

KAUNAS UNIVERSITY OF TECHNOLOGY

FACULTY OF INFORMATICS

T120B169 App Development for Smart Mobile Systems

*Car Deal*

|  |
| --- |
| *Group, Name and Surname:*  *IFU-8 Pedro García-Cañal Sánchez*  *IFU-8 José Antonio Pérez Miguel*  *IFE-8 Edvinas Kulboka*  *IFU-8 Mohanad Kher* |
| Date: *18/10/2021* |

Kaunas, 2021

Tables of Contents

[Description of Your app 3](#_Toc87952682)

[Functionality of your app 4](#_Toc87952683)

[List of functions (adapt to your own app) 4](#_Toc87952684)

[Solution 5](#_Toc87952685)

[Task #1. Login 5](#_Toc87952686)

[Task #2. Register 6](#_Toc87952687)

[Task #3. Add Car 8](#_Toc87952688)

[Task #4. Update Car Information 10](#_Toc87952689)

[Task #5. HTTP Scraping 11](#_Toc87952690)

[Task #e. Recycler View 16](#_Toc87952691)

[Defense 1 18](#_Toc87952692)

[Defense 2 19](#_Toc87952693)

[Reference list 19](#_Toc87952694)

# **Description of Your app**

1. **What type is your application/game?**

Car sales app

1. **Description.**

This car sales application will help users upload pictures, a description, defects, the specification of the car the user wants to sell. The goal of this application is to make it simpler and faster to create adverts for a car sale and to keep track of these adverts. It will use a database to store sales and users that are using the application. Notifications will be used to inform users if any new sales get created in the application, that the user might be interested in. The app will also show sales from other sources, by parsing ad sites. Additionally, each advert for a car sale can be shared to friends or followers using social media.

# **Functionality of your app**

## **List of functions (adapt to your own app)**

1. Login
2. Register
3. Recycler view(view the list of users from database)

# **Solution**

## **Task #1. Login**

Description of the implementation .

*The application’s first screen is the login window where you can sign in if you already have an account, if you don’t you can click on the “I don’t have an account…” button and the application will send you to the registration window.*

*To login, email and password will be required.*

UI components:

* Application Logo (Image View).
* Email (EditText).
* Password (EditText).
* Login (Button).
* I don’t have an account (Button).

All these components are inside of a linear layout.

Graphical user interface, application

Description automatically generated

Figure 1. Login

protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_login*);  
 emailInput = findViewById(R.id.*emailInput*);  
 passwordInput = findViewById(R.id.*passwordInput*);  
 toRegister = findViewById(R.id.*toRegisterBtn*);  
 loginButton = findViewById(R.id.*loginBtn*);  
 toRegister.setOnClickListener(v -> {  
 Intent toRegister = new Intent(LoginActivity.this, RegisterActivity.class);  
 startActivity(toRegister);  
 });  
  
 userInformationViewModel = new ViewModelProvider .AndroidViewModelFactory(LoginActivity.this  
 .getApplication()).create(UserInformationViewModel.class);  
  
 loginButton.setOnClickListener(view -> {String email = emailInput.getText().toString().trim();  
 String password = passwordInput.getText().toString().trim();  
 UserInformation userInformation = new UserInformation(email,password);  
 if (!TextUtils.*isEmpty*(email) && !TextUtils.*isEmpty*(password))  
 {  
 if (userInformationViewModel.getUser(email)!=null){  
 Intent logSuccess = new Intent(LoginActivity.this, MainActivity.class);  
 startActivity(logSuccess);  
 }else {  
 Toast.*makeText*(this,R.string.*failLogin*,Toast.*LENGTH\_SHORT*).show();  
 }  
 }else {  
 Toast.*makeText*(this,R.string.*empty*,Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
}

## **Task #2. Register**

Description of the implementation .

In this screen you can see a form that you have to fill with your information to complete the registration. If you already have an account you can press the “Sign in instead” button and you will be sent to the login. Also when you press the register button to finish with this process, the app will send you to the login too, so you can log in in your newly created account.

UI Components:

* Application Logo (Image View).
* Email (EditText).
* Password (EditText).
* ConfirmPassword (EditText).
* Register (Button).
* Sign in instead (Button).

All these components are inside of a linear layout.

Graphical user interface

Description automatically generated with medium confidence

Figure 2. Registration

protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_register*);  
 registerButton = findViewById(R.id.*registerBtn*);  
 emailInput = findViewById(R.id.*emailInput*);  
 passwordInput = findViewById(R.id.*passwordInput*);  
 confirmPasswordInput = findViewById(R.id.*confPasswordInput*);  
 toLogin = findViewById(R.id.*toLoginBtn*);  
  
 userInformationViewModel = new ViewModelProvider.AndroidViewModelFactory(RegisterActivity.this  
 .getApplication()).create(UserInformationViewModel.class);  
  
 toLogin.setOnClickListener(view -> {Intent toLogin = new Intent(RegisterActivity.this,  
 LoginActivity.class);  
 startActivity(toLogin);  
 });  
  
 registerButton.setOnClickListener(v -> {  
 String email = emailInput.getText().toString().trim();  
 String password = passwordInput.getText().toString().trim();  
 String confPassword = confirmPasswordInput.getText().toString().trim();  
  
 if (!TextUtils.*isEmpty*(email) && !TextUtils.*isEmpty*(password) && !TextUtils.*isEmpty*(confPassword))  
 {  
 if (password.equals(confPassword))  
 {  
 UserInformation userInformation = new UserInformation(email,password);  
 UserInformationViewModel.*insert*(userInformation);  
 Intent toLogin = new Intent(RegisterActivity.this, LoginActivity.class);  
 startActivity(toLogin);  
 }else {  
 Toast.*makeText*(this,R.string.*passNotMatch*,Toast.*LENGTH\_SHORT*).show();  
 }  
 }else  
 {  
 Toast.*makeText*(this,R.string.*empty*,Toast.*LENGTH\_SHORT*).show();  
 }  
 });

## **Task #3. Add Car**

We created a window where you can add a new car ad. The user can see some inputs to write the model, a description, the manufacturer and the price.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_add\_car);  
 modelInput = findViewById(R.id.modelInput);  
 descInput = findViewById(R.id.descInput);  
 manufactureInput = findViewById(R.id.manufactureInput);  
 priceTx = findViewById(R.id.priceInput);  
 carImage = findViewById(R.id.carImage);  
 addCarBtn = findViewById(R.id.updateCarBtn);  
 selectImageBtn = findViewById(R.id.selectImage);  
 selectImageBtn.setOnClickListener(this);  
 carViewModel = new ViewModelProvider  
 .AndroidViewModelFactory(AddCarActivity.this.getApplication())  
 .create(CarViewModel.class);  
 addCarBtn.setOnClickListener(v -> {  
 String model = modelInput.getText().toString().trim();  
 String description = descInput.getText().toString().trim();  
 String price = priceTx.getText().toString().trim();  
 Double priceVal = Double.parseDouble(priceTx.getText().toString());  
 String manufacturer = manufactureInput.getText().toString().trim();  
 Bundle data = getIntent().getExtras();  
 if(data!=null){FK = data.getString(MainActivity.USER\_EMAIL);}  
 Log.d("FK", "onCreate:FK " + FK);  
 if (!TextUtils.isEmpty(model) && !TextUtils.isEmpty(description) && !TextUtils.isEmpty(price)  
 && !TextUtils.isEmpty(manufacturer))  
 {  
 CarEntity carEntity = new CarEntity(model,manufacturer,description,priceVal,FK);  
 CarViewModel.insert(carEntity);  
 Intent carAdded = new Intent(AddCarActivity.this, MainActivity.class);  
 startActivity(carAdded);  
 }else  
 {  
 Toast.makeText(this,R.string.empty,Toast.LENGTH\_SHORT).show();  
 }  
 });  
  
 Bundle data = getIntent().getExtras();  
 if(data!=null){  
 int id = data.getInt(MainActivity.CAR\_ID);  
 carViewModel.get(id).observe(this, carEntity -> {  
 modelInput.setText(carEntity.getModel());  
 descInput.setText(carEntity.getDescription());  
 manufactureInput.setText(carEntity.getManufacturer());  
 priceTx.setText(carEntity.getPrice().toString());  
 });  
 }  
}

## **Task #4. Update Car Information**

In this window you can change an ad information or delete it.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_update\_car\_info*);  
  
 modelInput = findViewById(R.id.*modelInput*);  
 descInput = findViewById(R.id.*descInput*);  
 manufactureInput = findViewById(R.id.*manufactureInput*);  
 priceTx = findViewById(R.id.*priceInput*);  
 updateCarBtn = findViewById(R.id.*updateCarBtn*);  
 deleteCarBtn = findViewById(R.id.*deleteCarBtn*);  
  
 carViewModel = new ViewModelProvider  
 .AndroidViewModelFactory(UpdateCarInfoActivity.this.getApplication())  
 .create(CarViewModel.class);  
  
 if(getIntent().hasExtra(MainActivity.*CAR\_ID*)){  
 carId = getIntent().getIntExtra(MainActivity.*CAR\_ID*,0);  
 carViewModel.get(carId).observe(this, carEntity -> {  
 if (carEntity!=null){  
 modelInput.setText(carEntity.getModel());  
 descInput.setText(carEntity.getDescription());  
 manufactureInput.setText(carEntity.getManufacturer());  
 priceTx.setText(carEntity.getPrice().toString());  
 }  
 });  
 isEdit = true;  
 }  
 deleteCarBtn.setOnClickListener(view -> {  
 edit(true);  
 });  
 updateCarBtn.setOnClickListener(view -> {  
 edit(false);  
 });  
  
}

## **Task #5. HTTP Scraping**

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

public class MainActivity extends AppCompatActivity {  
  
 private RecyclerView recyclerView;  
 private RecyclerViewAdapter recyclerViewAdapter;  
 private CarInformationViewModel carInformationViewModel;  
 private RequestQueue requestQueue;  
 private int PageNumber = 1;  
 private LinearLayoutManager manager;  
 private ProgressBar progressBar;  
  
 private boolean isButtonAvailable = false;  
 private boolean isScrolling = false;  
 int currentItems, totalItems, scrollOutItems;  
  
  
 private static final int *InternetRequestCode* = 1;  
  
 Button loadMoreVehiclesButton;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 requestQueue = Volley.newRequestQueue(MainActivity.this);  
 setContentView(R.layout.*activity\_main*);  
 manager = new LinearLayoutManager(this);  
  
 progressBar = findViewById(R.id.progressBar);  
 recyclerView = findViewById(R.id.*recyclerViewId*);  
 recyclerView.setHasFixedSize(true);  
 recyclerView.setLayoutManager(manager);  
 recyclerView.addOnScrollListener(new RecyclerView.OnScrollListener() {  
 @Override  
 public void onScrollStateChanged(RecyclerView recyclerView, int newState) {  
 super.onScrollStateChanged(recyclerView, newState);  
 if (newState == AbsListView.OnScrollListener.SCROLL\_STATE\_TOUCH\_SCROLL)  
 isScrolling = true;  
 }  
  
 @Override  
 public void onScrolled(RecyclerView recyclerView, int dx, int dy) {  
 super.onScrolled(recyclerView, dx, dy);  
  
 currentItems = manager.getChildCount();  
 totalItems = manager.getItemCount();  
 scrollOutItems = manager.findFirstVisibleItemPosition();  
  
 if (isScrolling && (currentItems + scrollOutItems == totalItems) && !isButtonAvailable)  
 {  
 isScrolling = false;  
 QueueRequestToOutsources();  
 }  
 }  
 });  
  
  
 carInformationViewModel = new ViewModelProvider  
 .AndroidViewModelFactory(MainActivity.this.getApplication())  
 .create(CarInformationViewModel.class);  
  
 carInformationViewModel  
 .getCars()  
 .observe(this, carInformation -> {  
 // Set the adapter  
 recyclerViewAdapter = new RecyclerViewAdapter(carInformation,MainActivity.this);  
 recyclerView.setAdapter(recyclerViewAdapter);  
 });  
  
 loadMoreVehiclesButton = findViewById(R.id.loadMoreVehiclesButton);  
 loadMoreVehiclesButton.setOnClickListener(v -> {  
 loadMoreVehiclesButtonPressed();  
 });  
 }  
  
 private void loadMoreVehiclesButtonPressed()  
 {  
 if (ContextCompat.checkSelfPermission( this, Manifest.permission.INTERNET) == PackageManager.PERMISSION\_GRANTED)  
 {  
 RemoveMoreVehiclesButton();  
 QueueRequestToOutsources();  
 }  
 else  
 ActivityCompat.requestPermissions(this, new String[] {Manifest.permission.INTERNET}, *InternetRequestCode*);  
 }  
  
 @SuppressLint("MissingSuperCall")  
 @Override  
 public void onRequestPermissionsResult(int requestCode, String permissions[], int[] grantResults)  
 {  
 switch (requestCode)  
 {  
 case *InternetRequestCode*:  
 if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION\_GRANTED)  
 QueueRequestToOutsources();  
 else  
 RemoveMoreVehiclesButton();  
  
 return;  
 }  
 }  
  
 private void RemoveMoreVehiclesButton()  
 {  
 ViewGroup parentView = (ViewGroup) loadMoreVehiclesButton.getParent();  
 parentView.removeView(loadMoreVehiclesButton);  
 isButtonAvailable = false;  
 }  
  
 private void QueueRequestToOutsources()  
 {  
 progressBar.setVisibility(View.*VISIBLE*);  
 String request = "https://en.autoplius.lt/ads/used-cars?order\_by=3&order\_direction=DESC&page\_nr=" + PageNumber;// for further requests add &page\_nr=2  
 PageNumber = PageNumber + 1;  
 StringRequest stringRequest = new StringRequest(Request.Method.GET, request,  
 new Response.Listener<String>() {  
 @Override  
 public void onResponse(String response) {  
 ParseVehicles(response);  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
  
 }  
 });  
  
 requestQueue.add(stringRequest);  
 }  
  
 private void ParseVehicles(String response)  
 {  
 Scanner scanner = new Scanner(response);  
 boolean started = false;  
 List list = new ArrayList();  
 while(scanner.hasNextLine())  
 {  
 String line = scanner.nextLine();  
 if (started)  
 {  
 if(line.contains("</a>"))  
 {  
 try{  
 AddCarFromStrings(list);  
 }  
 catch (Exception ex) {  
 Toast.*makeText*(this, "Failed to load vehicle", Toast.*LENGTH\_SHORT*);  
 }  
 list.clear();  
 }  
 else if(!line.trim().isEmpty())  
 list.add(line.trim());  
 }  
 else if (line.contains("<div class=\"list-items\">"))  
 started = true;  
 }  
 progressBar.setVisibility(View.*GONE*);  
 recyclerViewAdapter.notifyDataSetChanged();  
 }  
  
 private void AddCarFromStrings(List<String> list)  
 {  
 String description = "";  
 Double price = 0.0;  
 String city = "";  
 String link = "";  
 String imageLink = "";  
  
 for(int i = 0; i<list.size(); i++)  
 {  
 String line = list.get(i);  
 if (line.contains("<div class=\"line1\">"))  
 {  
 String descriptionLine = list.get(i+1).trim();  
 if (descriptionLine.contains("</div>"))  
 descriptionLine = descriptionLine.replace("</div>", "");  
 description = descriptionLine.trim();  
 }  
 else if (line.contains("<div class=\"pricing-container\">"))  
 {  
 String priceString = list.get(i+2).trim();  
 price = Double.*parseDouble*(priceString.substring(0, priceString.indexOf("&")-1).replace(" ", ""));  
 }  
 else if (line.contains("<div class=\"item-parameters\">"))  
 {  
 String gasLine = list.get(i+2);  
 String gasType = gasLine.substring(0, gasLine.length() - 7).trim();  
  
 String volumeAndPowerLine = list.get(i+4);  
 String volumeAndPower = volumeAndPowerLine.substring(0, volumeAndPowerLine.length() - 7).trim();  
  
 String gearboxLine = list.get(i+5);  
 String gearbox = gearboxLine.substring(6, gearboxLine.length() - 7);  
  
 String kilometrageLine = list.get(i+6);  
 String kilometrage = kilometrageLine.substring(6, kilometrageLine.length() - 7);  
  
 String bodyTypeLine = list.get(i+7);  
 String bodyType = bodyTypeLine.substring(6, bodyTypeLine.length() - 7);  
  
 String cityLine = list.get(i+8).trim();  
 city = cityLine.substring(6,cityLine.length()-7);  
 }  
 else if(line.contains("href"))  
 {  
 String trimmed = line.trim();  
 link = trimmed.substring(trimmed.indexOf("=")+2, trimmed.length() - 1);  
 }  
 else if(line.contains("img class"))  
 {  
 String identifier = "src=\"";  
 imageLink = line.substring(line.indexOf(identifier)+identifier.length()-1, line.length() - 8);  
 }  
 }  
 if (price != 0.0 && !description.isEmpty())  
 {  
 CarInformation carInfo = new CarInformation(description, city, price, link, imageLink);  
  
 recyclerViewAdapter.addItem(carInfo);  
 }  
 }  
  
 @Override  
 protected void onStop()  
 {  
 super.onStop();  
 if(requestQueue != null)  
 requestQueue.cancelAll(*InternetRequestCode*);  
 }  
}

**Task #6. Design for the ads cards**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

@Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 addPost = findViewById(R.id.*addCadrId*);  
 recyclerView = findViewById(R.id.*recyclerViewId*);  
 recyclerView.setHasFixedSize(true);   
  
 carViewModel = new ViewModelProvider  
 .AndroidViewModelFactory(MainActivity.this.getApplication())  
 .create(CarViewModel.class);  
  
 carViewModel.getCars().observe(this, carEntities -> {  
 // Set the adapter  
 recyclerViewAdapter = new RecyclerViewAdapter(carEntities,MainActivity.this,this);  
 recyclerView.setAdapter(recyclerViewAdapter);  
 });  
  
 Bundle userData = getIntent().getExtras();  
 if(userData!=null){  
 CarFK = userData.getString(LoginActivity.*USER\_EMAIL*);  
 }  
 addPost.setOnClickListener(view -> {  
 Intent addCar = new Intent(MainActivity.this,AddCarActivity.class);  
 startActivity(addCar);  
 });  
 }

## **Task #e. Recycler View**

For this to work we created to classes as shown below in the code.

Class RecyclerViewAdapter., and ViewHolder within it.

public class RecyclerViewAdapter extends RecyclerView.Adapter <RecyclerViewAdapter.ViewHolder>{  
 private List<UserInformation> userList;  
 private Context context;  
  
 public RecyclerViewAdapter(List<UserInformation> userList, Context context) {  
 this.userList = userList;  
 this.context = context;  
 }  
  
 @NonNull  
 @Override  
 public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext())  
 .inflate(R.layout.*recycler\_row*,parent,false);  
 return new ViewHolder(view);  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull ViewHolder holder, int position) {  
 UserInformation userInformation = Objects.*requireNonNull*(userList.get(position));  
 holder.email.setText(userInformation.getEmail());  
 }  
  
 @Override  
 public int getItemCount() {  
 return Objects.*requireNonNull*(userList.size());  
 }  
  
 public class ViewHolder extends RecyclerView.ViewHolder {  
 public TextView email;  
 public ViewHolder(@NonNull View itemView) {  
 super(itemView);  
 email = itemView.findViewById(R.id.*recyclerEmailId*);  
 }  
 }  
}

And in the MainActivity we get the data from the database and pass them to the recycler view adapter and then to the recycler view as shown in the code.

private RecyclerView recyclerView;  
 private RecyclerViewAdapter recyclerViewAdapter;  
 private UserInformationViewModel userInformationViewModel;  
   
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 recyclerView = findViewById(R.id.*recyclerViewId*);  
 recyclerView.setHasFixedSize(true);  
 recyclerView.setLayoutManager(new LinearLayoutManager(this));  
  
  
 userInformationViewModel = new ViewModelProvider.AndroidViewModelFactory(MainActivity.this  
 .getApplication()).create(UserInformationViewModel.class);  
  
 userInformationViewModel.getAllUsers().observe(this, userInformation -> {  
 // Set the adapter  
 recyclerViewAdapter = new RecyclerViewAdapter(userInformation,MainActivity.this);  
 recyclerView.setAdapter(recyclerViewAdapter);  
 });  
 }  
}

# **Defense 1**

Tasks:

* Add another table "cars"
* Display it
* It should have manufacturer and model fields.

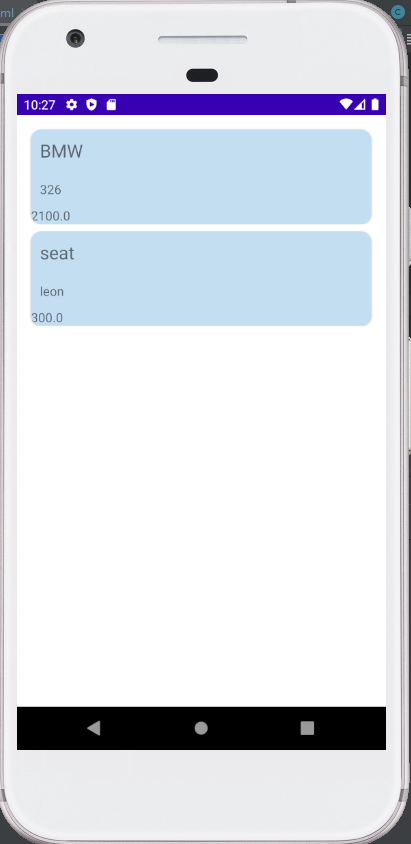


Figure 3 Application showing current cars for sale.

Code:

public class MainActivity extends AppCompatActivity {  
  
 private LiveData<List<UserInformation>> userList;  
 private RecyclerView recyclerView;  
 private RecyclerViewAdapter recyclerViewAdapter;  
 private UserInformationViewModel userInformationViewModel;  
 private CarInformationViewModel carInformationViewModel;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 getSupportActionBar().hide();  
 recyclerView = findViewById(R.id.*recyclerViewId*);  
 recyclerView.setHasFixedSize(true);  
 recyclerView.setLayoutManager(new LinearLayoutManager(this));  
  
 carInformationViewModel = new ViewModelProvider  
 .AndroidViewModelFactory(MainActivity.this.getApplication())  
 .create(CarInformationViewModel.class);  
  
 carInformationViewModel.getCars().observe(this, carInformation -> {  
 // Set the adapter  
 recyclerViewAdapter = new RecyclerViewAdapter(carInformation,MainActivity.this);  
 recyclerView.setAdapter(recyclerViewAdapter);  
 });  
 }  
}

# **Defense 2**

Task: Create an animation.

We decided to create a zoom out animation for the Add Post button.

XML animation code.

<?xml version="1.0" encoding="utf-8"?>  
<set xmlns:android="http://schemas.android.com/apk/res/android"  
 android:fillAfter="true">  
  
 <scale  
 android:duration="800"  
 android:fromXScale="1"  
 android:fromYScale="1"  
 android:pivotX="50%"  
 android:pivotY="50%"  
 android:toXScale="0"  
 android:toYScale="0"  
 />  
</set>

Java animation code.

addPost.setOnClickListener(view -> {  
 addPost.startAnimation((AnimationUtils.loadAnimation(getApplicationContext(), R.anim.close));  
 Intent addCar = new Intent(MainActivity.this,AddCarActivity.class);  
 startActivity(addCar);  
});

# **Reference list**