Politechnika Warszawska
Cyberbezpieczeństwo

UAIM

Zadanie 5

Android Studio. Lista TODO

Daria Shevchenko 335901

May 15, 2025

Contents

1 Wstęp

Aplikacja TODO List została zrealizowana zgodnie z założeniami projektu. Głównym celem było stworzenie funkcjonalnej listy zadań z możliwością śledzenia ich statusu. W aplikacji zaimplementowano wyświetlanie listy zadań z kolorowymi ikonami wskazującymi status każdego zadania. Po kliknięciu na zadanie użytkownik przechodzi do widoku szczegółowego, gdzie może przeczytać pełny opis i zobaczyć termin wykonania. Istnieje możliwość zmiany statusu zadania między "wykonane" a "niewykonane". System automatycznie oznacza zadania jako przeterminowane, jeśli termin minął, a zadanie nie zostało wykonane.

2 Opis implementacji

2.1 Struktura projektu

- **MainActivity.java**

```
listView.setOnItemClickListener((parent, view, position, id) -> {
            Task task = taskList.get(position);
            Intent intent = new Intent(MainActivity.this, TaskDetailActivity
    .class);
            intent.putExtra("task", task);
            intent.putExtra("position", position);
            startActivityForResult(intent, 1);
            });
```

Opis: Przechodzenie do ekranu szczegółów po kliknięciu zadania. Przekazuje obiekt zadania przez Intent.

- **Task.java**

```
public class Task implements Serializable {
    public enum Status { NOT_DONE, DONE, OVERDUE }

    private String title;
    private String description;
    private Date dueDate;
    private Status status;

    // Sprawdza czy zadanie jest przeterminowane
   public boolean isOverdue() {
        return status == Status.NOT_DONE && dueDate.before(new Date());
    }
}
```

Opis: Klasa modelu przechowuje dane zadania i logikę sprawdzania przeterminowania. Statusy są zdefiniowane jako enum.

- **Task.Adapter.java**

```
public class TaskAdapter extends ArrayAdapter<Task> {
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        Task task = getItem(position);

        if (convertView == null)
            convertView = LayoutInflater.from(getContext()).inflate(R.layout
    .task_list_item, parent, false);

        TextView title = convertView.findViewById(R.id.task_title);
        TextView dueDate = convertView.findViewById(R.id.task_due_date);
        ImageView icon = convertView.findViewById(R.id.task_status_icon);

        title.setText(task.getTitle());
        dueDate.setText(sdf.format(task.getDueDate()));

        int iconRes;
        boolean isEffectivelyOverdue = task.getStatus() == Task.Status.
    NOT_DONE && task.getDueDate().before(new java.util.Date());

        if (isEffectivelyOverdue) {
            iconRes = R.drawable.ic_overdue;
        } else if (task.getStatus() == Task.Status.DONE) {
            iconRes = R.drawable.ic_done;
        } else {
```

```
            iconRes = R.drawable.ic_not_done;
        }

        icon.setImageResource(iconRes);

        return convertView;
    }
}
```

Opis: Adapter odpowiedzialny za wyświetlanie elementów listy. Ustawia odpowiednie ikony w zależności od statusu zadania.

- **TaskDetailActivity.java**

```
doneButton.setOnClickListener(v -> {
    task.setStatus(Task.Status.DONE);
    returnResult();
});

notDoneButton.setOnClickListener(v -> {
    task.setStatus(Task.Status.NOT_DONE);
    returnResult();
});
```

Opis: Prosta obsługa przycisków zmieniających status zadania. Wynik jest zwracany do głównej aktywności.

- **activity_main.xml** - układ głównego ekranu z ListView
- **activity_task_detail.xml** - układ ekranu szczegółów zadania
- **task_list_item.xml** - wygląd pojedynczego elementu na liście

3 Efekty działania aplikacji

3.1 a. Widok zbiorczej listy zadań

Jak widać na Rysunku 1, aplikacja wyświetla wszystkie zadania w przejrzystej liście. Każde zadanie pokazuje tytuł zadania, termin wykonania i ikonę statusu (zielona - wykonane, czerwona - niewykonane, czarna - przeterminowane).

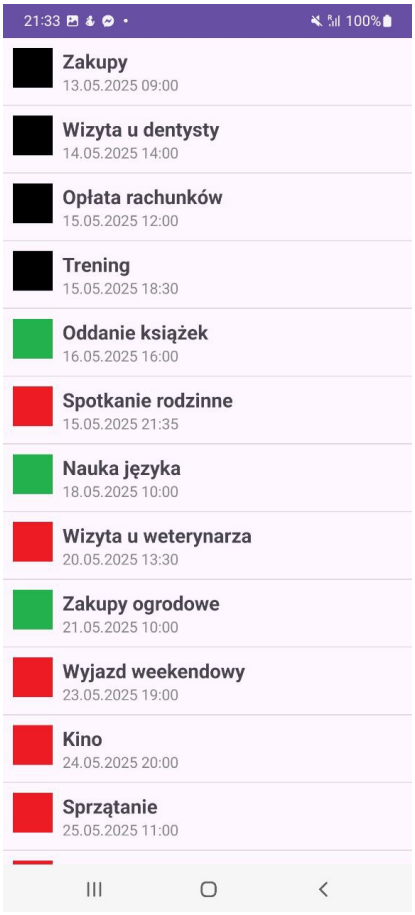

Figure 1: Zbiorcza lista zadań

3.2 b. Widok pojedynczego zadania ze szczegółami zadania do wykonania

Na screenie Figure 2 widać ekran szczegółów zadania (nie przeterminowanego), który pokazuje tytuł, opis, termin oraz przyciski do zmiany statusu. Obok niego na Figure 3 widać, że w tym przypadku termin jest wyświetlony na czerwono, a przyciski nie są aktywne.

Figure 2: Widok zadania do zrobienia

Figure 3: Widok przeterminowanego

3.3 c. Widok dokumentujący zmianę statusu zadania z "wykonane" na "niewykonane"

Aplikacja pozwala na zmianę statusu z wykonanego na niewykonany. Taką zmianę od razu widać zmianą ikony zadania z zielonej na czerwoną (Figure 5).

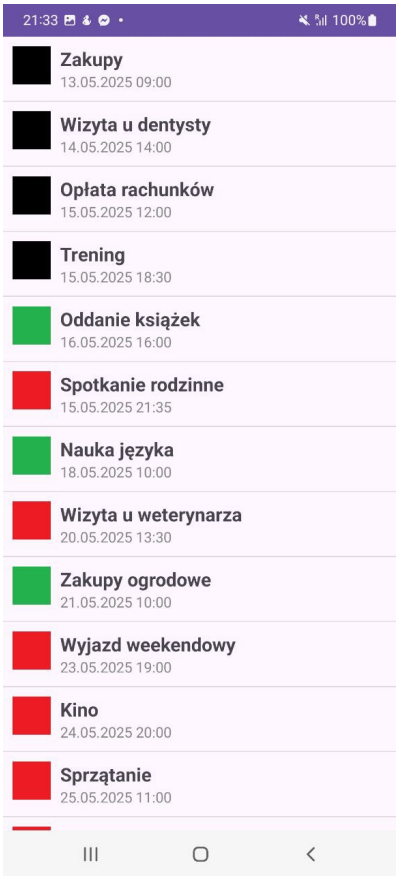

Figure 4: Lista zadań: było

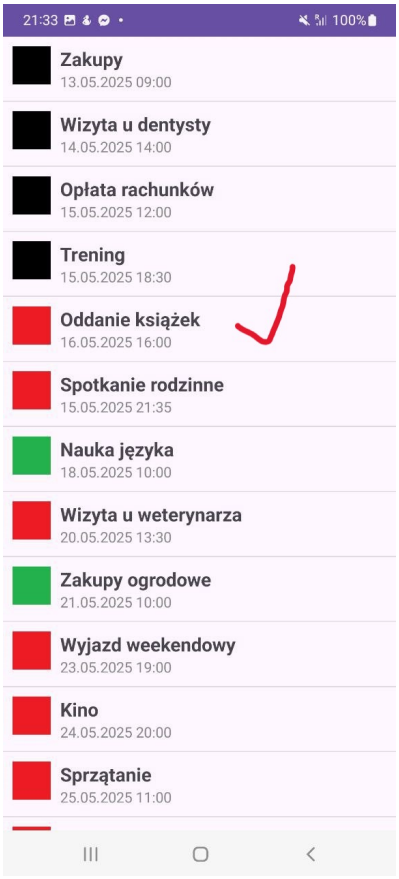

Figure 5: Lista zadań: stało

3.4 d. Widok dokumentujący zmianę statusu zadania z "niewykonane" na "wykonane"

Można zmienić zadaniie niewykonane na wykonane (Figure 7).

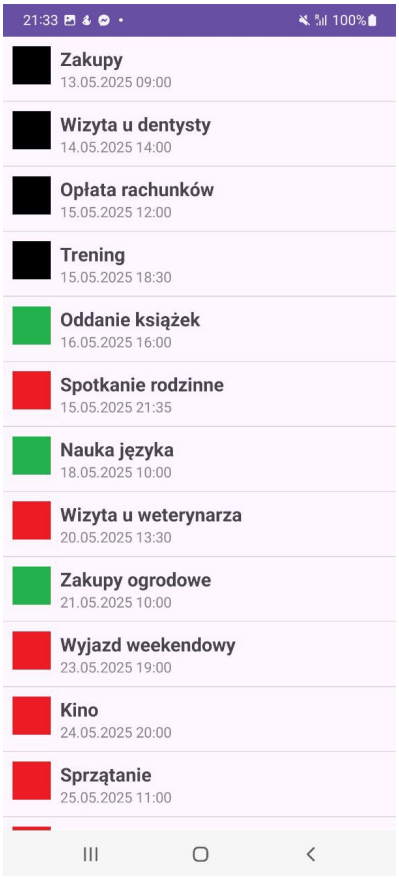

Figure 6: Lista zadań: było

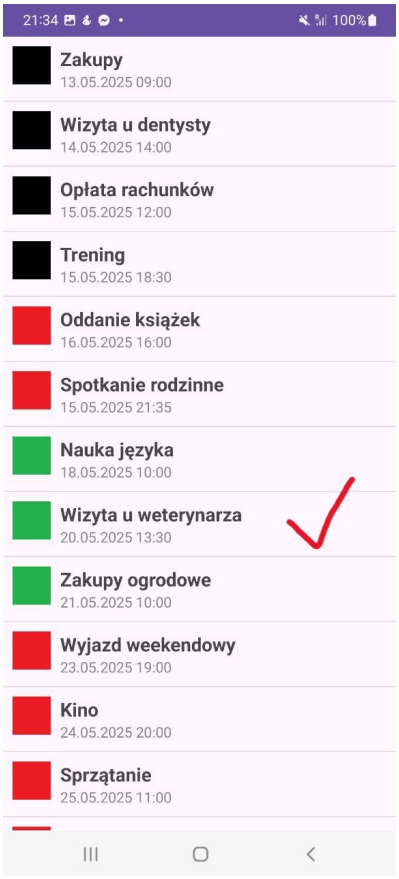

Figure 7: Lista zadań: stało

3.5 e. Widok dokumentujący zmianę statusu zadania z "niewykonane" na "przeterminowane"

Na Figure 9 pokazano, że status z niewykonanego zmienia się na przeterminowany, kiedy termin zrobienia zadania dochodzi do końca.

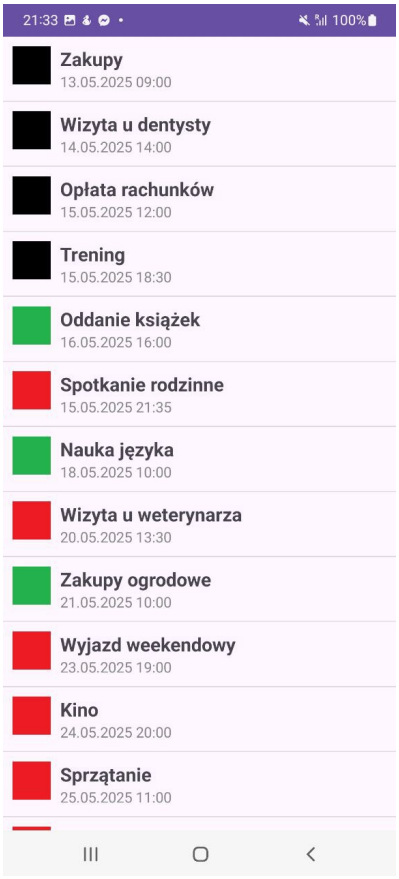

Figure 8: Lista zadań: było

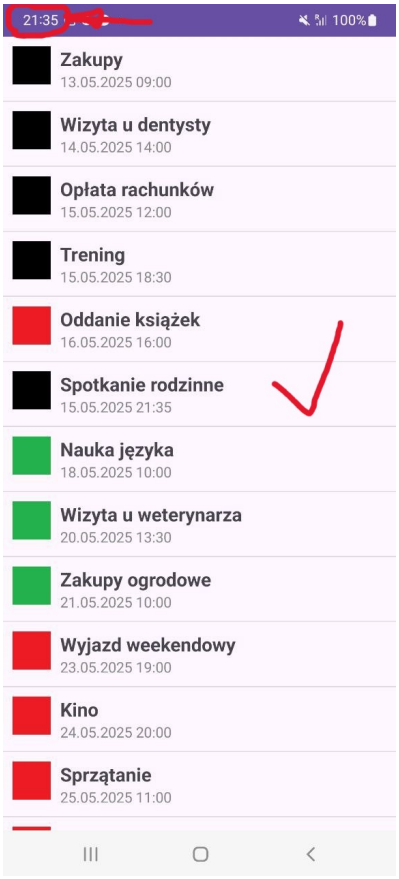

Figure 9: Lista zadań: stało

4 Podsumowanie

Wszystkie wymagania funkcjonalne zostały w pełni zrealizowane, a aplikacja działa poprawnie na różnych urządzeniach z systemem Android.