

# AKADEMIA GÓRNICZO-HUTNICZA

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ  
KIERUNEK INFORMATYKA, III ROK, 2018/2019



INŻYNIERIA OPROGRAMOWANIA

---

## Sprawozdanie z projektu

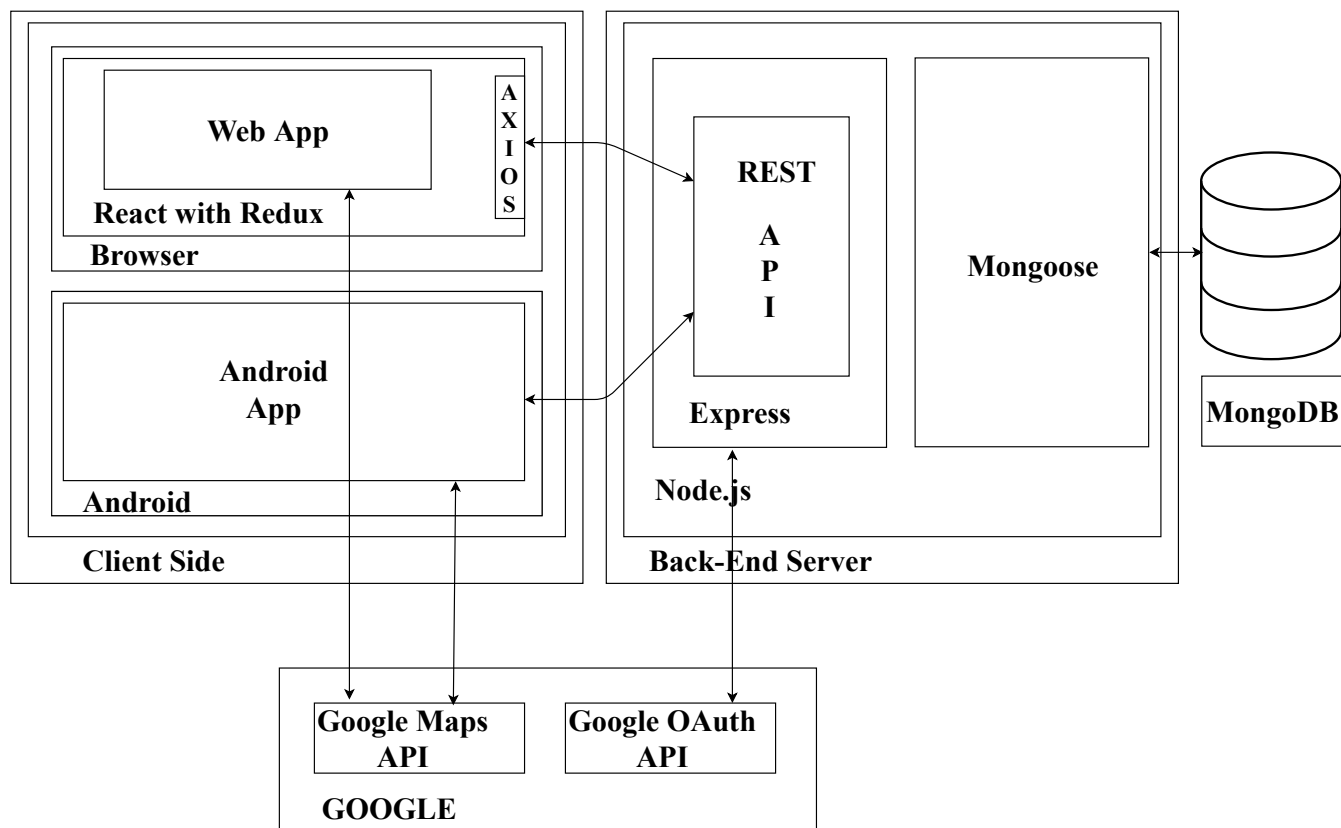
„Gdzie jest moje dziecko?”

---

Agnieszka Zadworny, Piotr Morawiecki, Tomasz Pęczak, Maciej Bielech

Kraków, 7 listopada 2018

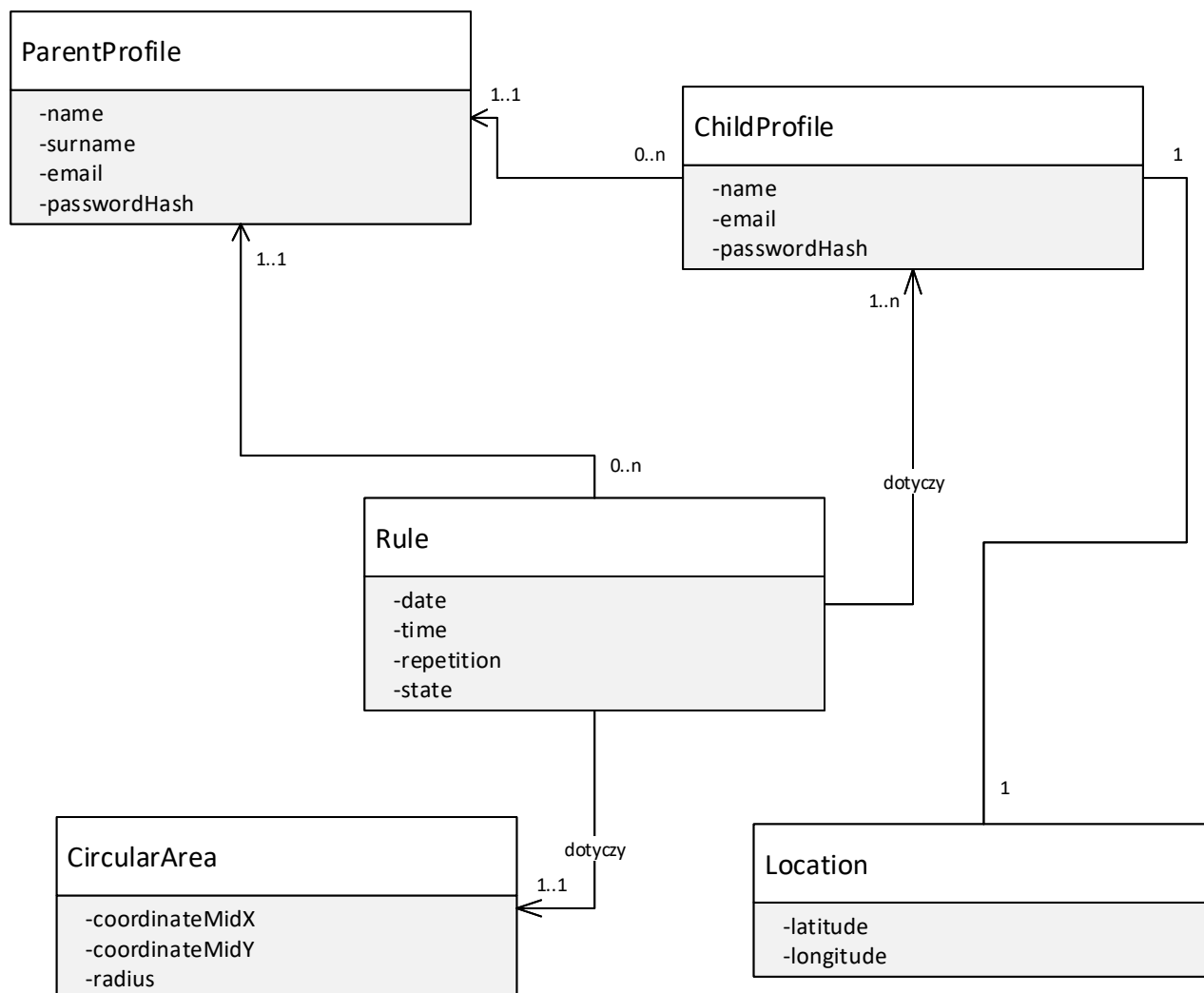
# 1 Moduły i komunikacja między nimi



Rysunek 1: Moduły i interfejsy

- Node.js - środowisko uruchomieniowe do pisania aplikacji webowych przy użyciu języka JavaScript
- Express - framework wykorzystany w celu implementacji API serwera.
- Mongoose - biblioteka ułatwiająca współpracę serwera i bazy danych MongoDB
- MongoDB - obiektowa baza danych
- Web App - aplikacja webowa działająca na przeglądarce klienckiej
- React - framework służący prostemu tworzeniu dynamicznego wyglądu aplikacji webowych
- Redux - framework ułatwiający komunikację pomiędzy komponentami React
- Android App - aplikacja mobilna działająca pod systemem android
- Google Maps API - interfejs udostępniony przez google używany przy korzystaniu z Google Maps
- Google OAuth Api - interfejs udostępniony przez google wykorzystany do logowania w aplikacji przy użyciu konta google użytkownika.

## 2 Schemat bazy danych



Rysunek 2: Schemat bazy

## 3 Projekt testów

Rodzaj testów	Co testujemy?	Technologie
Jednostkowe	metody backendu i frontendu	Jasmine
Integracyjne	komunikacja między modułami	Jasmine
Funkcjonalne	scenariusze użycia	Jasmine, Selenium
Manualne	Aplikacja mobina	manualnie
Manualne	API	postman

Tablica 1: Projekt testów

## 4 Ogólny opis systemu

### 4.1 Cel systemu

Celem naszej aplikacji, jest udostępnienie rodzicom możliwości kontroli lokalizacji swoich dzieci. Mogą oni tworzyć reguły, składające się z informacji o obszarze, w którym dziecko powinno przebywać w danym czasie. Jeśli dziecko złamie regułę,

rodzic zostaje powiadomiony poprzez wiadomość email. W przypadku wyłączenia aplikacji dziecka, rodzic zostaje powiadomiony oraz może wyświetlić ostatnią zarejestrowaną lokalizację dziecka.

## 4.2 Udziałowcy i ich cele

Użytkownikami systemu są rodzice, którzy chcą zadbać o bezpieczeństwo swoich dzieci. Udziałowcami systemu są:

- System Google OAuth, który odpowiada za autentyfikację użytkownika w aplikacji,
- System Google Maps, który odpowiada za zarządzanie i wyświetlanie lokalizacji dziecka.

Biernymi użytkownikami systemu są dzieci, których aplikacja mobilna dostarcza jedynie informację o ich bieżącym położeniu.

Udziałowiec	Cel	Priorytet
Rodzic	tworzenie reguł	wysoki
Rodzic	sprawdzanie bieżącej lokalizacji dziecka	wysoki
Dziecko	udostępnienie swojej lokalizacji	wysoki
Google OAuth	udostępnienie API logowania	niski
Google Maps	wyświetlanie lokalizacji dziecka na mapie	wysoki
Google Maps	udostępnienie API zarządzania lokalizacją	wysoki
SendGrid	wysłanie emaila do rodzica z powiadomieniem o złamaniu reguły przez dziecko	wysoki

Tablica 2: Udziałowcy i ich cele

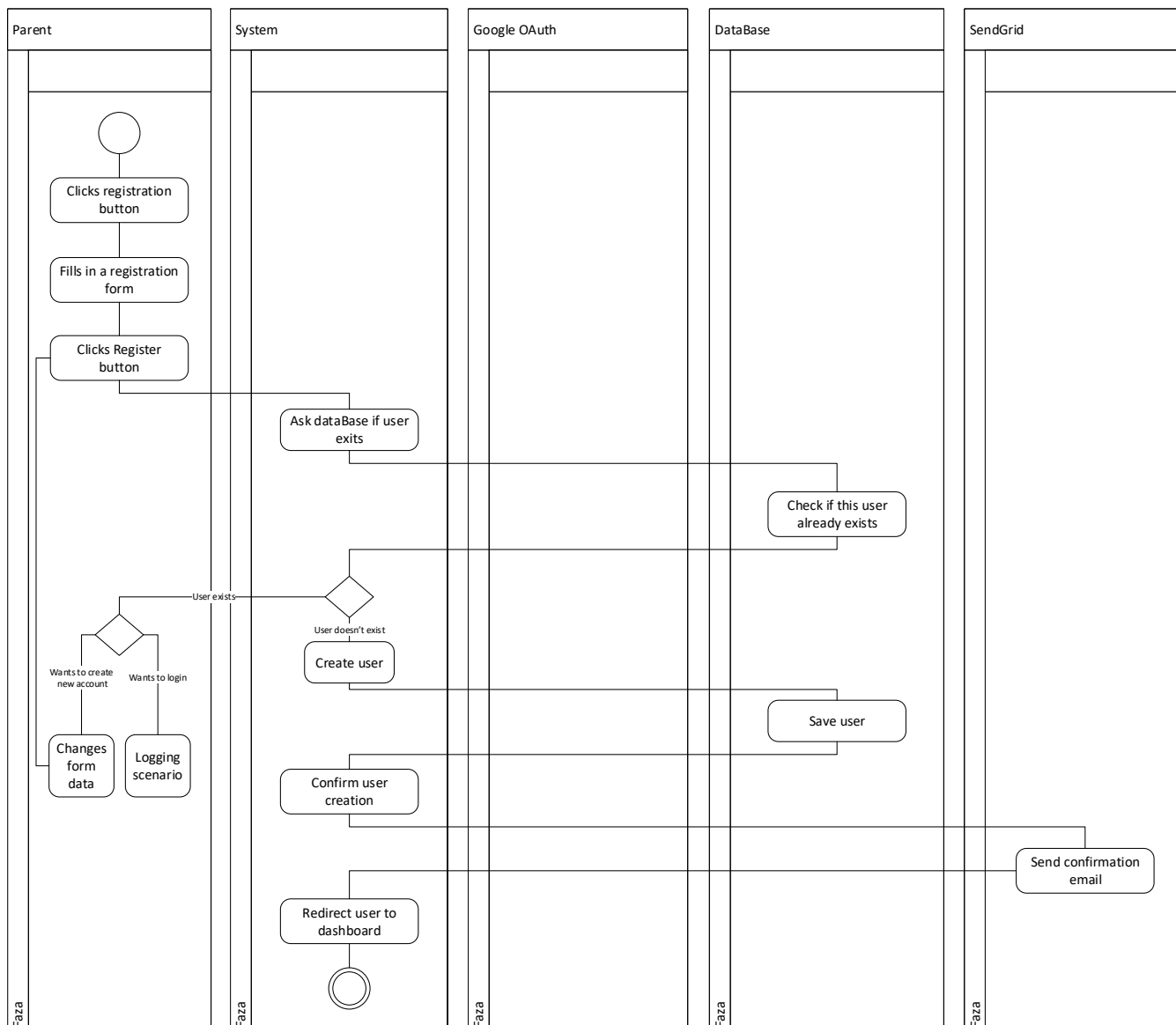
## 4.3 Granice systemu

Wyróżniamy następujących aktorów:

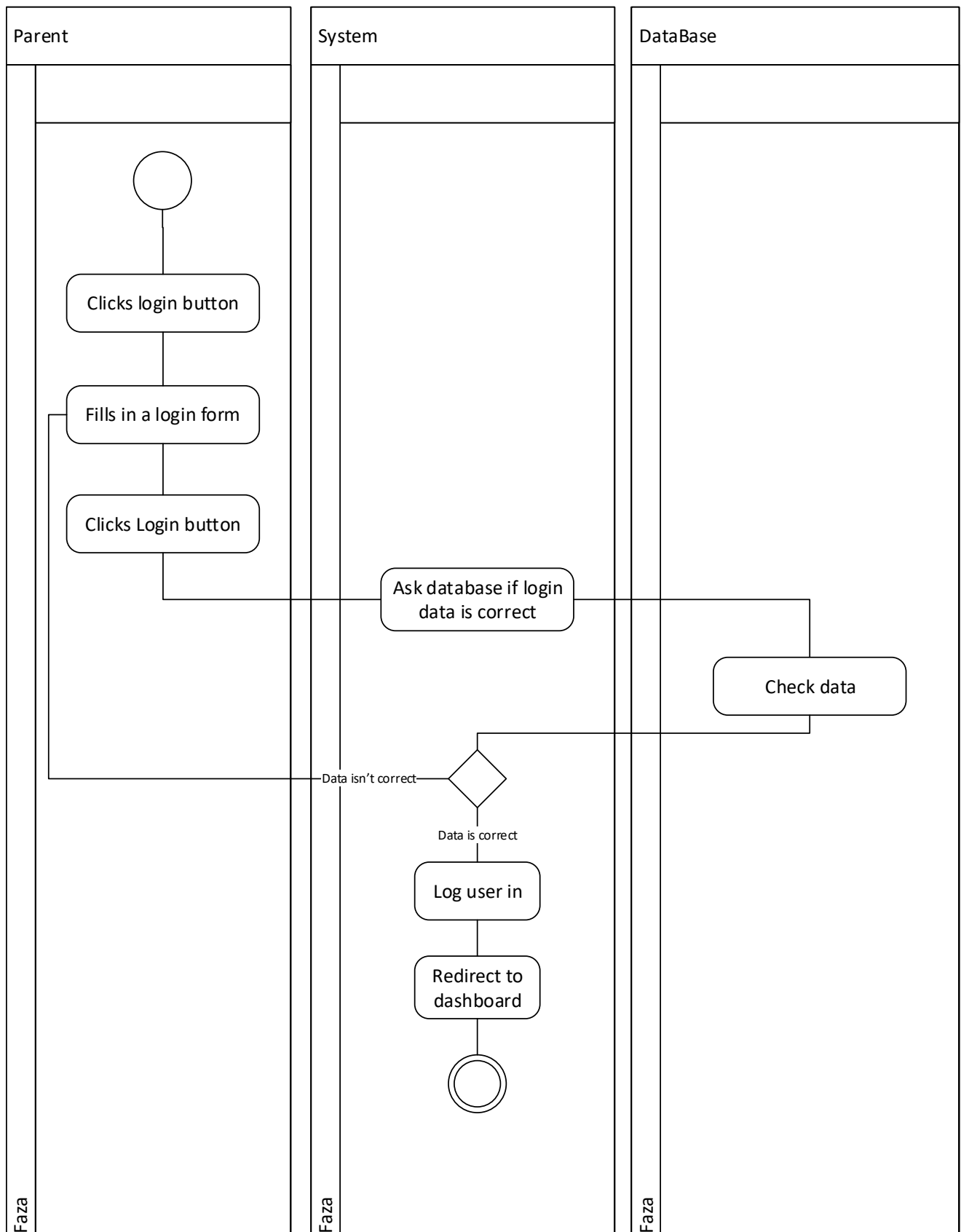
- Rodzic,
- Dziecko,
- System autentyfikacji Google OAuth,
- System Google Maps,
- System do wysyłania emaili SendGrid,
- Czas.

## 4.4 Lista możliwości

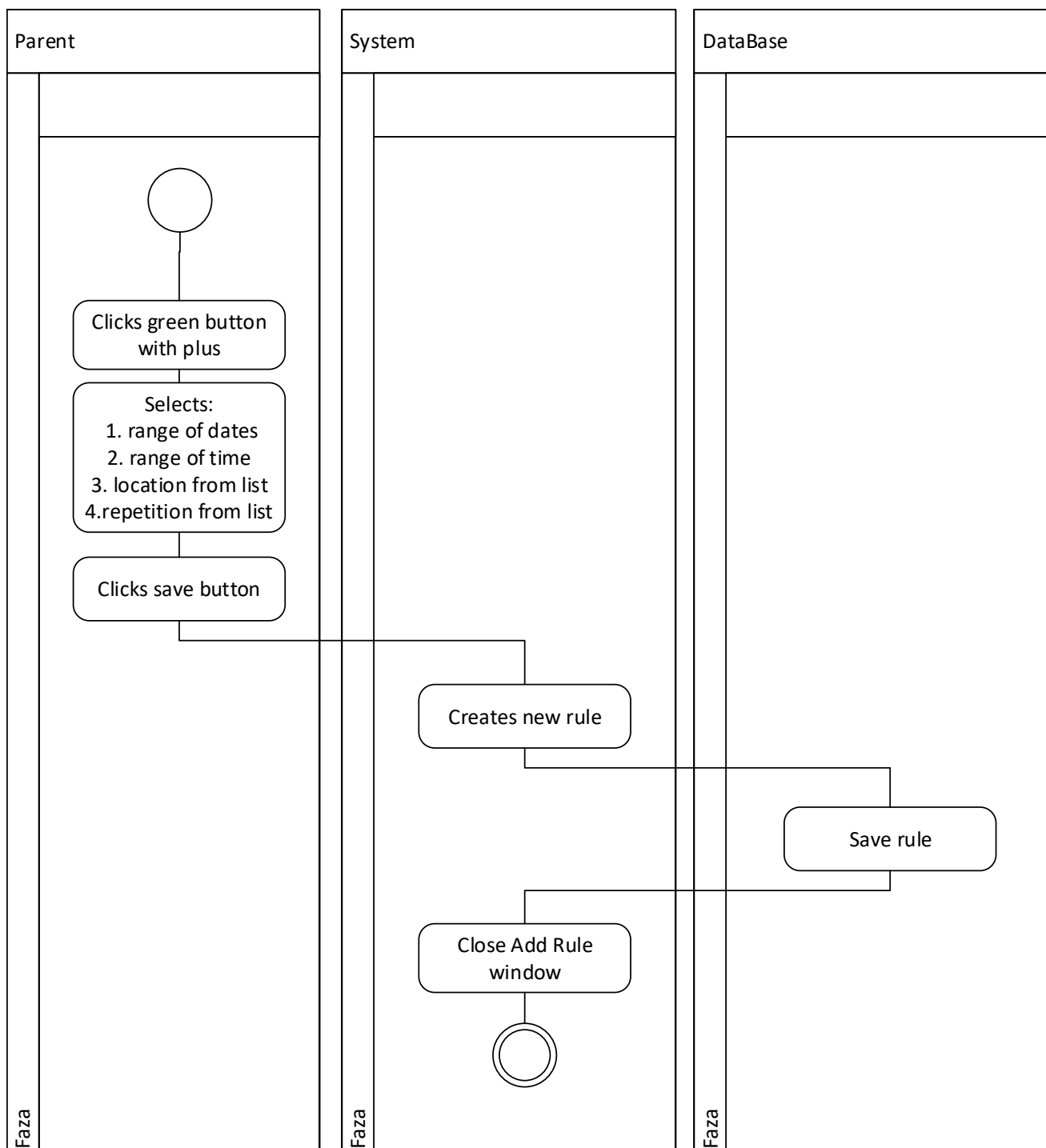
Lista możliwości została przedstawiona w postaci diagramów aktywności. Diagramy te reprezentują typowe sekwencje działań wykonywanych w systemie. W kolejnych sekcjach zostaną one zaprezentowane w postaci przypadków użycia i scenariuszy.



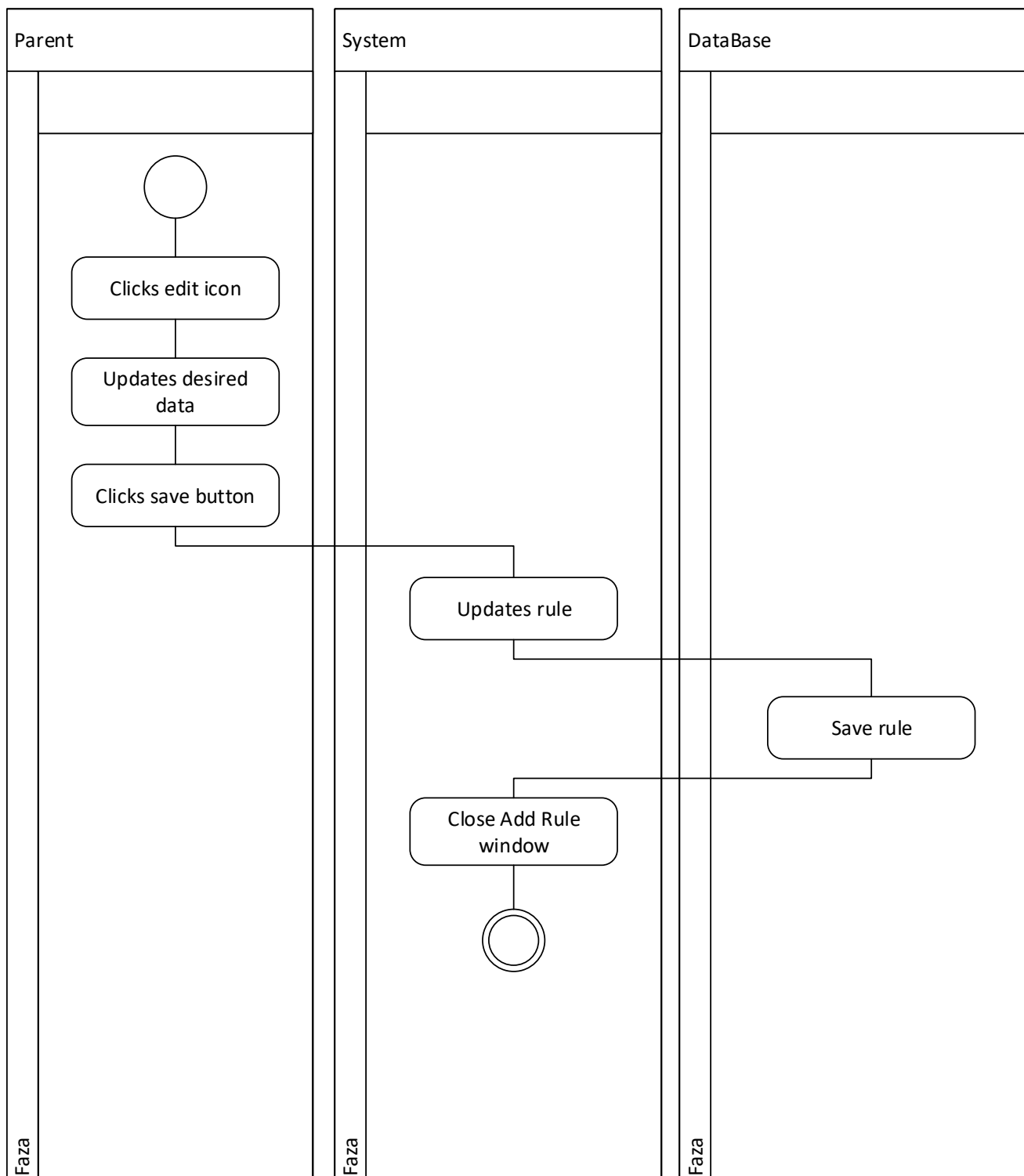
Rysunek 3: Diagram aktywności dla rejestracji



Rysunek 4: Diagram aktywności dla logowania

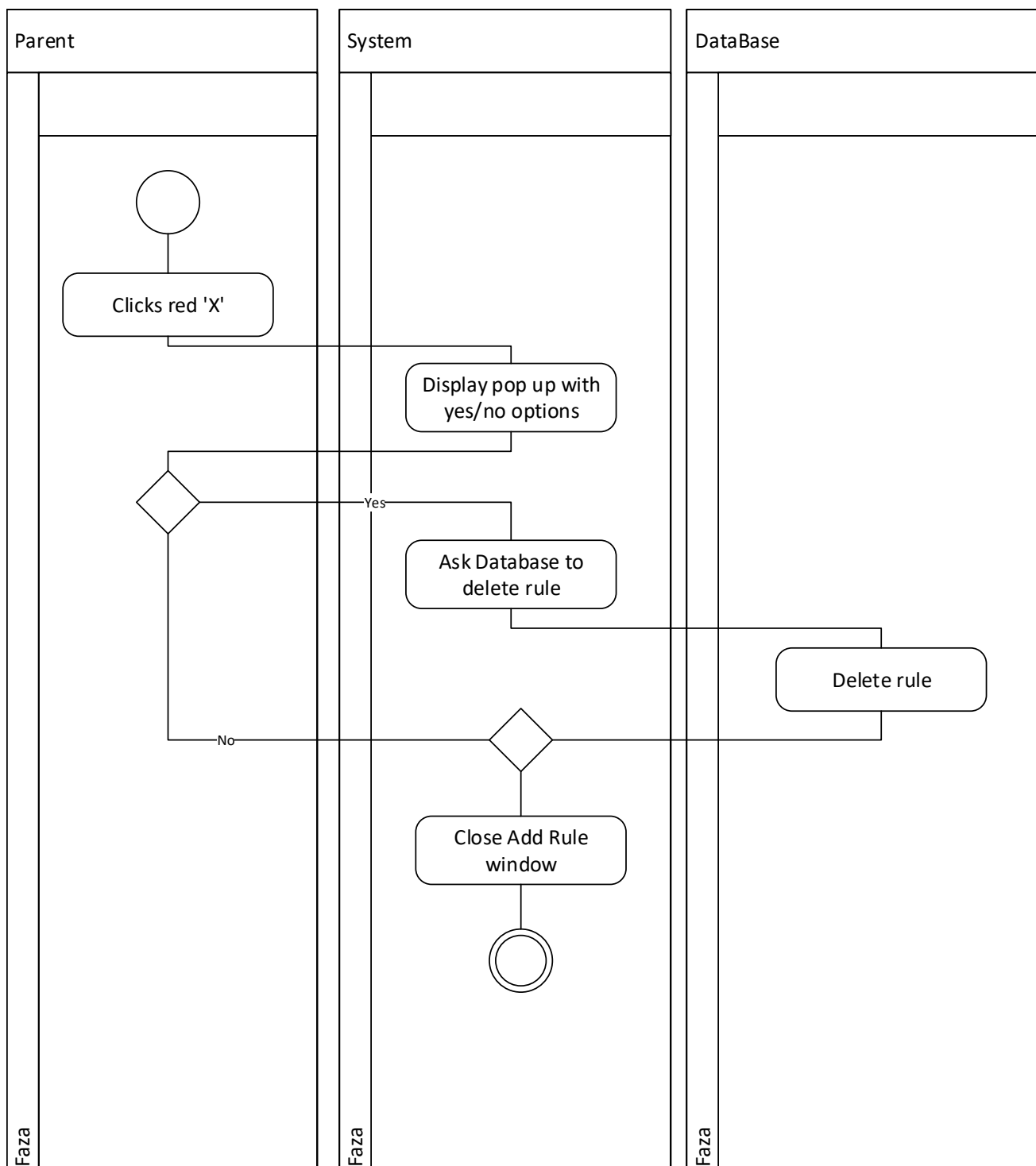


Rysunek 5: Diagram aktywności dla tworzenia nowej reguły



Rysunek 6: Diagram aktywności dla modyfikacji reguły





Rysunek 7: Diagram aktywności dla usuwania reguły

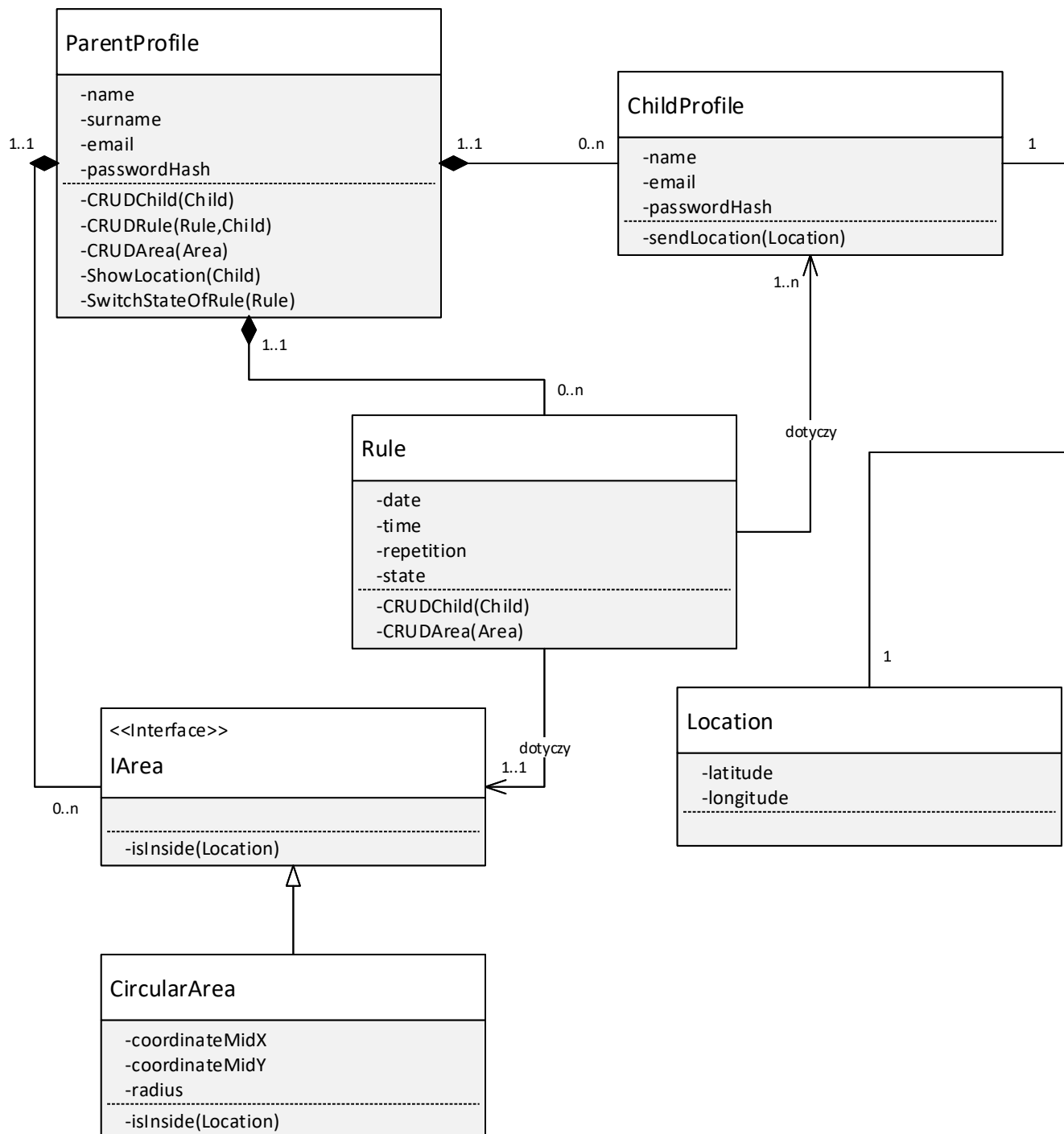
W diagramach wyróżniliśmy operacje CRUD (tworzenie, odczyt, modyfikacja i usuwanie) dla zarządzania regułami. Nie umieszczaliśmy pozostałych, gdyż realizowane są one w analogiczny sposób.

## 5 Analiza dziedziny

Klasy zidentyfikowane w aplikacji:

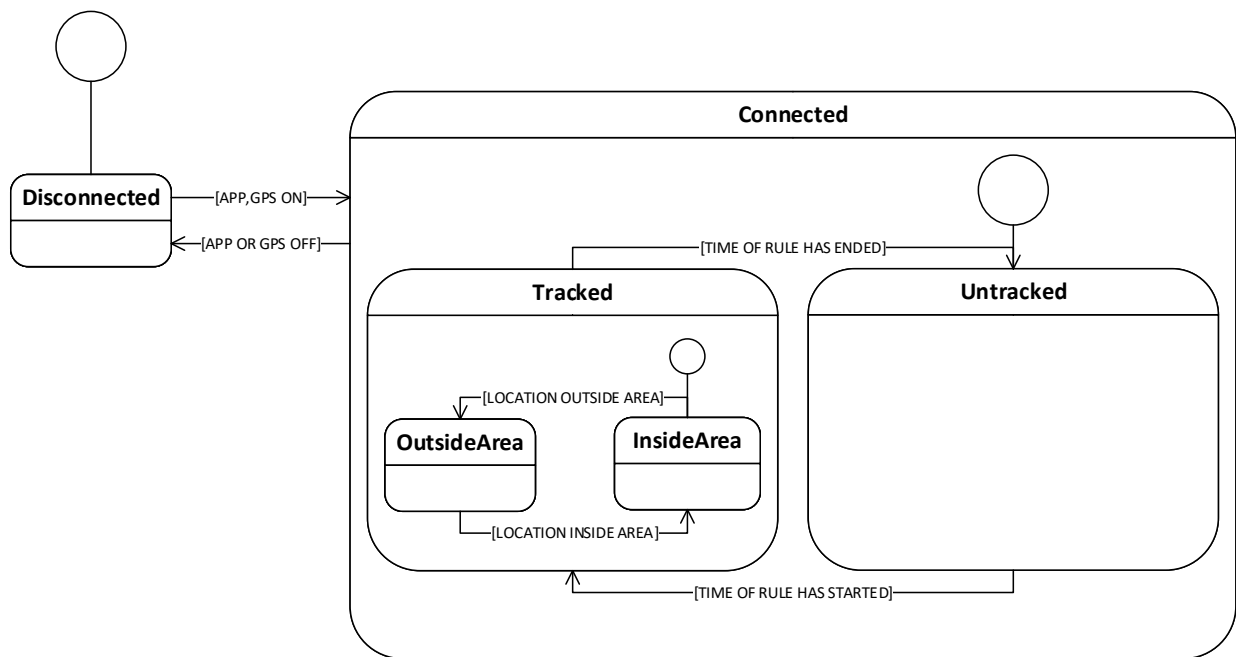
- KontoRodzica - klasa odpowiadająca za przechowywanie informacji o rodzicu takie jak imię, nazwisko, email, suma kontrolna hasła,
- KontoDziecka - klasa odpowiadająca za przechowywanie informacji o dziecku takie jak nazwa, email, suma kontrolna hasła,

- Reguła - klasa odpowiadająca za przechowywanie informacji o regułach ustalonych dla danego dziecka, dotyczących danego obszaru. Data pierwszego wydarzenia, czas pierwszego wydarzenia, powtarzanie wydarzeń (do wyboru: codziennie, co tydzień, itd.), czas trwania wydarzenia,
- Lokalizacja - klasa związana bezpośrednio z klasą KontoDziecka przechowująca lokalizację dziecka. Parametrami jest długość i szerokość geograficzna,
- Interfejs Obszaru - klasa interfejsu dla klas Obszarów,
- OkrągłyObszar - klasa przechowująca koordynaty X i Y środka okręgu, jego promień opisująca obszar na mapie w którym powinno znajdować się dziecko



Rysunek 8: Diagram klas

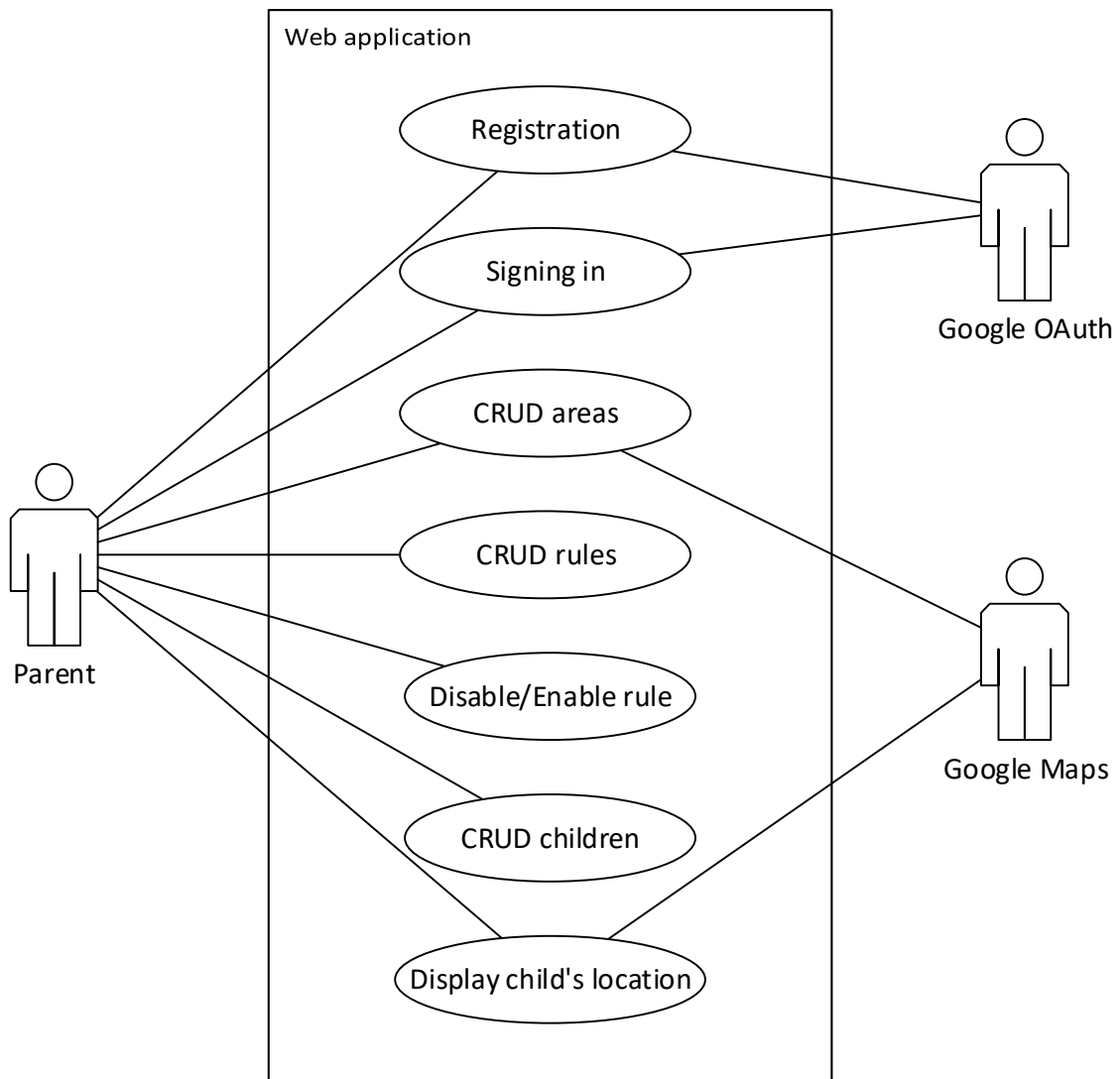
## 5.1 Diagram stanów dla telefonu dziecka



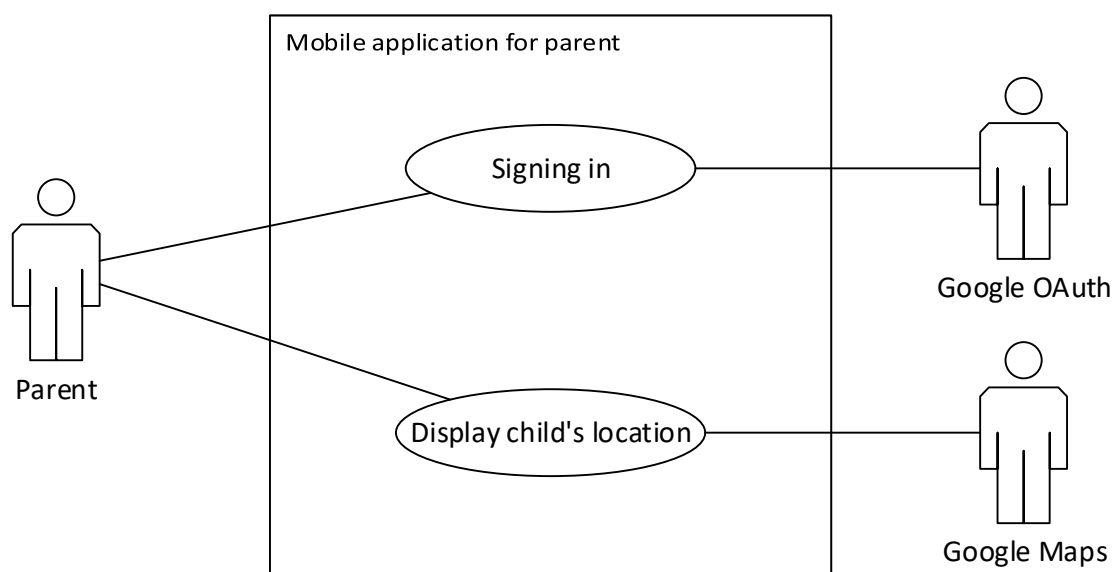
Rysunek 9: Diagram stanów

## 6 Specyfikacja wymagań

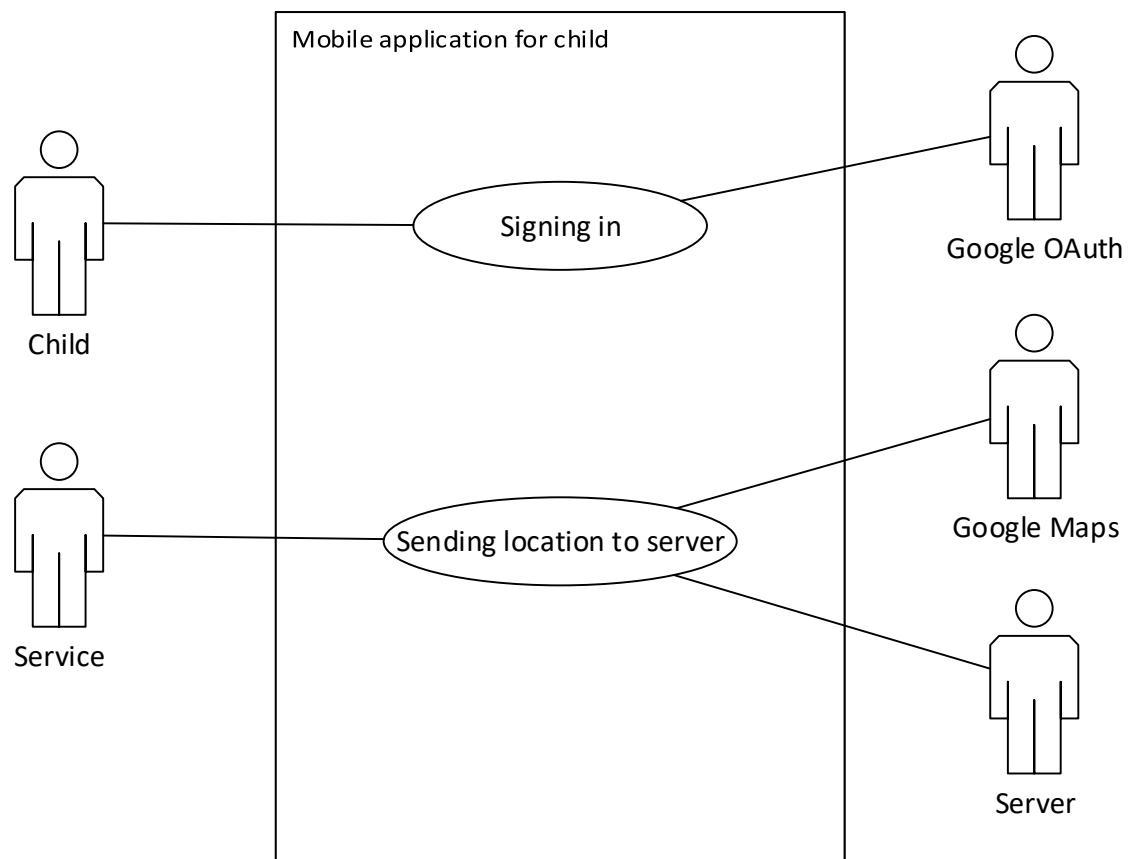
Wymagania stawiane systemowi przedstawione zostały w formie przypadków użycia wraz z scenariuszami.



Rysunek 10: Przypadki użycia dla aplikacji webowej



Rysunek 11: Przypadki użycia dla aplikacji mobilnej rodzica



Rysunek 12: Przypadki użycia dla aplikacji mobilnej dziecka

## Scenarios

---

### Registration

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to register
<b>Trigger:</b>	Parent clicks on Registration button
<b>Initial conditions:</b>	Parent is on our root route on our website
<b>Final conditions for success:</b>	Parent receives email with confirmation link. Account is created.
<b>Final conditions for failure:</b>	If parent already has account then we display an error message.

#### Main scenario: Registration

- 1.Parent clicks on Registration button.
- 2.Parent fills in a registration form.
- 3.Parent clicks Register button.
- 4.Account is created. Confirmation email is sent.

#### Alternative scenario: Parent already has an account.

- 3a.*Error message is displayed. Parent is asked to login, or change email.*
- 3a.1a.Step 2.
- 3a.1b.Logging scenario.
- 

### Logging

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to login
<b>Trigger:</b>	Parent clicks Login button
<b>Initial conditions:</b>	Parent is on our root route on our website
<b>Final conditions for success:</b>	Parent is logged in. Dashboard is displayed.
<b>Final conditions for failure:</b>	Parent is not logged in. Error message is displayed.

#### Main scenario: Logging

- 1.Parent clicks on Login button.

- 2.Parent fills in a login form.
- 3.Parent clicks Login button.
- 4.Parent is logged in. Dashboard is displayed.

**Alternative scenario: Parent filled in wrong email or password.**

*3a.Error message is displayed. Parent is asked to login, or change email.*

3a.1.Step 2.

**Alternative scenario: Parent want to login with Google.**

*1a.Parent clicks Login with Google.*

1a.1.Parent is redirected to GoogleOAuth.

1a.2.Parent is logged in. Dashboard is displayed.

---

**CRUD Areas: create**

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to create area
<b>Trigger:</b>	Parent clicks green button with plus
<b>Initial conditions:</b>	Parent is on areas route
<b>Final conditions for success:</b>	Area is created and displayed in list of areas.

**Main scenario: Creating area**

- 1.Parent clicks green button with plus.
- 2.Parent type name of area.
- 3.Parent type address of location.
- 4.Parent sets radius of area.
- 5.Parent adds child/children.
- 6.Parent saves changes.
- 7.Area is created and displayed in list of areas

**CRUD Areas: update**

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to update area
<b>Trigger:</b>	Parent clicks edit icon on the area
<b>Initial conditions:</b>	Parent is on areas route
<b>Final conditions for success:</b>	Area is updated and displayed in list of areas.

**Main scenario: Updating area**

- 1.Parent clicks edit icon on the area.
- 2.Parent updates name of area or location address or radius or list of children.
- 3.Parent saves changes.
- 4.Area is updated and displayed in list of areas.

**CRUD Areas: delete**

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to delete area
<b>Trigger:</b>	Parent clicks red 'X' icon on the area
<b>Initial conditions:</b>	Parent is on areas route
<b>Final conditions for success:</b>	Area is deleted and list of areas is updated.
<b>Final conditions for failure:</b>	Area isn't deleted.

**Main scenario: Deleting area**

- 1.Parent clicks red 'X' icon on the area.
- 2.Pop up is displayed with question "Are you sure?" with "Yes", "No" buttons.
- 3.Parent clicks "Yes".
- 4.Area is deleted and displayed and list of areas is updated.

**Alternative scenario: Parent don't want to delete.**

3a.Parent clicks "No".

3a.1.Area isn't deleted.

-----

**CRUD Children: create**

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to create child
<b>Trigger:</b>	Parent clicks green button with plus
<b>Initial conditions:</b>	Parent is on children route
<b>Final conditions for success:</b>	Child is created and displayed in list of children

**Main scenario: Creating child**

- 1.Parent clicks green button with plus.



- 2.Parent type name of child.
- 3.Parent chooses initials.
- 4.Parent chooses color for icon of child.
- 5.Parent saves changes.
- 6.Child is created and displayed in list of children.

#### CRUD Children: update

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to update child
<b>Trigger:</b>	Parent clicks edit icon on the child
<b>Initial conditions:</b>	Parent is on children route
<b>Final conditions for success:</b>	Child is updated and displayed in list of children.

#### Main scenario: Updating child

- 1.Parent clicks edit icon on the child.
- 2.Parent updates name of child or initials or color for icon.
- 3.Parent saves changes.
- 4.Child is updated and displayed in list of children.

#### CRUD Children: delete

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to delete child
<b>Trigger:</b>	Parent clicks red 'X' icon on the child
<b>Initial conditions:</b>	Parent is on children route
<b>Final conditions for success:</b>	Child is deleted and list of children is updated.
<b>Final conditions for failure:</b>	Child isn't deleted.

#### Main scenario: Deleting child

- 1.Parent clicks red 'X' icon on the child.
- 2.Pop up is displayed with question "Are you sure?" with "Yes", "No" buttons.
- 3.Parent clicks "Yes".
- 4.Child is deleted and displayed and list of children is updated.

#### Alternative scenario: Parent don't want to delete.

- 3a.Parent clicks "No".
- 3a.1.Child isn't deleted.

---

### CRUD Rules: create

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to create rule
<b>Trigger:</b>	Parent clicks green button with plus
<b>Initial conditions:</b>	Parent is on specific child route
<b>Final conditions for success:</b>	Rule is created and displayed in list of rules.

#### Main scenario: Creating rule

- 1.Parent clicks green button with plus.
- 2.Parent chooses range of dates.
- 3.Parent chooses range of time.
- 4.Parent chooses location from list.
- 5.Parent chooses repetition from list.
- 6.Parent saves changes.
- 7.Rule is created and displayed in list of rules.

### CRUD Rules: update

<b>Actors:</b>	Parent
<b>Range:</b>	Web application
<b>Goals:</b>	<b>Parent</b> wants to update rule
<b>Trigger:</b>	Parent clicks edit icon on the rule
<b>Initial conditions:</b>	Parent is on rules route
<b>Final conditions for success:</b>	Rule is updated and displayed in list of rules.

#### Main scenario: Updating rule

- 1.Parent clicks edit icon on the rule.
- 2.Parent updates range of dates or range of time or location or repetition.
- 3.Parent saves changes.
- 4.Rule is updated and displayed in list of rules.

### CRUD Rules: delete

<b>Actors:</b>	Parent
<b>Range:</b>	Web application

<b>Goals:</b>	<b>Parent</b> wants to delete rule
<b>Trigger:</b>	Parent clicks red 'X' icon on the rule
<b>Initial conditions:</b>	Parent is on rules route
<b>Final conditions for success:</b>	Rule is deleted and list of rules is updated.
<b>Final conditions for failure:</b>	Rule isn't deleted.

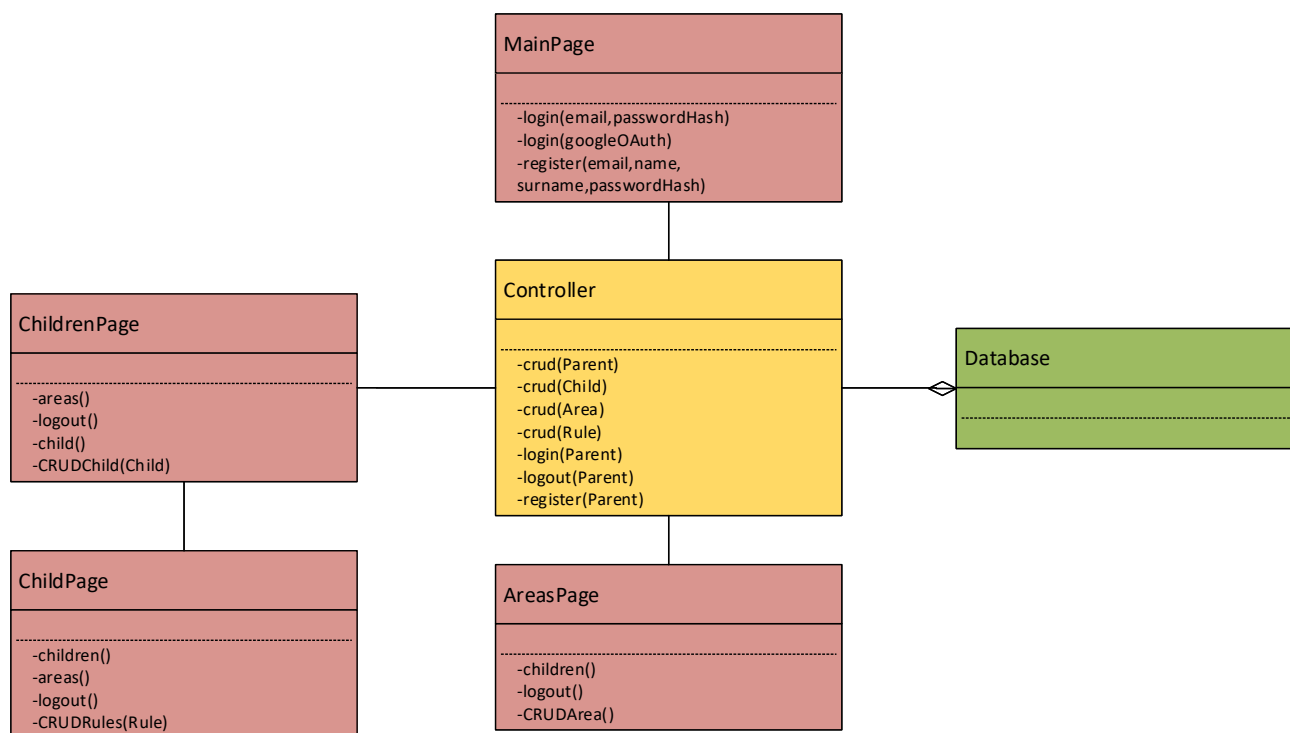
**Main scenario: Deleting rule**

1. Parent clicks red 'X' icon on the rule.
2. Pop up is displayed with question "Are you sure?" with "Yes", "No" buttons.
3. Parent clicks "Yes".
4. Rule is deleted and displayed and list of rules is updated.

**Alternative scenario: Parent don't want to delete.**

- 3a. *Parent clicks "No".*
- 3a.1. *Rule isn't deleted.*

## 7 Architektura systemu



Rysunek 13: Diagram implementacji modelu MVC

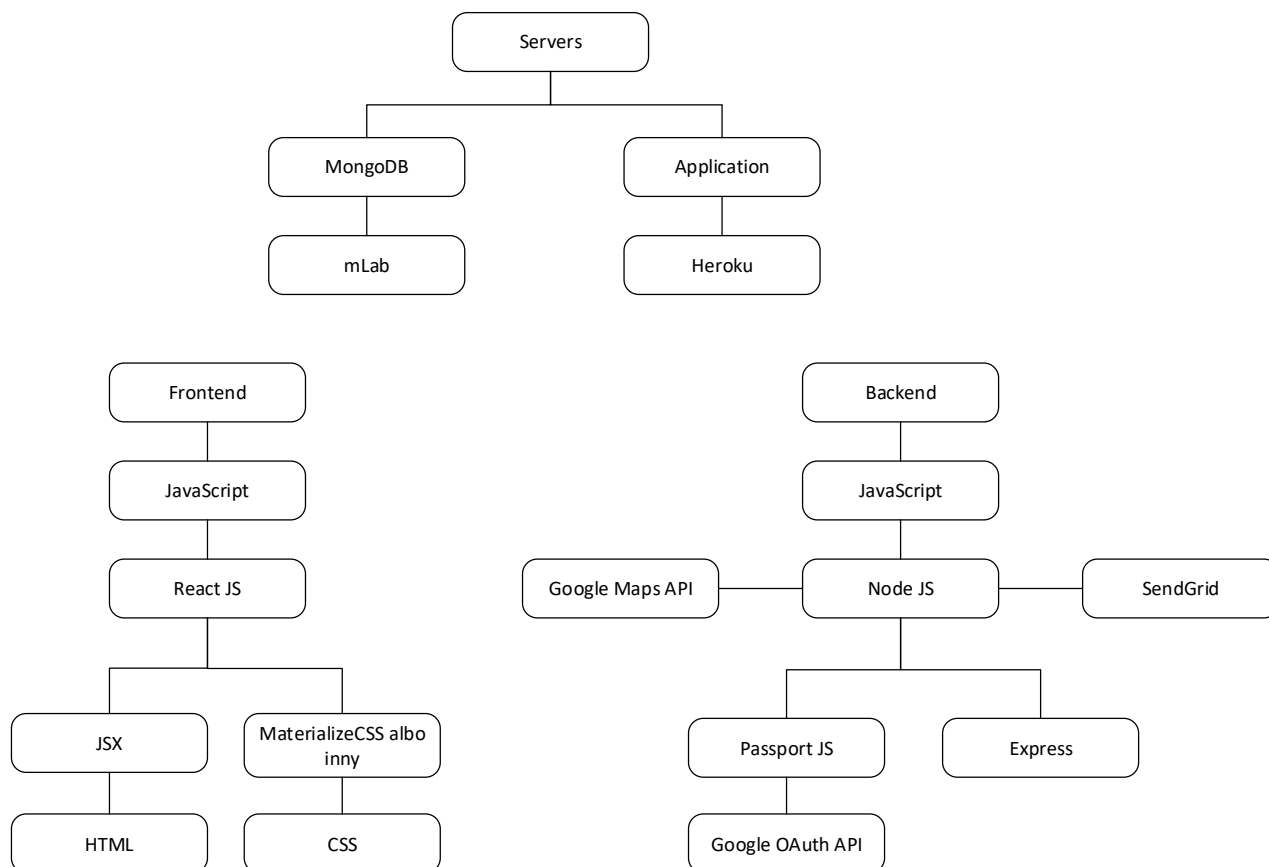
Komunikacja w aplikacji będzie obsługiwana przez serwer, który udostępnia api. Użytkownik poprzez wypełnianie różnych formularzy, czy klikanie przycisków, będzie wysyłał zapytania do serwera. Serwer uruchomi odpowiednie działanie na bazie danych i zwróci właściwą odpowiedź. Na podstawie informacji z serwera komponenty strony zaktualizują się lub użytkownik zostanie przekierowany na odpowiedni adres.

### 7.1 Opis API

- `\api\current_user` - zwraca bieżącego użytkownika, aby kontrolować czy użytkownik jest zalogowany,
- `\api\logout` - wylogowuje użytkownika,
- `\api\googleAuth` - logowanie z Google,
- `\api\register` - rejestrowanie nowych użytkowników,
- `\api\login` - logowanie użytkowników,
- `\api\rules` - pobieranie reguł dla danego rodzica
- `\api\rules\create` - dodawanie reguły
- `\api\rules\read` - odczyt reguły
- `\api\rules\update` - modyfikowanie reguły
- `\api\rules\delete` - usuwanie reguły
- `\api\areas` - pobieranie obszarów dla danego rodzica
- `\api\areas\create` - dodawanie obszaru
- `\api\areas\read` - odczyt obszaru
- `\api\areas\update` - modyfikowanie obszaru
- `\api\areas\delete` - usuwanie obszaru
- `\api\children` - pobieranie dzieci dla danego rodzica

- `\api\children\create` - dodawanie dziecka
- `\api\children\read` - odczyt dziecka
- `\api\children\update` - modyfikowanie dziecka
- `\api\children\delete` - usuwanie dziecka

## 8 Projekt oprogramowania



Rysunek 14: Stos technologiczny

### 8.1 Lista narzędzi używanych przy realizacji projektu

- GitHub,
- Trello,
- Slack,
- SonarQube,
- Latex

## 9 Projekt interfejsu użytkownika

### Rejestracja

LOGO

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec gravida tellus in turpis rhoncus sagittis. Morbi aliquet ante risus, sit amet placerat erat egestas sed. Nullam sodales dui lorem, sed blandit ligula volutpat id. Curabitur at luctus quam. Proin et mi blandit, pulvinar massa et volutpat.



REJESTRACJA

LOGOWANIE

Imię

Nazwisko

Email

Hasło

Potwierdź hasło

ZAREJESTRUJ SIĘ

GOOGLE

22

## Rejestracja Błąd

LOGO

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec gravida tellus in turpis rhoncus sagittis. Morbi aliquet ante risus, sit amet placerat erat egestas sed. Nullam sodales dui lorem, sed blandit ligula volutpat id. Curabitur at luctus quam. Proin et mi blandit, pulvinar massa et volutpat.



REJESTRACJA

LOGOWANIE

Imię

Nazwisko

Email

Hasło

Potwierdź hasło

Error message e.g Invalid email address!

ZAREJESTRUJ SIĘ

GOOGLE

## Logowanie

LOGO

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec gravida tellus in turpis rhoncus sagittis. Morbi aliquet ante risus, sit amet placerat erat egestas sed. Nullam sodales dui lorem, sed blandit ligula volutpat id. Curabitur at luctus quam. Proin et mi blandit, pulvinar massa et volutpat.



REJESTRACJA

LOGOWANIE

Adres Email

Hasło

ZALOGUJ SIĘ

lub zaloguj się za pomocą

GOOGLE

## Logowanie Błąd

LOGO

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec gravida tellus in turpis rhoncus sagittis. Morbi aliquet ante risus, sit amet placerat erat egestas sed. Nullam sodales dui lorem, sed blandit ligula volutpat id. Curabitur at luctus quam. Proin et mi blandit, pulvinar massa et volutpat.



REJESTRACJA

LOGOWANIE

Adres Email

Hasło

Error message e.g Invalid email address!

ZALOGUJ SIĘ

lub zaloguj się za pomocą

GOOGLE

## Dashboard

LOGO

Dzieci

Obszary

Profil

Wyloguj

11.11.2018

Szkoła

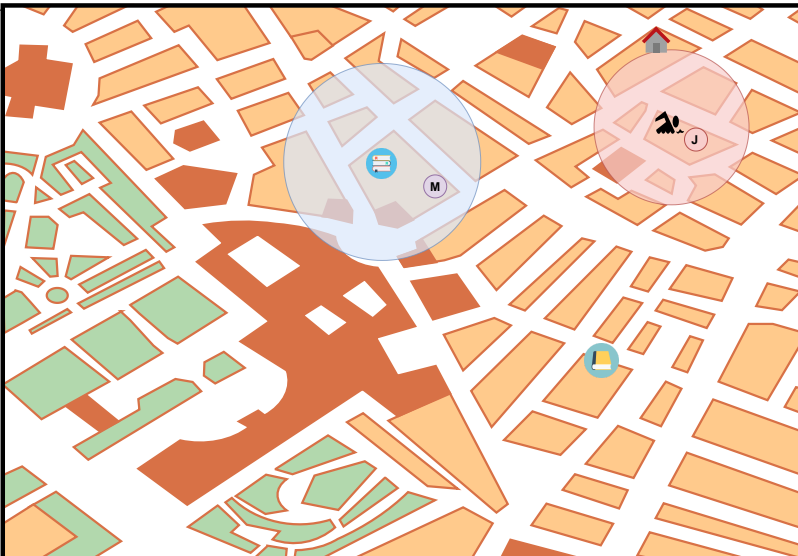
8:00-14:15

M

Basen

10:00-12:00

J





## Reguła - Create

LOGO

Jessica J

Obszary

Profil

Wyloguj

11.11.2018

Szkoła

8:00-14:15 J

Basen

16:00-18:00 J

+

Dodaj regułę

11.11.2018

11.11.2018

11.11.2018

11.11.2018

Wybierz obszar

Nie potarza się

Zapisz

Szkoła

Basen

Dom

Nie potarza się

Codziennie

Co tydzień

Co miesiąc

W dni powszednie

## Reguła - Read

LOGO

Dzieci

Obszary

Profil

Wyloguj

Jessica J

Brajan B

Sebastian M

+

## Reguła - Update

LOGO

Jessica J

Obszary

Profil

Wyloguj

11.11.2018

Szkoła

8:00-14:15

J

Basen

16:00-18:00

J

Edytuj regułę

11.11.2018

11.11.2018

11.11.2018

11.11.2018

Codziennie

Zapisz

Szkoła

Basen

Dom

Nie potarza się

Codziennie

Co tydzień

Co miesiąc

W dni powszednie

## Reguła - Delete

LOGO

Jessica J

Obszary

Profil

Wyloguj

11.11.2018

Szkoła

8:00-14:15

J

Basen

16:00-18:00

J

Czy na pewno chcesz usunąć

Szkoła

8:00-14:15

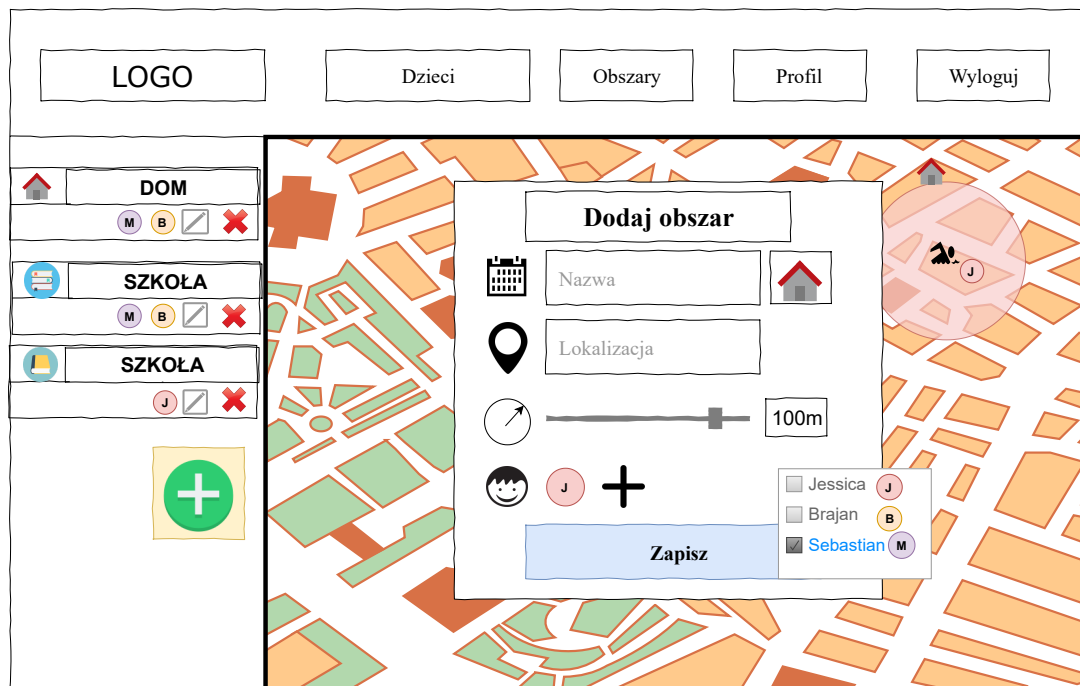
J

z listy reguł?

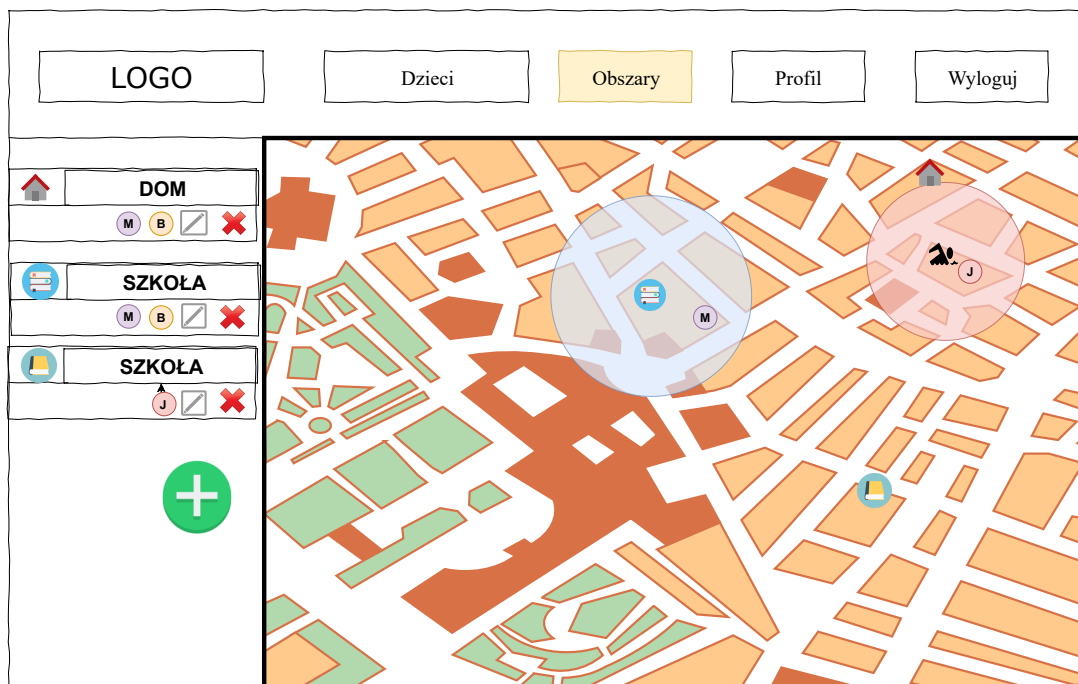
TAK

NIE

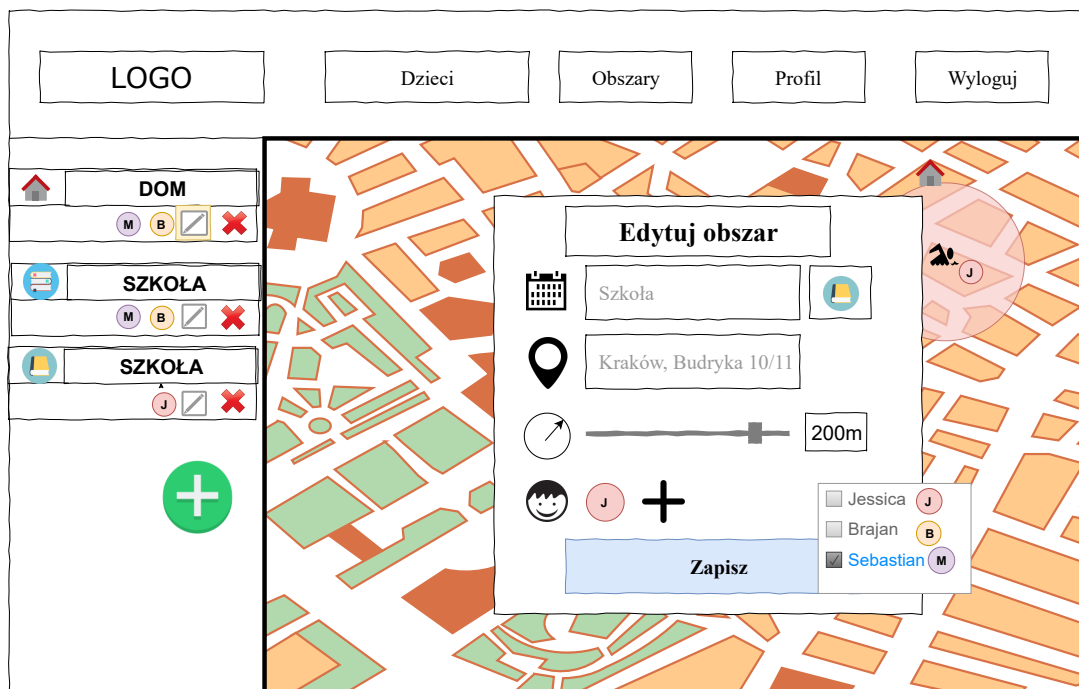
## Obszar - Create



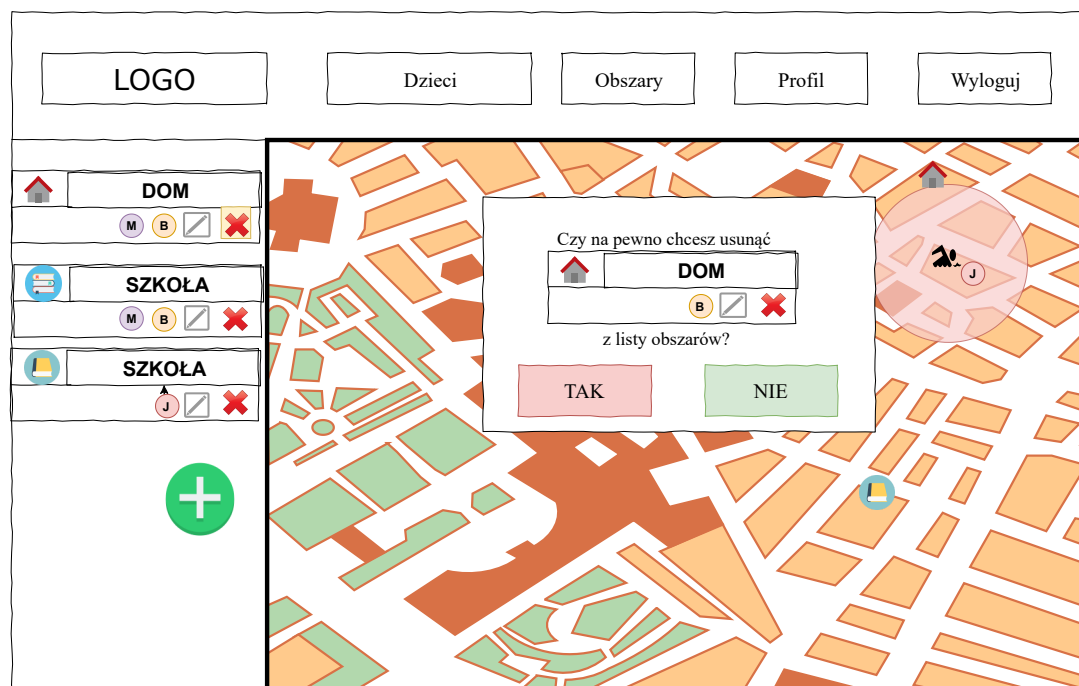
## Obszar - Read



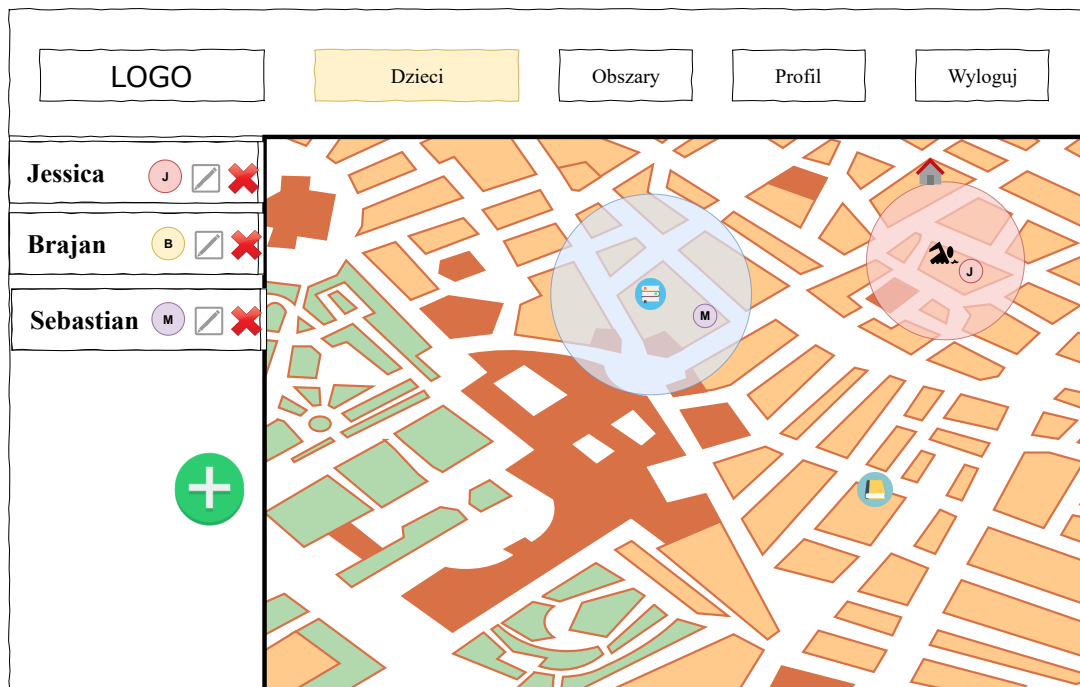
## Obszar - Update



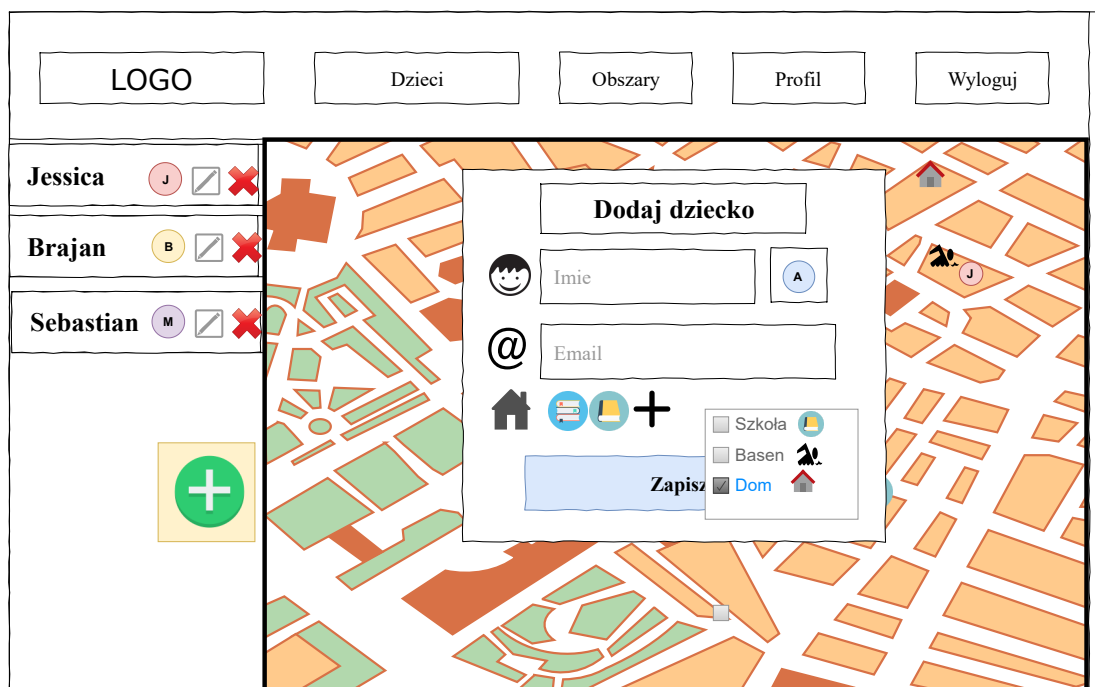
## Obszar - Delete



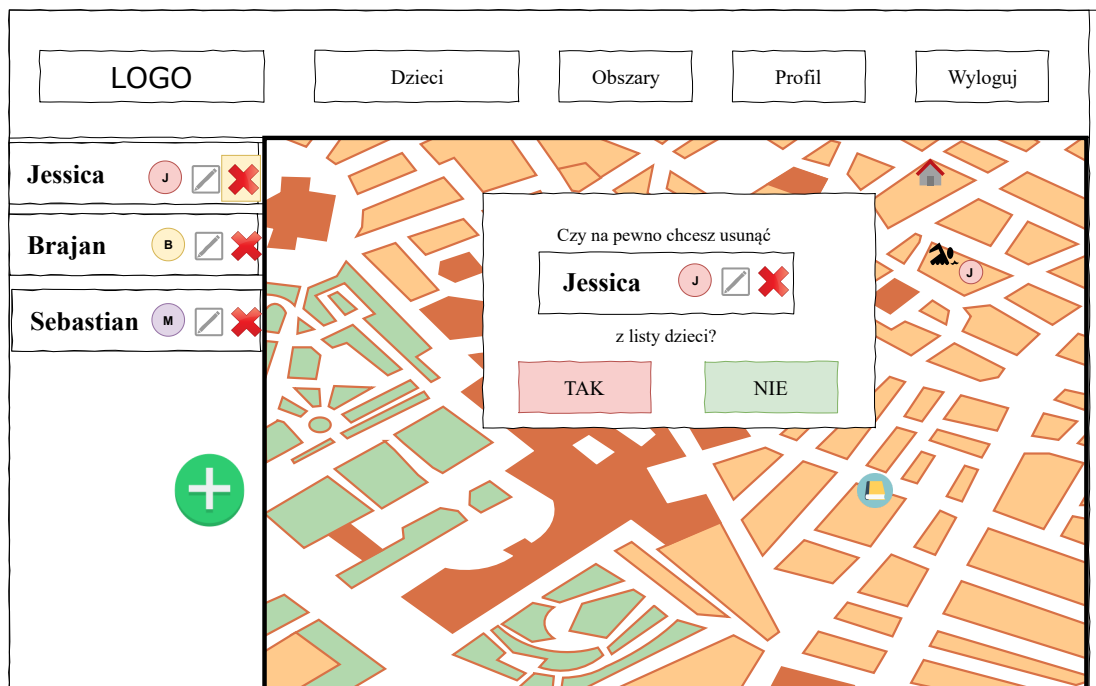
## Dziecko - Read



## Dziecko - Create



## Dziecko - Delete



## Dziecko - Update

