

*Technologie Obiektowe 2*  
*Zespół: Aniołki Charliego*  
*poniedziałek, godz. 11:15*

*Przemysław Jabłecki*  
*Arkadiusz Kraus*  
*Mateusz Naróg*  
*Filip Ślęzyk*

# Projekt - Kalendarz

## Opis projektu

Celem projektu jest stworzenie aplikacji “Kalendarz”, zbliżonej wyglądem i funkcjonalnością do aplikacji typu Google Calendar lub Calendar firmy Apple.

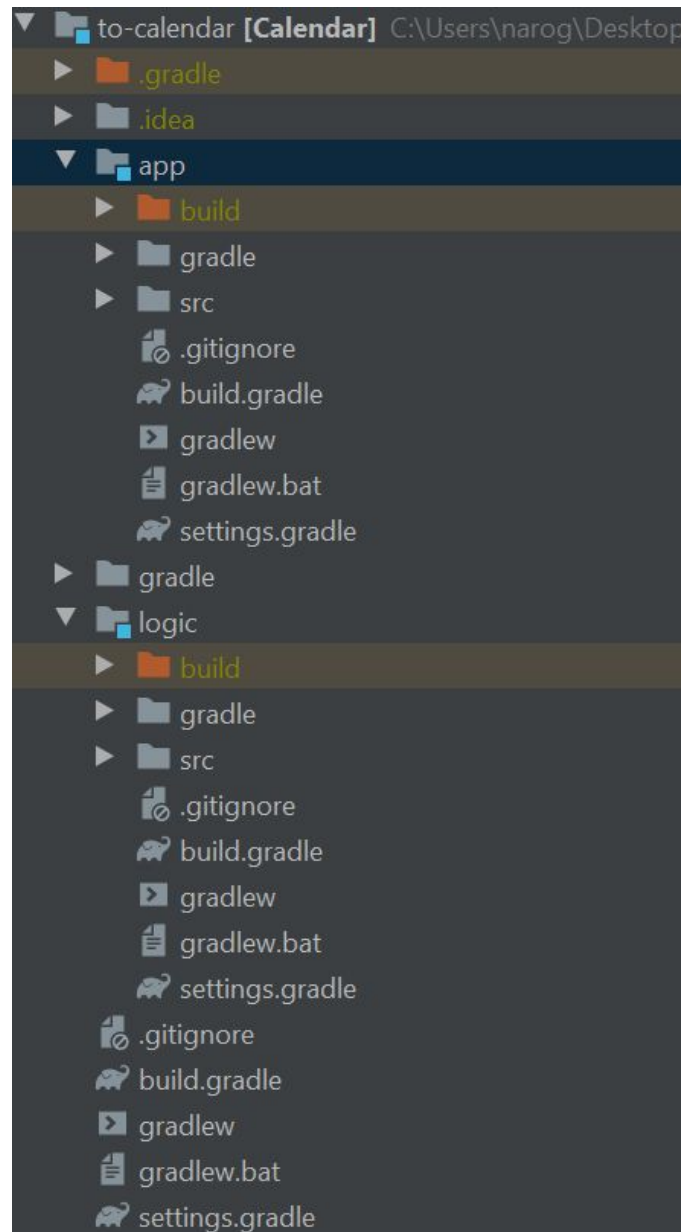
Aplikacja ma zapewniać:

- możliwość dodawania i kategoryzacji kalendarzy (np. praca, dydaktyka, zdrowie),
- jeden użytkownik może mieć przypisane wiele kalendarzy,
- wysyłanie powiadomień o nadchodzących wydarzeniach,
- rozwiązywanie konfliktów,
- widok dzienny, tygodniowy, miesięczny,
- z wydarzeniem można skojarzyć miejsce,
- wyszukiwanie.

Aplikacja zostanie stworzona w języku Java, przy pomocy frameworka JavaFX i narzędzia Gradle.

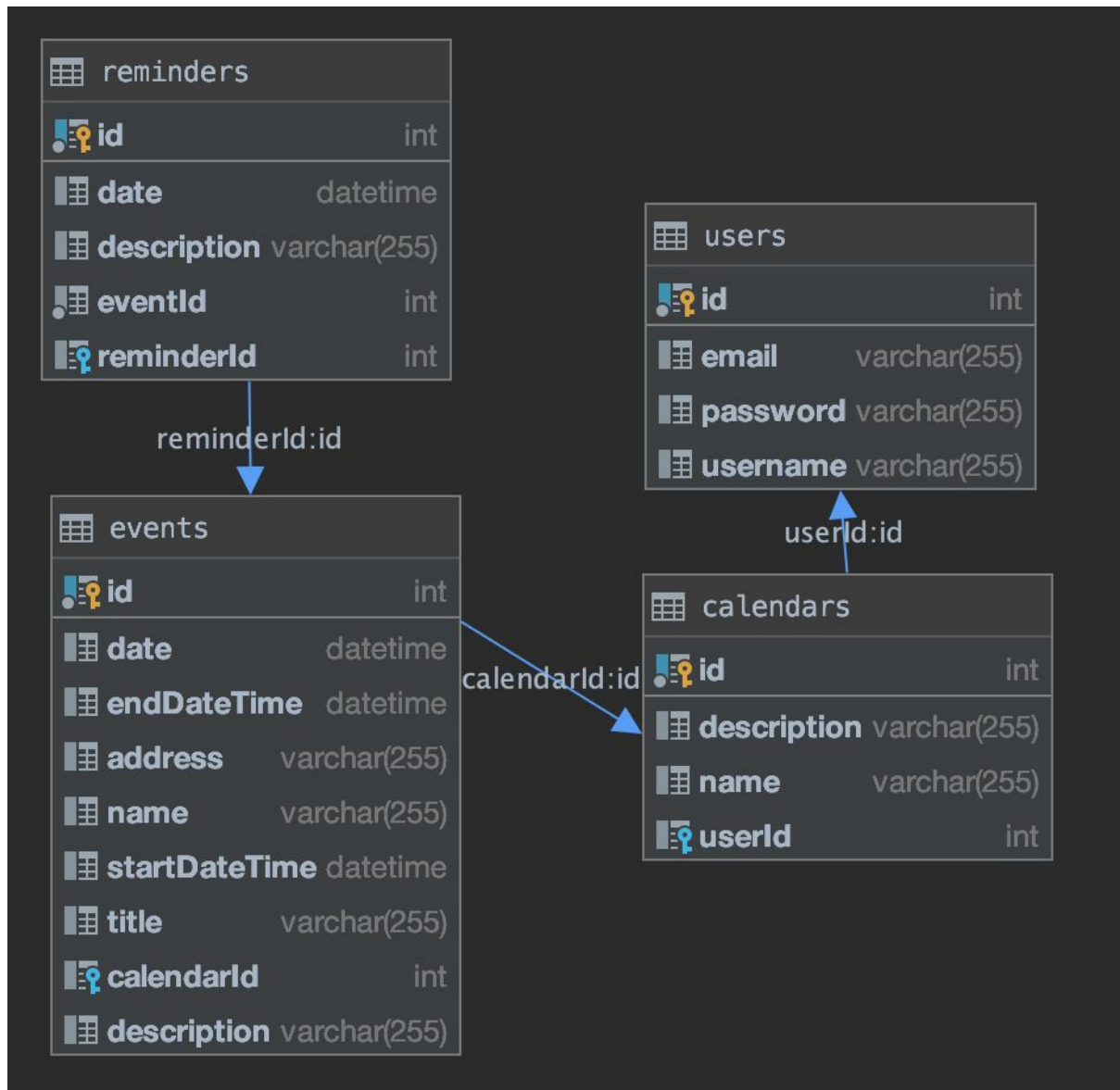
## Struktura projektu

Projekt jest podzielony na 2 części - logikę działania kalendarza oraz interfejs użytkownika, co jest odwzorowane za pomocą modułów narzędzia Gradle.



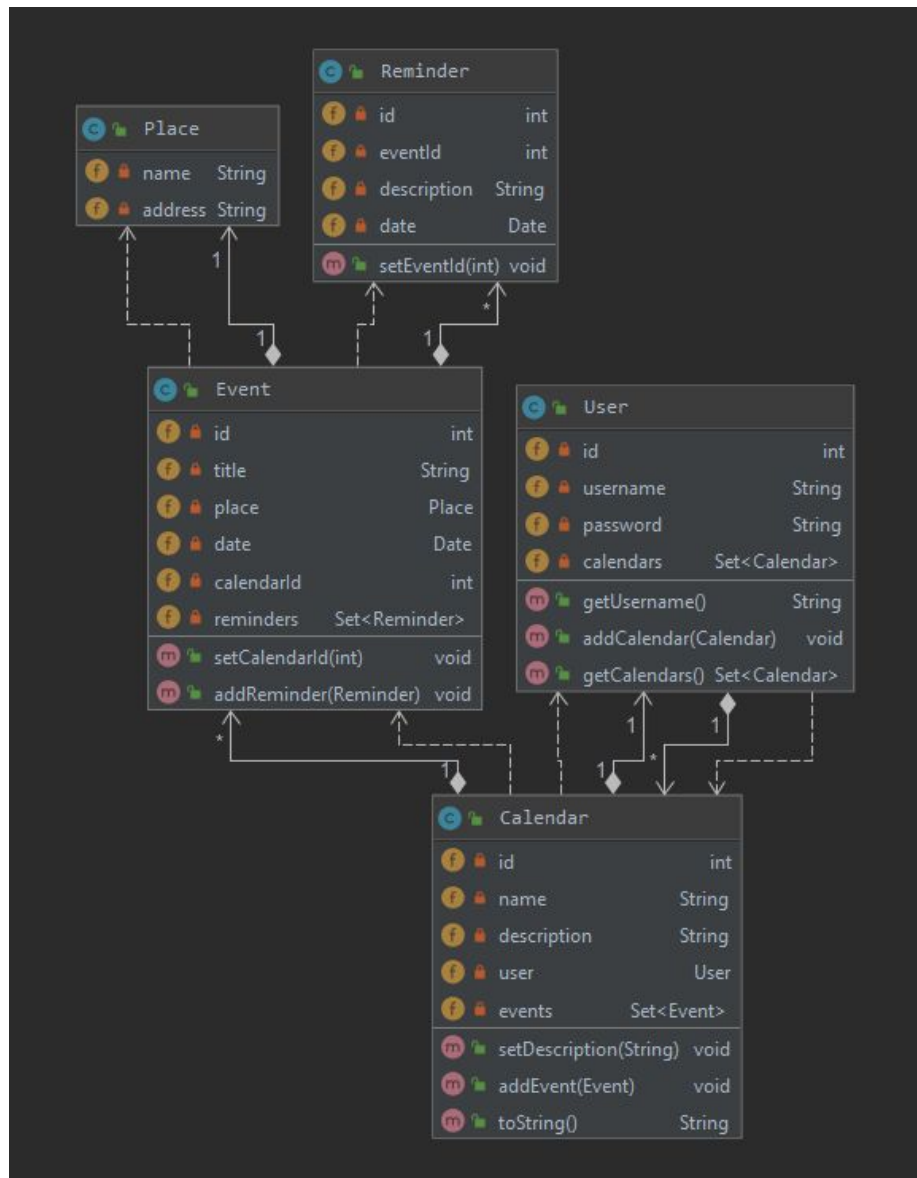
## Baza danych

Do projektu została wykorzystana baza danych Azure SQL Database, dostępna zewnętrznie dla wszystkich użytkowników. Do połączenia z bazą danych użyta jest biblioteka Hibernate.



## Diagram klas modelu

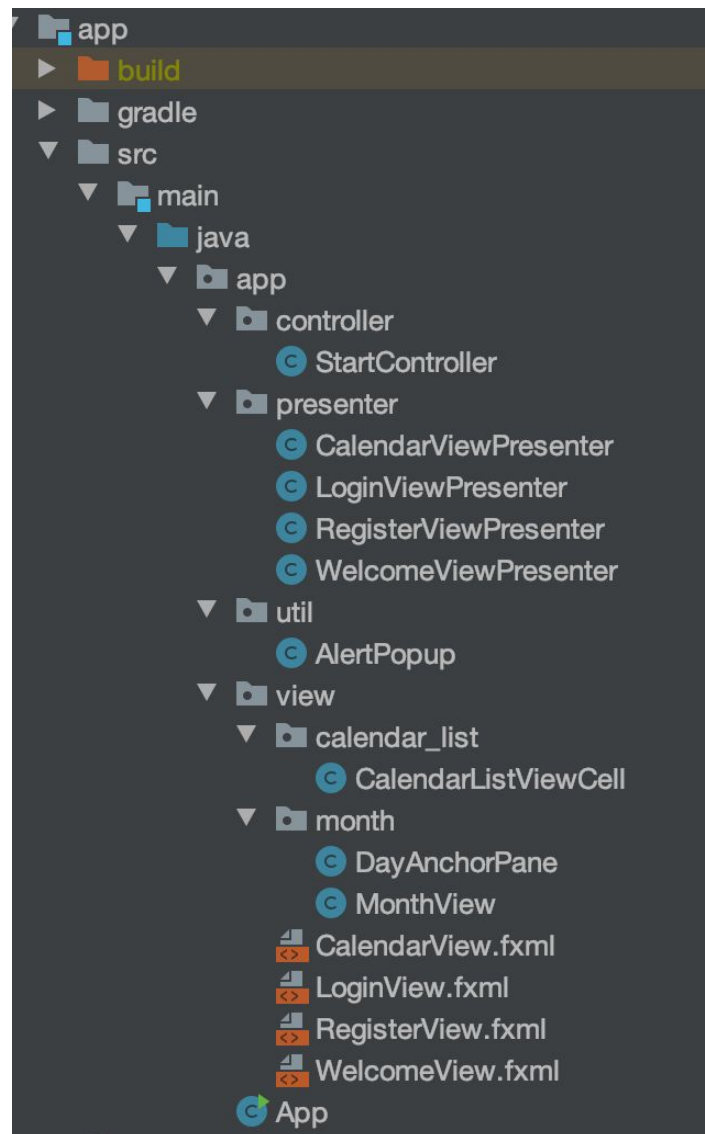
Diagram klas w dużym stopniu odpowiada schematowi bazy danych.



## Architektura aplikacji:

Aplikacja opiera się na wzorcu Model-View-Presenter. Poszczególne klasy Presenterów przypisane są do poszczególnych widoków (pliki .FXML i inne w pakiecie app.view). Poszczególne Presenter pobiera dane z Modelu dzięki klasom UserService i CalendarService (pakiet logic), a następnie dalej przekazuje do przypisanego widoku. Klasy Presenterów mają również zdefiniowaną logikę zachowania po zarejestrowaniu Eventów na obiektach View.

Uruchamianie poszczególnych Presenterów następuje w klasie StartController.



# Przebieg pracy nad projektem

## M1

### Zaimplementowane elementy i funkcjonalności aplikacji:

- stworzenie ekranu powitalnego aplikacji,
- możliwość zarejestrowania się nowego użytkownika (dodawanie osób),
- możliwość zalogowania się zarejestrowanego użytkownika.

Po zalogowaniu się:

- widok miesięczny kalendarza (aktualna data zaznaczona jest na różowo),
- możliwość wybrania obecnie przeglądanej daty,
- przycisk powrotu do aktualnej daty,
- przeglądanie listy kalendarzy użytkownika,
- dodawanie przez użytkownika nowego kalendarza,
- usuwanie kalendarza z listy danego użytkownika.

## Uruchamianie aplikacji

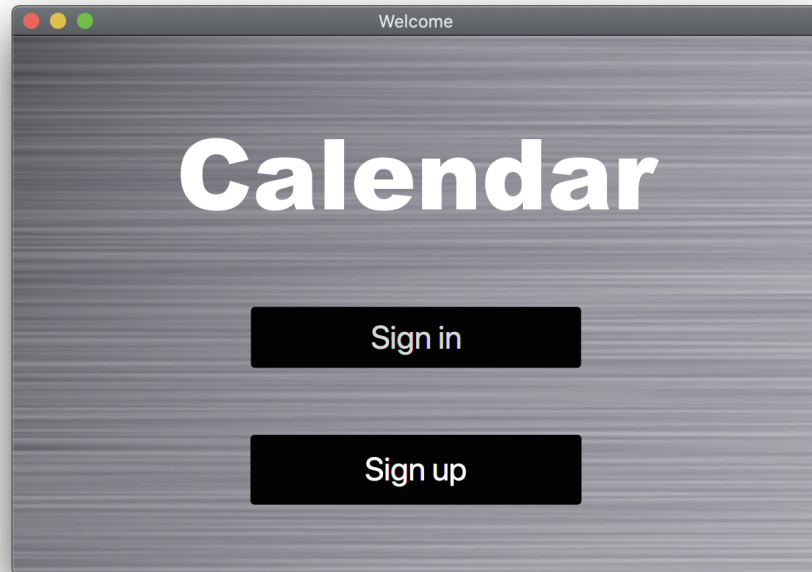
Aplikację uruchamiano przy pomocy JVM Java 11 i narzędzia Gradle w wersji 5.4. Po zaimportowaniu projektu, uruchomienie aplikacji następuje poprzez wykonanie komendy w wierszu poleceń w katalogu projektu:

**`./gradlew build :app:run`**

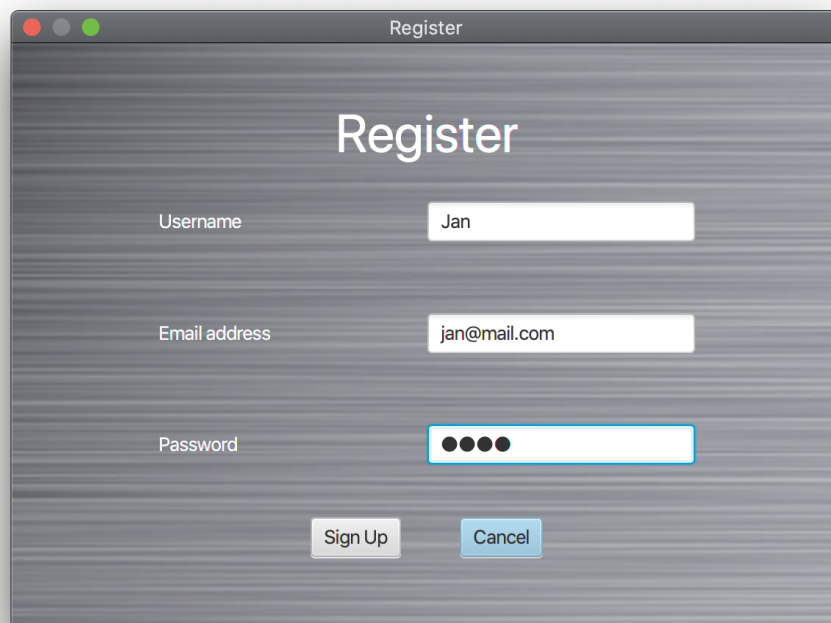
Do korzystania z aplikacji niezbędne jest połączenie z Internetem, ponieważ korzysta ona z zewnętrznej bazy danych.

## Demonstracja działania:

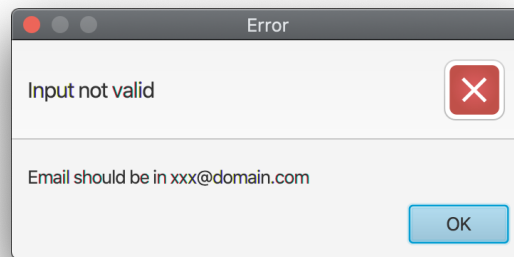
Po włączeniu aplikacji ukazuje się okno powitalne.



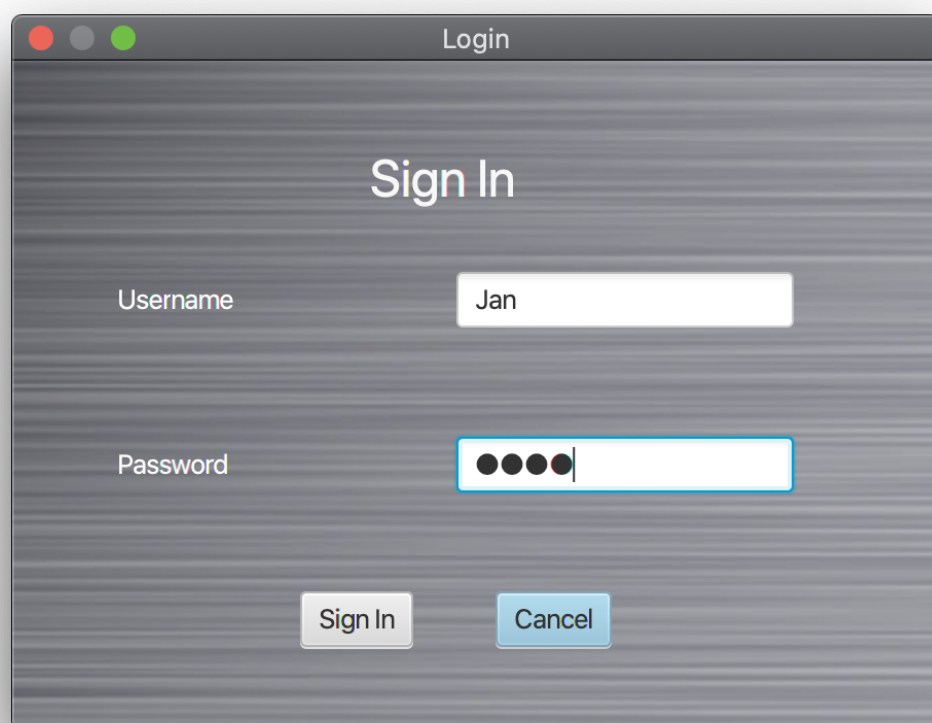
W celu skorzystania z głównej funkcjonalności kalendarza, należy się najpierw zarejestrować. Należy w tym celu kliknąć "Sign up". Pojawia się formularz rejestracyjny.

The image shows a macOS-style window titled "Register". The window has a dark gray background with a subtle horizontal line pattern. In the center, the word "Register" is displayed in a large, bold, white sans-serif font. Below the title, there are three input fields. The first field is labeled "Username" and contains the text "Jan". The second field is labeled "Email address" and contains the text "jan@mail.com". The third field is labeled "Password" and contains four black dots, indicating a password field. Below the input fields, there are two buttons: a light gray button labeled "Sign Up" and a light blue button labeled "Cancel". The window has standard macOS window controls (red, yellow, and green buttons) in the top-left corner.

Jeśli wprowadzono nieprawidłowe dane, na przykład niepoprawny adres email, pojawia się stosowny komunikat:

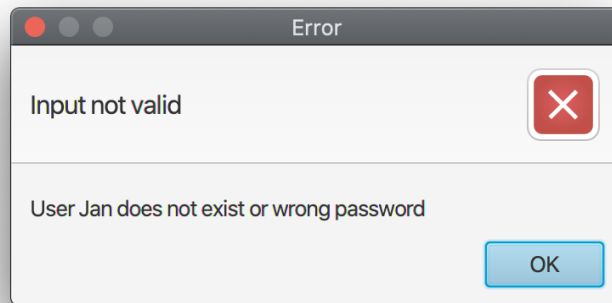


Po zarejestrowaniu się, można zalogować się do aplikacji.

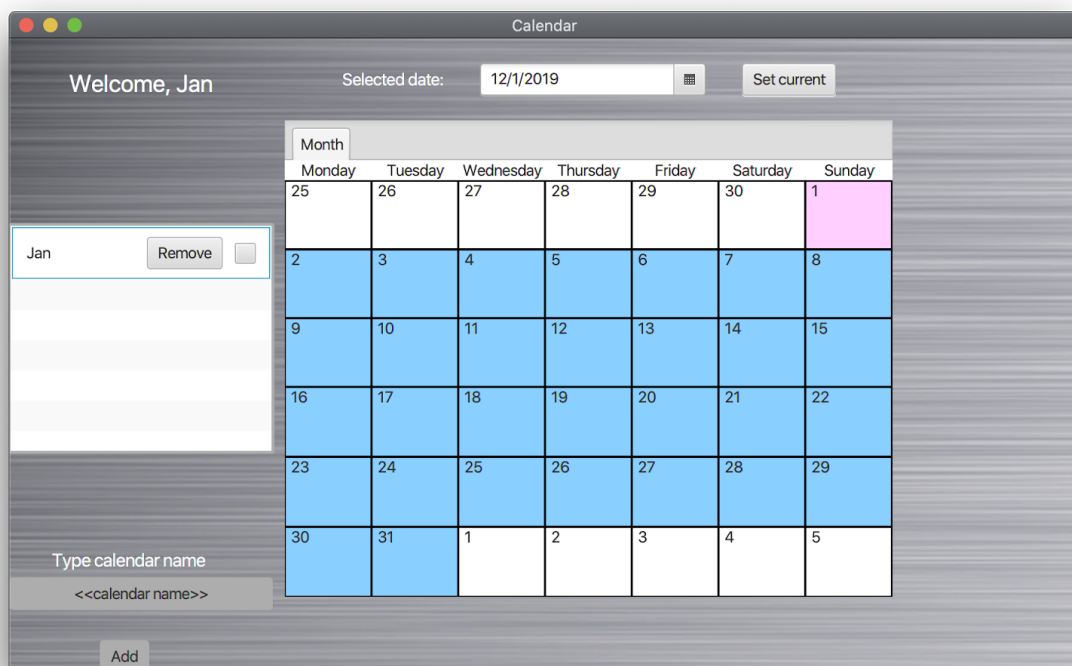




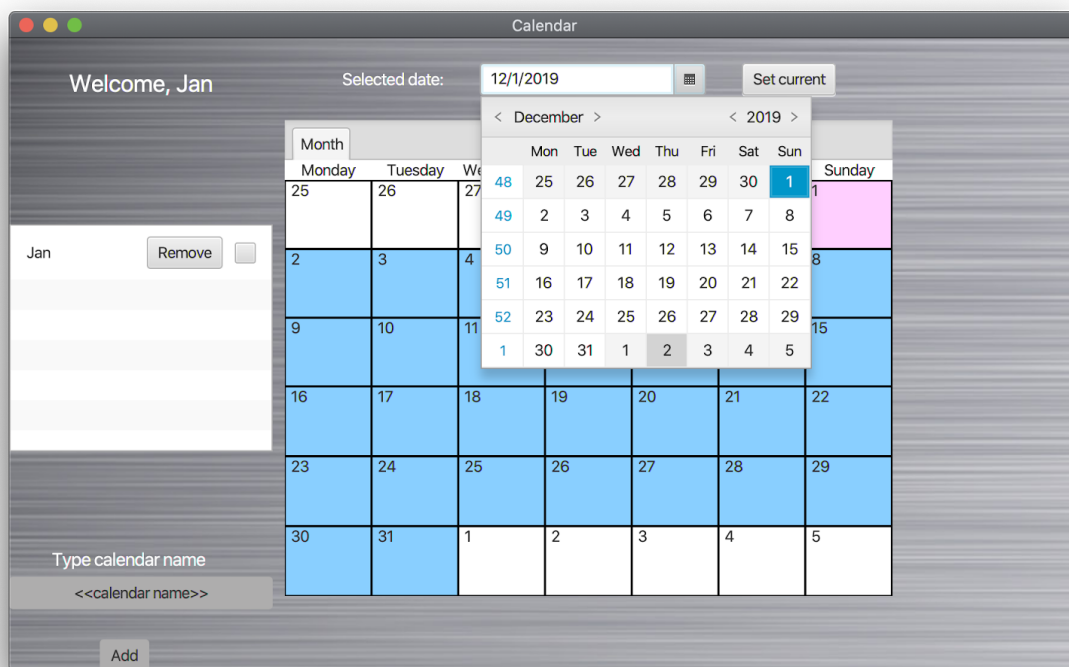
Jeśli dane są niepoprawne, pojawi się komunikat ostrzegawczy:



Po pomyślnym zalogowaniu, pojawia się główny ekran aplikacji:

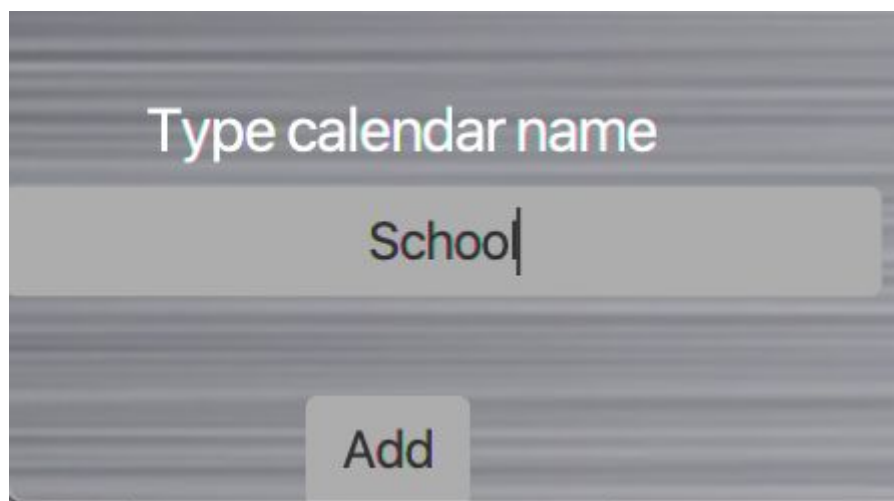


Na różowo zaznaczona jest aktualna data. Przejście do wybranej daty (miesiąca) następuje poprzez wybór jej z kontrolki na samej górze okna:

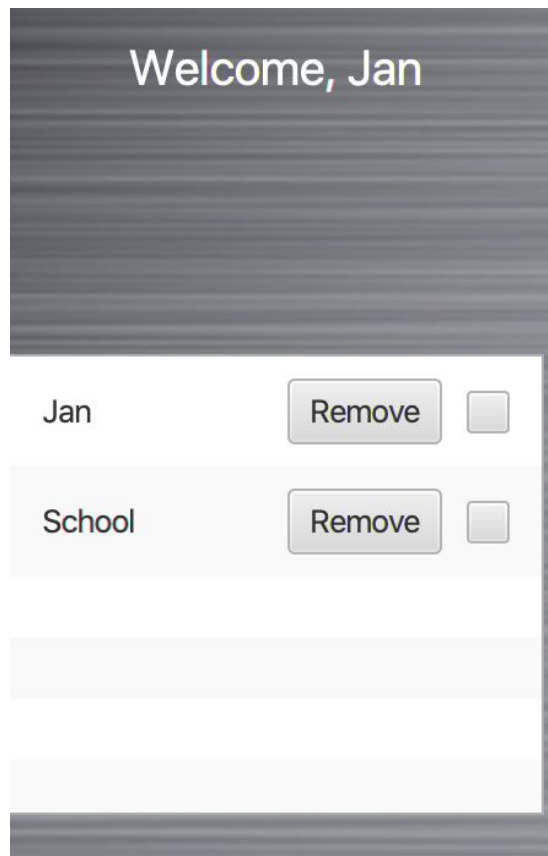


W każdej chwili można wrócić do aktualnej daty, klikając przycisk “Set current”. Można też wybierać dzień, poprzez kliknięcie na odpowiadający mu prostokąt w głównym widoku miesięcznym (w obecnym etapie projektu, może skutkować to zmianą wybranego miesiąca, jeśli dany dzień nie jest z obecnie wyświetlanego miesiąca).

Po lewej stronie okna zlokalizowano listę kalendarzy zalogowanego użytkownika. Dodanie nowego kalendarza następuje poprzez wpisanie jego nazwy w polu “<<calendar name>>”, a następnie kliknięcie przycisku “Add”.



Nowo dodany kalendarz zostaje wyświetlony na liście:



W kolejnej iteracji, do danego kalendarza będzie można przypisać wydarzenia, które ukażą się w głównym widoku po zaznaczeniu checkboxa. Kalendarz można usunąć, klikając na przycisk “Remove”.

### Przykładowe dane do logowania:

Username: Jan  
Password: 1234

## Podział pracy

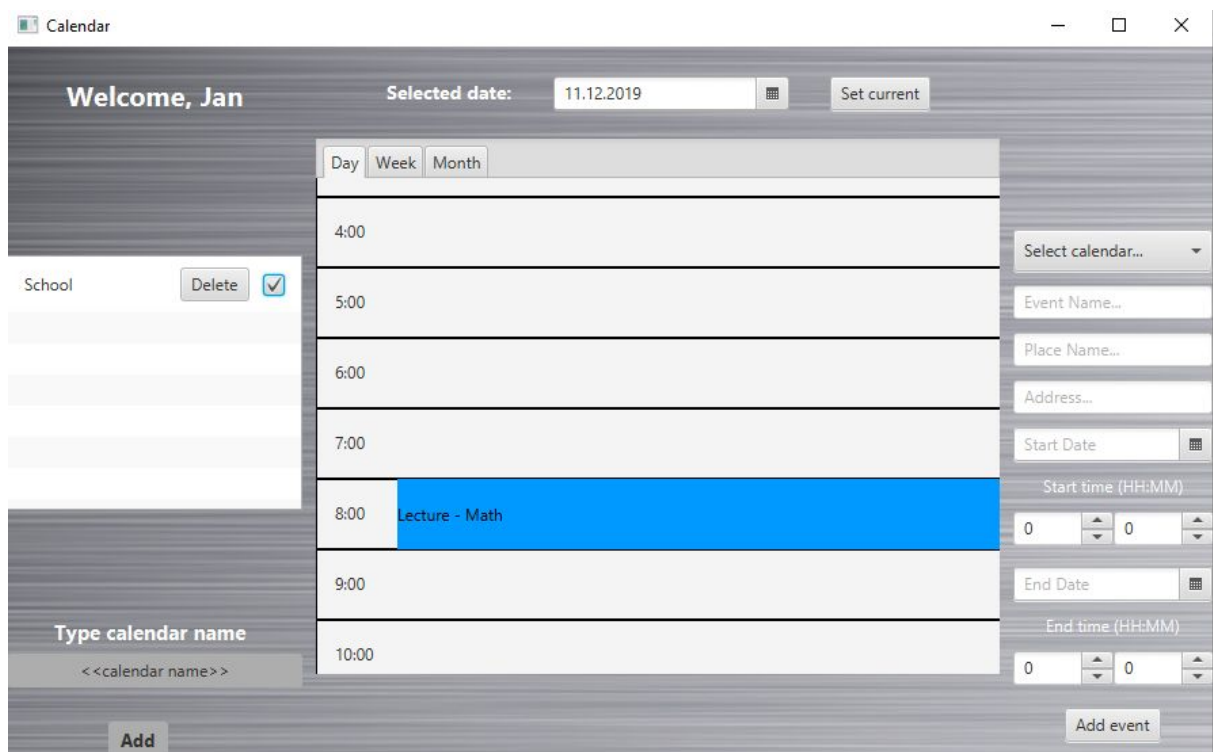
- stworzenie struktury projektu - Arkadiusz Kraus
- stworzenie bazy danych - Arkadiusz Kraus
- stworzenie modeli - Mateusz Naróg
- podpięcie Hibernate - Mateusz Naróg
- połączenie z bazą danych - Arkadiusz Kraus, Mateusz Naróg
- widok kalendarza - Filip Ślęzyk, Przemysław Jabłecki
- widoki definiujące użytkowników - Filip Ślęzyk, Przemysław Jabłecki
- zdefiniowanie prezenterów - Filip Ślęzyk, Przemysław Jabłecki

# M2

## Demonstracja działania

Obecnie kalendarz obsługuje trzy widoki - dzienny, tygodniowy, miesięczny. Dostępna jest funkcja dodawania nowych wydarzeń - w panelu z prawej strony kalendarza.

W widoku dziennym widoczna jest siatka godzin, a na niej wydarzenia skojarzone z wybranym kalendarzem. W celu pokazania wydarzeń z wybranego kalendarza, należy zaznaczyć checkbox w panelu po lewej stronie aplikacji.



W tej iteracji zaimplementowano także widok tygodniowy. Dzisiejszy dzień oznaczany jest w nim kolorem różowym, tak, jak w widoku miesięcznym.

Calendar

Welcome, Jan

Selected date: 11.12.2019

Set current

Day	Week	Month
Monday	Tuesday	Wednesday
0:00	0:00	0:00
1:00	1:00	1:00
2:00	2:00	2:00
3:00	3:00	3:00
4:00	4:00	4:00
5:00	5:00	5:00

School  ☒

Type calendar name

<<calendar name>>

Add

Select calendar...

Event Name...

Place Name...

Address...

Start Date

Start time (HH:MM)

0 0

End Date

End time (HH:MM)

0 0

Add event

Podobnie jak w przypadku kalendarza dziennego, zaznaczenie kalendarza po lewej stronie pozwala pokazać wydarzenia do niego przypisane.

Calendar

Event details

Event name: Lecture - Math

Place name: AGH

Address: Mickiewicza

Start date: 11.12.2019

Start time: 8 0

End date: 11.12.2019

End time: 9 0

Add

Selected date: 11.12.2019

Set current

5:00	5:00	5:00	5:00	5:00
6:00	6:00	6:00	6:00	6:00
7:00	7:00	7:00	7:00	7:00
Lecture - M...	8:00	Lecture	8:00	8:00
9:00	9:00	9:00	9:00	9:00
10:00	10:00	10:00	10:00	10:00
11:00	Lab	11:00	11:00	11:00

Select calendar...

Event Name...

Place Name...

Address...

Start Date

Start time (HH:MM)

0 0

End Date

End time (HH:MM)

0 0

Add event

Kliknięcie na kafelek wydarzenia pozwala wyświetlić jego detale. W kolejnej iteracji prawdopodobnie to okienko będzie również pozwalało na edycję szczegółów wydarzenia.

**Welcome, Jan**

Selected date: 11.12.2019 [Calendar icon] [Set current]

Day Week Month

0:00

1:00

2:00

3:00

4:00

5:00

6:00

School [Delete] [✓]

Appointments [Delete] [ ]

Type calendar name

Appointments

Add

Calendar Appoint... [v]

Appointment1

Space

Mickiewicza

11.12.2019 [Calendar icon]

Start time (HH:MM)

2 0

11.12.2019 [Calendar icon]

End time (HH:MM)

4 0

Add event

Dodawanie nowego wydarzenia polega na uzupełnieniu jego informacji w panelu po prawej stronie. Należy wybrać kalendarz do którego zapisujemy wydarzenie, podać nazwę wydarzenia, miejsce, adres, dni oraz godziny rozpoczęcia i zakończenia wydarzenia

**Welcome, Jan**

Selected date: 11.12.2019 [Calendar icon] [Set current]

Day Week Month

0:00

1:00

2:00 Appointment1

3:00

4:00

5:00

6:00

School [Delete] [✓]

Appointments [Delete] [✓]

Type calendar name

Appointments

Add

Calendar Appoint... [v]

Appointment1

Space

Mickiewicza

11.12.2019 [Calendar icon]

Start time (HH:MM)

2 0

11.12.2019 [Calendar icon]

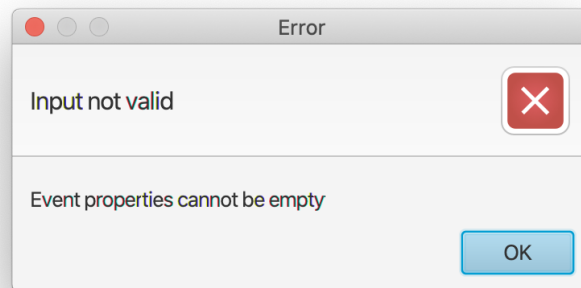
End time (HH:MM)

4 0

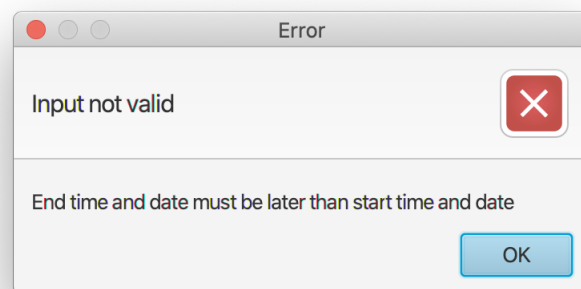
Add event

Po dodaniu nowego wydarzenia pokazuje ono się automatycznie w poszczególnych widokach (jeśli kalendarz jest wybrany).

W przypadku nieuzupełnienia jakichś danych wydarzenia pokazywany jest komunikat błędu.



Podobnie, jeśli wybrana data zakończenia wypada przed datą rozpoczęcia, sygnalizowany jest błąd.



Aktualnie jest ograniczenie, że tworzone eventy muszą zaczynać się i kończyć tego samego dnia.

## Podział pracy

- dodanie elementów reaktywnego programowania RxJava - Arkadiusz Kraus, Przemysław Jabłecki, Filip Ślęzyk, Mateusz Naróg
- możliwość dodawania wydarzeń - Przemysław Jabłecki, Filip Ślęzyk
- widok dzienny i tygodniowy - Arkadiusz Kraus, Przemysław Jabłecki, Mateusz Naróg
- wybór kalendarzy do wyświetlenia wydarzeń - Arkadiusz Kraus
- aktualizacja schematu bazy - wszyscy
- wyświetlanie eventów - Arkadiusz Kraus, Przemysław Jabłecki, Mateusz Naróg
- wyświetlanie detali eventów - Arkadiusz Kraus, Filip Ślęzyk
- inicjalizacja Hibernate przed wyświetleniem okna aplikacji - Przemysław Jabłecki, Arkadiusz Kraus

# M3

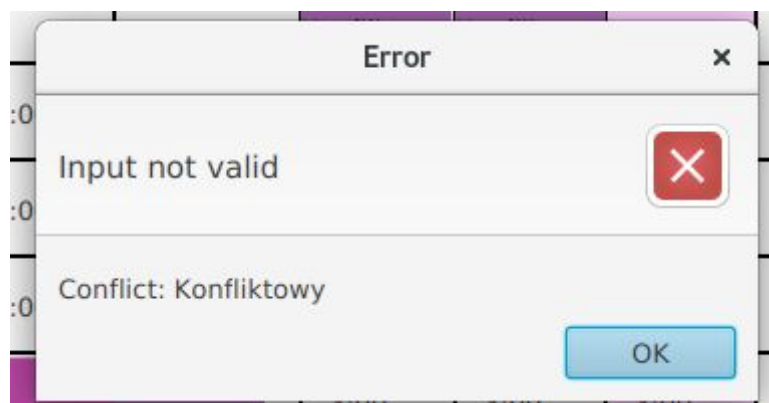
## Podsumowanie zmian

W ramach trzeciej części projektu przerefaktorowaliśmy kod, tak aby jego struktura była bardziej reaktywna. Zamiast powiadamiania teraz wszystkich miejsc o dodaniu np. kalendarza, poszczególne komponenty aplikacji same nasłuchują zmian. Dodaliśmy również następujące funkcjonalności: rozwiązywanie konfliktów, wysyłanie powiadomień, eventy całodniowe oraz kilkudniowe, możliwość edycji eventu, wyszukiwanie eventów.

## Demonstracja działania

- rozwiązywanie konfliktów

Aplikacja teraz sprawdza czy w czasie trwania wydarzenia, który chcemy utworzyć nie ma innego wydarzenia. Jeśli jest takie nie pozwala ona dodać nowego wydarzenia.



- wysyłanie powiadomień
- eventy całodniowe oraz kilkudniowe

Wydarzenia teraz mogą być dłuższe niż jeden dzień, a także mogą być oznaczone jako wydarzenie całodniowe.



Calendar

Selected date: 1/11/2020

Set current

Search...

Search

Welcome, Jan

Day Week Month

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
0:00	0:00	Dwudnio...	bbb	bbb	0:00	0:00
1:00	1:00	1:00	1:00	1:00	1:00	1:00
2:00	2:00	2:00	2:00	2:00	2:00	2:00
3:00	3:00	3:00	3:00	3:00	3:00	3:00
4:00	4:00	4:00	4:00	4:00	4:00	4:00
5:00	5:00	5:00	5:00	5:00	5:00	5:00

Test2

Delete

✓

Test1

Delete

✓

Test3

Delete

✓

Select calendar...

Event Name...

Place Name...

Address...

All day

Start Date

Start time (HH:MM)

00

End Date

End time (HH:MM)

00

Add event

Type calendar name

<<calendar name>>

Add

- możliwość edycji oraz usuwania eventu

Okno wydarzenia zostało wzbogacone o możliwość zapisu zmian oraz usunięcia wydarzenia.

Event details

Event details

Event name

Dwudniowy

Place name

Test

Address

test

All day

Start date

1/7/2020

Start time

30

End date

1/8/2020

End time

10

Description

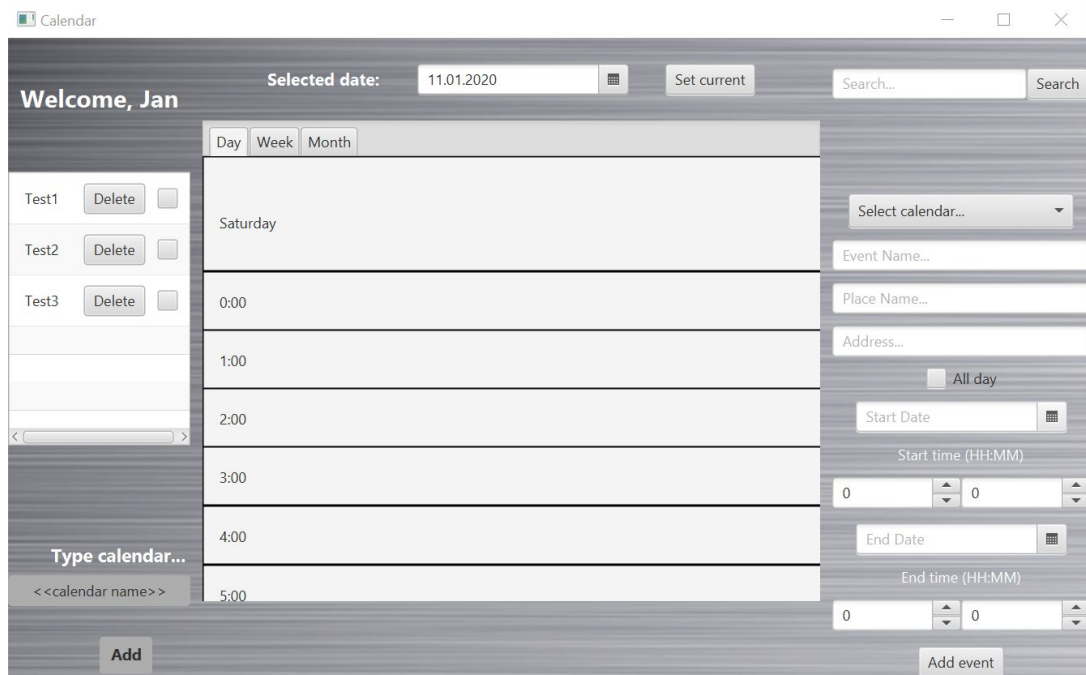
Save

Remove

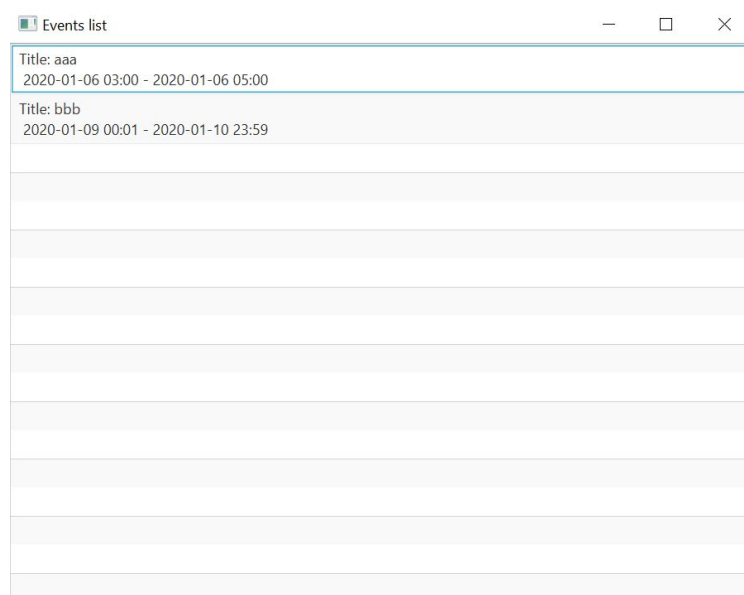
- wyszukiwanie eventów

W prawym górnym rogu pojawiło się nowe pole, w którym można wpisywać dowolny tekst, który będzie użyty do wyszukiwania eventów. Przy wyszukiwaniu brane są pod uwagę następujące pola eventu:

1. tytuł eventu,
2. nazwa miejsca,
3. adres,
4. opis eventu.

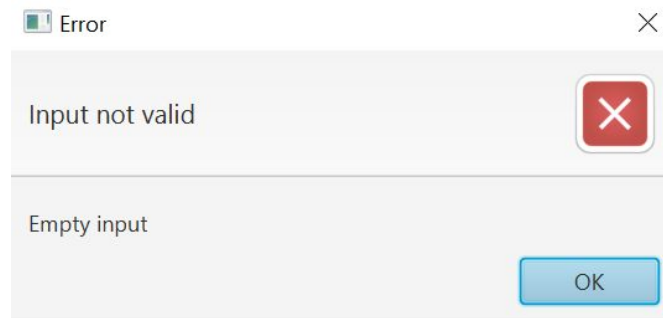


Następnie otwiera się nowe okno z listą eventów pasujących do podanego parametru.



Po naciśnięciu na wybrany event otwiera się okno ze szczegółami eventu.

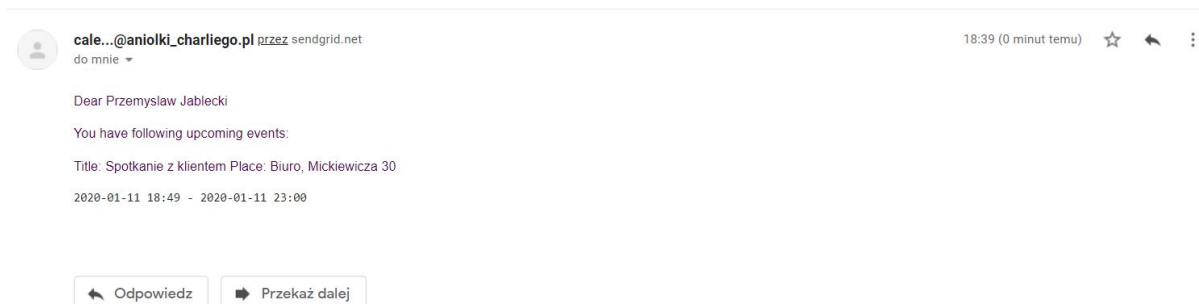
W przypadku braku wpisanego tekstu i próby wyszukania otrzymujemy następujący komunikat:



## Konfiguracja demona notyfikacji

1. Należy zdefiniować zmienną środowiskową o nazwie "API\_KEY", przechowującą klucz api SendGrida, przykładowa wartość klucza:  
"SG.XfGBROkrT2ilakHUMppQFg.F37JS4B9ZEDtO7qIRZXg\_qe0ytADIIICA7pAJ-\_BHLrg"
2. Demon jest uruchamiany jako osobny proces z poziomu gradle: "build run to-calendar:notifications"
3. Domyślnie demon wysyła powiadomienie o zdarzeniu z dziesięciominutowym wyprzedzeniem

Przykładowe powiadomienie o nadchodzącym zdarzeniu:



## Podział pracy

- ukończenie refaktoryzacji na reaktywną architekturę - wszyscy
- rozwiązywanie konfliktów - Arkadiusz Kraus
- wysyłanie powiadomień - Przemysław Jablęcki
- eventy całodniowe oraz kilkudniowe - Filip Ślęzyk
- możliwość edycji oraz usuwania eventu - Filip Ślęzyk
- wyszukiwanie eventów - Mateusz Naróg