



# Sprawozdanie końcowe

Grzybek

Opracował(a): L5 156315

## Spis treści

Praca w zespole .....	3
Za co odpowiadałem w projekcie.....	3
jaki był mój okres aktywności w projekcie .....	3
Co wykonałem w projekcie .....	3
Zespół, a praca .....	7
Jak przebiegała praca w zespole .....	7
za co w projekcie odpowiadali inni .....	7
za co w projekcie odpowiadali Project Manager'zy .....	8
za co w projekcie powinni odpowiadać Project Manager'zy .....	8
za co w projekcie odpowiadali interesariusze.....	8
za co w projekcie powinni odpowiadać interesariusze .....	8

## Praca w zespole

### ZA CO ODPOWIADAŁEM W PROJEKCIE

W projekcie „Grzybek” byłem odpowiedzialny za stronę wizualną aplikacji czyli szatę graficzną i wygląd interfejsu użytkownika.

### JAKI BYŁ MÓJ OKRES AKTYWNOŚCI W PROJEKCIE

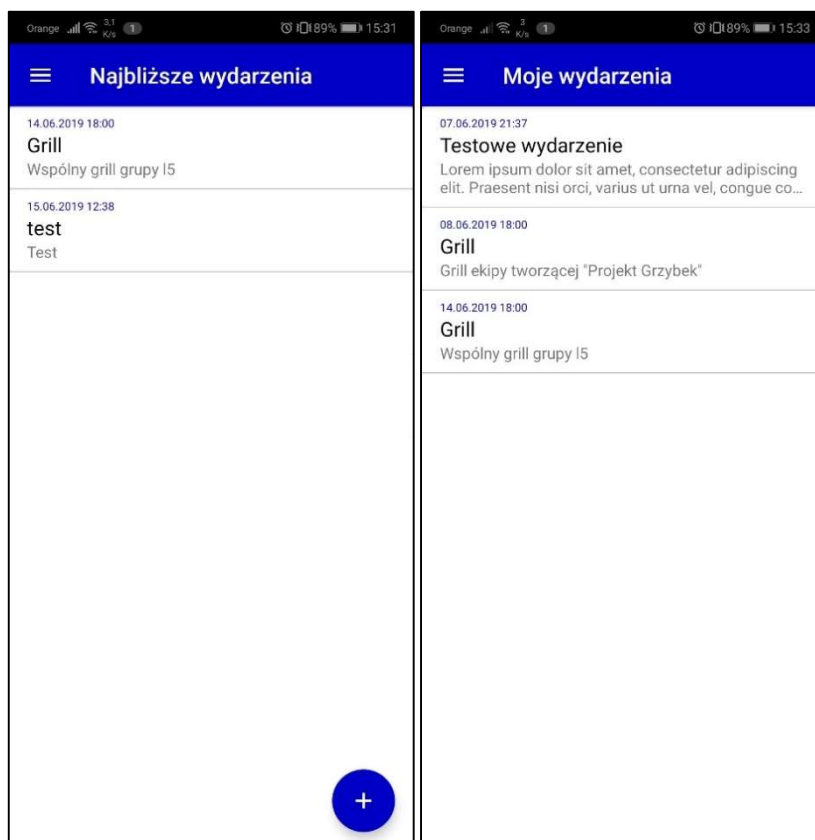
Swoją pracę rozpocząłem w połowie kwietnia. Najwięcej pracy przy projekcie wykonałem od połowy maja gdy prace zbliżały się ku końcowi i wymagane były ostateczne korekty wyglądu aplikacji.



### CO WYKONAŁEM W PROJEKCIE

Interfejs użytkownika został zaprojektowany w XML w oparciu o zasady Material Design dzięki czemu udało się uzyskać minimalistyczny oraz spójny interfejs przystosowany do obsługi na urządzeniach mobilnych. W wyglądzie dominuje kolor biały z ciemnoniebieskimi akcentami.

### Ekran główny aplikacji i ekran „Moje wydarzenia”



Na ekranie głównym znajduje się lista nadchodzących wydarzeń. Do utworzenia listy wykorzystano widget *RecyclerView* dostępny w środowisku Android Studio. Do obsługi wyświetlania obiektów listy utworzono specjalny adapter dzięki czemu udało się dostosować wygląd do wymogów aplikacji.

Kliknięcie w element listy przenosi użytkownika do ekranu ze szczegółami wydarzenia. Przycisk w prawym dolnym rogu ekranu odpowiada za przechodzenie z ekranu głównego do ekranu tworzenia wydarzenia.

Ekran „Moje wydarzenia” tak jak ekran główny wykorzystuje do działania widget *RecyclerView* wraz z dedykowanym adapterem. Na ekranie wyświetlane są wydarzenia, których organizatorem jest zalogowana osoba.

Dla każdego elementu zrealizowano wyświetlanie daty wydarzenia, tytułu oraz krótkiego opisu spotkania.

Wygląd poszczególnych elementów listy zdefiniowany jest w pliku `event_item_itemview.xml` i prezentuje się następująco:



```
1. package com.example.grzybekapk.view
2.
3. import android.support.v7.widget.RecyclerView
4. import android.view.LayoutInflater
5. import android.view.View
6. import android.view.ViewGroup
7. import android.widget.TextView
8. import com.example.grzybekapk.R
9.
10. class EventsAdapter(val eventsList: ArrayList<DataForEvents>) : RecyclerView.Adapter<EventsAdapter.ViewHolder>() {
11.     override fun onCreateViewHolder(p0: ViewGroup, p1: Int): ViewHolder {
12.         val v = LayoutInflater.from(p0?.context).inflate(R.layout.event_item_itemview, p0, false)
13.         return ViewHolder(v)
14.     }
15.
16.     lateinit var mClickListener: ClickListener
17.     fun setOnItemClickListener(aClickListener: ClickListener) {
18.         mClickListener = aClickListener
19.     }
20.
21.     interface ClickListener {
22.         fun onClick(pos: Int, aView: View)
23.     }
24.
25.     override fun getItemCount(): Int {
26.         return eventsList.size
27.     }
28.
29.     override fun onBindViewHolder(p0: ViewHolder, p1: Int) {
30.         val event: DataForEvents = eventsList[p1]
31.         p0?.textViewTitle.text = event.nameOfEvent
32.         p0?.textViewDate.text = event.getDate() + ' ' + event.getHour()
33.         p0?.textViewDescription.text = event.descriptionOfEvent
34.     }
35.
36.     inner class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView), View.OnClickListener {
37.         override fun onClick(v: View) {
38.             mClickListener.onClick(adapterPosition, v)
39.         }
40.
41.         val textViewTitle = itemView.findViewById(R.id.titleView) as TextView
42.         val textViewDate = itemView.findViewById(R.id.dateView) as TextView
43.         val textViewDescription = itemView.findViewById(R.id.descriptionView) as TextView
44.     }
45. }
```

```

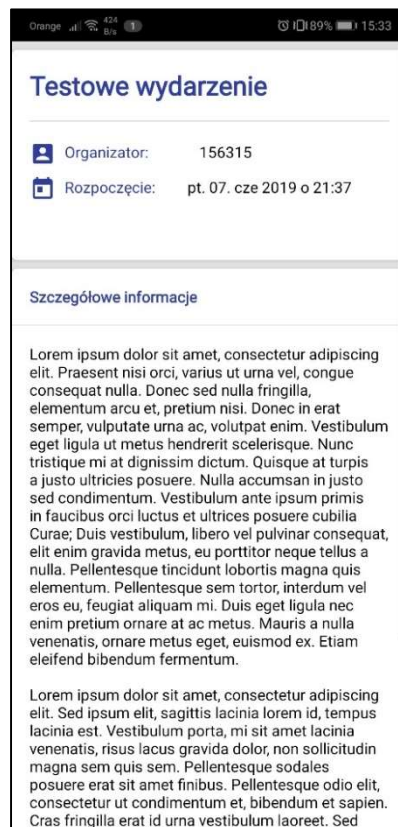
45.         init {
46.             itemView.setOnClickListener(this)
47.         }
48.     }
49.
50. }

```

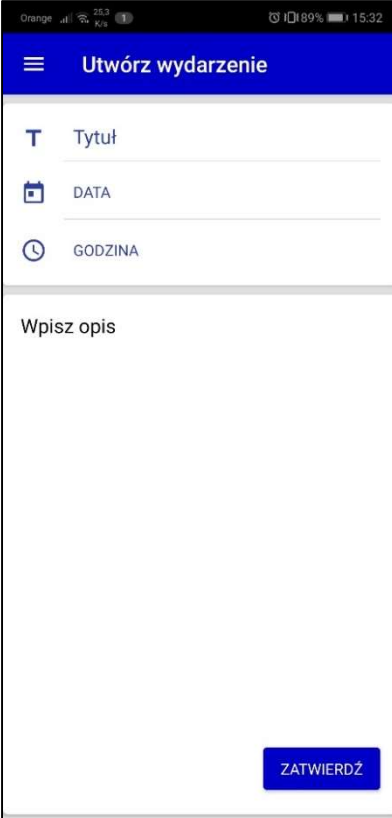
Powyższy kod pochodzi z klasy `EventsAdapter`, która definiuje wykorzystany adapter. Funkcja `getItemCount()` zwraca ilość elementów w wyświetlanej liście. Funkcja `onCreateViewHolder(p0: ViewGroup, p1: Int)` definiuje wygląd elementów w liście wykorzystując do tego szablon z pliku `event_item_itemview.xml`. W klasie `ViewHolder` pola `TextView` przypisywane są do zmiennych. W funkcji `onBindViewHolder` tworzona jest lista, do przechowywania wydarzeń.

### Ekran szczegółów wydarzenia

Na ekranie szczegółów wyświetlane są informacje o wydarzeniu czyli numer indeksu organizatora, data i godzina rozpoczęcia wydarzenia oraz opis. Ekran umożliwia przewijanie widoku jeśli wyświetlane treści nie mieszczą się na ekranie. Wykorzystane ikony pochodzą ze środowiska Android Studio.



## Ekran dodawania wydarzeń



Na ekranie tworzenia wydarzenia wyświetlane są pola do wprowadzenia tytułu, godziny i daty rozpoczęcia oraz opisu wydarzenia. Wykorzystane ikony pochodzą ze środowiska Android Studio.

Kliknięcie przycisku „Zatwierdź” zapisuje wydarzenie w bazie danych, wysyła powiadomienie oraz przenosi użytkownika do ekranu ze szczegółami. Za kod obsługujący zapisywanie wydarzeń w bazie danych stoją osoby odpowiedzialne za logikę aplikacji.

## Ekran kalendarza



Na ekranie kalendarza wyświetlany jest obecny miesiąc. Dni, w których odbywają się wydarzenia oznaczone są czerwonymi kropkami. Obecny dzień oznaczony jest okręgiem z białym obwodem a zaznaczony dzień – białym kołem.

## Zespół, a praca

### JAK PRZEBIEGAŁA PRACA W ZESPOLE

Praca przebiegała bardzo sprawnie. Podzieliliśmy zespół na dwie grupy według naszych mocnych stron i dotychczasowych umiejętności. Grupa pierwsza odpowiadała za logikę aplikacji a grupa druga za wygląd i interfejs użytkownika. Bardzo przydatna okazała się platforma GitHub, która w znacznym stopniu ułatwiła prace nad kodem aplikacji. Dodatkowym atutem okazała się możliwość rozpisania i rozdzielenia zadań w zespole, dzięki czemu każdy znał swoje zadania.

### ZA CO W PROJEKCIE ODPOWIADALI INNI

Zespół został podzielony na dwie grupy odpowiedzialne za:

- Front-end
  - Rafał Piszko – stworzenie szkieletu aplikacji, aktywności, fragmentów, bocznego menu i wiele innych
- Back-end
  - Bartłomiej Nawój – kontakt z PM’ami integracja z Firebase, integracja z Fabric, powiadomienia push
  - Mateusz Moch – logika aplikacji, pobieranie danych z Firebase, obsługa logowania po stronie aplikacji i wiele innych

- Marcin Pazowski – logowanie po stronie Firebase, CAS, utworzenie repozytorium
  - Artur Przysaś – kontakt z PM’ami, dodawanie wydarzeń
- Grafiki
  - Maciej Miśkowiec

#### ZA CO W PROJEKCIE ODPOWIADALI PROJECT MANAGER’ZY

Project managerzy odpowiadali jedynie za ogólną wizję projektu czyli pomysł na utworzenie aplikacji do organizacji wydarzeń na studenckiej altanie.

#### ZA CO W PROJEKCIE POWINNI ODPOWIADAĆ PROJECT MANAGER’ZY

Przed wszystkim project managerzy powinni być bardziej zaangażowani w projekt. Powinni odpowiadać za utworzenie i dostarczenie makiet obrazujących ogólny wygląd interfejsu aplikacji, schematów działania, oraz przemyślanych funkcjonalności. Ich zadaniem powinno być również rozliczanie nas z wykonanej pracy. W pewnym momencie można było zauważyć, że Project Managerzy zupełnie stracili zainteresowanie projektem i byliśmy zdani tylko na siebie. Aplikacje stworzyliśmy głównie w oparciu o początkowe założenia i wskazówki opiekuna projektu.

#### ZA CO W PROJEKCIE ODPOWIADALI INTERESARIUSZE

Prowadzący przedmiotu miał wpływ na ogólny przebieg projektu. Jego uwagi i sugestie były pomocne podczas tworzenia aplikacji. Zawsze służył pomocą w kwestiach technicznych. Dzięki jego inicjatywie możliwe było zaimplementowanie autoryzacji CAS. Na bieżąco kontrolował proces tworzenia aplikacji poprzez organizowanie specjalnych spotkań, na których omawialiśmy wykonane postępy.

#### ZA CO W PROJEKCIE POWINNI ODPOWIADAĆ INTERESARIUSZE

Interesariusze powinni mieć wpływ na proces tworzenia aplikacji. Ich opinie i sugestie powinny być brane pod uwagę podczas pracy nad projektem. Aktywność interesariusza z WEil była odpowiednia, natomiast interesariusz WZ powinien być bardziej zaangażowany.