Aplikacje mobilne

15 maja 2023

1. Aplikacja typu lista-szczegóły

   1.1 Aplikacja składa się z dwóch aktywności: głównej wyświetlającej listę potraw oraz aktywności szczegółów uruchamianej po kliknięciu wybranej potrawy z listy i wyświetlającej co najmniej listę składników oraz sposób przygotowania potrawy.

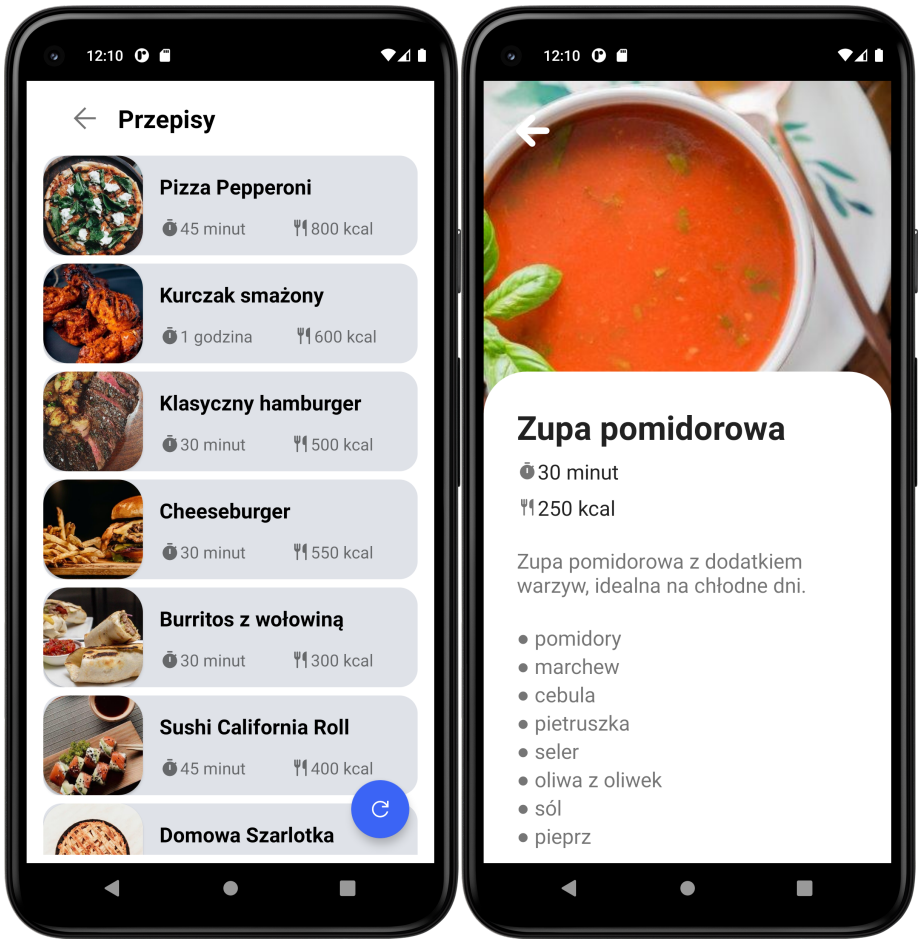
   1.1.1 Aktywność wyświetlająca listę potraw.

```
class RecipeListFragment : Fragment() {
    private var _binding: FragmentRecipeListBinding? = null
    private val binding get() = _binding!!

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?,
    ): View {
        _binding = FragmentRecipeListBinding.inflate(inflater, container, false)
        return binding.root
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        val recyclerView: RecyclerView = binding.itemList

        val itemDetailFragmentContainer: View? =
        view.findViewById(R.id.item_detail_nav_container)

        setupRecyclerView(recyclerView, itemDetailFragmentContainer)
    }
}
```

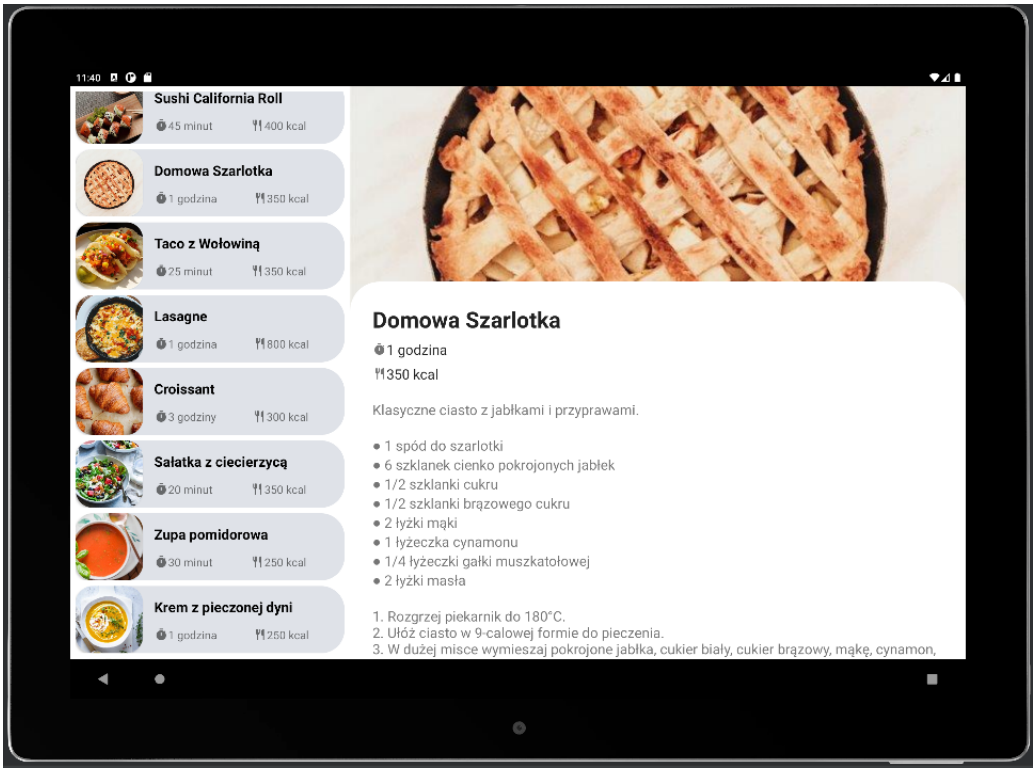
1.1.2 Wyświetlanie listy przepisów i pobieranie ich z serwera.

```
private fun setupRecyclerView(
    recyclerView: RecyclerView,
    itemDetailFragmentContainer: View?,
) {
    RetrofitInstance.api.getRecipes().enqueue(object : retrofit2.Callback<List<Recipe>> {
        override fun onResponse(
            call: retrofit2.Call<List<Recipe>>,
            response: retrofit2.Response<List<Recipe>>,
        ) {
            if (response.isSuccessful && response.body() != null) {
                val recipes = (response.body())!!
                for (recipe in recipes) {
                    Log.e(ContentValues.TAG, recipe.title)
                }
                recyclerView.adapter = SimpleItemRecyclerViewAdapter(
                    recipes, itemDetailFragmentContainer
                )
            } else {
                Log.e(ContentValues.TAG, "Response not successful")
            }
        }

        override fun onFailure(call: retrofit2.Call<List<Recipe>>, t: Throwable) {
            Log.e(ContentValues.TAG, "Response not successful")
        }
    })
}
```

1.2 Osobna wersja dla smartfonów i tabletów.



Rysunek 1: Wersja dla smartfonów



Rysunek 2: Wersja dla tabletów

2. Dodanie timera

```
class TimerFragment : Fragment() {
    private var _binding: FragmentTimerBinding? = null
    private val binding get() = _binding!!
    private lateinit var startButton: AppCompatButton
    private lateinit var resetTimerButton: AppCompatButton
    private lateinit var stopButton: AppCompatButton
    private lateinit var timerValue: TextView
    private var isTimerRunning: Boolean = false
    private var time: Long = 0L
    private var timeRemaining: Long = 0L
    private var timer: CountDownTimer? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        val totalTime = arguments?.getString("total_time")

        if (totalTime != null) {
            time = totalTime.toLong() * 60000
        }
        timeRemaining = time
    }

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?,
    ): View {
        _binding = FragmentTimerBinding.inflate(inflater, container, false)
        val rootView = binding.root

        startButton = binding.startTimerButton
        stopButton = binding.stopTimerButton
        resetTimerButton = binding.resetTimerButton
        timerValue = binding.timerValue

        val initialTime = time / 1000
        if (!isTimerRunning) {
            timerValue.text = String.format(
                "%02d:%02d:%02d",
                initialTime / 3600,
                (initialTime % 3600) / 60,
                initialTime % 60
            )
        }
```

2.1 Przycisk start

```
startButton.setOnClickListener {
        if (!isTimerRunning) {
            timer = object : CountDownTimer(timeRemaining, 1000L) {
                override fun onTick(millisUntilFinished: Long) {
                    timeRemaining = millisUntilFinished
                    updateTimerText()
                }

                override fun onFinish() {
                    startVibrations()
                    isTimerRunning = false
                    updateTimerText()
                    timeRemaining = time
                }
            }

            timer?.start()
            isTimerRunning = true
        } else {
            timer = object : CountDownTimer(timeRemaining, 1000L) {
                override fun onTick(millisUntilFinished: Long) {
                    timeRemaining = millisUntilFinished
                    updateTimerText()
                }

                override fun onFinish() {
                    startVibrations()
                    isTimerRunning = false
                    updateTimerText()
                    timeRemaining = time
                }
            }
            timer?.start()
        }
    }
```

2.2 Przycisk stop i reset

```
stopButton.setOnClickListener {
    onStop()
}

resetTimerButton.setOnClickListener {
    resetTimer()
}


override fun onStop() {
    super.onStop()
    if (isTimerRunning) {
        timer?.cancel()
        timer = null
    }
}

private fun resetTimer() {
    if (isTimerRunning) {
        timer?.cancel()
        timeRemaining = 0L
        updateTimerText()
        timer = null
        timeRemaining = time
    }
}
```

3. Elementy biblioteki wsparcia wzornictwa

3.1 Fragment z kategoriami przepisów działa analogicznie do listy przepisów tylko jest on przekazywany do GridLayoutManager.

```
private fun setupCategoryRecyclerView(
    recyclerView: RecyclerView,
    itemListFragmentContainer: View?,
) {
    RetrofitInstance.api.getCategories().enqueue(object : Callback<List<Category>> {
        override fun onResponse(
            call: Call<List<Category>>,
            response: Response<List<Category>>,
        ) {
            if (response.isSuccessful && response.body() != null) {
                val categories = (response.body())!!

                for (category in categories) {
                    Log.e(ContentValues.TAG, category.name)
                }

                val gridNum = 2;
                val layoutManager = GridLayoutManager(recyclerView.context, gridNum)

                recyclerView.layoutManager = layoutManager
                recyclerView.adapter = CategoryRecyclerViewAdapter(
                    categories, itemListFragmentContainer
                )
            } else {
                Log.e(ContentValues.TAG, response.code().toString())
                Log.e(ContentValues.TAG, "Response not successful")
            }
        }

        override fun onFailure(call: Call<List<Category>>, t: Throwable) {
            Log.e(ContentValues.TAG, "Response not successful")
            Log.e(ContentValues.TAG, t.message.toString())
        }
    })
}
```