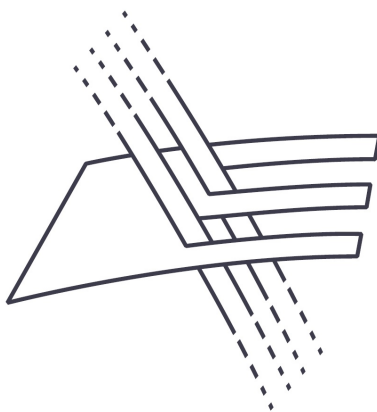


Sprawozdanie z Laboratorium 5

Architektura i inżynieria usług i aplikacji (ARIUS)

Mikołaj Roszczyk 318482



Wydział Elektroniki i Nauk Informatycznych
Inżynieria Internetu Rzeczy
Politechnika Warszawska
25 grudnia 2024

Spis treści

1	Wymagania	2
2	Opis implementacji	3
2.1	Android Manifest	3
2.2	Klasy w Javie	3
2.2.1	MainActivity	3
2.2.2	Task	5
2.2.3	TaskAdapter	6
2.2.4	TaskDetailsActivity	7
2.3	Pliki XML aktywności	9
2.3.1	activity main	9
2.3.2	activity task details	9
2.3.3	task item	11
2.4	Pozostałe	12
3	Efekty działania aplikacji	13
3.1	Widok zbiorczej listy zadań	13
3.2	Widok szczegółów zadania	13
3.3	Widok głównej listy po zmianie statusu	14
3.4	Zmiana statusu z wykonane na niewykonane	14
3.5	Status <i>przeterminowane</i>	15

1 Wymagania

Należało stworzyć aplikację, która:

- Posiada dwa widoki:
 - Widok główny przedstawiający listę zadań do wykonania
 - Widok szczegółów zadania
- Na głównym widoku mają być widoczne Nazwy zadań, data wykonania oraz graficzne przedstawienie stanu (wykonane, niewykonane, przeterminowane)
- Widok szczegółowy ma wyświetlać tytuł zadania, szczegóły, deadline, stan zadania (wykonane, niewykonane, przeterminowane) i umożliwiać zmianę stanu zadania (wykonane/niewykonane)

2 Opis implementacji

2.1 Android Manifest

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.roszczyk.arius_lab5">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MaterialComponents.DayNight.DarkActionBar">

        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity
            android:name=".TaskDetailsActivity"
            android:exported="false" />
    </application>
</manifest>
```

2.2 Klasy w Javie

Przygotowano aplikację z czterema klasami w Javie:

- MainActivity - odpowiada za obsługę głównej aktywności aplikacji
- Task - odpowiadająca za definicję zadania
- TaskAdapter - wykonywanie operacji i wyświetlanie zadań
- TaskDetailsActivity - odpowiada za obsługę drugiej aktywności

2.2.1 MainActivity

```
package com.roszczyk.arius_lab5;

import android.content.Intent;
import android.os.Bundle;
```

```
import android.widget.AdapterView;
import android.widget.ListView;
import android.view.View;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {
    private List<Task> taskList;
    private TaskAdapter taskAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        taskList = new ArrayList<>();
        taskList.add(new Task("Task 1", "Details for Task 1", "2024-12-20", false));
        taskList.add(new Task("Task 2", "Details for Task 2", "2024-12-20", true));
        taskList.add(new Task("Task 3", "Details for Task 3", "2024-12-30", false));
        taskList.add(new Task("Task 4", "Details for Task 4", "2024-12-30", true));

        taskAdapter = new TaskAdapter(this, taskList);
        ListView listView = findViewById(R.id.taskListView);
        listView.setAdapter(taskAdapter);

        listView.setOnItemClickListener((AdapterView<?> parent, View view, \\\
            int position, long id) -> {
            Task selectedTask = taskList.get(position);
            Intent intent = new Intent(MainActivity.this, TaskDetailsActivity.class);
            intent.putExtra("task", selectedTask);
            startActivityForResult(intent, 1);
        });
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == 1 && resultCode == RESULT_OK) {
            Task updatedTask = (Task) data.getSerializableExtra("updatedTask");
            if (updatedTask != null) {
                for (int i = 0; i < taskList.size(); i++) {
                    if (taskList.get(i).getTitle().equals(updatedTask.getTitle())) {
                        taskList.set(i, updatedTask);
                        break;
                    }
                }
            }
        }
    }
}
```

```
        }
        taskAdapter.notifyDataSetChanged();
    }
}
}
```

2.2.2 Task

```
package com.roszczyk.arius_lab5;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.io.Serializable;

public class Task implements Serializable{
    private String title;
    private String details;
    private String deadline;
    private boolean done;

    public Task(String title, String details, String deadline, boolean done) {
        this.title = title;
        this.details = details;
        this.deadline = deadline;
        this.done = done; // True - wykonane, False - niewykonane
    }

    public String getTitle() { return title; }
    public void setTitle(String title) { this.title = title; }
    public String getDetails() { return details; }
    public void setDetails(String details) { this.details = details; }
    public String getDeadline() { return deadline; }
    public void setDeadline(String deadline) {
        this.deadline = deadline;
    }
    public boolean getStatus() { return done; }
    public void setStatus(boolean status) { this.done = status; }
    public boolean isOverdue(){
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        Date date;
        try {
            date = sdf.parse(this.deadline);
        } catch (ParseException e) {
            System.err.println("Invalid date format: " + this.deadline);
        }
    }
}
```

```
        return false;
    }
    return !this.done && date.before(new Date());
}
}
```

2.2.3 TaskAdapter

```
package com.roszczyk.arius_lab5;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import java.util.List;

public class TaskAdapter extends ArrayAdapter<Task> {

    public TaskAdapter(Context context, List<Task> tasks) {
        super(context, 0, tasks);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View itemView = convertView;
        if (itemView == null) {
            itemView = LayoutInflater.from(getContext()).inflate(R.layout.task_item,\\
                parent, false);
        }

        Task currentTask = getItem(position);

        TextView titleTextView = itemView.findViewById(R.id.taskTitle);
        titleTextView.setText(currentTask.getTitle());

        TextView deadlineTextView = itemView.findViewById(R.id.taskDeadline);
        deadlineTextView.setText(currentTask.getDeadline());

        ImageView statusIcon = itemView.findViewById(R.id.taskStatusIcon);
        if (currentTask.getStatus()) {
            statusIcon.setImageResource(R.drawable.ic_completed);
        } else if (currentTask.isOverdue()) {
```

```
        statusIcon.setImageResource(R.drawable.ic_overdue);
    } else {
        statusIcon.setImageResource(R.drawable.ic_pending);
    }

    return itemView;
}
}
```

2.2.4 TaskDetailsActivity

```
package com.roszczyk.arius_lab5;

import com.roszczyk.arius_lab5.Task;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.view.MenuItem;

import androidx.appcompat.app.AppCompatActivity;

public class TaskDetailsActivity extends AppCompatActivity {

    private TextView titleTextView;
    private TextView descriptionTextView;
    private TextView deadlineTextView;
    private TextView statusTextView;
    private Button completeButton;
    private Button pendingButton;
    private Task task;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_task_details);

        if (getSupportActionBar() != null) {
            getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        }

        titleTextView = findViewById(R.id.taskDetailsTitle);
        descriptionTextView = findViewById(R.id.taskDetailsDescription);
        deadlineTextView = findViewById(R.id.taskDetailsDeadline);
    }
}
```



```
statusTextView = findViewById(R.id.taskDetailsStatus);
completeButton = findViewById(R.id.completeButton);
pendingButton = findViewById(R.id.pendingButton);

Intent intent = getIntent();
task = (Task) intent.getSerializableExtra("task");

if (task != null) {
    titleTextView.setText(task.getTitle());
    descriptionTextView.setText(task.getDetails());
    deadlineTextView.setText(task.getDeadline());
    updateStatusText(task);
}

completeButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (task != null) {
            task.setStatus(true);
            updateStatusText(task);
        }
    }
});

pendingButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (task != null) {
            task.setStatus(false);
            updateStatusText(task);
        }
    }
});
}

private void updateStatusText(Task task) {
    if (task.getStatus()) {
        statusTextView.setText("completed");
    } else if (task.isOverdue()) {
        statusTextView.setText("overdue");
    } else {
        statusTextView.setText("pending");
    }
}

// Obsługa strzałki powrotu
@Override
```

```

public boolean onOptionsItemSelected(MenuItem item) {
    if (item.getItemId() == android.R.id.home) {
        Intent resultIntent = new Intent();
        resultIntent.putExtra("updatedTask", task);
        setResult(RESULT_OK, resultIntent);
        finish();
        return true;
    }
    return super.onOptionsItemSelected(item);
}

@Override
protected void onPause() {
    super.onPause();
    Intent resultIntent = new Intent();
    resultIntent.putExtra("updatedTask", task);
    setResult(RESULT_OK, resultIntent);
}
}

```

2.3 Pliki XML aktywności

Przygotowano trzy pliki XML aktywności

- activity_main.xml - odpowiada za główny widok (aktywność)
- activity_task_details.xml - odpowiada za widok szczegółów zadania (aktywność poboczna)
- task_item.xml - odpowiada za widok pojedynczego zadania w głównym widoku

2.3.1 activity main

```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <ListView
        android:id="@+id/taskListView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>

```

2.3.2 activity task details

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout

```

```
xmlns:android="http://schemas.android.com/apk/res/android"  
android:layout_width="match_parent"  
android:layout_height="match_parent"  
android:orientation="vertical"  
android:padding="16dp">
```

```
<TextView  
    android:id="@+id/taskDetailsTitle"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Task Title"  
    android:textSize="24sp"  
    android:textStyle="bold"  
    android:paddingBottom="8dp" />
```

```
<TextView  
    android:id="@+id/taskDetailsDescription"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Task Description"  
    android:textSize="16sp"  
    android:paddingBottom="8dp" />
```

```
<TextView  
    android:id="@+id/taskDetailsDeadline"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Deadline: 2024-12-30"  
    android:textSize="16sp"  
    android:paddingBottom="8dp" />
```

```
<TextView  
    android:id="@+id/taskDetailsStatus"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Status: Pending"  
    android:textSize="16sp"  
    android:paddingBottom="16dp" />
```

```
<Button  
    android:id="@+id/completeButton"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Mark as Complete"  
    android:backgroundTint="@color/teal_200"  
    android:padding="8dp"  
    android:textColor="@android:color/white" />
```

```
<Button
    android:id="@+id/pendingButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Mark as Pending"
    android:backgroundTint="@color/purple_200"
    android:padding="8dp"
    android:textColor="@android:color/white"
    android:layout_marginTop="8dp" />

</LinearLayout>
```

2.3.3 task item

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:padding="8dp">

    <ImageView
        android:id="@+id/taskStatusIcon"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:layout_marginEnd="8dp"
        android:contentDescription="@string/status_icon"
        android:src="@drawable/ic_overdue" />

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:orientation="vertical">

        <TextView
            android:id="@+id/taskTitle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Task Title"
            android:textAppearance="?attr/textAppearanceListItem"
            android:textStyle="bold" />

        <TextView
            android:id="@+id/taskDeadline"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Due: YYYY-MM-DD HH:mm"
        android:textAppearance="?android:attr/textAppearanceSmall" />
    </LinearLayout>

</LinearLayout>
```

2.4 Pozostałe

Przygotowano poza tym:





- 5 obiektów *drawable*
- kolory
- *themes*
- *strings*

3 Efekty działania aplikacji

Do testowania implementacji dodano w kodzie 4 zadania testowe.

```
taskList = new ArrayList<>();
taskList.add(new Task( title: "Task 1", details: "Details for Task 1", deadline: "2024-12-20", done: false));
taskList.add(new Task( title: "Task 2", details: "Details for Task 2", deadline: "2024-12-20", done: true));
taskList.add(new Task( title: "Task 3", details: "Details for Task 3", deadline: "2024-12-30", done: false));
taskList.add(new Task( title: "Task 4", details: "Details for Task 4", deadline: "2024-12-30", done: true));
```

3.1 Widok zbiorczej listy zadań

Arius-Lab5	
	Task 1 2024-12-20
	Task 2 2024-12-20
	Task 3 2024-12-30
	Task 4 2024-12-30





3.2 Widok szczegółów zadania

Poniżej po lewej stronie szczegóły Task 1 przed kliknięciem *Mark as complete*, a po prawej po kliknięciu.

Arius-Lab5	
Task 1 Details for Task 1 2024-12-20 overdue	Task 1 Details for Task 1 2024-12-20 completed
MARK AS COMPLETE	MARK AS COMPLETE
MARK AS PENDING	MARK AS PENDING


3.3 Widok głównej listy po zmianie statusu

Po zmianie statusu zadania w szczegółach zadania, zmienia się także ikona.

Arius-Lab5	
	Task 1 2024-12-20
	Task 2 2024-12-20
	Task 3 2024-12-30
	Task 4 2024-12-30

3.4 Zmiana statusu z wykonane na niewykonane


W szczegółach można zmienić status zadania *Task 4*

 Arius-Lab5

Task 4
Details for Task 4
2024-12-30
completed

MARK AS COMPLETE

MARK AS PENDING

 Arius-Lab5

Task 4
Details for Task 4
2024-12-30
pending

MARK AS COMPLETE

MARK AS PENDING

W widoku także się zmienia dla tego zadania ikona.

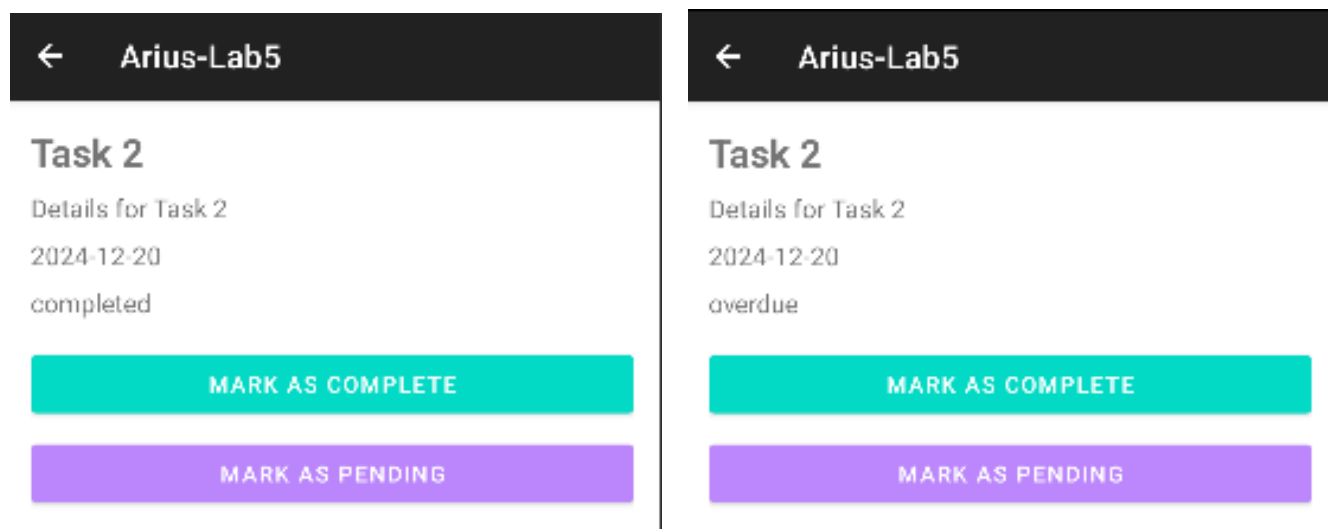
Arius-Lab5	Arius-Lab5
<div>✓ Task 1</div> <div>2024-12-20</div>	<div>✓ Task 1</div> <div>2024-12-20</div>
<div>✓ Task 2</div> <div>2024-12-20</div>	<div>✓ Task 2</div> <div>2024-12-20</div>
<div>! Task 3</div> <div>2024-12-30</div>	<div>! Task 3</div> <div>2024-12-30</div>
<div>✓ Task 4</div> <div>2024-12-30</div>	<div>! Task 4</div> <div>2024-12-30</div>

3.5 Status *przeterminowane*

Status *przeterminowane* jest przyznawany automatycznie, kiedy zadanie jest niewykonane i przekroczony został termin wykonania.

```
public boolean isOverdue(){
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
    Date date;
    try {
        date = sdf.parse(this.deadline);
    } catch (ParseException e) {
        System.err.println("Invalid date format: " + this.deadline);
        return false;
    }
    return !this.done && date.before(new Date());
}
```

Po przyznaniu statusu *niewykonane* dla zadania, którego termin ważności upłynął, status sam się zmienia na *przeterminowane*.



Na liście ikona także się zmienia na czerwoną:

