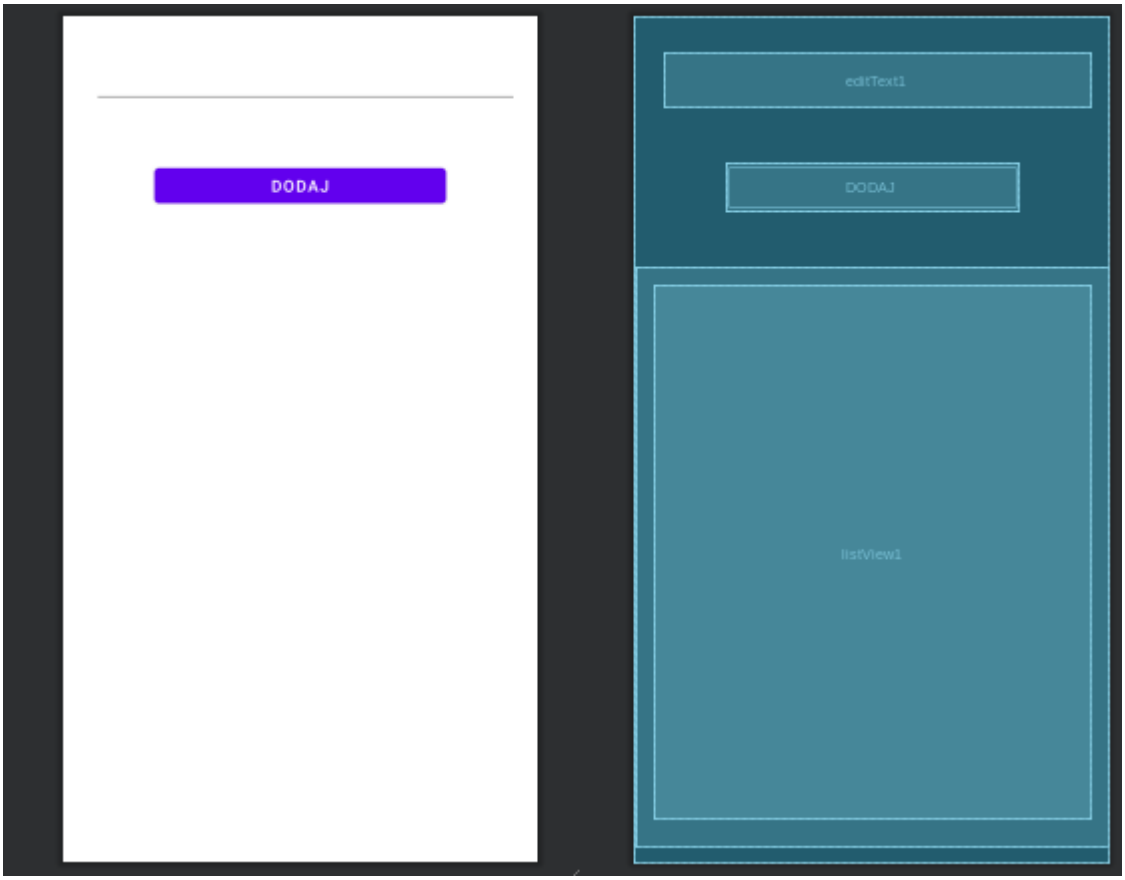
Laboratorijska vježba 2

Arhitektura Android aplikacije, resursi i napredni layout

Zadatak 1.

Omogućiti da se uneseni tekst klikom na dugme doda u listu koja će biti prikazana korisniku. Riješiti zadatak korištenjem Android adaptera, te korištenjem vlastitog adaptera.

Omogućimo prvo dodavanje elementa u listu korištenjem Android adaptera. U prvom koraku ćemo umjesto TextView-a u našoj aplikaciji dodati ListView s ID `listview1`.



Napomena: `ListView` je smješten u `FrameLayout` kako bi se spriječilo pomjeranje odgovarajućih elemenata uključivanjem tastature.

Kôd layout-a je dat u nastavku:

```
<?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">
    <EditText
        android:id="@+id/editText1"
        android:layout_width="368dp"
        android:layout_height="46dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="17dp"
        android:layout_marginBottom="16dp"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toTopOf="@+id/button1"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />
    <Button
        android:id="@+id/button1"
        android:layout_width="253dp"
        android:layout_height="42dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="16dp"
        android:layout_marginBottom="16dp"
        android:text="Dodaj"
        app:layout_constraintBottom_toTopOf="@+id/frameLayout"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/editText1" />
    <FrameLayout
        android:id="@+id/frameLayout"
        android:layout_width="409dp"
        android:layout_height="500dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="16dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/button1"
        app:layout_constraintVertical_bias="0.0">
        <ListView
            android:id="@+id/listView1"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_marginStart="16dp"
            android:layout_marginTop="16dp"
            android:layout_marginEnd="16dp"
            android:layout_marginBottom="24dp" />
```

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

Pređimo na odgovarajući kôd. Svi elementi layout-a će nam trebati. Iste ćemo definisati unutar klase s ključnom riječi `lateinit` zato što ih možemo inicijalizirati tek nakon što postavimo layout. Ova ključna riječ se najčešće koristi za `View` elemente koji se naknadno moraju inicijalizirati, te olakšava korištenje varijabli, odnosno ne mora se koristiti `??` ili `!!` potrebni za null vrijednosti. Pored ovog načina, postoje i drugi korištenjem ključne riječi `lazy` ili kreiranjem `get()` metode, te delegata. Međutim, ovi načini nisu prilagođeni radu s `Fragment` ili `Activity`. Pored `View` ćemo definisati adapter i listu vrijednosti pri čemu ćemo koristiti konstrukciju `arrayListOf` ili `mutableListOf`.

```
private lateinit var listView: ListView
private lateinit var editText: EditText
private lateinit var button: Button
// može se koristiti i mutableListOf
private val  listaVrijednosti = arrayListOf<String>()
private lateinit var adapter : ArrayAdapter<String>
```

Napomena: Razmislite zašto se koristi ovakva konstrukcija i koje su još moguće.

Incijalizirat ćemo odgovarajuće elemente i dodijeliti listi adapter, te listener dugmetu. Adapter koji koristimo je `ArrayAdapter` androida kojem proslijeđujemo context, layout koji želimo koristiti i listu vrijednosti. Layout koji proslijeđujemo je interni Android layout elementa liste.

```
button = findViewById<Button>(R.id.button1);
editText = findViewById<EditText>(R.id.editText1)
listView = findViewById<ListView>(R.id.listView1)
//Koristi se androidov layout
adapter = ArrayAdapter(this, android.R.layout.simple_list_item_1, listaVrijednosti)
listView.adapter=adapter
//Definisat ćemo akciju u slučaju klik akcije
button.setOnClickListener {
    addToList()
}
```

U sklopu funkcije `addToList` element se dodaje u listu, pošalje se obavijest da je došlo do promjene adapteru i obriše se uneseni tekst.

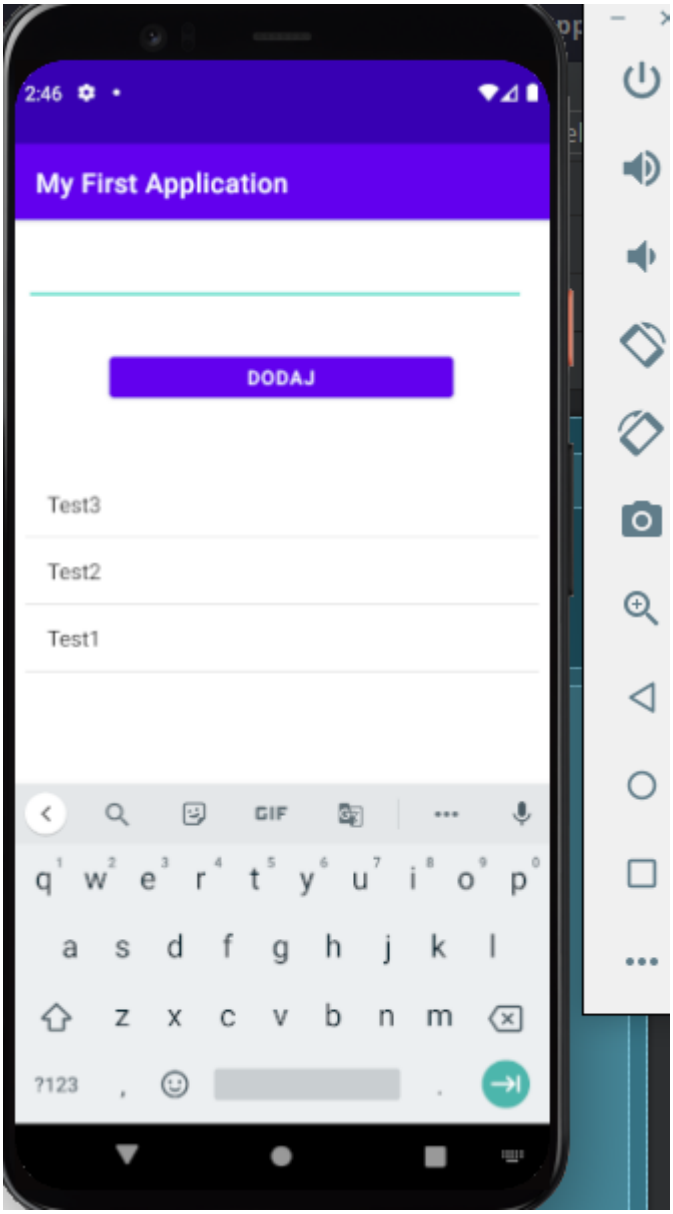
```
//Poziva se na klik dugmenta
 private fun addToList() {
    // Novi tekst se dodaje kao prvi element
    listaVrijednosti.add(0,editText.text.toString())
    adapter.notifyDataSetChanged();
    editText.setText("");
}
```

Cjelokupni kôd klase je dat u nastavku:

```
class MainActivity : AppCompatActivity() {
    private lateinit var listView: ListView
```

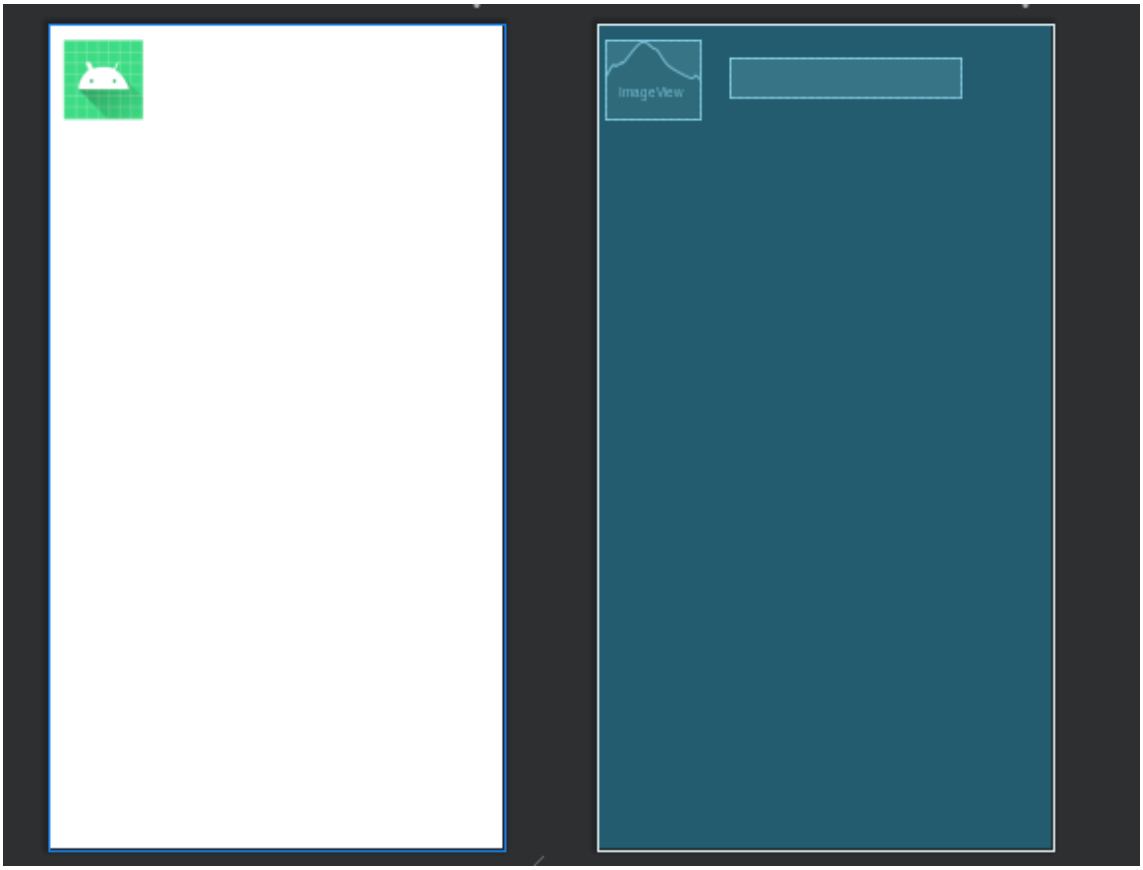
```
    private lateinit var editText: EditText
    private lateinit var button: Button
    // može se koristiti i mutableListOf
    private val  listaVrijednosti = arrayListOf<String>()
    private lateinit var adapter : ArrayAdapter<String>
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        //Inicijalizirat ćemo elemente
        button = findViewById<Button>(R.id.button1);
        editText = findViewById<EditText>(R.id.editText1)
        listView = findViewById<ListView>(R.id.listView1)
        //Koristi se androidov layout
        adapter = ArrayAdapter(this, android.R.layout.simple_list_item_1,
listaVrijednosti)
        listView.adapter=adapter
        //Definisat ćemo akciju u slučaju klik akcije
        button.setOnClickListener {
            addToList()
        }
    }
    //Poziva se na klik dugmenta
     private fun addToList() {
        // Novi tekst se dodaje kao prvi element
        listaVrijednosti.add(0,editText.text.toString())
        adapter.notifyDataSetChanged();
        editText.setText("");
    }
}
```

Rezultat nakon pokretanja i testiranja:


2:46
My First Application
DODAJ
Test3
Test2
Test1
GIF
?123

Definišimo sada izgled elementa liste, takav da pored svakog teksta se nalazi slika. Obzirom da imamo poseban layout, kreirat ćemo vlastiti `ArrayAdapter`.

U prvom koraku definišimo izgled našeg elementa liste. U odgovarajućem `res/layout` folder-u dodajmo novi layout kroz desni klik `New->XML->XML Layout File`. U layout ćemo dodati `ImageView` i postaviti neku default sliku, te ćemo dodati jedan `TextView` s ID `textElement`. Izgled layout-a `element_list.xml` je dat u nastavku.



Kod layout-a je sljedeći:

```
<?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="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="horizontal" >
    <ImageView
        android:id="@+id/icon"
        android:layout_width="102dp"
        android:layout_height="85dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@+id/textElement"
        app:layout_constraintHorizontal_bias="0.1"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.012"
        app:srcCompat="@mipmap/ic_launcher" />
    <TextView
```

```
        android:id="@+id/textElement"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="32dp"
        android:layout_marginTop="36dp"
        android:layout_marginEnd="98dp"
        android:paddingTop="10dp"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toEndOf="@+id/icon"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Definišimo sada klasu vlastitog `ArrayAdapter-a`. U nastavku je prikazan kôd iste kreiran praćenjem uputa datih u teoretskom uvodu.

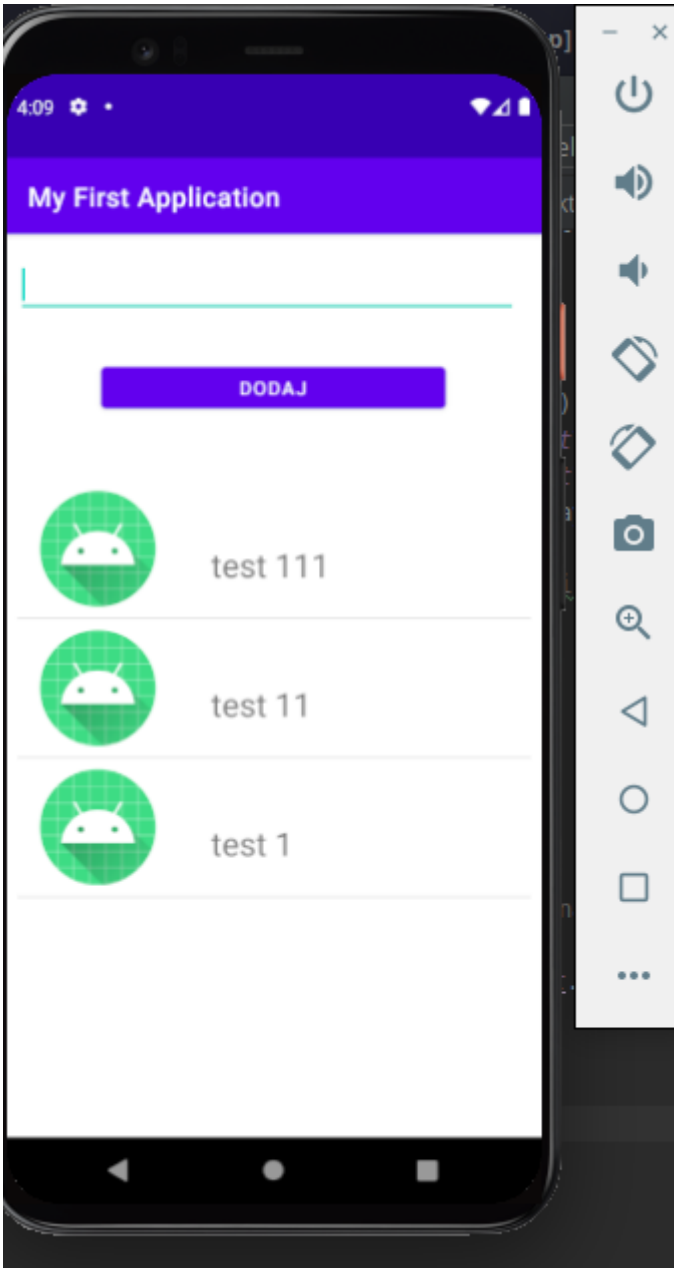
```
class MyArrayAdapter(context: Context, @LayoutRes private val layoutResource: Int,
private val elements: ArrayList<String>):
    ArrayAdapter<String>(context, layoutResource, elements) {
    override fun getView(position: Int, newView: View?, parent: ViewGroup): View {
        var newView = newView
        newView = LayoutInflater.from(context).inflate(R.layout.element_list, parent,
false)
        val textView = newView.findViewById<TextView>(R.id.textElement)
        val element = elements.get(position)
        textView.text=element
        return newView
    }
}
```

Izmjene nad glavnom klasom se odnose na definisanje i inicijalizaciju adaptera i date su u nastavku:

```
private lateinit var adapter : MyArrayAdapter


adapter = MyArrayAdapter(this, R.layout.element_list, listaVrijednosti)
```

Rezultat izvršenja i unosa par vrijednosti je dat u nastavku:

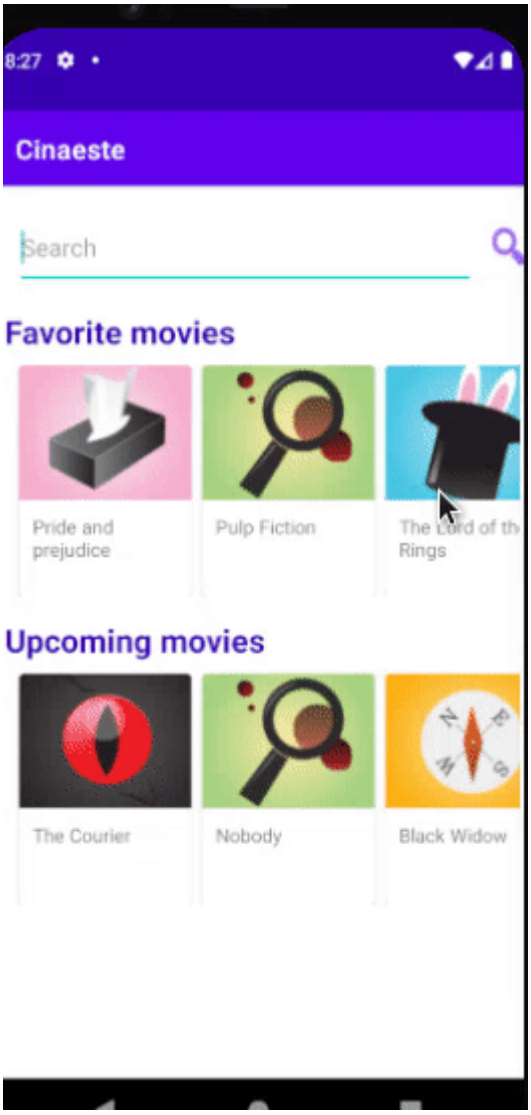

4:09
My First Application
DODAJ
test 111
test 11
test 1

Napomena: U praksi se sve više primjenjuje `RecyclerView` i odgovarajući adapter, te ćemo i mi primijenjivati isti. S ovim zadatkom zvanično završavamo sa pokaznom aplikacijom i prelazimo na zvaničnu aplikaciju koju kreiramo na vježbama. Za izradu ostalih zadataka i vježbi možete kreirati novu aplikaciju ili koristiti istu.

Movie App

Studenti u sklopu vježbi će implementirati `Cineaste` aplikaciju. Ova aplikacija prikazuje najnovije filmove, omiljene filmove, detalje o filmovima, omogućava pretragu filmova putem TMDB API, te dodavanje istih u favorite. U sklopu iste studenti će naučiti osnovne principe razvoja moderne Android aplikacije.

Aplikacija nakon ove vježbe će izgledati ovako:



Zadatak 1.

Kreirati data klasu Movie, koja sadrži naziv filma (title - string), žanr (genre- string), datum izdavanja (releaseDate- string), službenu web stranicu (homepage - string) i kratki tekst sadržaja (overview - string). Napraviti file MoviesStaticData u kojoj se nalazi funkcije koja vraćaju listu testnih podataka za omiljene filmove i najnovije filmove. Napisati Repository klasu i ViewModel klasu za potrebe dobavljanja podataka.

Kreiramo `Movie` klasu:

```
data class Movie(
    val id: Long,
    val title: String,
    val overview: String,
    val releaseDate: String,
    val homepage: String,
```

```
    val genre: String
)
```

Kreiramo `MovieStaticData` file s funkcijama za dobavljanje omiljenih filmova i najnovijih filmova:

```
fun favoriteMovies(): List<Movie> {
    return listOf(
         Movie(1,"Pride and prejudice",
            "Sparks fly when spirited Elizabeth Bennet meets single, rich, and proud Mr. Darcy. But Mr. Darcy reluctantly finds himself falling in love with a woman beneath his class. Can each overcome their own pride and prejudice?",
            "16.02.2005.","https://www.imdb.com/title/tt0414387/",
             "drama"),
       //Dodajte filmove po želji
    )
}
fun recentMovies(): List<Movie> {
    return listOf(
        Movie(1,"The Courier",
            "Cold War spy Greville Wynne and his Russian source try to put an end to the Cuban Missile Crisis.",
            "17.05.2021.","https://www.imdb.com/title/tt8368512/",
            "thriller")
            //Dodajte filmove po želji
    )
}
```

Kreiramo `MovieRepository` objekat:

```
object MovieRepository {
    fun getFavoriteMovies() : List<Movie> {
        return favoriteMovies();
    }
    fun getRecentMovies() : List<Movie> {
        return recentMovies();
    }
}
```

Napomena: Razmislite zašto smo kreirali objekat, a ne klasu.

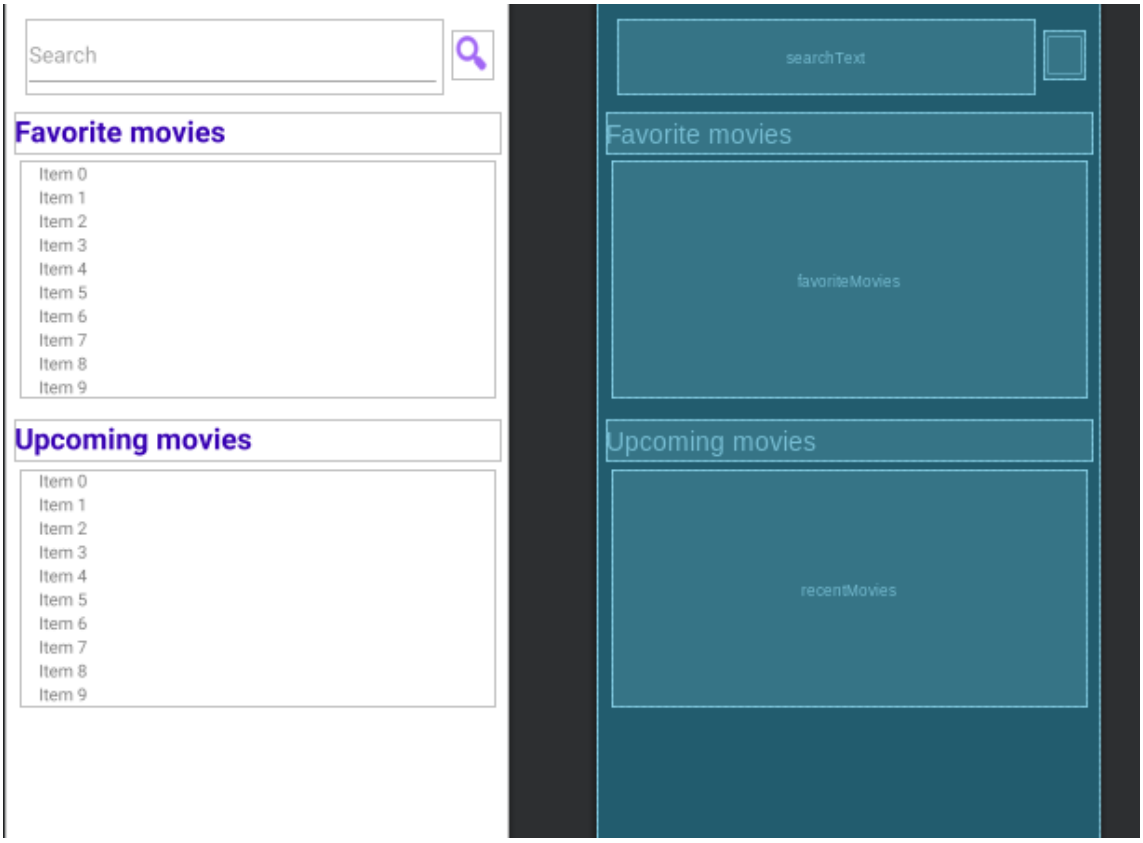
Kreiramo klasu `MovieListModel`:

```
class MovieListViewModel {
    fun getFavoriteMovies():List<Movie>{
        return MovieRepository.getFavoriteMovies();
    }
    fun getRecentMovies():List<Movie>{
        return MovieRepository.getRecentMovies();
    }
}
```

Zadatak 2.

Kreirati početni layout aplikacije u sklopu kojeg će postojati `EditText` i `Button` za pretragu filmova, te dvije horizontalne liste omiljenih filmova i najnovijih filmova.

Kreiramo početni layout aplikacije kao na slici:



XML kôd layout-a:

```
<?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">
    <EditText
        android:id="@+id/searchText"
        android:layout_width="341dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="4dp"
        android:hint="@string/search"
        app:layout_constraintEnd_toStartOf="@+id/searchButton"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
    <Button
        android:id="@+id/searchButton"
        android:layout_width="34dp"
        android:layout_height="39dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="25dp"
        android:background="@android:drawable/ic_menu_search"
        app:layout_constraintStart_toEndOf="@+id/searchText"
        app:layout_constraintTop_toTopOf="parent" />
    <TextView
        android:id="@+id/textView"
        android:layout_width="398dp"
        android:layout_height="33dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:text="@string/favorite"
        android:textColor="@color/purple_700"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/searchText" />
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/favoriteMovies"
        android:layout_width="388dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:clipToPadding="false"
        android:paddingStart="16dp"
        android:paddingEnd="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView" />
    <TextView
        android:id="@+id/textView1"
        android:layout_width="398dp"
        android:layout_height="33dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:text="@string/recent"
        android:textColor="@color/purple_700"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/favoriteMovies" />
```

```
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recentMovies"
        android:layout_width="388dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:clipToPadding="false"
        android:paddingStart="16dp"
        android:paddingEnd="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView1" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Zadatak 3.

Kreirati layout za listu omiljenih filmova, te `RecyclerViewAdapter` za istu.

Kreiran je layout za element liste koji će biti u obliku kartice. Sadržavat će sliku žanr-a filma koju je potrebno dodati u `res/drawable` folder (Kreirat ćemo više ikonica za različite žanrove filma) i naziv filma. Kôd layout-a je dat u nastavku:

```
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="128dp"
    android:layout_height="172dp"
    android:layout_marginEnd="8dp"
    app:cardCornerRadius="4dp">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <ImageView
            android:id="@+id/movieImage"
            android:layout_width="match_parent"
            android:layout_height="100dp"
            android:scaleType="centerCrop"
            app:srcCompat="@drawable/picture1" />
        <TextView
            android:id="@+id/movieTitle"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_gravity="center_horizontal"
            android:layout_margin="10dp"
            android:textSize="14sp" />
    </LinearLayout>
</androidx.cardview.widget.CardView>
```

Kreiran je odgovarajući `RecyclerViewAdapter` po uputama iz teoretskog uvoda. U sklopu istog se bind-aju naziv filma i slika žanr-a.

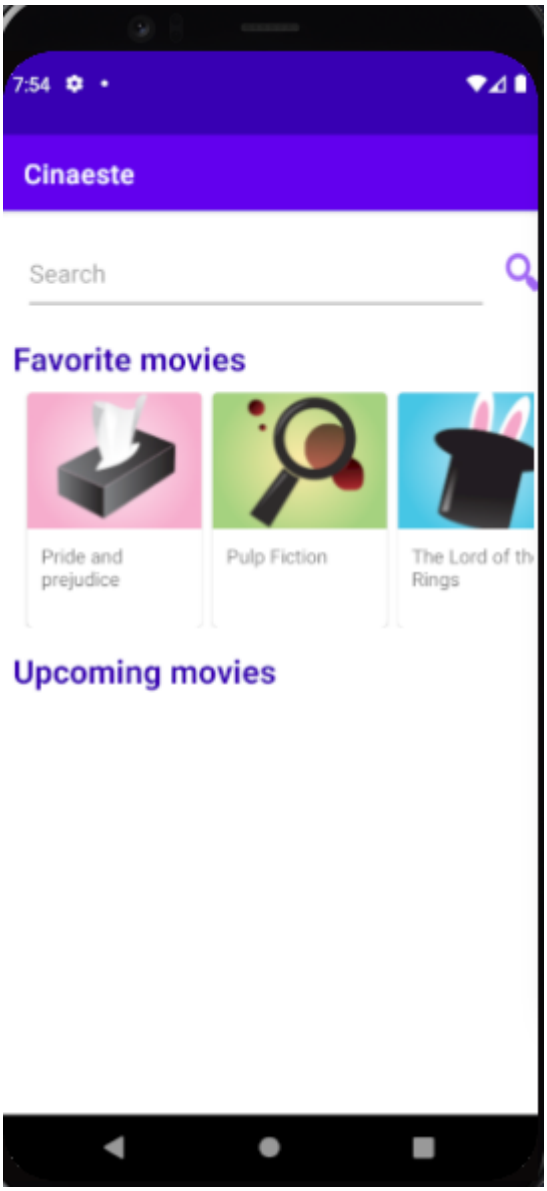
```
class MovieListAdapter(
    private var movies: List<Movie>
) : RecyclerView.Adapter<MovieListAdapter.MovieViewHolder>() {
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): MovieViewHolder
{
        val view = LayoutInflater
            .from(parent.context)
            .inflate(R.layout.item_movie, parent, false)
        return MovieViewHolder(view)
    }
    override fun getItemCount(): Int = movies.size
    override fun onBindViewHolder(holder: MovieViewHolder, position: Int) {
        holder.movieTitle.text = movies[position].title;
        val genreMatch: String = movies[position].genre
        //Pronalazimo id drawable elementa na osnovu naziva žanra
        val context: Context = holder.movieImage.getContext()
        var id: Int = context.getResources()
            .getIdentifier(genreMatch, "drawable", context.getPackageName())
        if (id===0) id=context.getResources()
            .getIdentifier("picture1", "drawable", context.getPackageName())
        holder.movieImage.setImageResource(id)
    }
    fun updateMovies(movies: List<Movie>) {
        this.movies = movies
        notifyDataSetChanged()
    }
    inner class MovieViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
        val movieImage: ImageView = itemView.findViewById(R.id.movieImage)
        val movieTitle: TextView = itemView.findViewById(R.id.movieTitle)
    }
}
```

U sklopu `MainActivity` klase ćemo definisati i inicijalizirati `RecyclerView` i `RecyclerViewAdapter` te ćemo dodijeliti filmove. `RecyclerView` ćemo dodijeliti `LinearLayoutManager` i to horizontalni.

```
class MainActivity : AppCompatActivity() {
    private lateinit var favoriteMovies: RecyclerView
    private lateinit var favoriteMoviesAdapter: MovieListAdapter
    private var movieListViewModel =  MovieListViewModel()
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        favoriteMovies = findViewById(R.id.favoriteMovies)
        favoriteMovies.layoutManager = LinearLayoutManager(
            this,
            LinearLayoutManager.HORIZONTAL,
            false
        )
        favoriteMoviesAdapter = MovieListAdapter(listOf())
        favoriteMovies.adapter = favoriteMoviesAdapter
        favoriteMoviesAdapter.updateMovies(movieListViewModel.getFavoriteMovies())
```

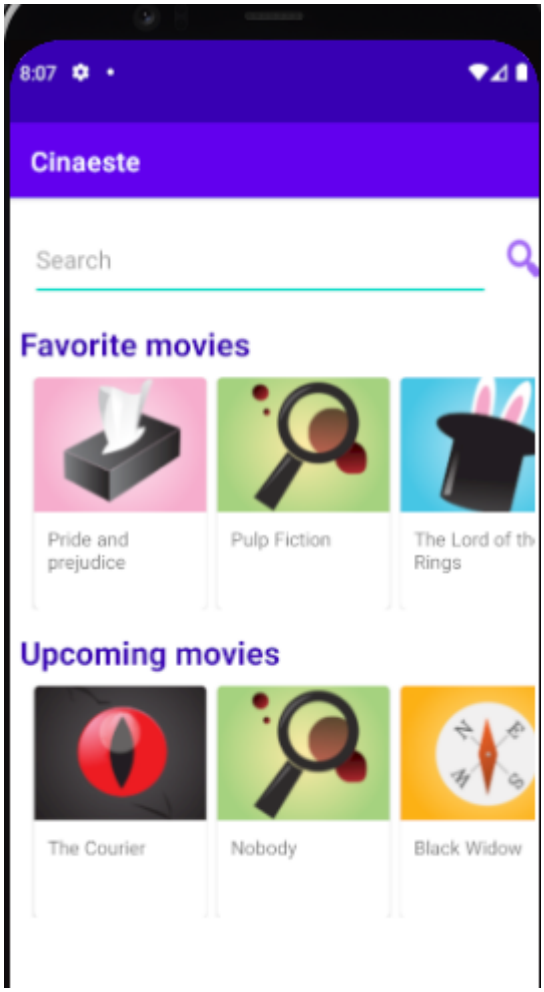
```
    }
}
```

Rezultat izvršavanja je dat u nastavku:



ZSR

- Kreirajte potrebne klase za listu popularnih filmova.
- Izdvojite sve labele kao stringove u resursima i napravite prevod svih stringova i na engleski i na bosanski jezik.
- Organizujte sve slike koje se koriste u aplikaciji tako da su podržani uređaji sa svim gustinama prikaza (Tip: pogledajte stranicu)



Rezultantna aplikacija je data na slici: