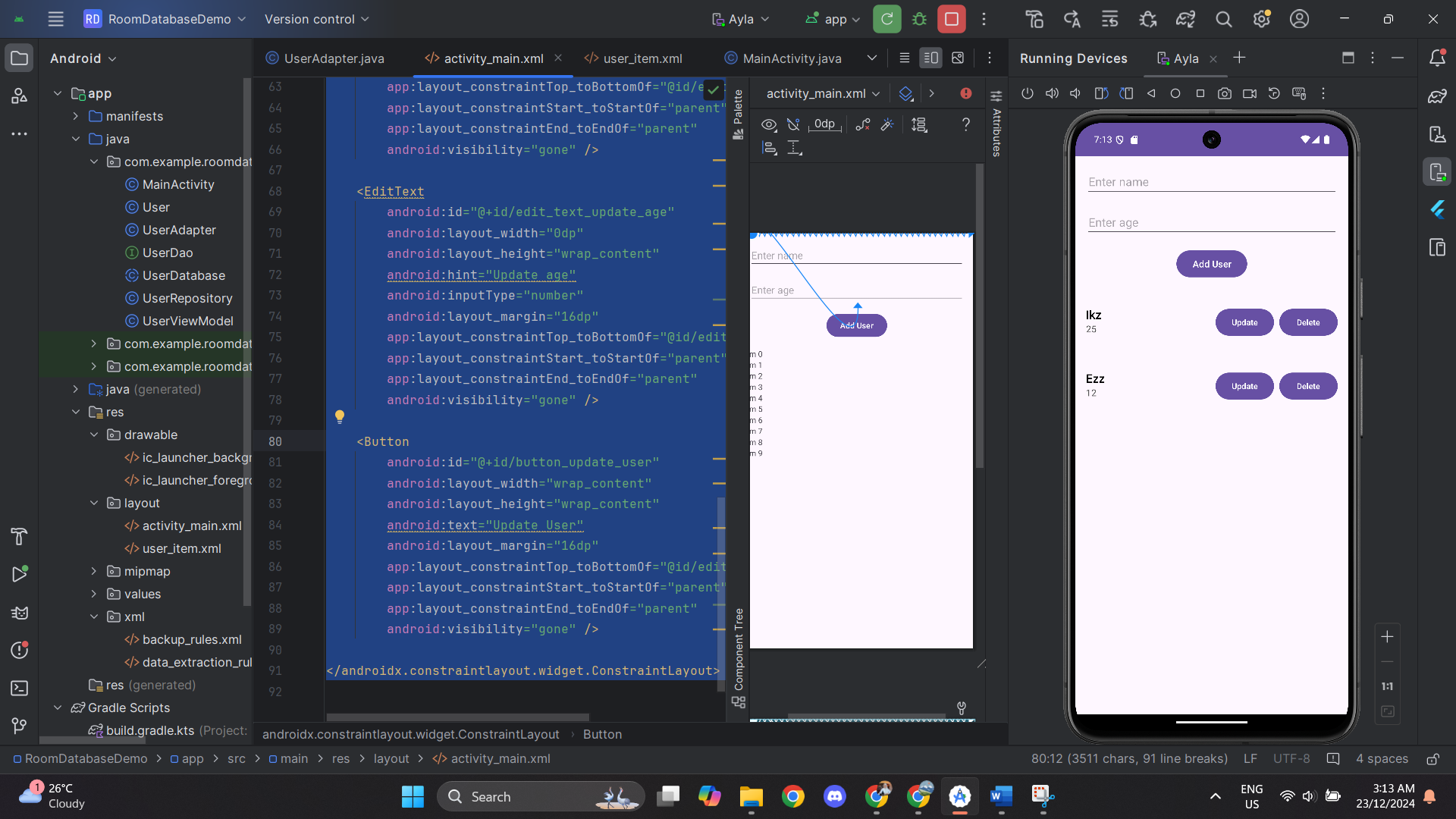
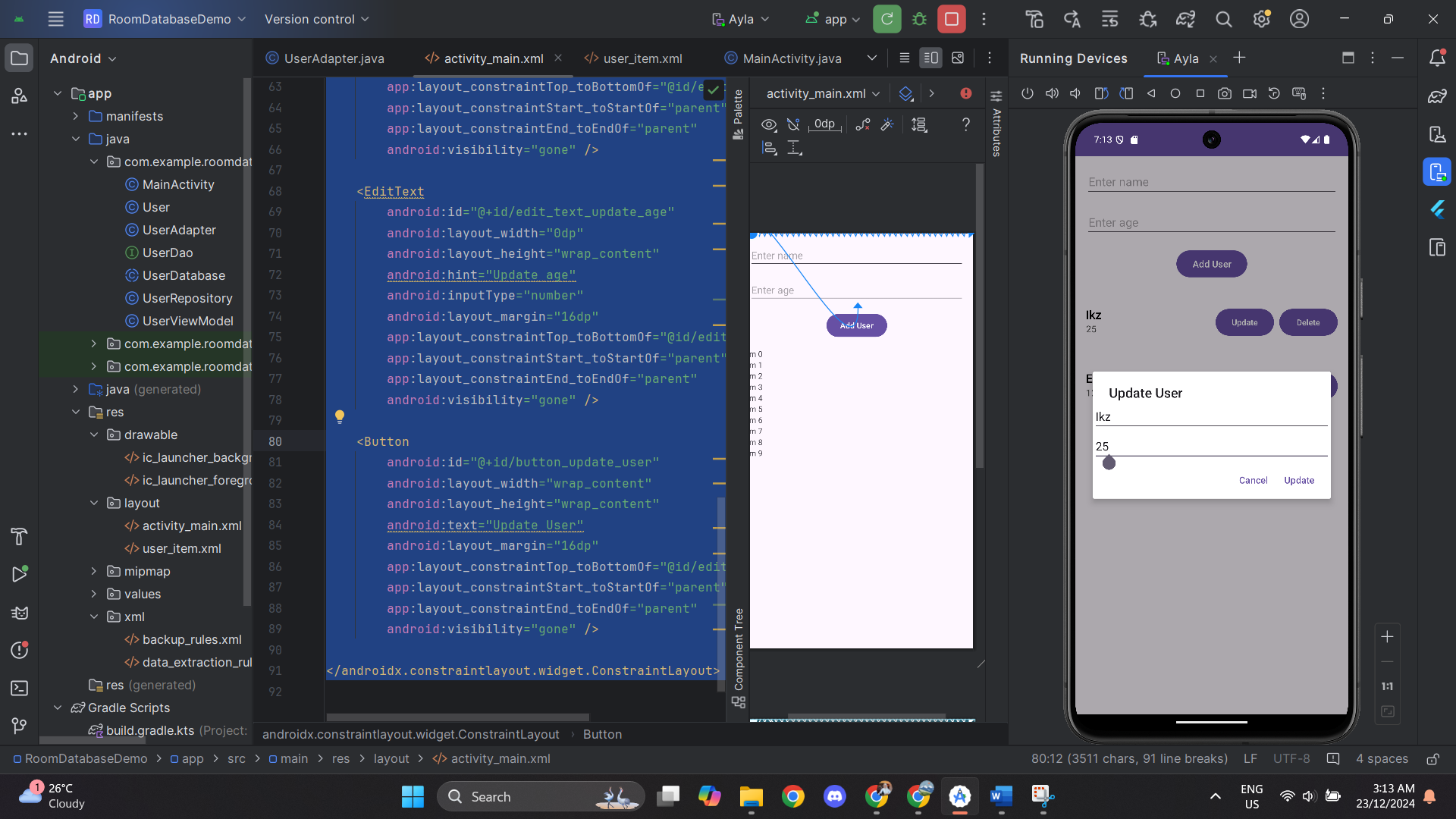
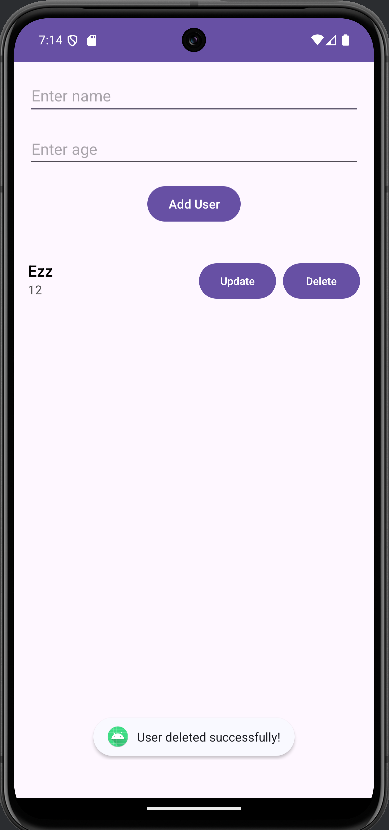
User\_item.xml

|  |
| --- |
| <?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="wrap\_content"  android:padding="16dp"  android:layout\_marginTop="8dp"  android:layout\_marginBottom="8dp"  android:elevation="2dp"  android:background="?android:attr/selectableItemBackground"  tools:context=".MainActivity">   <!-- User Name TextView -->  <TextView  android:id="@+id/text\_view\_name"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:text="User Name"  android:textSize="18sp"  android:textColor="#000000"  android:fontFamily="sans-serif-medium"  android:layout\_marginEnd="8dp"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintEnd\_toStartOf="@id/button\_update"/>   <!-- User Age TextView -->  <TextView  android:id="@+id/text\_view\_age"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:text="User Age"  android:textSize="14sp"  android:textColor="#555555"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@id/text\_view\_name"  app:layout\_constraintEnd\_toStartOf="@id/button\_update"/>   <!-- Update Button -->  <Button  android:id="@+id/button\_update"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Update"  android:textSize="12sp"  android:layout\_marginStart="8dp"  app:layout\_constraintStart\_toEndOf="@id/text\_view\_age"  app:layout\_constraintTop\_toTopOf="@id/text\_view\_name"  app:layout\_constraintEnd\_toStartOf="@id/button\_delete"/>   <!-- Delete Button -->  <Button  android:id="@+id/button\_delete"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Delete"  android:textSize="12sp"  android:layout\_marginStart="8dp"  app:layout\_constraintStart\_toEndOf="@id/button\_update"  app:layout\_constraintTop\_toTopOf="@id/text\_view\_name"  app:layout\_constraintEnd\_toEndOf="parent"/>  </androidx.constraintlayout.widget.ConstraintLayout> |

UserAdapter.java

|  |
| --- |
| package com.example.roomdatabasedemo;  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.ArrayList; import java.util.List;  public class UserAdapter extends RecyclerView.Adapter<UserAdapter.UserHolder> {   private List<User> users = new ArrayList<>();  private OnItemClickListener listener;   // Interface for button click events  public interface OnItemClickListener {  void onUpdateClick(User user);  void onDeleteClick(User user);  }   public void setOnItemClickListener(OnItemClickListener listener) {  this.listener = listener;  }   @NonNull  @Override  public UserHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  View itemView = LayoutInflater.*from*(parent.getContext())  .inflate(R.layout.*user\_item*, parent, false);  return new UserHolder(itemView);  }   @Override  public void onBindViewHolder(@NonNull UserHolder holder, int position) {  User currentUser = users.get(position);  holder.textViewName.setText(currentUser.getName());  holder.textViewAge.setText(String.*valueOf*(currentUser.getAge()));   // Update button click event  holder.buttonUpdate.setOnClickListener(v -> {  if (listener != null) {  listener.onUpdateClick(currentUser);  }  });   // Delete button click event  holder.buttonDelete.setOnClickListener(v -> {  if (listener != null) {  listener.onDeleteClick(currentUser);  }  });  }   @Override  public int getItemCount() {  return users.size();  }   public void setUsers(List<User> users) {  this.users = users;  notifyDataSetChanged();  }   static class UserHolder extends RecyclerView.ViewHolder {  private TextView textViewName;  private TextView textViewAge;  private Button buttonUpdate;  private Button buttonDelete;   public UserHolder(@NonNull View itemView) {  super(itemView);  textViewName = itemView.findViewById(R.id.*text\_view\_name*);  textViewAge = itemView.findViewById(R.id.*text\_view\_age*);  buttonUpdate = itemView.findViewById(R.id.*button\_update*);  buttonDelete = itemView.findViewById(R.id.*button\_delete*);  }  } } |

Output:

Button update Button delete

**Activity 2: Add a Search Feature**

UserDao.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import androidx.lifecycle.LiveData; import androidx.room.Dao; import androidx.room.Delete; import androidx.room.Insert; import androidx.room.Query; import androidx.room.Update;  import java.util.List;  @Dao public interface UserDao {  @Insert  void insert(User user);   @Update  void update(User user);    @Delete  void delete(User user);   @Query("SELECT \* FROM user\_table ORDER BY id ASC")  LiveData<List<User>> getAllUsers();   @Query("SELECT \* FROM user\_table WHERE name LIKE :searchQuery")  LiveData<List<User>> searchUsers(String searchQuery); } |

UserRepository.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.app.Application;  import androidx.lifecycle.LiveData;  import java.util.List;  public class UserRepository {  private UserDao userDao;  private LiveData<List<User>> allUsers;   public UserRepository(Application application) {  UserDatabase database = UserDatabase.*getInstance*(application);  userDao = database.userDao();  allUsers = userDao.getAllUsers();  }   public void insert(User user) {  new Thread(() -> userDao.insert(user)).start();  }   public void update(User user) {  new Thread(() -> userDao.update(user)).start();  }   public void delete(User user) {  new Thread(() -> userDao.delete(user)).start();  }   public LiveData<List<User>> getAllUsers() {  return allUsers;  }   public LiveData<List<User>> searchUsers(String query) {  return userDao.searchUsers(query);  } } |

UserViewModel.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.app.Application;  import androidx.annotation.NonNull; import androidx.lifecycle.AndroidViewModel; import androidx.lifecycle.LiveData;  import java.util.List;  public class UserViewModel extends AndroidViewModel {  private UserRepository repository;  private LiveData<List<User>> allUsers;   public UserViewModel(@NonNull Application application) {  super(application);  repository = new UserRepository(application);  allUsers = repository.getAllUsers();  }   public void insert(User user) {  repository.insert(user);  }   public void update(User user) {  repository.update(user);  }   public void delete(User user) {  repository.delete(user);  }   public LiveData<List<User>> getAllUsers() {  return allUsers;  }   public LiveData<List<User>> searchUsers(String query) {  return repository.searchUsers(query);  } } |

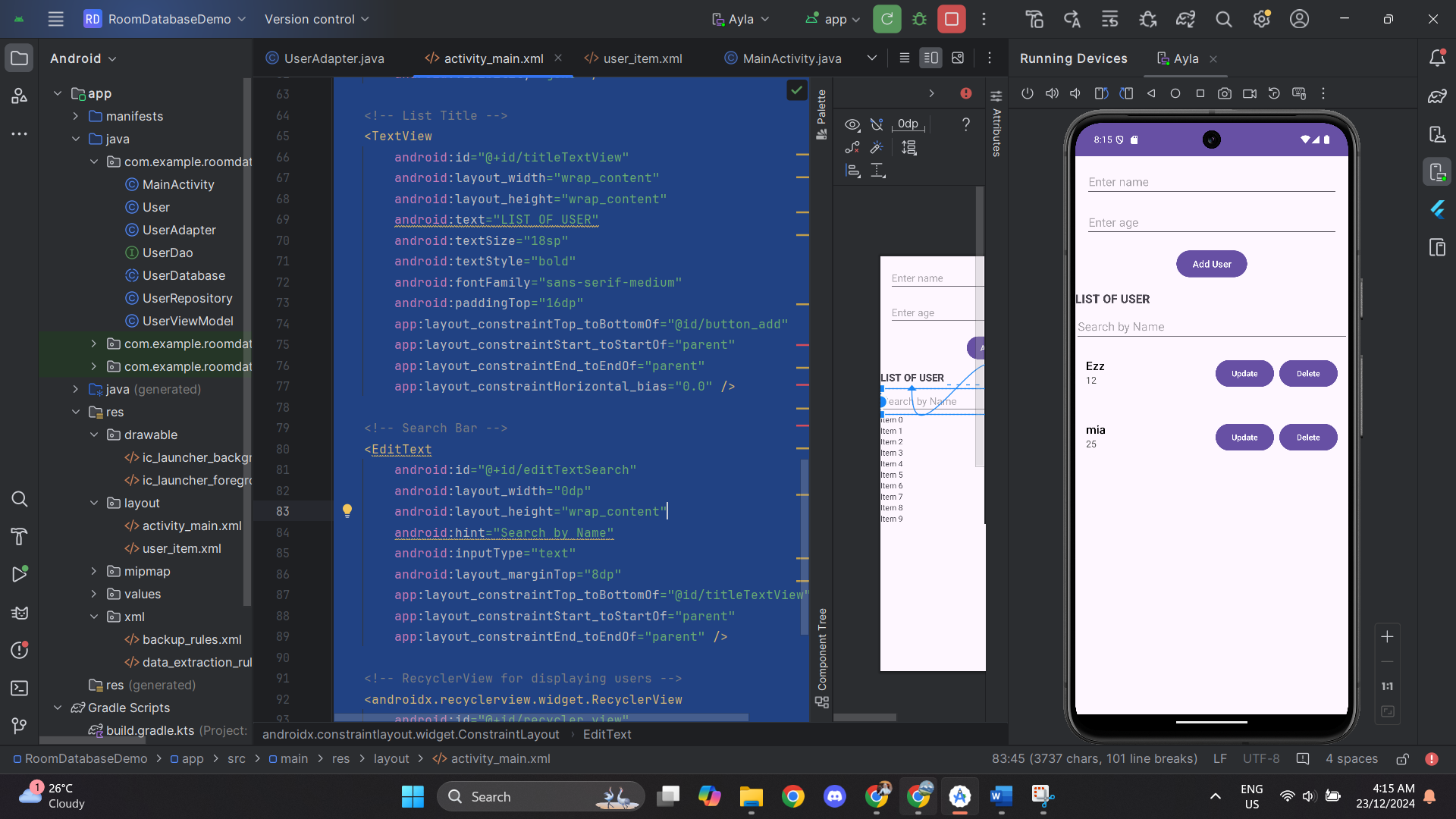
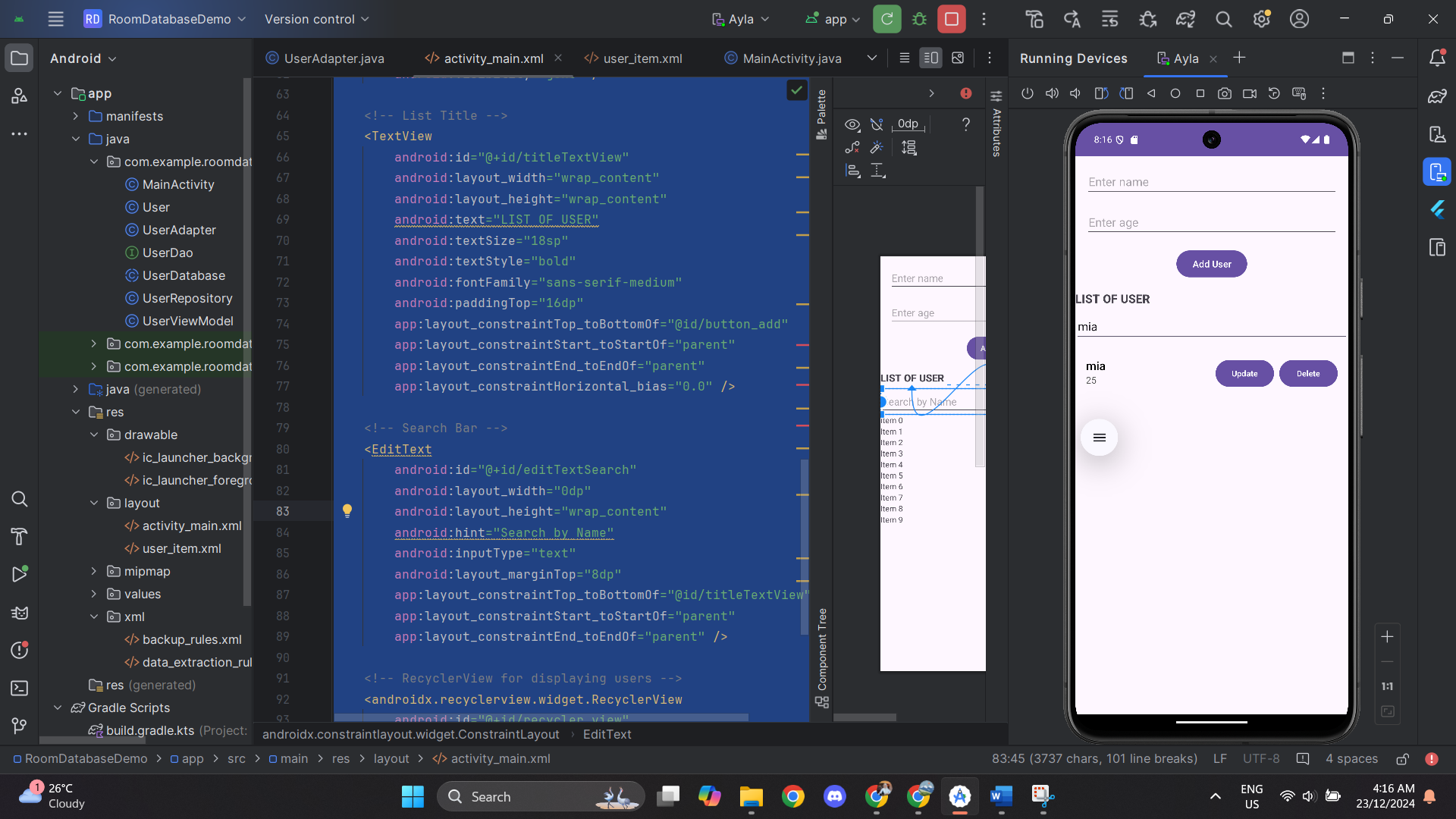
MainActivity.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.annotation.SuppressLint;  import android.os.Bundle;  import android.text.Editable;  import android.text.TextUtils;  import android.text.TextWatcher;  import android.view.View;  import android.widget.Button;  import android.widget.EditText;  import android.widget.Toast;  import androidx.appcompat.app.AppCompatActivity;  import androidx.lifecycle.ViewModelProvider;  import androidx.recyclerview.widget.LinearLayoutManager;  import androidx.recyclerview.widget.RecyclerView;  public class MainActivity extends AppCompatActivity {  private UserViewModel userViewModel;  private EditText editTextName, editTextAge, editTextSearch, editTextUpdateName, editTextUpdateAge;  private Button buttonAddUser, buttonUpdateUser;  private User selectedUserForUpdate; // Tracks the user selected for update  @SuppressLint("MissingInflatedId")  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_main);  // Initialize UI components  editTextName = findViewById(R.id.edit\_text\_name);  editTextAge = findViewById(R.id.edit\_text\_age);  editTextSearch = findViewById(R.id.editTextSearch);  editTextUpdateName = findViewById(R.id.edit\_text\_update\_name); // Initialize the update name EditText  editTextUpdateAge = findViewById(R.id.edit\_text\_update\_age); // Initialize the update age EditText  buttonAddUser = findViewById(R.id.button\_add);  buttonUpdateUser = findViewById(R.id.button\_update\_user);  RecyclerView recyclerView = findViewById(R.id.recycler\_view);  recyclerView.setLayoutManager(new LinearLayoutManager(this));  recyclerView.setHasFixedSize(true);  // Set up Adapter  UserAdapter adapter = new UserAdapter();  recyclerView.setAdapter(adapter);  // Initialize ViewModel  userViewModel = new ViewModelProvider(this).get(UserViewModel.class);  userViewModel.getAllUsers().observe(this, adapter::setUsers);  // Add User button functionality  buttonAddUser.setOnClickListener(v -> addUser());  // Update User button functionality  buttonUpdateUser.setOnClickListener(v -> updateUser());  // Handle RecyclerView item click listeners for update and delete  adapter.setOnItemClickListener(new UserAdapter.OnItemClickListener() {  @Override  public void onUpdateClick(User user) {  selectedUserForUpdate = user;  // Show Update fields and populate with selected user data  editTextUpdateName.setVisibility(View.VISIBLE);  editTextUpdateAge.setVisibility(View.VISIBLE);  buttonUpdateUser.setVisibility(View.VISIBLE);  editTextUpdateName.setText(user.getName());  editTextUpdateAge.setText(String.valueOf(user.getAge()));  }  @Override  public void onDeleteClick(User user) {  userViewModel.delete(user);  Toast.makeText(MainActivity.this, "User deleted successfully!", Toast.LENGTH\_SHORT).show();  }  });  // Search functionality  editTextSearch.addTextChangedListener(new TextWatcher() {  @Override  public void beforeTextChanged(CharSequence charSequence, int start, int count, int after) {}  @Override  public void onTextChanged(CharSequence charSequence, int start, int before, int count) {  String searchQuery = charSequence.toString().trim();  if (TextUtils.isEmpty(searchQuery)) {  // If search is cleared, show all users  userViewModel.getAllUsers().observe(MainActivity.this, adapter::setUsers);  } else {  // If search query is not empty, show filtered users  userViewModel.searchUsers(searchQuery).observe(MainActivity.this, adapter::setUsers);  }  }  @Override  public void afterTextChanged(Editable editable) {}  });  }  // Method to add a new user  private void addUser() {  String name = editTextName.getText().toString().trim();  String ageText = editTextAge.getText().toString().trim();  if (TextUtils.isEmpty(name) || TextUtils.isEmpty(ageText)) {  Toast.makeText(this, "Please enter both name and age!", Toast.LENGTH\_SHORT).show();  return;  }  int age;  try {  age = Integer.parseInt(ageText);  } catch (NumberFormatException e) {  Toast.makeText(this, "Age must be a valid number!", Toast.LENGTH\_SHORT).show();  return;  }  User user = new User(name, age);  userViewModel.insert(user);  // Clear input fields after adding user  editTextName.setText("");  editTextAge.setText("");  Toast.makeText(this, "User added successfully!", Toast.LENGTH\_SHORT).show();  }  // Method to update an existing user  private void updateUser() {  if (selectedUserForUpdate == null) {  Toast.makeText(this, "No user selected for update!", Toast.LENGTH\_SHORT).show();  return;  }  String updatedName = editTextUpdateName.getText().toString().trim();  String updatedAgeText = editTextUpdateAge.getText().toString().trim();  if (TextUtils.isEmpty(updatedName) || TextUtils.isEmpty(updatedAgeText)) {  Toast.makeText(this, "Update fields cannot be empty!", Toast.LENGTH\_SHORT).show();  return;  }  int updatedAge;  try {  updatedAge = Integer.parseInt(updatedAgeText);  } catch (NumberFormatException e) {  Toast.makeText(this, "Age must be a valid number!", Toast.LENGTH\_SHORT).show();  return;  }  // Update the user object with the new name and age  selectedUserForUpdate.setName(updatedName);  selectedUserForUpdate.setAge(updatedAge);  userViewModel.update(selectedUserForUpdate);  // Hide the update fields and clear them  editTextUpdateName.setVisibility(View.GONE);  editTextUpdateAge.setVisibility(View.GONE);  buttonUpdateUser.setVisibility(View.GONE);  editTextUpdateName.setText("");  editTextUpdateAge.setText("");  selectedUserForUpdate = null;  Toast.makeText(this, "User updated successfully!", Toast.LENGTH\_SHORT).show();  }  } |

Activity\_main.java

|  |
| --- |
| <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">   <!-- Input Fields -->  <EditText  android:id="@+id/edit\_text\_name"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:hint="Enter name"  android:layout\_margin="16dp"  app:layout\_constraintTop\_toTopOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />   <EditText  android:id="@+id/edit\_text\_age"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:hint="Enter age"  android:inputType="number"  android:layout\_margin="16dp"  app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_name"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />   <!-- Add User Button -->  <Button  android:id="@+id/button\_add"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Add User"  android:layout\_margin="16dp"  app:layout\_constraintTop\_toBottomOf="@id/edit\_text\_age"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />   <!-- Update Fields -->  <EditText  android:id="@+id/edit\_text\_update\_name"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Update Name"  android:visibility="gone" />   <EditText  android:id="@+id/edit\_text\_update\_age"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Update Age"  android:visibility="gone" />   <Button  android:id="@+id/button\_update\_user"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Update User"  android:visibility="gone" />   <!-- List Title -->  <TextView  android:id="@+id/titleTextView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="LIST OF USER"  android:textSize="18sp"  android:textStyle="bold"  android:fontFamily="sans-serif-medium"  android:paddingTop="16dp"  app:layout\_constraintTop\_toBottomOf="@id/button\_add"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintHorizontal\_bias="0.0" />   <!-- Search Bar -->  <EditText  android:id="@+id/editTextSearch"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:hint="Search by Name"  android:inputType="text"  android:layout\_marginTop="8dp"  app:layout\_constraintTop\_toBottomOf="@id/titleTextView"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent" />   <!-- RecyclerView for displaying users -->  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/recycler\_view"  android:layout\_width="0dp"  android:layout\_height="0dp"  app:layout\_constraintTop\_toBottomOf="@id/editTextSearch"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintBottom\_toBottomOf="parent" />  </androidx.constraintlayout.widget.ConstraintLayout> |

Output:

Search bar

**Activity 3: Add Relationships Between Entities**

MainActivity.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.os.Bundle; import android.text.TextUtils; import android.view.View; import android.widget.Button; import android.widget.EditText; import android.widget.Toast;  import androidx.appcompat.app.AppCompatActivity; import androidx.lifecycle.ViewModelProvider; import androidx.recyclerview.widget.LinearLayoutManager; import androidx.recyclerview.widget.RecyclerView;  public class MainActivity extends AppCompatActivity {   private UserViewModel userViewModel;  private EditText editTextName, editTextAge, editTextSearch;  private Button buttonAddUser;  private RecyclerView recyclerView;  private UserAdapter adapter;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);   // Initialize UI components  editTextName = findViewById(R.id.*edit\_text\_name*);  editTextAge = findViewById(R.id.*edit\_text\_age*);  editTextSearch = findViewById(R.id.*editTextSearch*);  buttonAddUser = findViewById(R.id.*button\_add*);   recyclerView = findViewById(R.id.*recycler\_view*);  recyclerView.setLayoutManager(new LinearLayoutManager(this));  recyclerView.setHasFixedSize(true);   // Set up Adapter  adapter = new UserAdapter();  recyclerView.setAdapter(adapter);   // Initialize ViewModel  userViewModel = new ViewModelProvider(this).get(UserViewModel.class);  userViewModel.getAllUsers().observe(this, adapter::setUsers);   // Add User button functionality  buttonAddUser.setOnClickListener(v -> addUser());   // Handle RecyclerView item click listeners for user and task display  adapter.setOnItemClickListener(new UserAdapter.OnItemClickListener() {  @Override  public void onUpdateClick(User user) {  // Handle user update functionality if needed  }   @Override  public void onDeleteClick(User user) {  userViewModel.delete(user);  Toast.*makeText*(MainActivity.this, "User deleted successfully!", Toast.*LENGTH\_SHORT*).show();  }  });  }   // Method to add a new user  private void addUser() {  String name = editTextName.getText().toString().trim();  String ageText = editTextAge.getText().toString().trim();   if (TextUtils.*isEmpty*(name) || TextUtils.*isEmpty*(ageText)) {  Toast.*makeText*(this, "Please enter both name and age!", Toast.*LENGTH\_SHORT*).show();  return;  }   int age;  try {  age = Integer.*parseInt*(ageText);  } catch (NumberFormatException e) {  Toast.*makeText*(this, "Age must be a valid number!", Toast.*LENGTH\_SHORT*).show();  return;  }   User user = new User(name, age);  userViewModel.insert(user);   // Clear input fields after adding user  editTextName.setText("");  editTextAge.setText("");   Toast.*makeText*(this, "User added successfully!", Toast.*LENGTH\_SHORT*).show();  } } |

Task.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import androidx.room.Entity; import androidx.room.PrimaryKey; import androidx.room.ForeignKey;  @Entity(tableName = "task\_table", foreignKeys = @ForeignKey(entity = User.class, parentColumns = "id", childColumns = "user\_id", onDelete = ForeignKey.*CASCADE*)) public class Task {  @PrimaryKey(autoGenerate = true)  private int id;   private String taskName;  private String taskDescription;   // Foreign key for the User that the task belongs to  private int userId;   // Constructor, getters, and setters  public Task(String taskName, String taskDescription, int userId) {  this.taskName = taskName;  this.taskDescription = taskDescription;  this.userId = userId;  }   public int getId() {  return id;  }   public void setId(int id) {  this.id = id;  }   public String getTaskName() {  return taskName;  }   public void setTaskName(String taskName) {  this.taskName = taskName;  }   public String getTaskDescription() {  return taskDescription;  }   public void setTaskDescription(String taskDescription) {  this.taskDescription = taskDescription;  }   public int getUserId() {  return userId;  }   public void setUserId(int userId) {  this.userId = userId;  } } |

UserWithTask.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import androidx.room.Embedded; import androidx.room.Relation;  import java.util.List;  public class UserWithTask {   @Embedded  public User user;   @Relation(  parentColumn = "id",  entityColumn = "userId"  )  public List<Task> tasks; } |

User.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import androidx.room.Entity; import androidx.room.PrimaryKey;  @Entity(tableName = "user\_table") public class User {  @PrimaryKey(autoGenerate = true)  private int id;  private String name;  private int age;   public User(String name, int age) {  this.name = name;  this.age = age;  }   // Getter and Setter for ID  public int getId() {  return id;  }   public void setId(int id) {  this.id = id;  }   // Getter and Setter for Name  public String getName() {  return name;  }   public void setName(String name) {  this.name = name;  }   // Getter and Setter for Age  public int getAge() {  return age;  }   public void setAge(int age) {  this.age = age;  } } |

UserDao.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import androidx.lifecycle.LiveData; import androidx.room.Dao; import androidx.room.Delete; import androidx.room.Insert; import androidx.room.Query; import androidx.room.Update; import androidx.room.Transaction;  import java.util.List;  @Dao public interface UserDao {   @Insert  void insert(User user);   @Update  void update(User user);   @Delete  void delete(User user);   @Query("SELECT \* FROM user\_table ORDER BY id ASC")  LiveData<List<User>> getAllUsers();   @Query("SELECT \* FROM user\_table WHERE name LIKE :searchQuery")  LiveData<List<User>> searchUsers(String searchQuery);   // Insert task  @Insert  void insertTask(Task task);   // Get tasks for a specific user  @Transaction  @Query("SELECT \* FROM user\_table WHERE id = :userId")   LiveData<List<UserWithTask>> getUserWithTask(int userId); } |

UserRepository.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.app.Application;  import androidx.lifecycle.LiveData;  import java.util.List;  public class UserRepository {  private UserDao userDao;  private LiveData<List<User>> allUsers;   public UserRepository(Application application) {  UserDatabase database = UserDatabase.*getInstance*(application);  userDao = database.userDao();  allUsers = userDao.getAllUsers();  }   // Insert User  public void insert(User user) {  new Thread(() -> userDao.insert(user)).start();  }   // Update User  public void update(User user) {  new Thread(() -> userDao.update(user)).start();  }   // Delete User  public void delete(User user) {  new Thread(() -> userDao.delete(user)).start();  }   // Get all users  public LiveData<List<User>> getAllUsers() {  return allUsers;  }   // Search users  public LiveData<List<User>> searchUsers(String query) {  return userDao.searchUsers(query);  }   // Insert Task  public void insertTask(Task task) {  new Thread(() -> userDao.insertTask(task)).start();  }   // Get tasks for a specific user  public LiveData<List<UserWithTask>> getUserWithTasks(int userId) {  return userDao.getUserWithTask(userId);  } } |

UserViewModel.java

|  |
| --- |
| package com.example.roomdatabasedemo;  import android.app.Application;  import androidx.annotation.NonNull; import androidx.lifecycle.AndroidViewModel; import androidx.lifecycle.LiveData;  import java.util.List;  public class UserViewModel extends AndroidViewModel {  private UserRepository repository;  private LiveData<List<User>> allUsers;   public UserViewModel(@NonNull Application application) {  super(application);  repository = new UserRepository(application);  allUsers = repository.getAllUsers();  }   // Insert User  public void insert(User user) {  repository.insert(user);  }   // Update User  public void update(User user) {  repository.update(user);  }   // Delete User  public void delete(User user) {  repository.delete(user);  }   // Get all users  public LiveData<List<User>> getAllUsers() {  return allUsers;  }   // Search users  public LiveData<List<User>> searchUsers(String query) {  return repository.searchUsers(query);  }   // Insert Task  public void insertTask(Task task) {  repository.insertTask(task);  }   // Get tasks for a specific user  public LiveData<List<UserWithTask>> getUserWithTasks(int userId) {  return repository.getUserWithTasks(userId);  } } |