Aplikacje mobilne – sprawozdanie

1. Autorzy
    - Andrzej Kapczyński 145358
    - Jędrzej Smok 145286

2. Laboratorium 8 – Aplikacja typu lista szczegóły

Kod MainActivity.kt

```kotlin
package com.example.listdetail

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import androidx.fragment.app.add
import androidx.fragment.app.commit

class MainActivity : AppCompatActivity(R.layout.activity_main) {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        if (savedInstanceState == null) {
            supportFragmentManager.commit {
                setReorderingAllowed(true)
                add<ListFragment>(R.id.fragment_container)
            }
        }
    }

    fun goBack(view: View){
        val fragmentManager = supportFragmentManager
        val fragmentTransaction = fragmentManager.beginTransaction()
        fragmentTransaction.replace(R.id.fragment_container, ListFragment())
        fragmentTransaction.commit()
    }
}
```

Kod ListFragment.kt

```kotlin
package com.example.listdetail

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ArrayAdapter
import android.widget.ListView
import android.widget.Toast
import androidx.fragment.app.Fragment

import android.util.Log
import android.widget.TextView

class ListFragment : Fragment(R.layout.fragment_list) {
    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return LayoutInflater.from(container?.context).inflate(R.layout.fragment_list, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        val lv = view.findViewById<ListView>(R.id.listview)
        val routes = arrayOf(
            "Droga 1", "Droga 2", "Droga 3", "Droga 4", "Droga 5",
            "Droga 6", "Droga 7", "Droga 8", "Droga 9", "Droga 10",
            "Droga 11", "Droga 12", "Droga 13", "Droga 14", "Droga 15",
            "Droga 16", "Droga 17", "Droga 18", "Droga 19", "Droga 20",
        )
        val descriptions = mapOf(
            "Droga 1" to "Opis drogi 1", "Droga 2" to "Opis drogi 2", "Droga 3" to "Opis drogi 3",
            "Droga 4" to "Opis drogi 4", "Droga 5" to "Opis drogi 5", "Droga 6" to "Opis drogi 6",
            "Droga 7" to "Opis drogi 7", "Droga 8" to "Opis drogi 8", "Droga 9" to "Opis drogi 9",
            "Droga 10" to "Opis drogi 10", "Droga 11" to "Opis drogi 11", "Droga 12" to "Opis drogi 12",
            "Droga 13" to "Opis drogi 13", "Droga 14" to "Opis drogi 14", "Droga 15" to "Opis drogi 15",
            "Droga 16" to "Opis drogi 16", "Droga 17" to "Opis drogi 17", "Droga 18" to "Opis drogi 18",
            "Droga 19" to "Opis drogi 19", "Droga 20" to "Opis drogi 20",
        )


        val adapter = ArrayAdapter(view.context, android.R.layout.simple_list_item_1, routes)
        lv.adapter = adapter

        lv.setOnItemClickListener { parent, view, position, id ->
            val route = adapter.getItem(position)
            val description = descriptions[route]
            replaceFragment(DetailFragment.newInstance(route.toString(), description.toString()))
        }
    }

    private fun replaceFragment(fragment: Fragment) {
        val smallestWidth = resources.configuration.smallestScreenWidthDp
        val fragmentManager = requireActivity().supportFragmentManager
        val fragmentTransaction = fragmentManager.beginTransaction()

        if (smallestWidth < 720)
            fragmentTransaction.replace(R.id.fragment_container, fragment)
        else
            fragmentTransaction.replace(R.id.fragment_container2, fragment)

        fragmentTransaction.commit()
    }
}
```

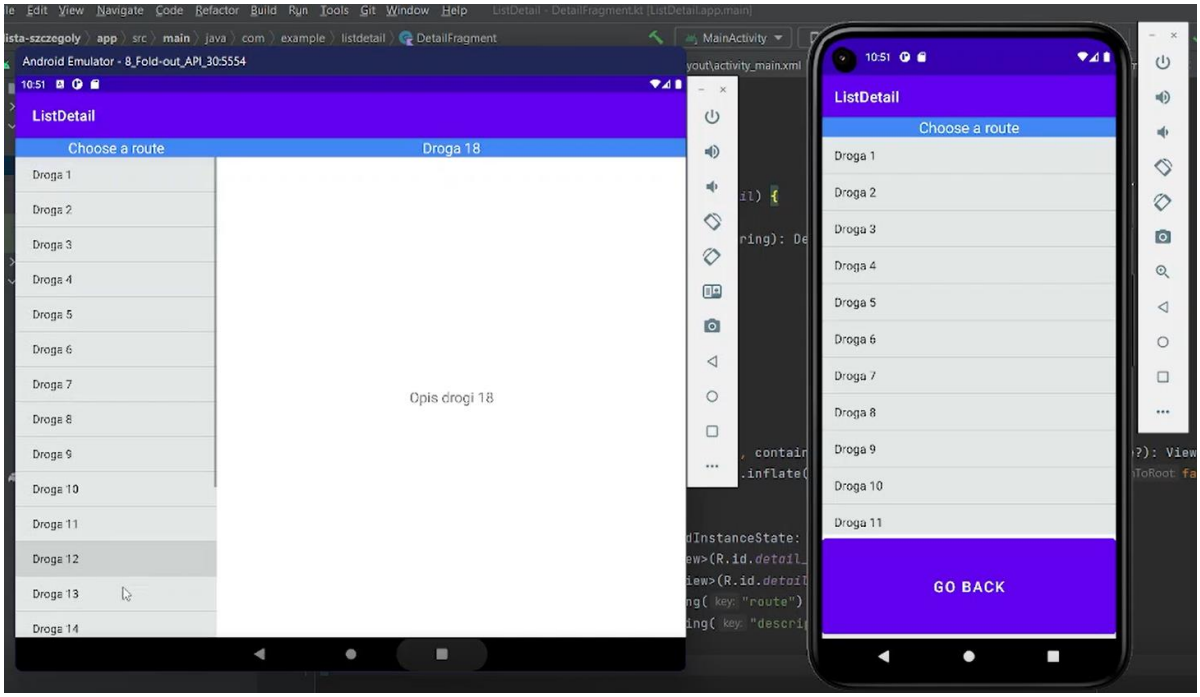
Kod DetailFragment.kt

```kotlin
package com.example.listdetail

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.fragment.app.Fragment


class DetailFragment : Fragment(R.layout.fragment_detail) {
    companion object {
        fun newInstance(route: String, description: String): DetailFragment {
            val fragment = DetailFragment()
            val args = Bundle()
            args.putString("route", route)
            args.putString("description", description)
            fragment.arguments = args
            return fragment
        }
    }

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return LayoutInflater.from(container?.context).inflate(R.layout.fragment_detail, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        val textView = view.findViewById<TextView>(R.id.detail_textview)
        val textView2 = view.findViewById<TextView>(R.id.detail_textview2)
        textView.text = this.arguments?.getString("route")
        textView2.text = this.arguments?.getString("description")
    }
}
```

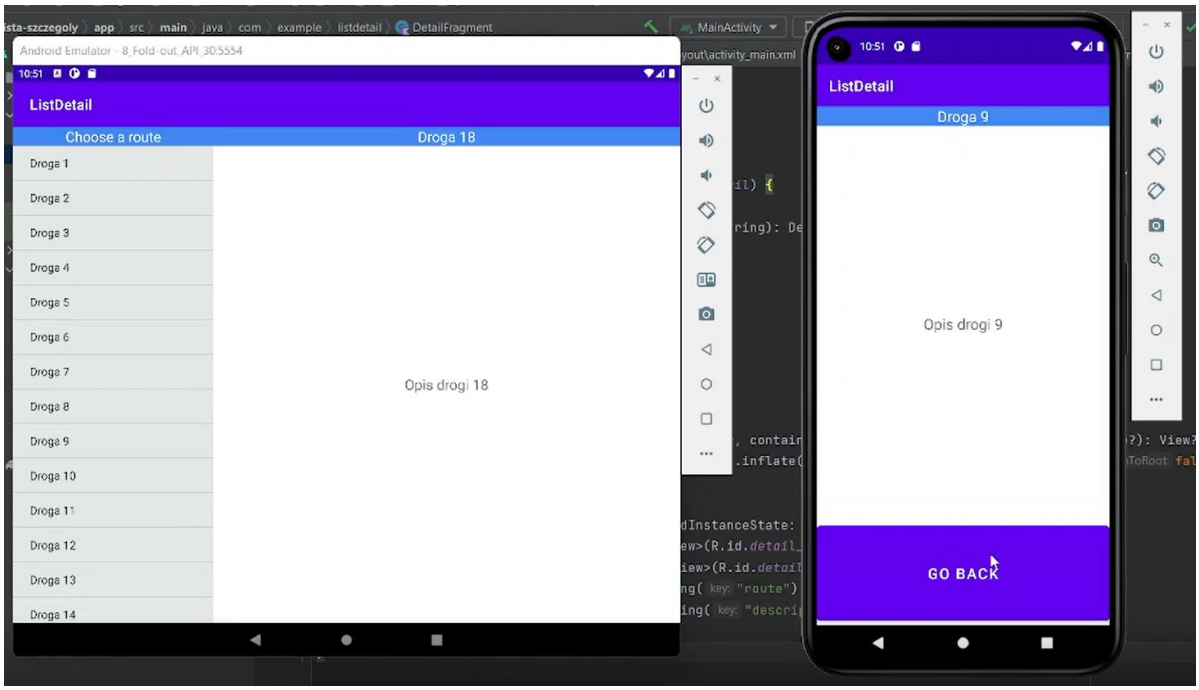
Aplikacja składa się z jednej aktywności i dwóch fragmentów – pierwszego odpowiadającego za wyświetlenie listy tras oraz drugiego odpowiadającego za wyświetlenie szczegółów trasy.

Zrzuty ekranów:

Lista:



Szczegóły:



3. Laboratorium - Aplikacja typu lista-szczegóły z fragmentem dynamicznym

Kod DetailFragment.kt

```kotlin
package com.example.listdetail

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.fragment.app.Fragment
import androidx.fragment.app.add
import androidx.fragment.app.commit


class DetailFragment : Fragment(R.layout.fragment_detail) {
    companion object {
        fun newInstance(route: String, description: String): DetailFragment {
            val fragment = DetailFragment()
            val args = Bundle()
            args.putString("route", route)
            args.putString("description", description)
            fragment.arguments = args
            return fragment
        }
    }

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return LayoutInflater.from(container?.context).inflate(R.layout.fragment_detail, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        val textView = view.findViewById<TextView>(R.id.detail_textview)
        val textView2 = view.findViewById<TextView>(R.id.detail_textview2)
        val textView3 = view.findViewById<TextView>(R.id.detail_textview3)
        val textView4 = view.findViewById<TextView>(R.id.detail_textview4)
        textView.text = this.arguments?.getString("route")
        textView2.text = this.arguments?.getString("description")

        val sharedTime = requireActivity().getSharedPreferences("com.example.listdetail.shared",0)
        textView3.text = "Best time: ${sharedTime.getString(textView.text.toString(), "None")}"
        textView4.text = "Last time: ${sharedTime.getString("${textView.text.toString()} last", "None")}"

        if (savedInstanceState == null) {
            parentFragmentManager.commit {
                setReorderingAllowed(true)
                add<StopWatchFragment>(
                    R.id.fragment_container3,
                    "SS",
                    Bundle().apply { putString("route", textView.text.toString()) }
                )
            }
        }

    }
}
```

Dodanie fragmentu odpowiadającego za wyświetlanie stopera.

Kod StopWatchFragment.kt

```kotlin
class StopWatchFragment : Fragment(R.layout.fragment_stop_watch) {
    private var timerStarted = false
    private lateinit var serviceIntent: Intent
    private var time = 0.0
    private lateinit var timeTv: TextView
    private lateinit var startStopButton: MaterialButton
    private lateinit var resetButton: MaterialButton
    private lateinit var saveButton: MaterialButton

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        startStopButton = view.findViewById(R.id.startStopButton)
        resetButton = view.findViewById(R.id.resetButton)
        saveButton = view.findViewById(R.id.saveButton)
        timeTv = view.findViewById(R.id.timeTV)

        // If screen was rotated the instance is recreated so we need to restore values
        if (savedInstanceState != null) {
            startStopButton.text = savedInstanceState.getString("startStopButtonText")
            startStopButton.icon = requireActivity().getDrawable(savedInstanceState.getInt("startStopButtonIconId"))
            timerStarted = savedInstanceState.getBoolean("timerStarted")
            time = savedInstanceState.getDouble("time")
            timeTv.text = getTimeStringFromDouble(time)
        }

        val updateTime: BroadcastReceiver = object : BroadcastReceiver() {
            override fun onReceive(context: Context, intent: Intent) {
                time = intent.getDoubleExtra(TimerService.TIMER_EXTRA, 0.0)
                timeTv.text = getTimeStringFromDouble(time)
            }
        }

        startStopButton.setOnClickListener { startStopTimer() }
        resetButton.setOnClickListener { resetTimer() }
        saveButton.setOnClickListener {
            stopTimer()
            val route = this.arguments?.getString("route")
            val sharedTime = requireActivity().getSharedPreferences("com.example.listdetail.shared",0)
            val record = sharedTime.getFloat("$route time", POSITIVE_INFINITY)
            val sdf = SimpleDateFormat("dd/M/yyyy")
            val currentDate = sdf.format(Date())

            val edit = sharedTime.edit()
            if (record > time.toFloat()) {
                edit.putString(route, timeTv.text.toString() + " at " + currentDate)
                edit.putFloat("$route time", time.toFloat())
                edit.apply()
                Toast.makeText(view.context, "This is the best time ever for $route! Time has been saved.", Toast.LENGTH_LONG).show()
            }
            edit.putString("$route last", timeTv.text.toString() + " at " + currentDate)
            edit.apply()
        }

        serviceIntent = Intent(requireActivity().applicationContext, TimerService::class.java)
        requireActivity().registerReceiver(updateTime, IntentFilter(TimerService.TIMER_UPDATED))
    }

    private fun resetTimer() {
        time = 0.0
        timeTv.text = getTimeStringFromDouble(time)
        if (timerStarted)
            stopTimer()
    }

    private fun startStopTimer() {
        if (timerStarted)
            stopTimer()
        else
            startTimer()
    }
```

```
89      private fun startTimer() {
90          serviceIntent.putExtra(TimerService.TIMER_EXTRA, time)
91          requireActivity().startService(serviceIntent)
92          startStopButton.text = "  Stop"
93          startStopButton.icon = requireActivity().getDrawable(R.drawable.ic_baseline_pause_24)
94          timerStarted = true
95      }
96
97      private fun stopTimer() {
98          requireActivity().stopService(serviceIntent)
99          startStopButton.text = "Start"
100         startStopButton.icon = requireActivity().getDrawable(R.drawable.ic_baseline_play_arrow_24)
101         timerStarted = false
102     }
103
104     private fun getTimeStringFromDouble(time: Double): String {
105         val resultInt = time.roundToInt()
106         val hours = resultInt % 86400 / 3600
107         val minutes = resultInt % 86400 % 3600 / 60
108         val seconds = resultInt % 86400 % 3600 % 60
109
110         return makeTimeString(hours, minutes, seconds)
111     }
112
113     private fun makeTimeString(hour: Int, min: Int, sec: Int): String = String.format("%02d:%02d:%02d", hour, min, sec)
114
115     override fun onSaveInstanceState(outState: Bundle) {
116         super.onSaveInstanceState(outState)
117         // Save timer data before screen rotate
118         if (timerStarted)
119             outState.putInt("startStopButtonIconId", R.drawable.ic_baseline_pause_24)
120         else
121             outState.putInt("startStopButtonIconId", R.drawable.ic_baseline_play_arrow_24)
122
123         outState.putString("startStopButtonText", startStopButton.text.toString())
124         outState.putBoolean("timerStarted", timerStarted)
125         outState.putDouble("time", time)
126     }
127 }
```

Fragment zagnieżdżony w DetailFragment odpowiadający za wyświetlanie stopera. Oparty na serwisie TimerService pozwalającym na odmierzanie czasu.

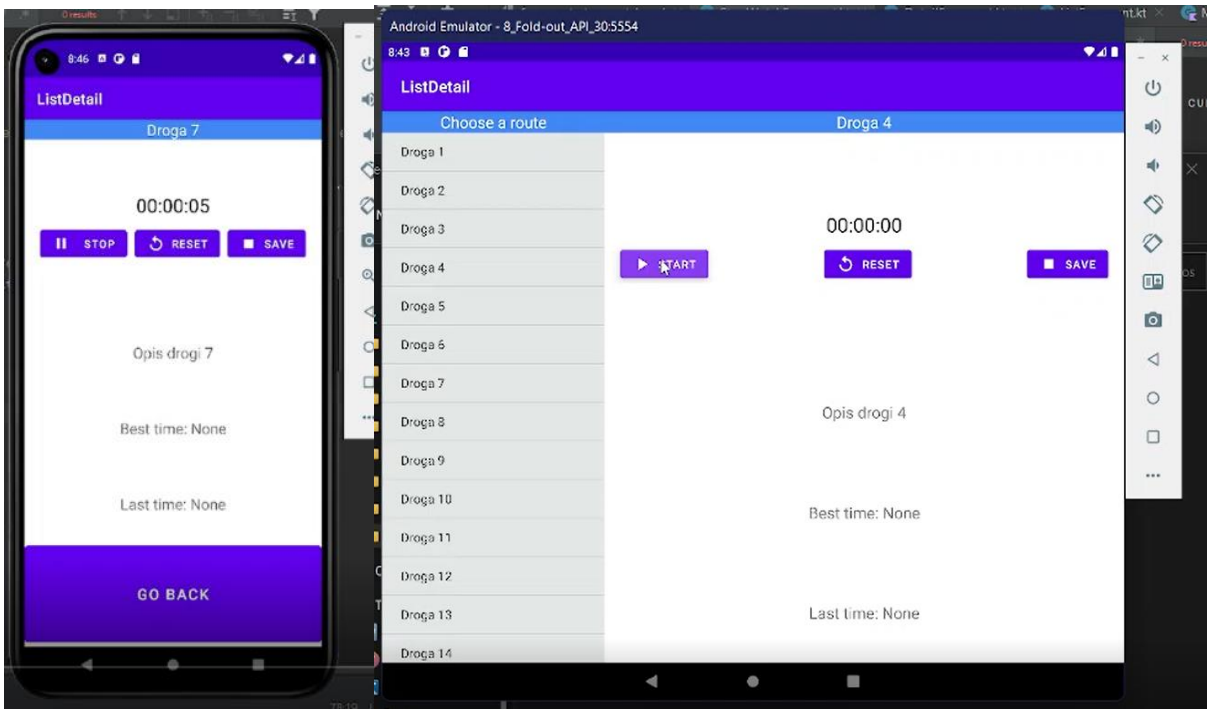
Kod TimerService.kt

```kotlin
class TimerService : Service() {
    override fun onBind(p0: Intent?): IBinder? = null

    private val timer = Timer()

    override fun onStartCommand(intent: Intent, flags: Int, startId: Int): Int {
        val time = intent.getDoubleExtra(TIMER_EXTRA, 0.0)
        timer.scheduleAtFixedRate(TimeTask(time), 0, 1000)
        return START_NOT_STICKY
    }

    override fun onDestroy() {
        timer.cancel()
        super.onDestroy()
    }

    private inner class TimeTask(private var time: Double) : TimerTask() {
        override fun run() {
            val intent = Intent(TIMER_UPDATED)
            time++
            intent.putExtra(TIMER_EXTRA, time)
            sendBroadcast(intent)
        }
    }

    companion object {
        const val TIMER_UPDATED = "timerUpdated"
        const val TIMER_EXTRA = "timerExtra"
    }


}
```

Licznik zostaje inkrmentowany co 1000ms.

Zrzuty ekranów:

Stoper:



Dodany stoper w widoku szczegółowym.

4. Laboratorium 10-11 - Aplikacja typu lista-szczegóły c.d. - wykorzystanie biblioteki wsparcia

Kod InfoFragment.kt

```
11  class InfoFragment : Fragment(R.layout.fragment_info) {
12      override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
13          val switch = view.findViewById<Switch>(R.id.switch1)
14
15          val sharedTime = requireActivity().getSharedPreferences("com.example.listdetail.shared",0)
16          val darkMode = sharedTime.getBoolean("dark_mode", false)
17
18          if (darkMode)
19              AppCompatDelegate.setDefaultNightMode(MODE_NIGHT_YES)
20
21          switch.isChecked = darkMode
22
23          val edit = sharedTime.edit()
24          switch.setOnCheckedChangeListener { _, checked ->
25              if (checked) {
26                  AppCompatDelegate.setDefaultNightMode(MODE_NIGHT_YES)
27                  edit.putBoolean("dark_mode", true)
28              } else {
29                  AppCompatDelegate.setDefaultNightMode(MODE_NIGHT_NO)
30                  edit.putBoolean("dark_mode", false)
31              }
32              edit.apply()
33          }
34      }
35  }
```

Kod ListFragment.kt

```kotlin
class ListFragment : Fragment(R.layout.fragment_list) {

    companion object {
        fun newInstance(difficulty: String): ListFragment {
            val fragment = ListFragment()
            val args = Bundle()
            args.putString("difficulty", difficulty)
            fragment.arguments = args
            return fragment
        }
    }

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return LayoutInflater.from(container?.context).inflate(R.layout.fragment_list, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        val difficulty = this.arguments?.getString("difficulty")
        var routes = arrayOf(
            "Droga 1", "Droga 2", "Droga 3", "Droga 4", "Droga 5",
            "Droga 6", "Droga 7", "Droga 8", "Droga 9", "Droga 10",
            "Droga 11", "Droga 12", "Droga 13", "Droga 14", "Droga 15",
            "Droga 16", "Droga 17", "Droga 18", "Droga 19", "Droga 20",
        )

        when (difficulty) {
            "easy" -> routes = getEasyRoutes(routes)
            "difficult" -> routes = getDifficultRoutes(routes)
        }

        val recyclerView = view.findViewById<RecyclerView>(R.id.recyclerview)
        recyclerView.layoutManager = LinearLayoutManager(context)
        recyclerView.adapter = RecyclerAdapter(routes)
    }

    private fun getDifficultRoutes(routes: Array<String>): Array<String> {
        val list = mutableListOf<String>()
        routes.forEachIndexed{ i, v -> if (i%2==0) list.add(v)}
        return list.toTypedArray()
    }

    private fun getEasyRoutes(routes: Array<String>): Array<String> {
        val list = mutableListOf<String>()
        routes.forEachIndexed{ i, v -> if (i%2==1) list.add(v)}
        return list.toTypedArray()
    }
}
```

Zmieniono ListView na RecyclerView.

Kod DetailFragment.kt

```kotlin
class DetailFragment : Fragment(R.layout.fragment_detail) {
    private lateinit var rotateOpen: Animation
    private lateinit var rotateClose: Animation
    private lateinit var fromBottom: Animation
    private lateinit var toBottom: Animation
    private lateinit var mainBtn: FloatingActionButton
    private lateinit var statBtn: FloatingActionButton
    private lateinit var timerBtn: FloatingActionButton
    private var clicked = false

    private lateinit var textView3: TextView
    private lateinit var textView4: TextView
    private lateinit var fragmentContainer: FragmentContainerView

    companion object {
        fun newInstance(route: String, description: String, imageId: Int): DetailFragment {
            val fragment = DetailFragment()
            val args = Bundle()
            args.putString("route", route)
            args.putString("description", description)
            args.putInt("imageId", imageId)
            fragment.arguments = args
            return fragment
        }
    }

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return LayoutInflater.from(container?.context).inflate(R.layout.fragment_detail, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        val textView = view.findViewById<TextView>(R.id.detail_textview)
        val textView2 = view.findViewById<TextView>(R.id.detail_textview2)
        val imageView = view.findViewById<ImageView>(R.id.detailimage)
        textView3 = view.findViewById(R.id.detail_textview3)
        textView4 = view.findViewById(R.id.detail_textview4)
        fragmentContainer = view.findViewById(R.id.fragment_container3)

        textView.text = this.arguments?.getString("route")
        textView2.text = this.arguments?.getString("description")
        val imageId = this.arguments?.getInt("imageId")
        val image = if (imageId == 1) R.drawable.route else R.drawable.route2
        imageView.setImageResource(image)

        val sharedTime = requireActivity().getSharedPreferences("com.example.listdetail.shared",0)
        "Best time: ${sharedTime.getString(textView.text.toString(), "None")}".also { textView3.text = it }
        "Last time: ${sharedTime.getString("${textView.text} last", "None")}".also { textView4.text = it }

        if (savedInstanceState == null) {
            parentFragmentManager.commit {
                setReorderingAllowed(true)
                add<StopWatchFragment>(
                    R.id.fragment_container3,
                    "SS",
                    Bundle().apply { putString("route", textView.text.toString()) }
                )
            }
        } else {
            textView3.visibility = savedInstanceState.getInt("statsVisibility")
            textView4.visibility = savedInstanceState.getInt("statsVisibility")
            fragmentContainer.visibility = savedInstanceState.getInt("timerVisibility")
        }

        rotateOpen = AnimationUtils.loadAnimation(view.context, R.anim.rotate_open_anim)
        rotateClose = AnimationUtils.loadAnimation(view.context, R.anim.rotate_close_anim)
        fromBottom = AnimationUtils.loadAnimation(view.context, R.anim.from_bottom_anim)
        toBottom = AnimationUtils.loadAnimation(view.context, R.anim.to_bottom)

        mainBtn = view.findViewById(R.id.main_btn)
        statBtn = view.findViewById(R.id.stat_btn)
        timerBtn = view.findViewById(R.id.timer_btm)

        mainBtn.setOnClickListener {
            onMainButtonClicked()
        }
```

```
        statBtn.setOnClickListener {
            textView3.visibility = if (textView3.visibility == View.VISIBLE) View.INVISIBLE else View.VISIBLE
            textView4.visibility = if (textView4.visibility == View.VISIBLE) View.INVISIBLE else View.VISIBLE
        }

        timerBtn.setOnClickListener {
            fragmentContainer.visibility = if (fragmentContainer.visibility == View.VISIBLE) View.INVISIBLE else View.VISIBLE
        }

    }

    override fun onSaveInstanceState(outState: Bundle) {
        super.onSaveInstanceState(outState)

        // Save timer data before screen rotate
        outState.putInt("timerVisibility", fragmentContainer.visibility)
        outState.putInt("statsVisibility", textView3.visibility)
    }

    private fun onMainButtonClicked() {
        setVisibility()
        setAnimation()
        setClickable()
        clicked = !clicked
    }

    private fun setAnimation() {
        if (clicked) {
            statBtn.visibility = View.INVISIBLE
            timerBtn.visibility = View.INVISIBLE
        } else {
            statBtn.visibility = View.VISIBLE
            timerBtn.visibility = View.VISIBLE
        }
    }
```

```
    private fun setVisibility() {
        if (clicked) {
            statBtn.startAnimation(toBottom)
            timerBtn.startAnimation(toBottom)
            mainBtn.startAnimation(rotateClose)
        } else {
            statBtn.startAnimation(fromBottom)
            timerBtn.startAnimation(fromBottom)
            mainBtn.startAnimation(rotateOpen)
        }
    }

    private fun setClickable() {
        if (clicked) {
            statBtn.isClickable = false
            timerBtn.isClickable = false
        } else {
            statBtn.isClickable = true
            timerBtn.isClickable = true
        }
    }
}
```

W widoku szczegółowym dodano FloatingActionButton umożliwiający wyświetlenie stopera i statystyk danej trasy a także powiększony obrazek trasy.

Implementacja wykrywania gestów w MainActivity.kt

```
58      override fun onTouchEvent(event: MotionEvent?): Boolean {
59          gestureDetector.onTouchEvent(event)
60
61          when (event?.action) {
62              0 -> {
63                  x1 = event.x
64              }
65              1 -> {
66                  x2 = event.x
67
68                  if (kotlin.math.abs(x2 - x1) > MIN_DISTANCE) {
69                      if (x2 > x1) {
70                          when (tabId) {
71                              0 -> {
72                                  replaceFragment(easyRoutes)
73                                  tabId = 1
74                              }
75                              1 -> {
76                                  replaceFragment(difficultRoutes)
77                                  tabId = 2
78                              }
79                              2 -> {
80                                  replaceFragment(InfoFragment())
81                                  tabId = 0
82                              }
83                          }
84                      } else {
85                          when (tabId) {
86                              0 -> {
87                                  replaceFragment(difficultRoutes)
88                                  tabId = 2
89                              }
90                              1 -> {
91                                  replaceFragment(InfoFragment())
92                                  tabId = 0
93                              }
94                              2 -> {
95                                  replaceFragment(easyRoutes)
96                                  tabId = 1
```

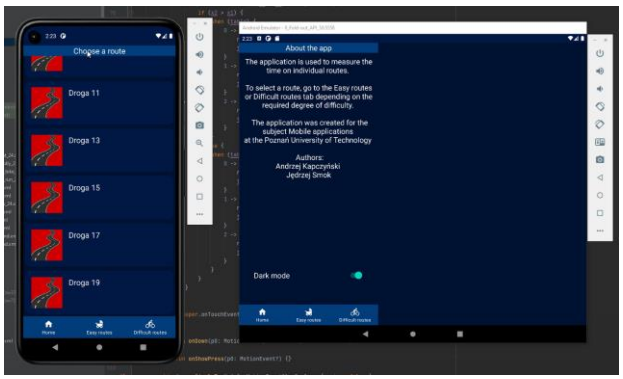
```
97                          }
98                      }
99                  }
100             }
101         }
102     }
103     return super.onTouchEvent(event)
104 }
105
106 override fun onDown(p0: MotionEvent?): Boolean {  return false }
107
108 override fun onShowPress(p0: MotionEvent?) {}
109
110 override fun onSingleTapUp(p0: MotionEvent?): Boolean { return false }
111
112 override fun onScroll(p0: MotionEvent?, p1: MotionEvent?, p2: Float, p3: Float): Boolean { return false }
113
114 override fun onLongPress(p0: MotionEvent?) {}
115
116 override fun onFling(p0: MotionEvent?, p1: MotionEvent?, p2: Float, p3: Float): Boolean { return false }
117 }
```

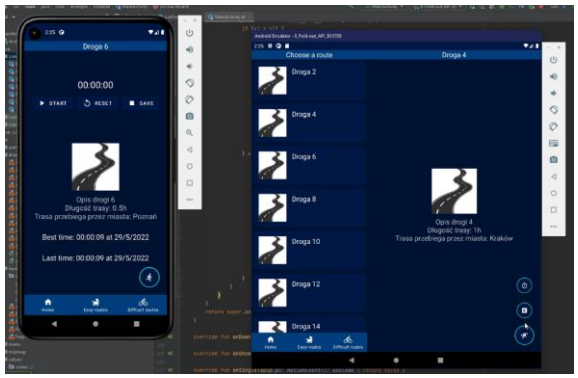
Layout CardView

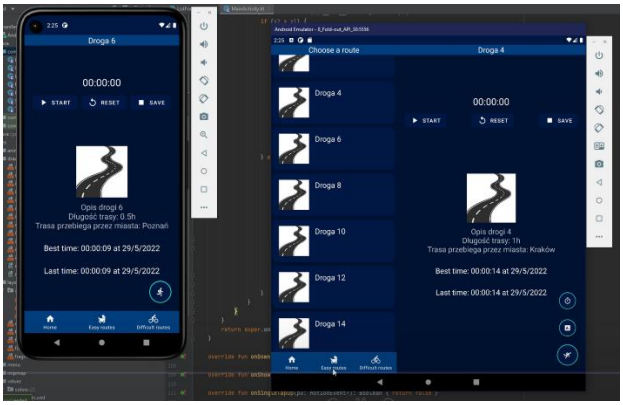
```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/card_view"
    android:layout_margin="5dp"
    app:cardBackgroundColor="@color/secondaryColor"
    app:cardCornerRadius="12dp"
    app:cardElevation="3dp"
    app:contentPadding="4dp">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/relativelayout"
        android:padding="12dp">

        <ImageView
            android:id="@+id/item_image"
            android:layout_width="80dp"
            android:layout_height="80dp"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintLeft_toLeftOf="parent"/>

        <TextView
            android:id="@+id/item_title"
            android:textColor="@color/textColor"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            app:layout_constraintWidth_percent="0.85"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintLeft_toRightOf="@+id/item_image"
            android:layout_marginStart="16dp"
            android:textSize="20sp"/>

    </androidx.constraintlayout.widget.ConstraintLayout>

</androidx.cardview.widget.CardView>
```

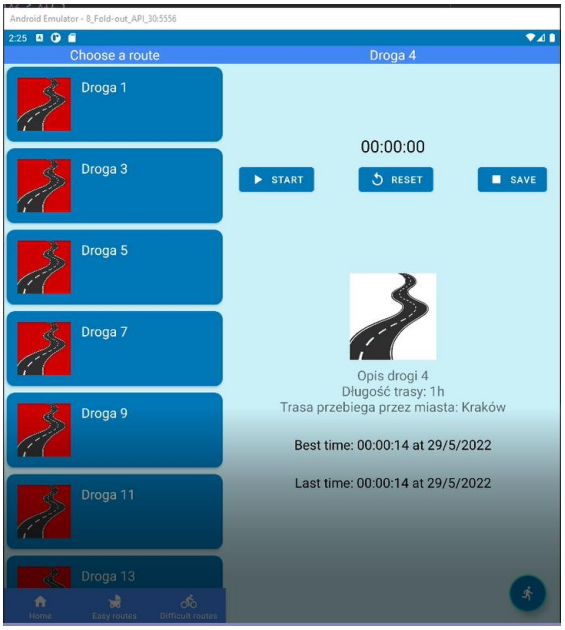
W aplikacji zamieniono ListView na RecyclerView prezentującego pozycje zawarte w CardView – obrazka oraz nazwy trasy. W widoku szczegółowym umieszczono FAB, pozwalający na wyświetlenie/ukrycie stopera oraz statystyk. Wprowadzono też motyw jasny oraz ciemny aplikacji. Poruszanie się pomiędzy kartami jest także możliwe za pomocą gestu przeciągnięcia.

Zrzuty ekranów:









5. Laboratorium 12 – Animacja

Kod ShoeFragment.kt

```kotlin
class ShoeFragment: Fragment(R.layout.fragment_shoe) {
    private lateinit var mSceneView: View
    private lateinit var mLoadView: View
    private lateinit var mShoeView: View
    private lateinit var mSkyView: View

    companion object {
        fun newInstance(): ShoeFragment{
            return ShoeFragment()
        }
    }


    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        val view: View = inflater.inflate(R.layout.fragment_shoe, container, false)
        mSceneView = view
        mLoadView = view.findViewById(R.id.loader)
        mShoeView = view.findViewById(R.id.shoe)
        mSkyView = view.findViewById(R.id.sky)

        startAnimation(mSceneView, 1500)

        return view
    }
    private fun startAnimation(view: View, duration: Long){
        val heightAnimator: ObjectAnimator = ObjectAnimator.ofFloat(mLoadView, "scaleX", 1f, 20f)
        heightAnimator.duration = duration*2
        val  shoeScaleAnimator = ObjectAnimator.ofFloat(mShoeView, "rotation", 50f, 0f)
        shoeScaleAnimator.duration = duration
        val shoeXAnimator = ObjectAnimator.ofFloat(mShoeView, "translationX", 300f, 0f)
        shoeXAnimator.duration = duration
        val shoeYAnimator = ObjectAnimator.ofFloat(mShoeView, "translationY", -150f, 0f)
        shoeYAnimator.duration = duration
        val animatorSet = AnimatorSet()
        animatorSet.play(heightAnimator).with(shoeScaleAnimator).with(shoeXAnimator).with(shoeYAnimator)
        animatorSet.start()
    }
}
```

Dodano fragment odpowiadający za wyświetlenie animacji.

Kod AnimActivity.kt

```
7   class AnimActivity : AppCompatActivity() {
8       override fun onCreate(savedInstanceState: Bundle?) {
9           super.onCreate(savedInstanceState)
10          setContentView(R.layout.activity_anim)
11
12          val thread = Thread {
13              run{
14                  val fm = supportFragmentManager
15                  var fragment = fm.findFragmentById(R.id.fragment_container)
16                  if (fragment == null){
17                      // here you can change animation
18                      fragment = ShoeFragment.newInstance()
21                      fm.beginTransaction().add(R.id.fragment_container, fragment).commit()
22                  }
23                  Thread.sleep(4000)
24              }
25              runOnUiThread {
26                  val intent = Intent(this, MainActivity::class.java)
27                  startActivity(intent)
28                  finish()
29              }
30          }
31          thread.start()
32      }
33  }
```

Dodano aktywność odpowiadającą za wyświetlenie fragmentu z animacją a następnie przejście do głownej aktywności.

Zrzuty ekranu: