НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №4

з дисципліни

“Програмування мобільних систем”

Виконав:

студент групи ІО-81

ЗК ІО-8118

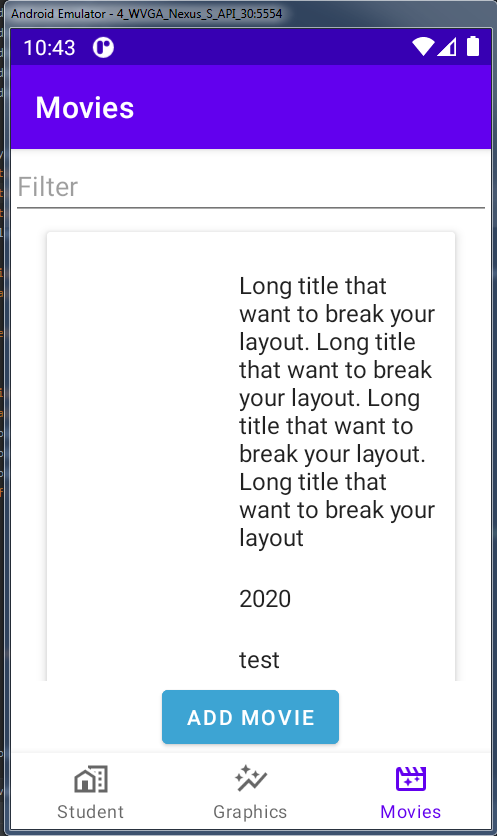
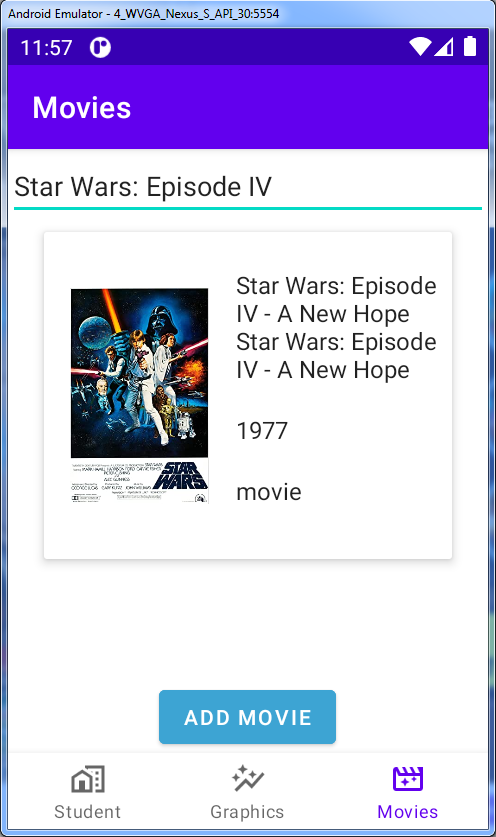
Лялін Ігор

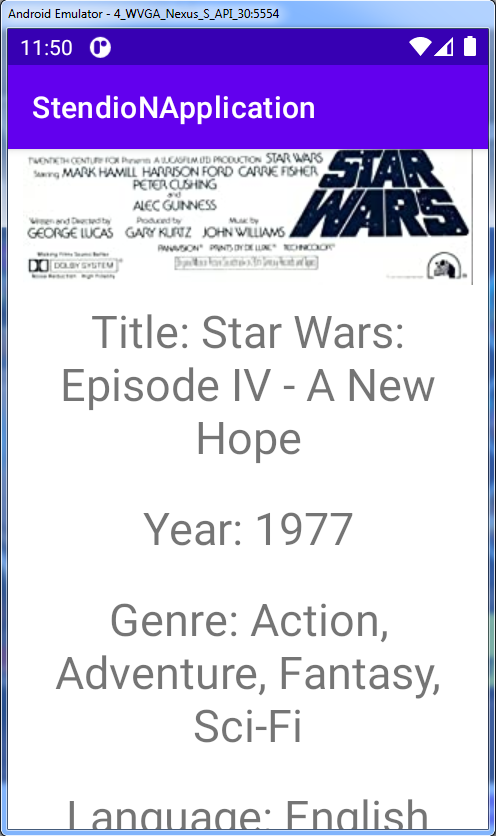
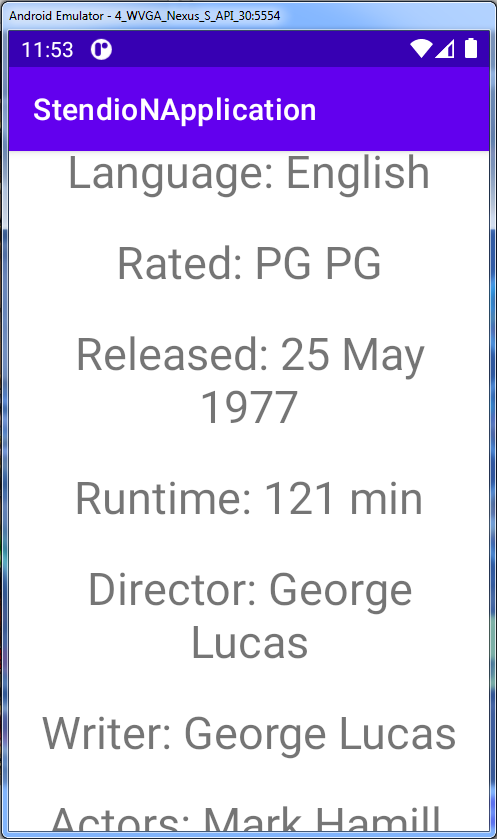
Київ 2021

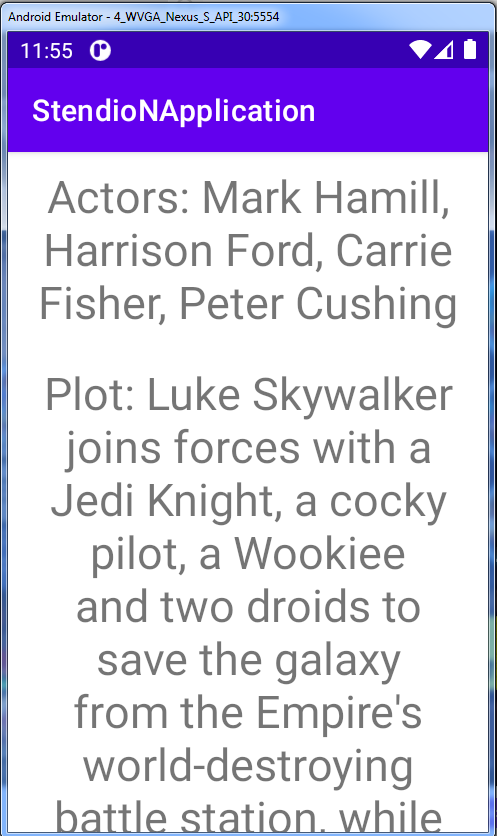
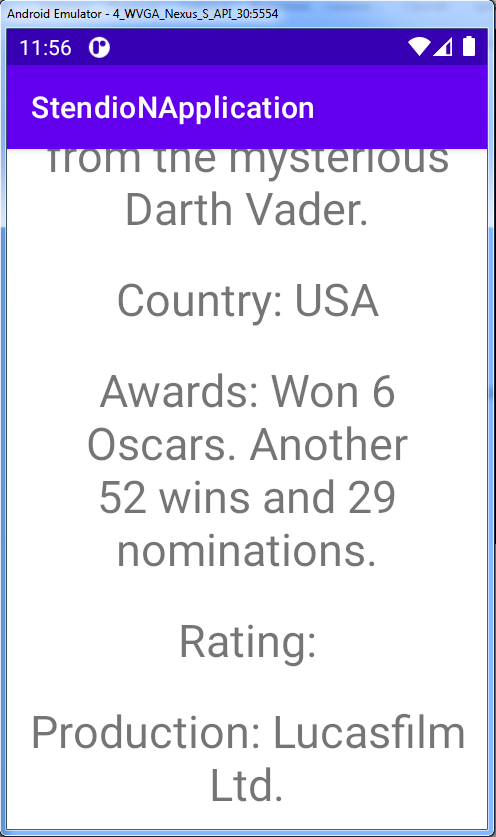
**(8118%6)+1 = Варіант 1**

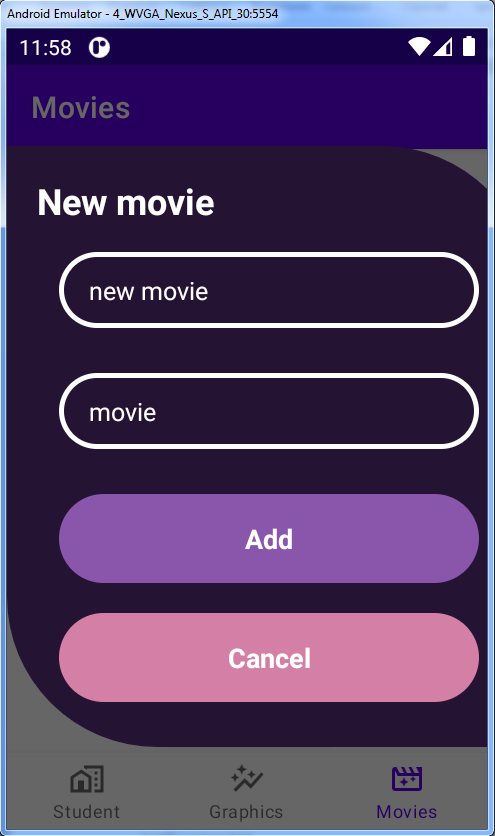
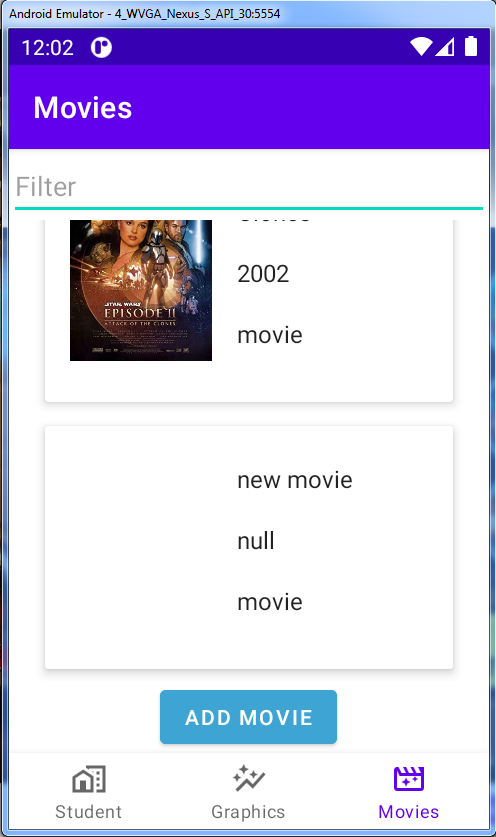


**Скріншоти роботи програми**









**Лістинг коду**

RecyclerViewAdapter.kt

package ua.kpi.comsys.io8118.ui.movieList  
  
import ua.kpi.comsys.io8118.R  
import android.content.Context  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.ImageView  
import android.widget.TextView  
import androidx.recyclerview.widget.RecyclerView  
  
  
class RecyclerViewAdapter(  
 private val context: Context,  
 private var values: List<Search>,  
 private val listener: (Search) -> Unit  
) : RecyclerView.Adapter<RecyclerViewAdapter.ViewHolder>() {  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {  
 val view = LayoutInflater.from(parent.*context*)  
 .inflate(R.layout.*fragment\_movie*, parent, false)  
 return ViewHolder(view)  
 }  
  
 override fun onBindViewHolder(holder: ViewHolder, position: Int) {  
 val item = values[position]  
 holder.title.*text* = item.title + "\n"  
 holder.year.*text* = item.year + "\n"  
 holder.type.*text* = item.type + "\n"  
 if (item.poster != null && item.poster!!.*isNotEmpty*()) {  
 val drawable = context.*resources*.getIdentifier(  
 item.poster!!.*replace*(".jpg", ""), "drawable",  
 context.*packageName* )  
 holder.poster.setImageDrawable(  
 context.*resources*.getDrawable(drawable)  
 )  
 } else {  
 holder.poster.setImageDrawable(  
 context.*resources*.getDrawable(android.R.color.*transparent*)  
 )  
 }  
 holder.itemView.setOnClickListener **{** listener(item) **}** }  
  
 override fun getItemCount(): Int = values.size  
  
 inner class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {  
  
 val title: TextView = view.findViewById(R.id.*title*)  
 val year: TextView = view.findViewById(R.id.*year*)  
 val type: TextView = view.findViewById(R.id.*type*)  
 val poster: ImageView = view.findViewById(R.id.*image*)  
 }  
  
 fun removeItem(position: Int) {  
 values.*toMutableList*().removeAt(position)  
 notifyItemRemoved(position)  
 }  
  
 fun restoreItem(item: Search, position: Int) {  
 values.*toMutableList*().add(position, item)  
 notifyItemInserted(position)  
 }  
  
 fun getData(): List<Search> {  
 return values  
 }  
  
 fun updateList(list: List<Search>) {  
 values = list  
 notifyDataSetChanged()  
 }  
}

ListFragment.kt

package ua.kpi.comsys.io8118.ui.movieList  
  
import ua.kpi.comsys.io8118.R  
import android.content.Intent  
import android.graphics.Color  
import android.os.Bundle  
import android.text.Editable  
import android.text.TextWatcher  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.Button  
import android.widget.EditText  
import android.widget.LinearLayout  
import androidx.coordinatorlayout.widget.CoordinatorLayout  
import androidx.fragment.app.Fragment  
import androidx.recyclerview.widget.GridLayoutManager  
import androidx.recyclerview.widget.ItemTouchHelper  
import androidx.recyclerview.widget.LinearLayoutManager  
import androidx.recyclerview.widget.RecyclerView  
import com.example.flatdialoglibrary.dialog.FlatDialog  
import com.fasterxml.jackson.databind.ObjectMapper  
import com.google.android.material.snackbar.Snackbar  
import java.lang.reflect.Field  
  
  
class ListFragment : Fragment() {  
  
 private var columnCount = 1  
 private var searchContainer: SearchContainer = SearchContainer()  
 private lateinit var view: LinearLayout  
 private lateinit var adapter: RecyclerViewAdapter  
 private lateinit var searchField: EditText  
 private lateinit var addButton: Button  
  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View {  
  
 view = inflater.inflate(R.layout.*fragment\_movie\_list*, container, false) as LinearLayout  
 return view  
 }  
  
 override fun onActivityCreated(savedInstanceState: Bundle?) {  
 super.onActivityCreated(savedInstanceState)  
  
 *with*(view.getChildAt(1) as RecyclerView) **{** *layoutManager* = when {  
 columnCount <= 1 -> LinearLayoutManager(*context*)  
 else -> GridLayoutManager(*context*, columnCount)  
 }  
 **}** initFilms()  
 initSearchField()  
 val arr = searchContainer.search  
 adapter = RecyclerViewAdapter(requireContext(), arr)**{** item **->** if (item.imdbID != null && item.imdbID!!.*isNotEmpty*()) {  
 val id = getResId(item.imdbID!!, R.string::class.*java*)  
 val jsonText = requireContext().*resources*.getString(id)  
 val intent = Intent(requireContext(), ActivityDetails::class.*java*)  
 intent.putExtra("info", jsonText)  
 startActivity(intent)  
 }  
 **}** view.findViewById<RecyclerView>(R.id.*list*).*adapter* = adapter  
 enableSwipeToDeleteAndUndo()  
 addButton = requireView().findViewById(R.id.*add*)  
 addButton.setOnClickListener **{** val flatDialog = FlatDialog(requireContext())  
 flatDialog.setTitle("New movie")  
 .setFirstTextFieldHint("Title")  
 .setSecondTextFieldHint("Type")  
 .setFirstButtonText("Add")  
 .setSecondButtonText("Cancel")  
 .withFirstButtonListner **{** val search = Search()  
 search.title = flatDialog.*firstTextField* search.type = flatDialog.*secondTextField* searchContainer.search.add(search)  
 flatDialog.dismiss()  
 **}** .withSecondButtonListner **{** flatDialog.dismiss()  
 **}** .show()  
 **}** }  
  
 private fun initFilms() {  
 val mapper = ObjectMapper()  
 val jsonText = requireView().*resources*.getString(R.string.*json\_movies*)  
 searchContainer = mapper.readValue(jsonText, SearchContainer::class.*java*)  
 }  
  
 private fun initSearchField() {  
 searchField = requireView().findViewById(R.id.*filter*)  
 searchField.addTextChangedListener(object : TextWatcher {  
 override fun onTextChanged(s: CharSequence, start: Int, before: Int, count: Int) {}  
  
 override fun beforeTextChanged(s: CharSequence, start: Int, count: Int, after: Int) {}  
  
 override fun afterTextChanged(s: Editable) {  
 filter(s.toString())  
 }  
 })  
 }  
  
 fun filter(text: String) {  
 val temp: MutableList<Search> = ArrayList()  
 for (search in searchContainer.search) {  
 if (search.title!!.*contains*(text)) {  
 temp.add(search)  
 }  
 }  
 adapter.updateList(temp)  
 }  
  
 private fun displaySnackBarWithBottomMargin(  
 snackbar: Snackbar,  
 sideMargin: Int,  
 marginBottom: Int  
 ) {  
 val snackBarView = snackbar.*view* val params = snackBarView.*layoutParams* as CoordinatorLayout.LayoutParams  
 params.setMargins(  
 params.leftMargin + sideMargin,  
 params.topMargin,  
 params.rightMargin + sideMargin,  
 params.bottomMargin + marginBottom  
 )  
 snackBarView.*layoutParams* = params  
 snackbar.show()  
 }  
  
 private fun enableSwipeToDeleteAndUndo() {  
 val swipeToDeleteCallback: MovieDeletion = object : MovieDeletion(  
 requireContext()  
 ) {  
 override fun onSwiped(viewHolder: RecyclerView.ViewHolder, i: Int) {  
 try {  
 val position = viewHolder.*adapterPosition* val item: Search = adapter.getData().get(position)  
 val suppData: Search = searchContainer.search[position]  
 adapter.removeItem(position)  
 searchContainer.search.removeAt(position)  
 val snackbar: Snackbar = Snackbar  
 .make(  
 view,  
 "Movie was deleted!",  
 Snackbar.*LENGTH\_LONG* )  
 snackbar.setAction("Undo", View.OnClickListener **{** adapter.restoreItem(item, position)  
 searchContainer.search.add(position, suppData)  
 view.findViewById<RecyclerView>(R.id.*list*).scrollToPosition(position)  
 **}**)  
 snackbar.setActionTextColor(Color.*WHITE*)  
 displaySnackBarWithBottomMargin(snackbar, 40, 40)  
 } catch (ignored: Exception) {  
 }  
 }  
 }  
 val itemTouchhelper = ItemTouchHelper(swipeToDeleteCallback)  
 itemTouchhelper.attachToRecyclerView(view.findViewById<RecyclerView>(R.id.*list*))  
 }  
  
 fun getResId(resName: String, c: Class<\*>): Int {  
 return try {  
 val idField: Field = c.getDeclaredField(resName)  
 idField.getInt(idField)  
 } catch (e: java.lang.Exception) {  
 e.printStackTrace()  
 -1  
 }  
 }  
}

ActivityDetails.kt

package ua.kpi.comsys.io8118.ui.movieList  
  
import ua.kpi.comsys.io8118.R  
import androidx.appcompat.app.AppCompatActivity  
import android.os.Bundle  
import android.widget.ImageView  
import android.widget.TextView  
import com.fasterxml.jackson.databind.ObjectMapper  
import java.lang.reflect.Field  
  
class ActivityDetails : AppCompatActivity() {  
  
 private lateinit var info: MovieDetails  
 private lateinit var poster: ImageView  
 private lateinit var year: TextView  
 private lateinit var genre: TextView  
 private lateinit var rated: TextView  
 private lateinit var released: TextView  
 private lateinit var runtime: TextView  
 private lateinit var director: TextView  
 private lateinit var actors: TextView  
 private lateinit var plot: TextView  
 private lateinit var country: TextView  
 private lateinit var awords: TextView  
 private lateinit var rating: TextView  
 private lateinit var prodaction: TextView  
 private lateinit var title: TextView  
 private lateinit var language: TextView  
 private lateinit var writer: TextView  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_detail*)  
 init()  
 }  
  
 fun init() {  
 val mapper = ObjectMapper()  
 val jsonText = *intent*.*extras*!!["info"].*toString*()  
 info = mapper.readValue(jsonText, MovieDetails::class.*java*)  
 poster = findViewById(R.id.*poster*)  
 val drawable = getResId(info.poster!!.*replace*(".jpg", ""), R.drawable::class.*java*)  
 if (drawable != -1)  
 poster.setImageDrawable(this.*resources*.getDrawable(drawable))  
  
 year = findViewById(R.id.*year*)  
 year.*text* = year.*text*.toString() + " " + info.year  
 genre = findViewById(R.id.*genre*)  
 genre.*text* = genre.*text*.toString() + " " + info.genre  
 rated = findViewById(R.id.*rated*)  
 rated.*text* = rated.*text*.toString() + " " + info.rated  
 released = findViewById(R.id.*released*)  
 released.*text* = released.*text*.toString() + " " + info.released  
 runtime = findViewById(R.id.*runtime*)  
 runtime.*text* = runtime.*text*.toString() + " " + info.runtime  
 director = findViewById(R.id.*director*)  
 director.*text* = director.*text*.toString() + " " + info.director  
 actors = findViewById(R.id.*actors*)  
 actors.*text* = actors.*text*.toString() + " " + info.actors  
 plot = findViewById(R.id.*plot*)  
 plot.*text* = plot.*text*.toString() + " " + info.plot  
 country = findViewById(R.id.*country*)  
 country.*text* = country.*text*.toString() + " " + info.country  
 awords = findViewById(R.id.*awards*)  
 awords.*text* = awords.*text*.toString() + " " + info.awards  
 rating = findViewById(R.id.*rated*)  
 rating.*text* = rating.*text*.toString() + " " + info.rated  
 prodaction = findViewById(R.id.*production*)  
 prodaction.*text* = prodaction.*text*.toString() + " " + info.production  
 title = findViewById(R.id.*title*)  
 title.*text* = title.*text*.toString() + " " + info.title  
 language = findViewById(R.id.*language*)  
 language.*text* = language.*text*.toString() + " " + info.language  
 writer = findViewById(R.id.*write*)  
 writer.*text* = writer.*text*.toString() + " " + info.writer  
 }  
  
 fun getResId(resName: String, c: Class<\*>): Int {  
 return try {  
 val idField: Field = c.getDeclaredField(resName)  
 idField.getInt(idField)  
 } catch (e: java.lang.Exception) {  
 e.printStackTrace()  
 -1  
 }  
 }  
}

MovieDeletion.kt

package ua.kpi.comsys.io8118.ui.movieList  
  
import ua.kpi.comsys.io8118.R  
import android.content.Context  
import android.graphics.\*  
import android.graphics.drawable.ColorDrawable  
import android.graphics.drawable.Drawable  
import androidx.core.content.ContextCompat  
import androidx.recyclerview.widget.ItemTouchHelper  
import androidx.recyclerview.widget.RecyclerView  
  
internal abstract class MovieDeletion(private val mContext: Context) :  
 ItemTouchHelper.Callback() {  
 private val mClearPaint: Paint  
 private val mBackground: ColorDrawable  
 private val backgroundColor: Int  
 private val deleteDrawable: Drawable?  
 private val intrinsicWidth: Int  
 private val intrinsicHeight: Int  
 override fun getMovementFlags(  
 recyclerView: RecyclerView,  
 viewHolder: RecyclerView.ViewHolder  
 ): Int {  
 return makeMovementFlags(0, ItemTouchHelper.*LEFT*)  
 }  
  
 override fun onMove(  
 recyclerView: RecyclerView,  
 viewHolder: RecyclerView.ViewHolder,  
 viewHolder1: RecyclerView.ViewHolder  
 ): Boolean {  
 return false  
 }  
  
 override fun onChildDraw(  
 c: Canvas,  
 recyclerView: RecyclerView,  
 viewHolder: RecyclerView.ViewHolder,  
 dX: Float,  
 dY: Float,  
 actionState: Int,  
 isCurrentlyActive: Boolean  
 ) {  
 super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState, isCurrentlyActive)  
 val itemView = viewHolder.itemView  
 val itemHeight = itemView.*height* val isCancelled = dX == 0f && !isCurrentlyActive  
 if (isCancelled) {  
 clearCanvas(  
 c, itemView.*right* + dX,  
 itemView.*top*.toFloat(), itemView.*right*.toFloat(), itemView.*bottom*.toFloat()  
 )  
 super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState, isCurrentlyActive)  
 return  
 }  
 mBackground.*color* = backgroundColor  
 mBackground.setBounds(  
 itemView.*right* + dX.toInt(),  
 itemView.*top*,  
 itemView.*right*,  
 itemView.*bottom* )  
 mBackground.draw(c)  
 val deleteIconTop = itemView.*top* + (itemHeight - intrinsicHeight) / 2  
 val deleteIconMargin = (itemHeight - intrinsicHeight) / 2  
 val deleteIconLeft = itemView.*right* - deleteIconMargin - intrinsicWidth  
 val deleteIconRight = itemView.*right* - deleteIconMargin  
 val deleteIconBottom = deleteIconTop + intrinsicHeight  
 deleteDrawable!!.setBounds(deleteIconLeft, deleteIconTop, deleteIconRight, deleteIconBottom)  
 deleteDrawable.draw(c)  
 super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState, isCurrentlyActive)  
 }  
  
 private fun clearCanvas(c: Canvas, left: Float, top: Float, right: Float, bottom: Float) {  
 c.drawRect(left, top, right, bottom, mClearPaint)  
 }  
  
 init {  
 mBackground = ColorDrawable()  
 backgroundColor = Color.parseColor("#ff0000")  
 mClearPaint = Paint()  
 mClearPaint.*xfermode* = PorterDuffXfermode(PorterDuff.Mode.CLEAR)  
 deleteDrawable = ContextCompat.getDrawable(mContext, R.drawable.*delete*)  
 intrinsicWidth = deleteDrawable!!.*intrinsicWidth* intrinsicHeight = deleteDrawable.*intrinsicHeight* }  
}

**Висновок**

Під час виконання роботи було створено новий функціонал для вкладки зі списком фільмів та їх постерів, а саме функція додавання нового фільму, функція видалення фільмів, пошук, а також додаткову інформацію про існуючі фільми. Мету роботи досягнуто.