

**Politechnika Śląska**

**Wydział Automatyki, Elektroniki i Informatyki**

**Kierunek Informatyka**

##### Projekt inżynierski

###### Domowy organizer

Autor: Piotr Kapski

Kierujący pracą: dr inż. Daniel Kostrzewa

Gliwice, Grudzień 2019

Załącznik Nr 2 do Zarz. Nr 97/08/09

**Oświadczenie**

Wyrażam zgodę/nie wyrażam\* zgody na udostępnienie mojej pracy dyplomowej/rozprawy doktorskiej\*

…………….………., dnia …………………………

……………………………………..……………….……

*(podpis)*

……………………………………………………………

*(poświadczenie wiarygodności podpisu przez Dziekanat)*

*\* właściwe podkreślić*

**Oświadczenie promotora**

Oświadczam, że praca „Tytuł pracy dyplomowej inżynierskiej” spełnia wymagania formalne pracy dyplomowej inżynierskiej.

|  |  |
| --- | --- |
| Gliwice, dnia ……………………… | ………………..……………….……  *(podpis)* |

Spis treści

[1. Wstęp 1](#_Toc526887085)

[2. [Analiza tematu] 5](#_Toc526887086)

[3. Wymagania i narzędzia 9](#_Toc526887087)

[4. Specyfikacja zewnętrzna 12](#_Toc526887088)

[5. Specyfikacja wewnętrzna 15](#_Toc526887089)

[6. Weryfikacja i walidacja 18](#_Toc526887090)

[7. Podsumowanie i wnioski 21](#_Toc526887091)

[Bibliografia i](#_Toc526887092)

[Spis skrótów i symboli ii](#_Toc526887093)

[Zawartość dołączonej płyty iii](#_Toc526887094)

[Spis rysunków iv](#_Toc526887095)

[Spis tabel v](#_Toc526887096)

# Wstęp

Jak wynika z raportu „Digital 2019”, ludzie na świecie spędzają średnio 6 godzin i 42 minuty dziennie w Internecie [1]. Ciągły rozwój i wzrost jego popularności wpływa również na nasze życie w domu, gdzie spędzamy większość swojego życia. Przez stale rosnąc pęd życia, staramy się przyśpieszać podstawowe czynności, by być jak najbardziej efektywnym.

Celem pracy jest więc napisanie aplikacji internetowej wspomagającej nas w codziennym życiu rodzinnym z użyciem Internetu. Będzie ona umożliwiała rodzinie tworzenie notatek, list oraz uzupełnianie wspólnego kalendarza. Umożliwi ona sprawniejszą komunikację w rodzinie, a także wpisze się w trend cyfryzacji, a co za tym idzie - rezygnacji z papieru.

Praca będzie obejmowała zarówno aplikację serwerową (logikę) jak i kliencką (wygląd).#TODO

Rozdział drugi przedstawia dokładniejszą analizę wybranego tematu oraz literatury, a także porównuje podobne istniejące już rozwiązania. W rozdziale trzecim

# [Analiza tematu]

Rozdział zawiera takie elementy, jak:

* analiza tematu,
* wprowadzenie do dziedziny (*state of the art*) – sformułowanie problemu,
* studia literaturowe [2, 3, 4, 1],
* opis znanych rozwiązań (także opisanych naukowo, jeżeli problem jest poruszany w publikacjach naukowych), algorytmów, osadzenie pracy w kontekście.

# Wymagania i narzędzia

## Wymagania funkcjonalne

* Użytkownik może zarejestrować się do systemu podając nazwę użytkownika, e-mail oraz hasło
* Użytkownik może uwierzytelniać się za pomocą nazwy użytkownika i hasła
* Użytkownik może zmienić wszystkie dane oraz hasło
* Każdy użytkownik ma równe prawa
* Użytkownicy mogą tworzyć lub dołączać do istniejącego gospodarstwa domowego (ang. household)

## Wymagania niefunkcjonalne

* Aplikacja kompatybilna z przeglądarkami: IE, Firefox oraz Chrome
* Aplikacja powinna być wieloplatformowa
* Serwer aplikacji powinien być zabezpieczony przed niezautoryzowanym dostępem
* Aplikacja powinna być jak najbardziej wydajna
* Aplikacja powinna być intuicyjna dla użytkownika i łatwa w obsłudze

## Opis narzędzi

### Java

Język programowania, w którym został napisany serwer aplikacji. Jest to jeden z najpopularniejszych języków na świecie do pisania aplikacji webowych, którego dużą zaletą jest multiplatformowość. Java posiada obszerną liczbę platform programistycznych (ang. Framework), które znacząco ułatwiają pracę.

### IntelliJ IDEA

Najpopularniejsze zintegrowane środowisko programistyczne dla Javy stworzone przez firmę JetBrains. Posiada ogromną liczbę narzędzi wspomagających pisanie kodu w takich sferach jak refaktoryzacja, automatyczne podpowiedzi, zaawansowany debugger oraz wsparcie dla licznych framework’ów, czy narzędzi jak np. GitHub. IntelliJ został stworzony tak, aby zmaksymalizować produktywność programistów i ograniczyć użycie myszki przez stworzenie licznych skrótów klawiszowych. Jedynym większym konkurentem tego środowiska jest Eclipse, którego największą zaletą jest posiadanie otwartych źródeł (ang. Open source) i to, że jest całkowicie darmowe nawet do zastosować komercyjnych. Środowisko stworzone przez JetBrains jest jednak zdaniem autora zdecydowanie wygodniejsze, dlatego zostało wybrane do tego projektu.

### Spring framework

Spring jest szkieletem tworzenia aplikacji dla platformy Java EE (Enterprise Edition) [2]. Przez ogrom możliwości i usprawnień, które oferuje, jest on jednym z ważniejszych frameworków dla Javy i jest używany przez większość programistów tego języka. Spring składa się z wielu projektów, które mogą być używane niezależnie np. Spring Cloud, Spring Data czy Spring Batch. W tym projekcie, wykorzystano:

* Spring Boot – zdaniem autora, jeden z najważniejszych części frameworku. Oferuje szybkie stworzenie gotowej do użycia aplikacji, która zawiera wszystkie potrzebne komponenty. W trakcie programowania oferuje znaczne uproszczenia kodu, np. przez rezygnacje z plików XML do konfiguracji.
* Spring Data – używany do komunikacji oraz obsługi bazy danych, a także do mapowania obiektowo-relacyjnego. Jest alternatywą do popularnego frameworku Hibernate, który jednak przez sporą liczbę wad, zostaje powoli wyparty przez nowsze technologie takie jak Spring Data.
* Spring Security – zapewnia autoryzację oraz autentykacje dla aplikacji. Jest to potężny framework o dużych możliwościach konfiguracji, przy jednoczesnej łatwości w obsłudze.

### Angular

Angular to otwarty framework napisany w języku TypeScript, stworzony i wspierany przez Google. Można dzięki niemu szybko i łatwo zbudować aplikacje internetową, nie tracąc przy tym na ilości możliwości, którymi dysponuje. W projekcie był on używany w wersji 8 do napisania aplikacji klienckiej. Alternatywą dla Angulara jest między innymi React, który jest biblioteką JavaScript. Obie technologie są jednymi z ważniejszych w front-endzie oraz mają swoje plusy i minusy, dlatego każdy wybór oferował wystarczające możliwości.

### Baza danych MongoDB

MongoDB jest nierelacyjną bazą danych, czyli nie posiada sztywno określonej struktury danych w niej przechowywanych. W przeciwieństwie do klasycznego, relacyjnego podejścia, dane składowane są w postaci dokumentów w formacie JSON, który jest dla aplikacji bardzo naturalną strukturą. Mongo jest przy tym bezpieczne, szybkie i bardzo skalowalne, a także jest dobrze wspierane przez Spring Data. Jest to aktualnie najczęściej używana nierelacyjna baza danych, a jej popularność rośnie z roku na rok [4].

### Swagger

Swagger jest znany przede wszystkim jako narzędzie do tworzenia dokumentacji. W projekcie został jednak również użyty do generowania API (Interfejs Programowania Aplikacji) wraz z encjami modelu. Połączenie tych dwóch funkcjonalności daje bardzo pomocne narzędzie w tego typu aplikacjach.

### Robo 3T

Robo 3T to graficzny interfejs do zarządzania bazą danych [5]. Jest to lekki i darmowy odpowiednik bardziej rozbudowanego Studio 3T, jednak na potrzeby projektu Robo jest zdecydowanie wystarczające.

## Metodyka pracy

Podczas pracy nad projektem wykorzystywane było narzędzie do kontroli wersji Git. Każde nowe zadanie niosło za sobą stworzenie nowej gałęzi (ang. branch), by zasymulować pracę w realnym projekcie jak i umożliwiało rozwijanie paru pomysłów jednocześnie. Praca rozpoczęta została od stworzenia wstępnego serwera, który był później na bieżąco modyfikowany na potrzeby nowo powstałych wymagań.

# Specyfikacja zewnętrzna

Aplikacja do działania wymaga jedynie dostępu do Internetu oraz posiadania jednej z następujących przeglądarek:

* IE 11+
* Google Chrome
* Mozilla Firefox
* Safari 7+
* Edge 13+

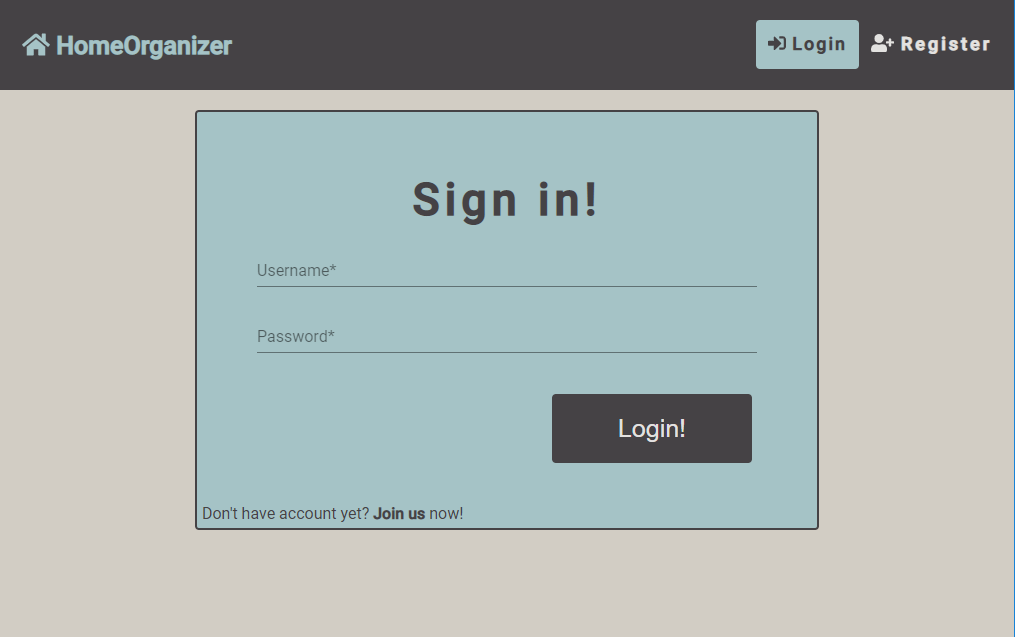
W systemie wyróżniamy tylko jedną kategorie użytkowników – użytkownik systemu. Może on być w dwóch stanach:

* Użytkownik niezalogowany – jedynymi akcjami jakie może wykonać jest rejestracja oraz logowanie.
* Użytkownik zalogowany – gość po zalogowaniu otrzymuje dostęp do wszystkich funkcji. Wszyscy użytkownicy mają równe prawa.

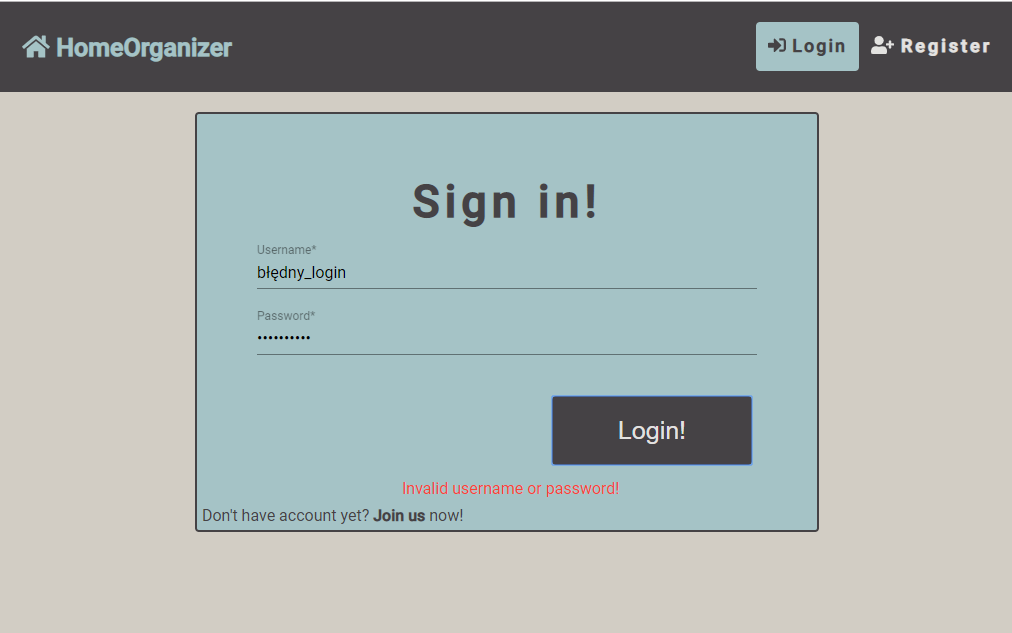
## Użytkownik niezalogowany

### Logowanie

Po wejściu do aplikacji, pierwszą stroną ukazującą się gościowi jest formularz logowania przedstawiony na Rys. 1. Musi on podać poprawną nazwę użytkownika i hasło, w przeciwnym razie, zostanie wyświetlony stosowny komunikat (Rys.2). W przypadku sukcesu, użytkownik zostaje zalogowany i przekierowany do menu głównego (rys. 5).

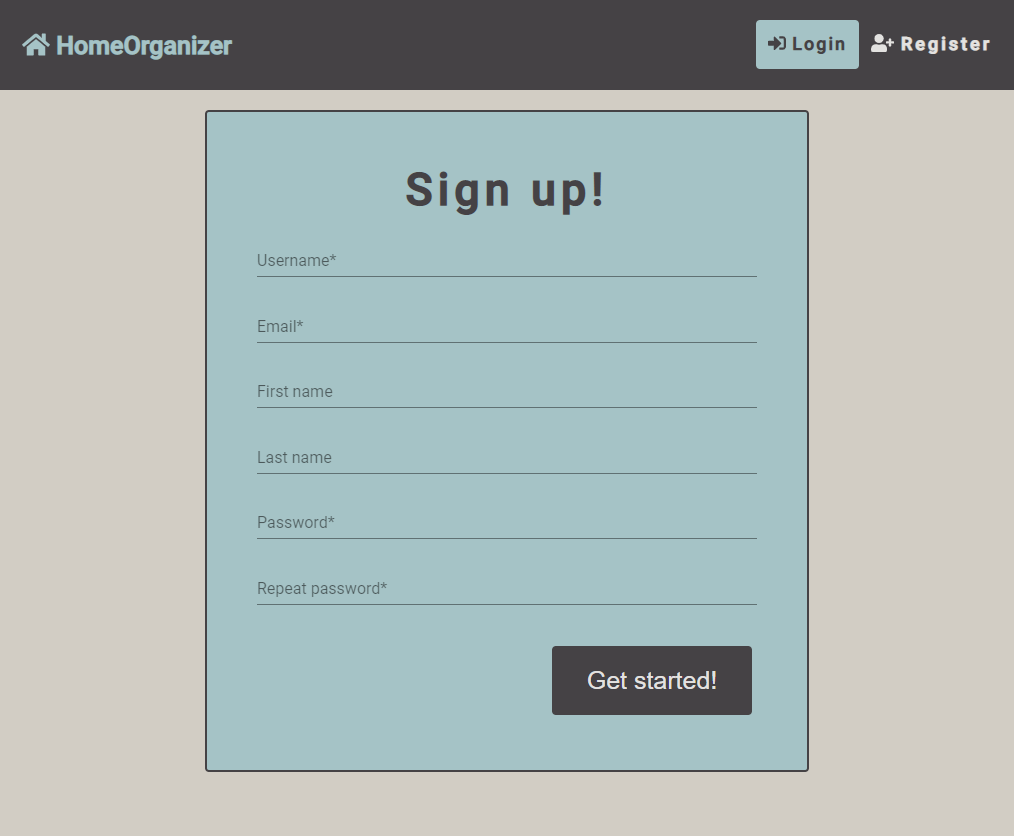


|  |
| --- |
| Rys.1 *Formularz logowania* |
|  |

Rys.2 *Formularz logowania z błędem*

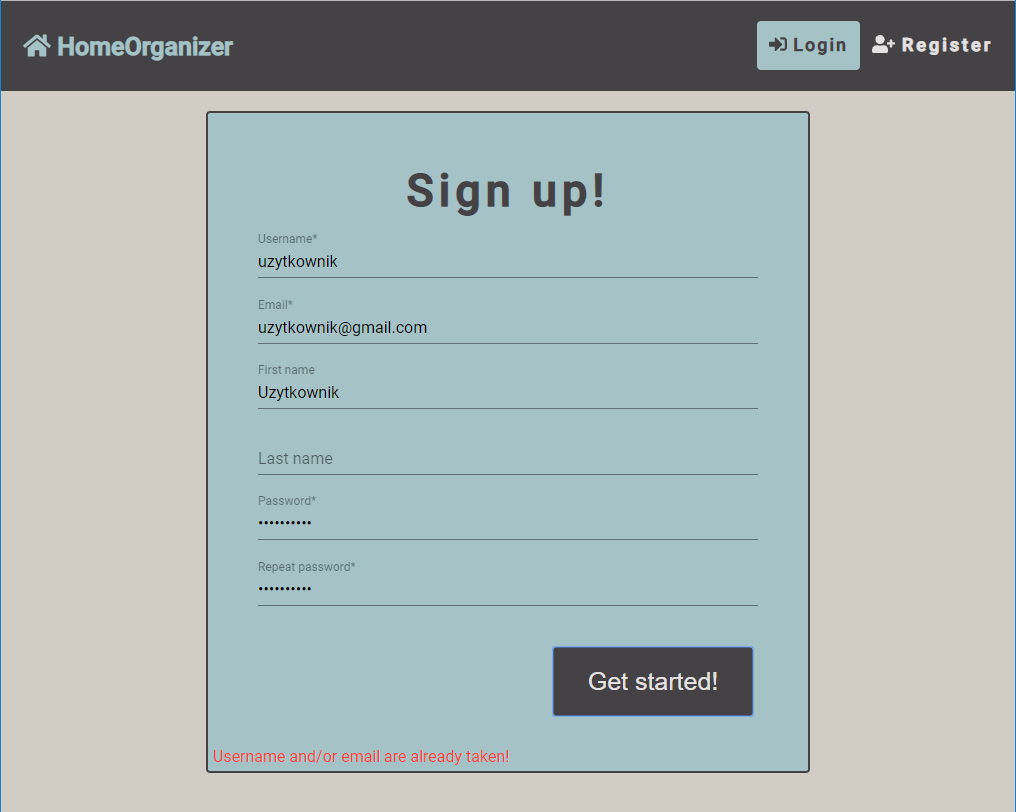
### Rejestracja

W przypadku, gdy użytkownik nie ma jeszcze utworzonego konta, należy nacisnąć na przycisk Register w nagłówku lub na napis **Join us** na dole formularza. Aplikacja przekierowuje wtedy do strony rejestracji z Rys. 3.



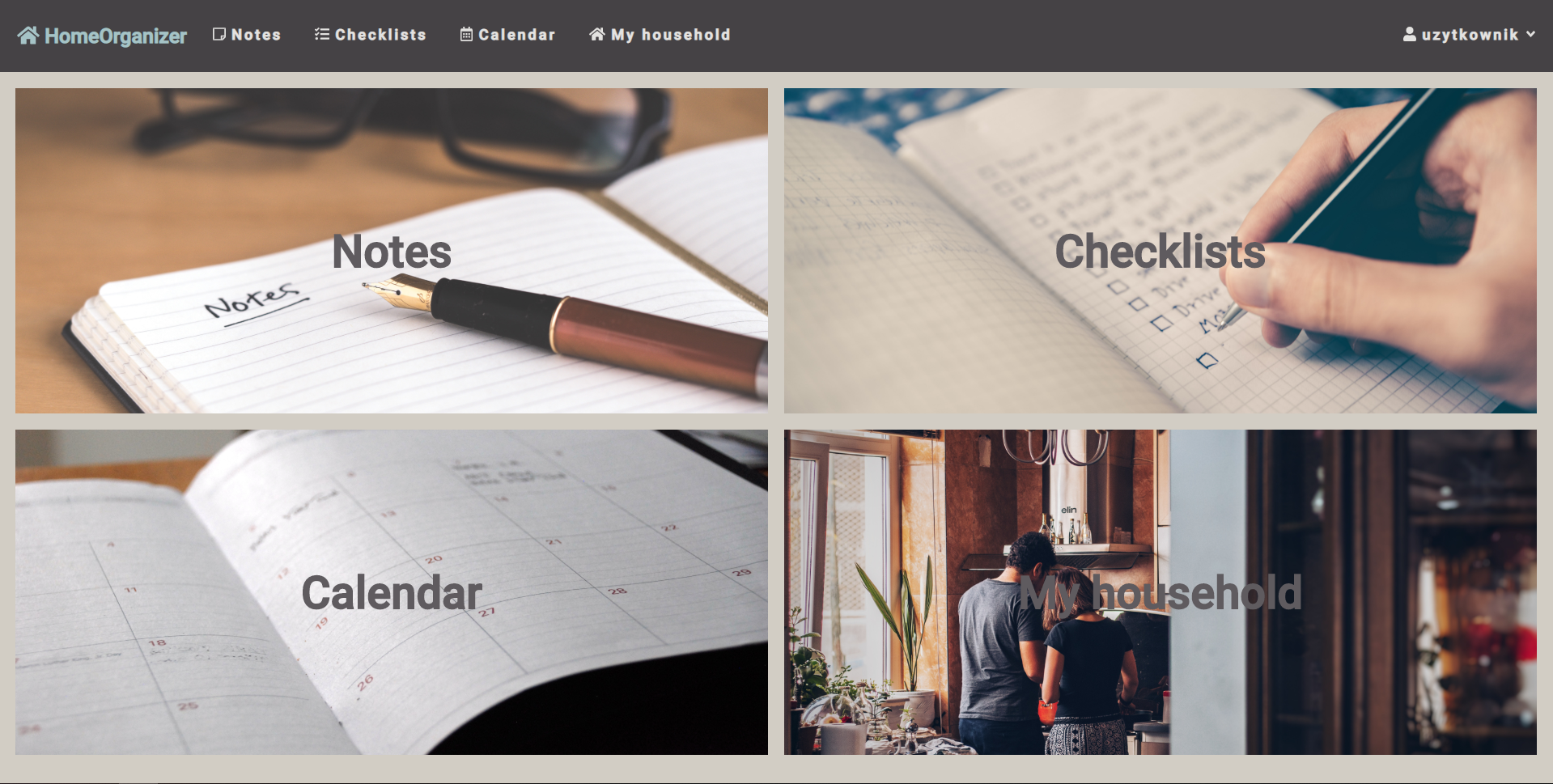
Rys.3 *Formularz rejestracji*

Należy w nim podać unikalną nazwę użytkownika (min. 5 znaków), unikalny email, opcjonalne imię i nazwisko oraz dwukrotnie hasło (min. 5 znaków). Po wypełnieniu pól, naciśnięcie przycisku **Get started!** w przypadku porażki wyświetli komunikat błędu (Rys. 4), w przeciwnym razie użytkownik zostaje utworzony i ma możliwość zalogowania się.



Rys.4 *Formularz rejestracji z błędem*

## Użytkownik zalogowany



Rys.5 *Strona główna użytkownika*

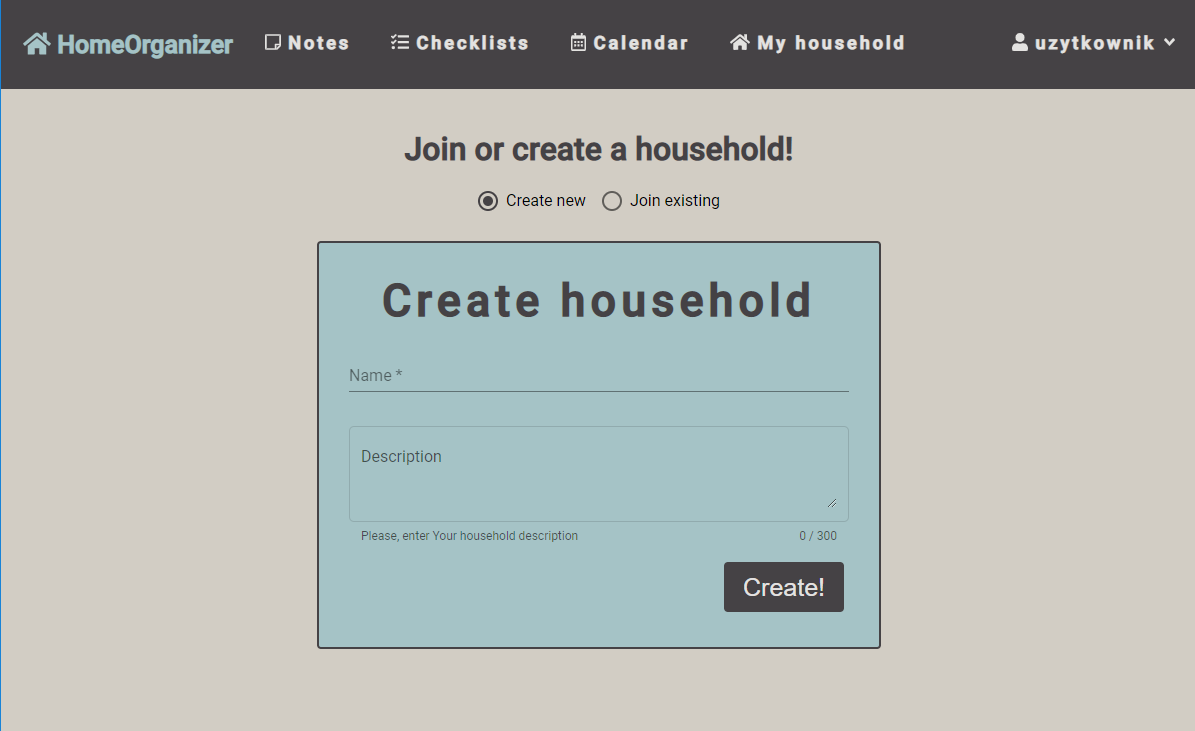
Będąc zalogowanym, użytkownik może wykonać następujące akcje:

* Zarządzać notatkami swojego household’u
* Zarządzać listami swojego household’u
* Zarządzać kalendarzem swojego household’u
* Zarządzać household’em
* Edytować swoje dane i hasło
* Wylogować się

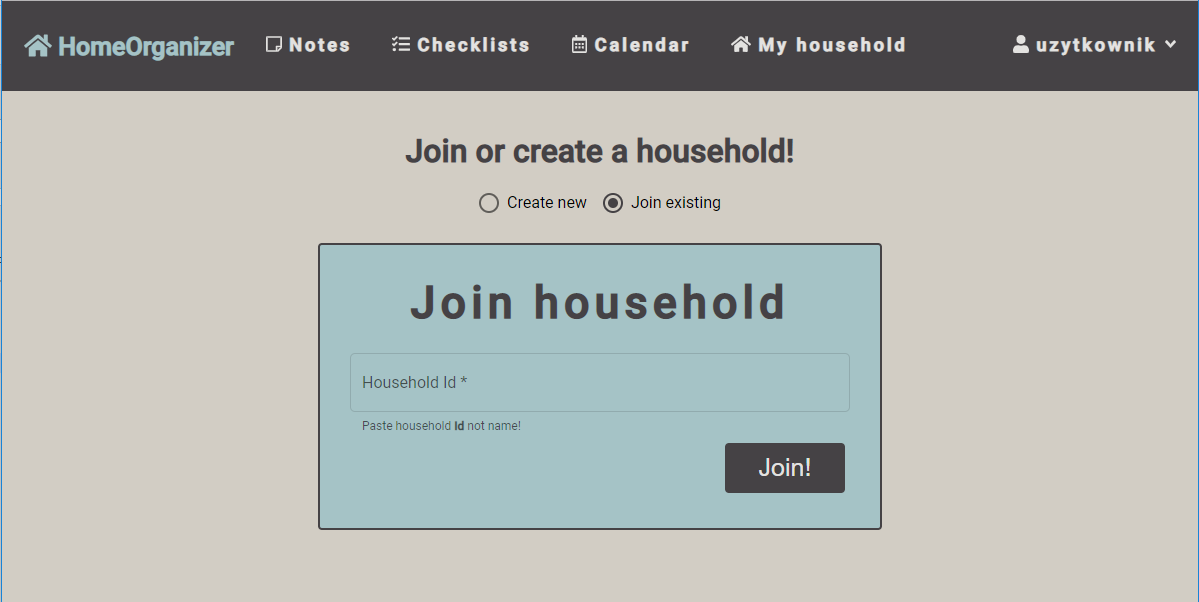
### Household

Household w aplikacji pełni role wirtualnego domu. Zawiera on nazwę, opis, unikalny identyfikator nadawany przez aplikacje oraz listę członków. Wszystkie akcje za wyjątkiem edycji konta odbywają się w obrębie householdu i tylko jego członkowie mogą widzieć dodane notatki, listy oraz uzupełniony kalendarz. Użytkownik nie będący w żadnym domu, może jedynie edytować swoje dane.

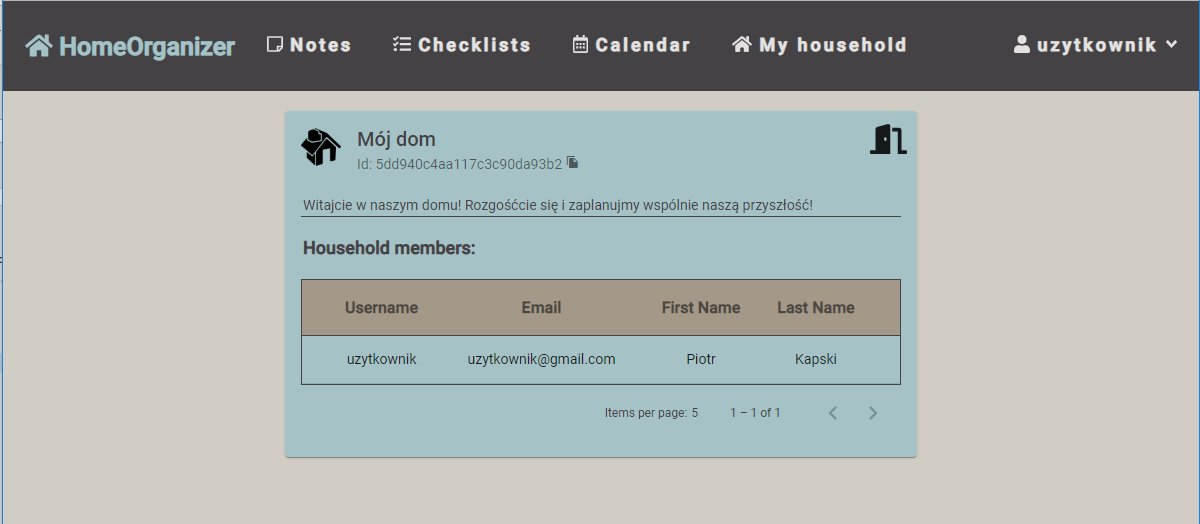
Chcąc wyświetlić swój obecny household, stworzyć nowy bądź dołączyć do już istniejącego, należy wybrać zakładkę **My Household** w nagłówku lub menu głównym. Jeśli nie jesteśmy jeszcze w żadnym domu, zostaniemy przekierowani do formularza tworzenia nowego jak przedstawiono na rysunku 6. Należy podać nazwę, która nie musi być unikalna, oraz opcjonalnie opis, by po kliknięciu przycisku **Create!** stworzyć household i być jego pierwszym członkiem (Rys.8). Jest również możliwość dołączenia do już istniejącego household’u, przez wybranie opcji **Join existing** (rysunek 7). Należy wtedy w puste pole wpisać identyfikator domu otrzymany od członka znajdującego się już w householdzie (Id pod nazwą na rysunku 8).



Rys.6 *Formularz tworzenia household’u*



Rys.7 *Dołączanie do istniejącego household’u*



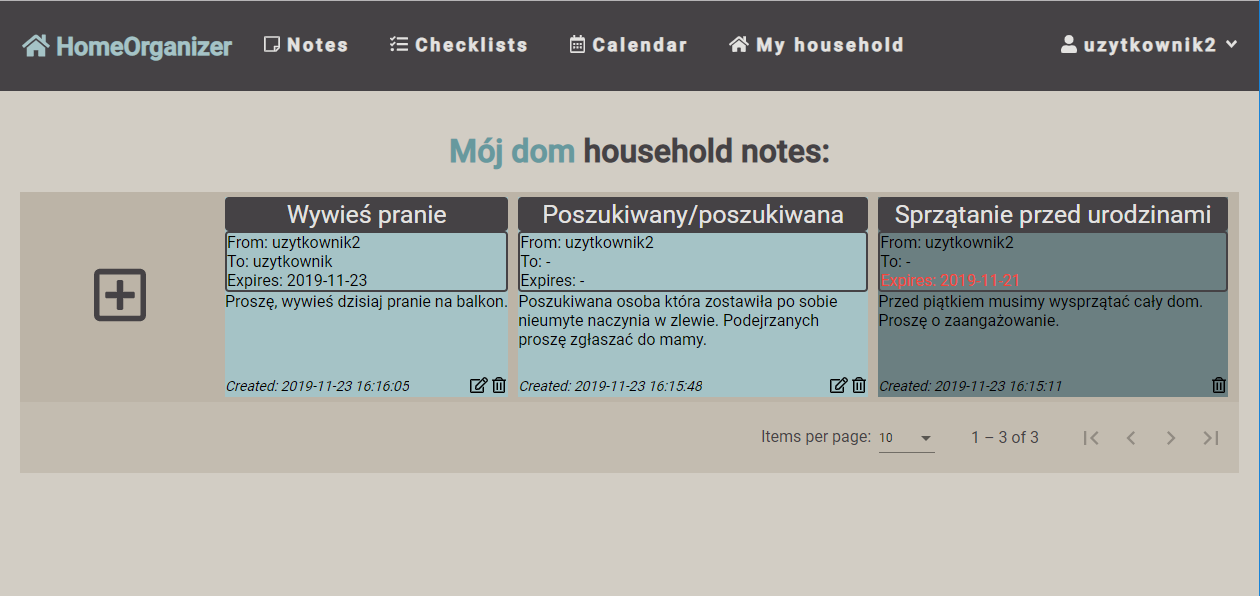
Rys.8 *Widok aktualnego household’u*

W widoku na rysunku 8 wyświetlają się szczegóły aktualnego household’u, to jest: nazwa (Mój dom), identyfikator, który możemy skopiować po kliknięciu na ikonę po jego prawej, opis (Witajcie w …) oraz lista członków. W prawym górnym roku, po kliknięciu w otwarte drzwi, jest możliwość opuszczenia domu.

### Notatki

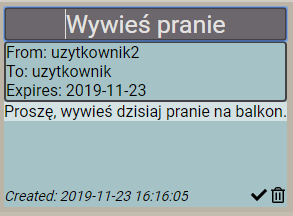
Notatki są krótkimi wiadomościami skierowanymi do całego household’u lub konkretnego jego członka. Można nimi zarządzać przez wejście z zakładkę **Notes**. Przykładowe dodane notatki przedstawione są na rysunku 9. Każda notatka zawiera tytuł, twórcę, odbiorcę, datę wygaśnięcia oraz wiadomość. Jeśli termin notatki się skończył, staje się ona ciemniejsza (ostatnia notatka na rysunku 9) i nie można jej już edytować. Każdą karteczkę można edytować (zmienić można tylko tytuł i wiadomość, przykład na rysunku 10) oraz usuwać przez ikony w prawym dolnym rogu. Uczynić to może tylko i wyłącznie jej twórca. Po usunięciu notatki, na dole strony pojawi się o tym odpowiednia wiadomość, wraz z przyciskiem do cofnięcia operacji (rys. 11). Jeśli odbiorca lub data wygaśnięcia nie zostaną ustalone, wyświetlany jest w odpowiednim miejscu znak „**-**” (druga notatka na rysunku 9).

W przypadku większej liczby notatek są one dzielone na strony. Ich ilość na jednej stronie możemy ustalić przez zmianę parametru „Items per page”.

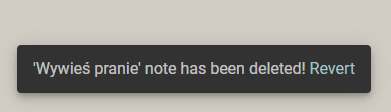


Rys.9 *Zakładka notes*

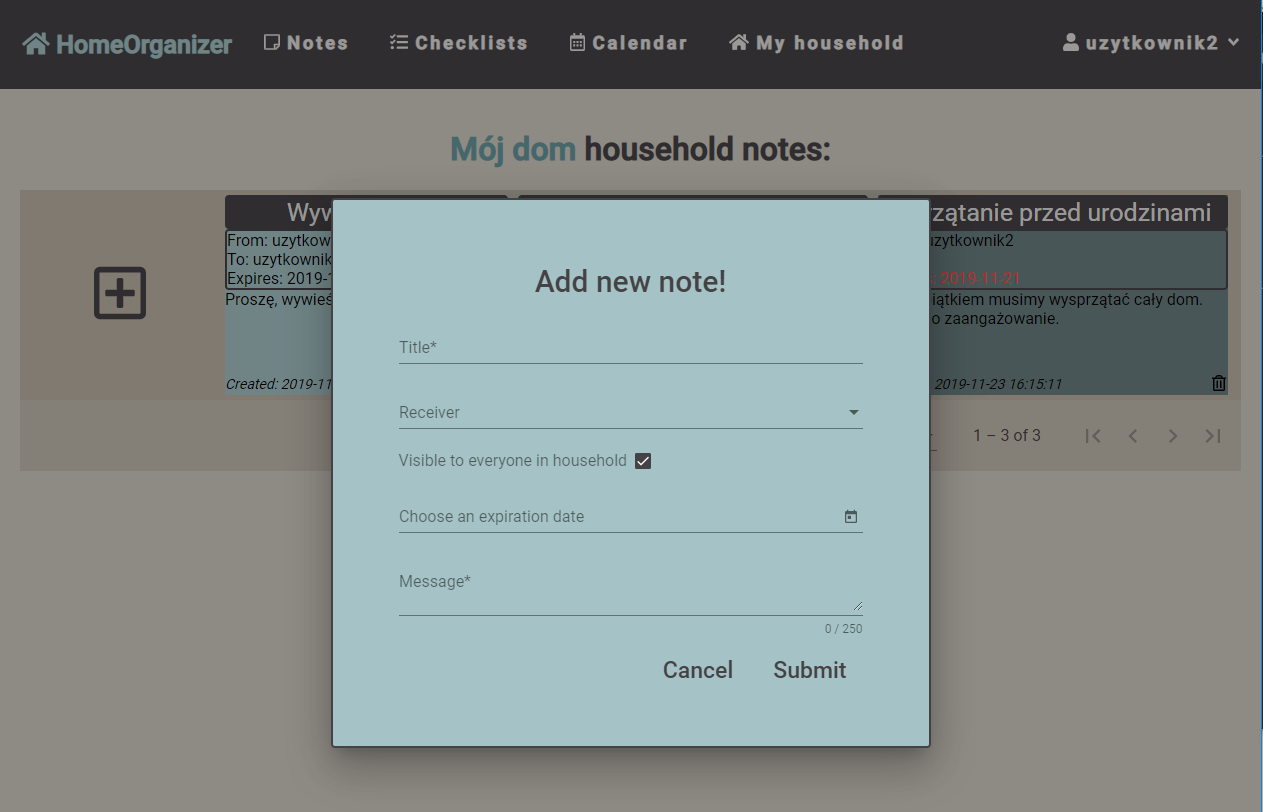
Plus po lewej stronie rysunku 9, umożliwia użytkownikowi dodanie nowej notatki. Po naciśnięciu przycisku, wyświetla się nowe okno przedstawione na rysunku 12. W kolejne pola użytkownik wpisuje kolejno: tytuł, opcjonalnie odbiorcę oraz datę wygaśnięcia, a także wiadomość (do 250 znaków). Domyślnie, odbiorcą jest cały household. Zmiana tego pola, niesie za sobą również możliwość ustalenia czy wiadomość ma być widoczna tylko dla odbiorcy, czy również dla reszty członków domu. Po wpisaniu wymaganych informacji, dodanie notatki następuje po naciśnięciu przycisku **Submit**.



Rys.10 *Edytowanie notatki*



Rys.11 *Wiadomość po usunięciu wiadomości*

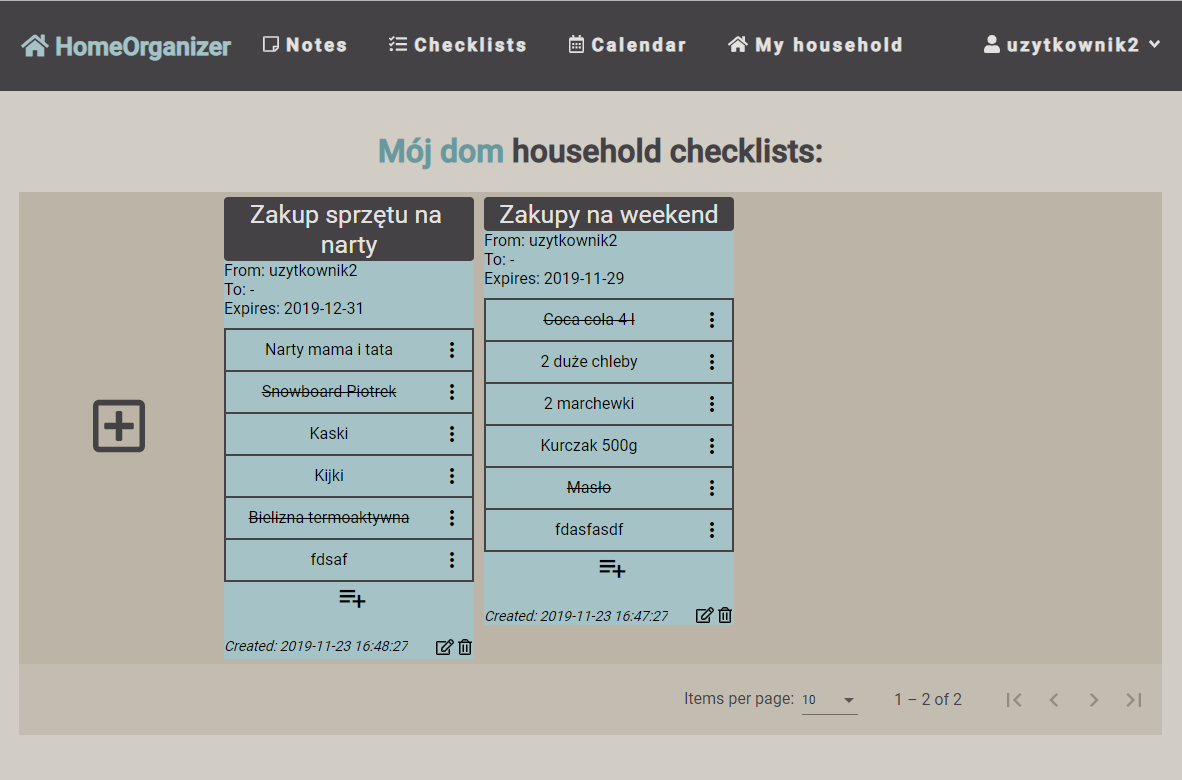


Rys.12 *Okno dodawania nowej notatki*

### Listy

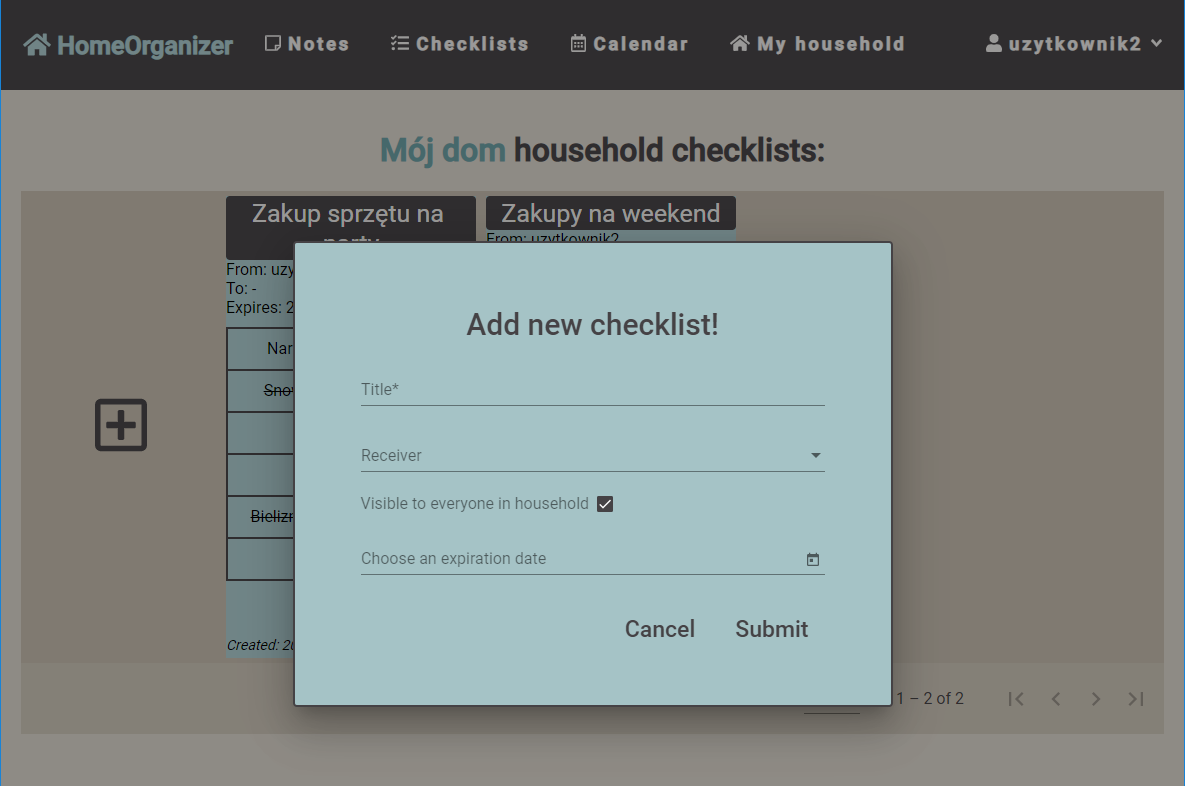
Listy są listami kontrolnymi tworzonymi przez cały household lub konkretnego jego członka. Przykładem zastosowania takiej struktury może być np. wspólna lista zakupów. Panel zarządzania nimi dostępny jest po wejściu w zakładkę **Checklists** (rysunek 13). Każda lista zawiera tytuł, twórcę, odbiorcę, datę wygaśnięcia oraz listę przedmiotów (ang. Item). Jeśli odbiorcą jest cały household (znak „**-**”), wtedy edytować oraz dodawać nowe elementy może każdy. W sytuacji jednak, gdy adresatem jest konkretna osoba, tylko on i twórca może zmieniać listę.

Cała checklist’a może być edytowana (tytuł) oraz usunięta (z możliwością przywrócenia jak w notatkach, rysunek 11). Przedmiot natomiast może być skreślony (odkreślony), edytowany oraz usunięty. Nowe elementy dodaje się znakiem notatki z plusem znajdującym się na dole spisu przedmiotów, natomiast nową listę – plusem w kwadracie. Listy są również stronnicowane.



Rys.13 *Zakładka checklists*

Po naciśnięciu przycisku dodania nowej listy, wyświetla się nowe okno przedstawione na rysunku 14. Pola formatki są analogiczne do notatek, a chęć dodania potwierdzana jest przyciskiem **Submit**.

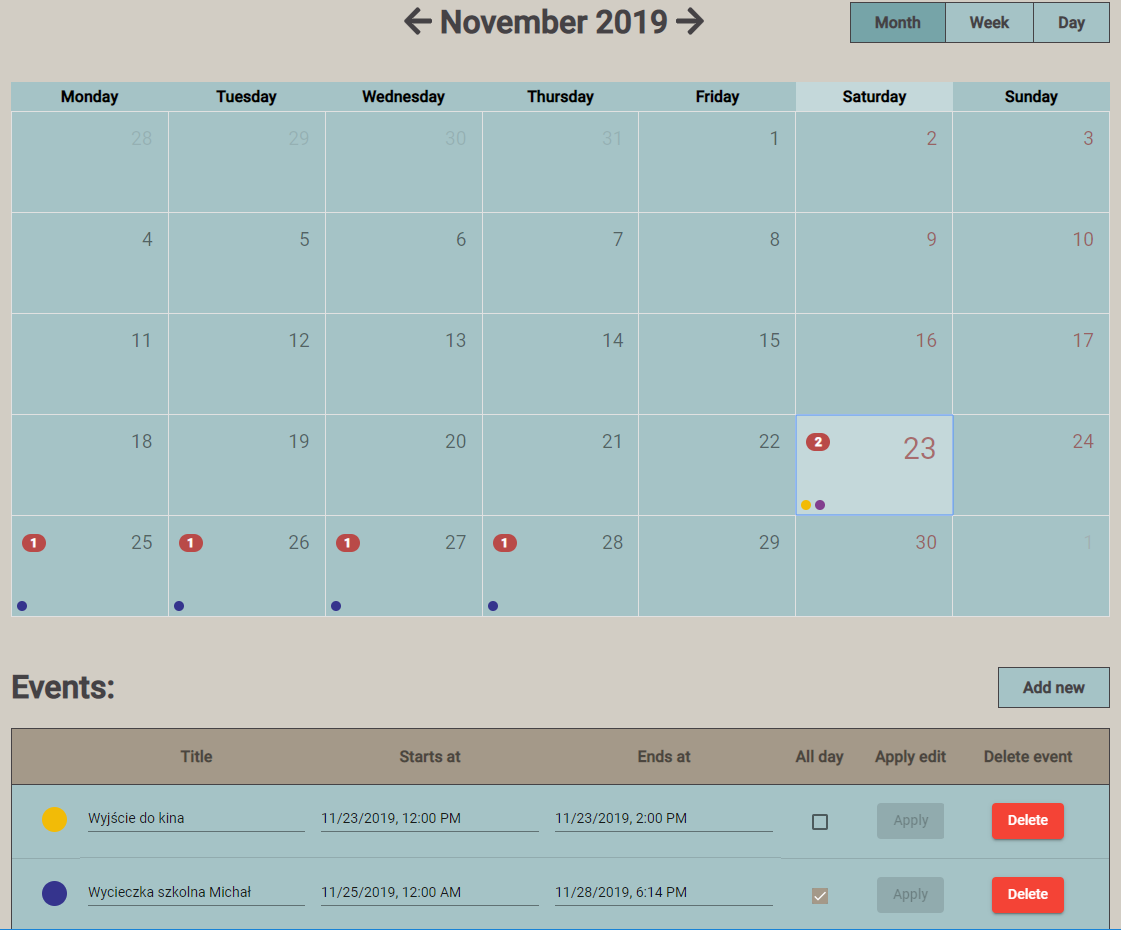


Rys.14 *Okno dodawania nowej listy*

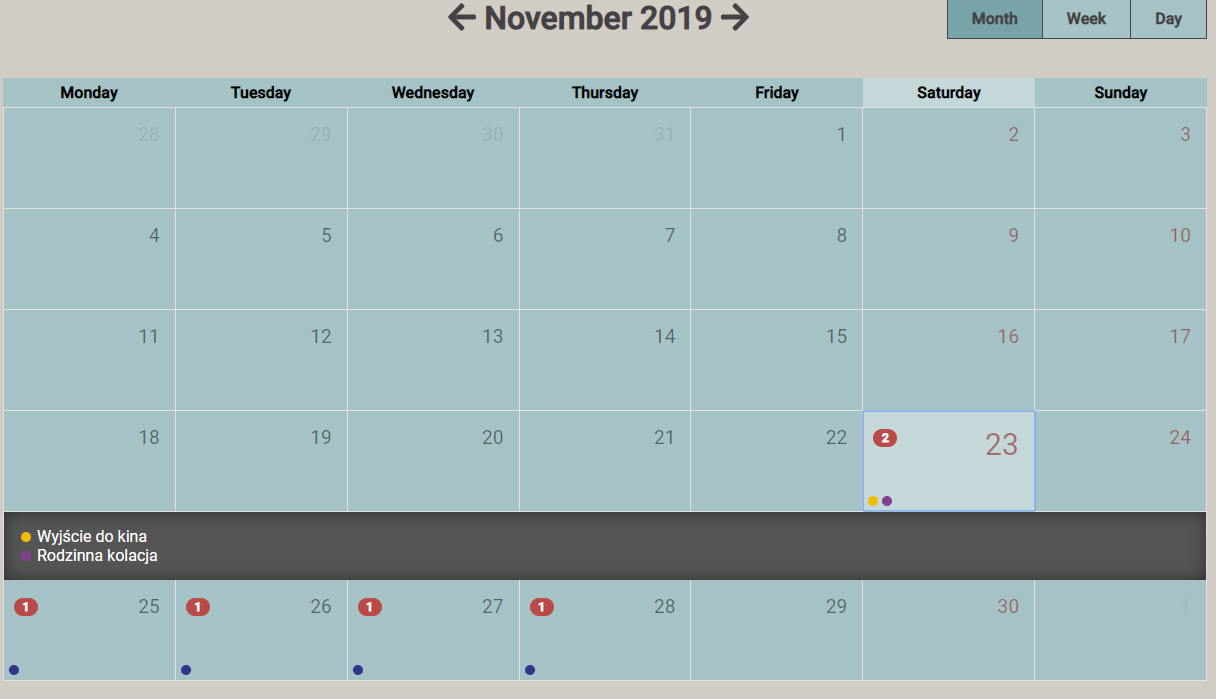
### Kalendarz

Do wspólnego kalendarza można dostać się przez wybranie zakładki **Calendar**. Umożliwia on dodawanie nowych wydarzeń (ang. Event) (również tych trwających parę dni) oraz przeglądanie ich w widoku miesięcznym, tygodniowym oraz dniowym. Jak widać na rysunku 15 (widok miesięczny), wydarzenia zaznaczane są na kalendarzu kolorowymi kropkami o kolorach takich jak w tabelce poniżej. Dodanie nowego wydarzenia następuje po naciśnięciu przycisku **Add new**. Możemy je następnie edytować i zapisać zmiany przyciskiem **Apply** lub usunąć przyciskiem **Delete**. Jeśli event trwa przez pare dni, na każdym dniu w kalendarzu pojawi się o nim informacja. Przeglądany okres może być zmieniany przez strzałki nad kalendarzem.

Po naciśnięciu na dzień, w którym jest już zaplanowane jakieś zdarzenie, wyświetla się ich lista z nazwami (rys. 16). Wydarzenia z tej listy można przeciągać na inny dzień, by zmienić ich dzień startu.

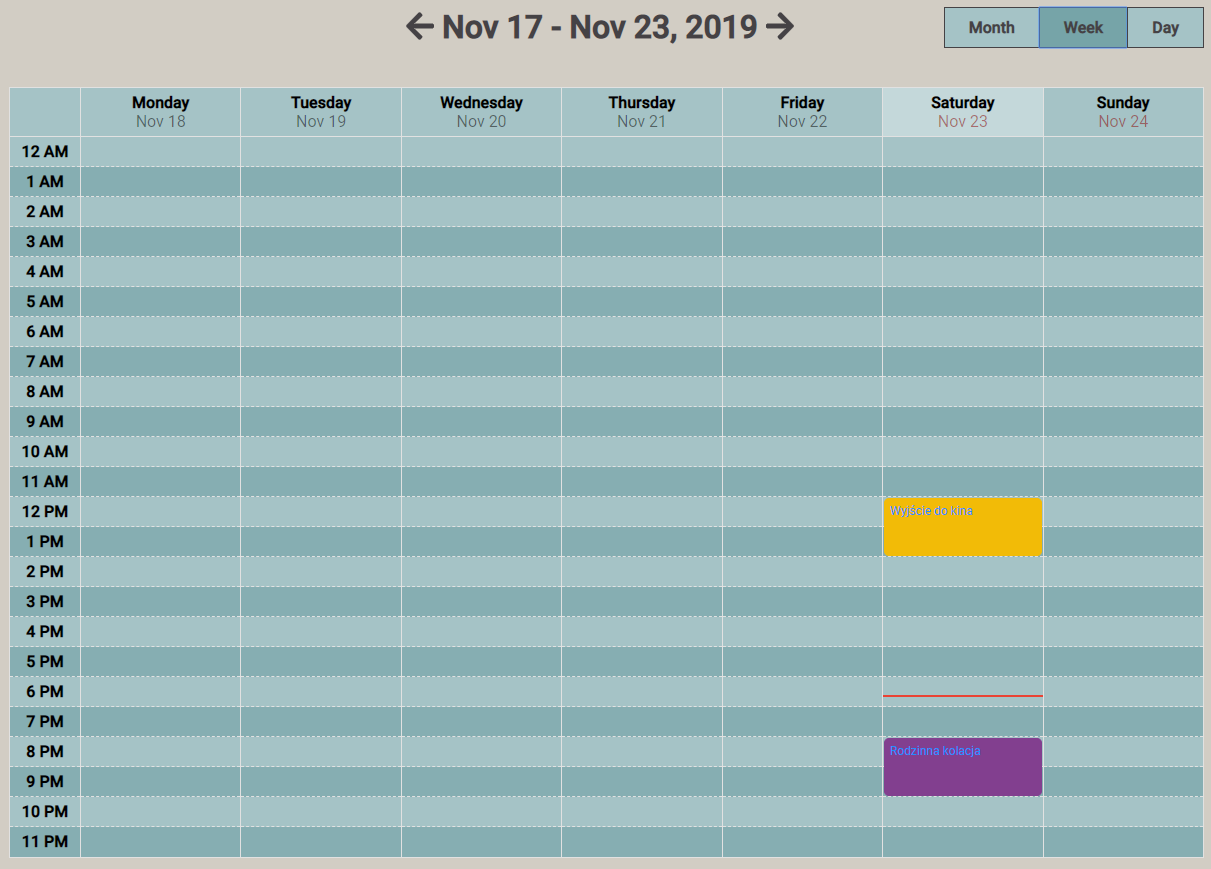


Rys.15 *Zakładka calendar, widok miesięczny*

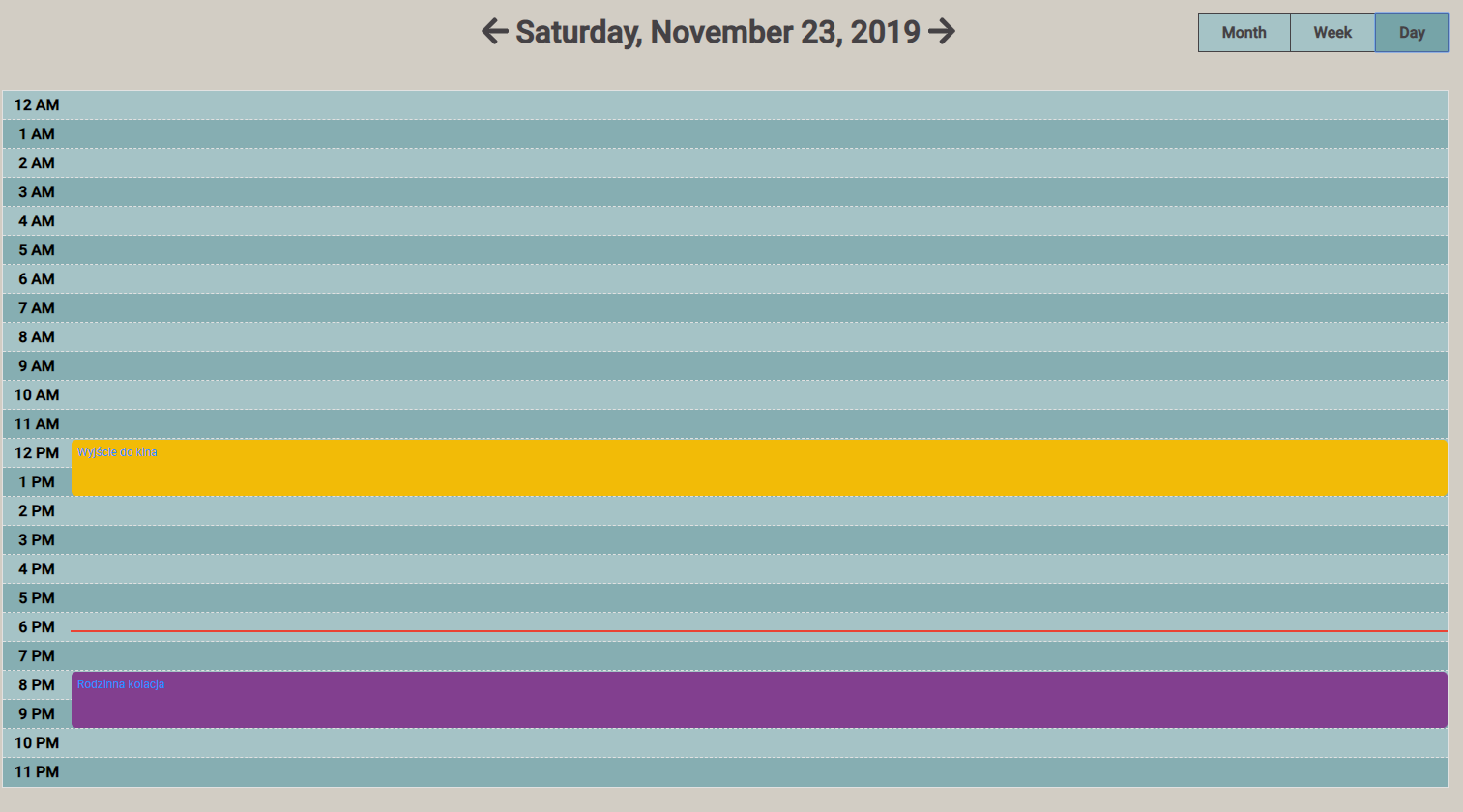


Rys.16 *Lista wydarzeń w dany dzień*

Kalendarz można również przełączać przyciskami w prawym górnym roku na widok miesięczny, tygodniowy (rys.17) i dzienny (rys.18).



Rys.17 *Kalendarz – widok tygodniowy*

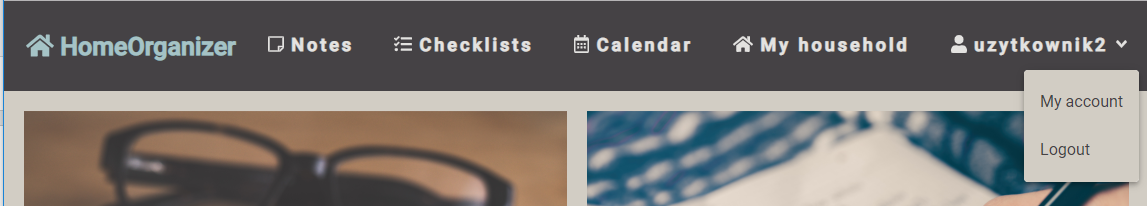


Rys.18 *Kalendarz – widok dzienny*

Zarówno w widoku dziennym jak i tygodniowym, wszystkie wydarzenia mogą być skracane/wydłużane przez przeciągnięcie górnej lub dolnej krawędzi bloczka, jak również przeciągane na inne godziny lub dni.

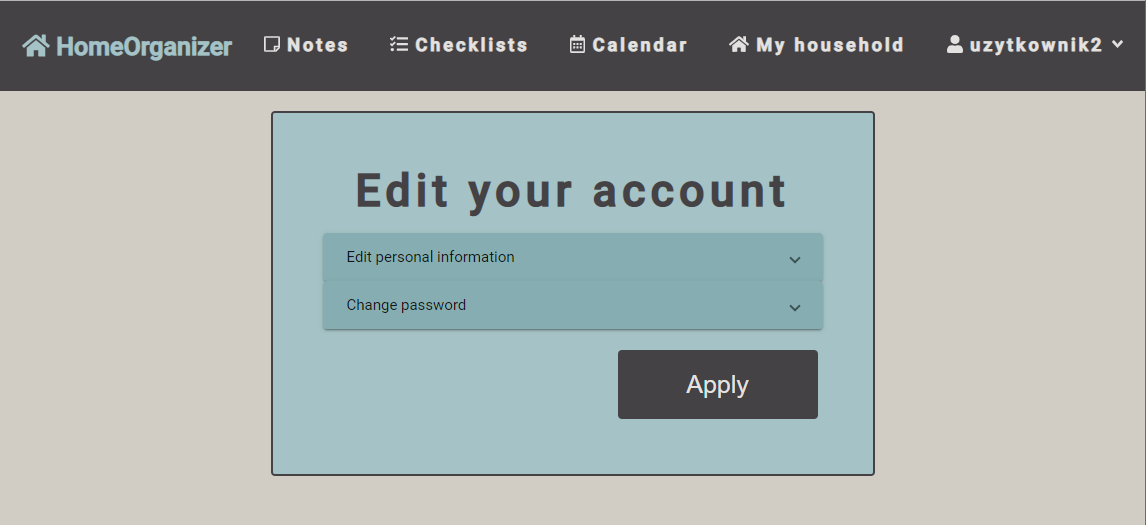
### Edycja konta i wylogowanie

Użytkownik w panelu edycji może zmienić wszystkie swoje dane, łączenie z hasłem. By dostać się do tej zakładki, należy nacisnąć na nazwę swoją nazwę użytkownika w prawym górnym rogu i z listy rozwijanej wybrać pozycje **My account** (rysunek 19). W liście tej znajduje się również pozycje **Logout** umożliwiająca wylogowanie się użytkownika.



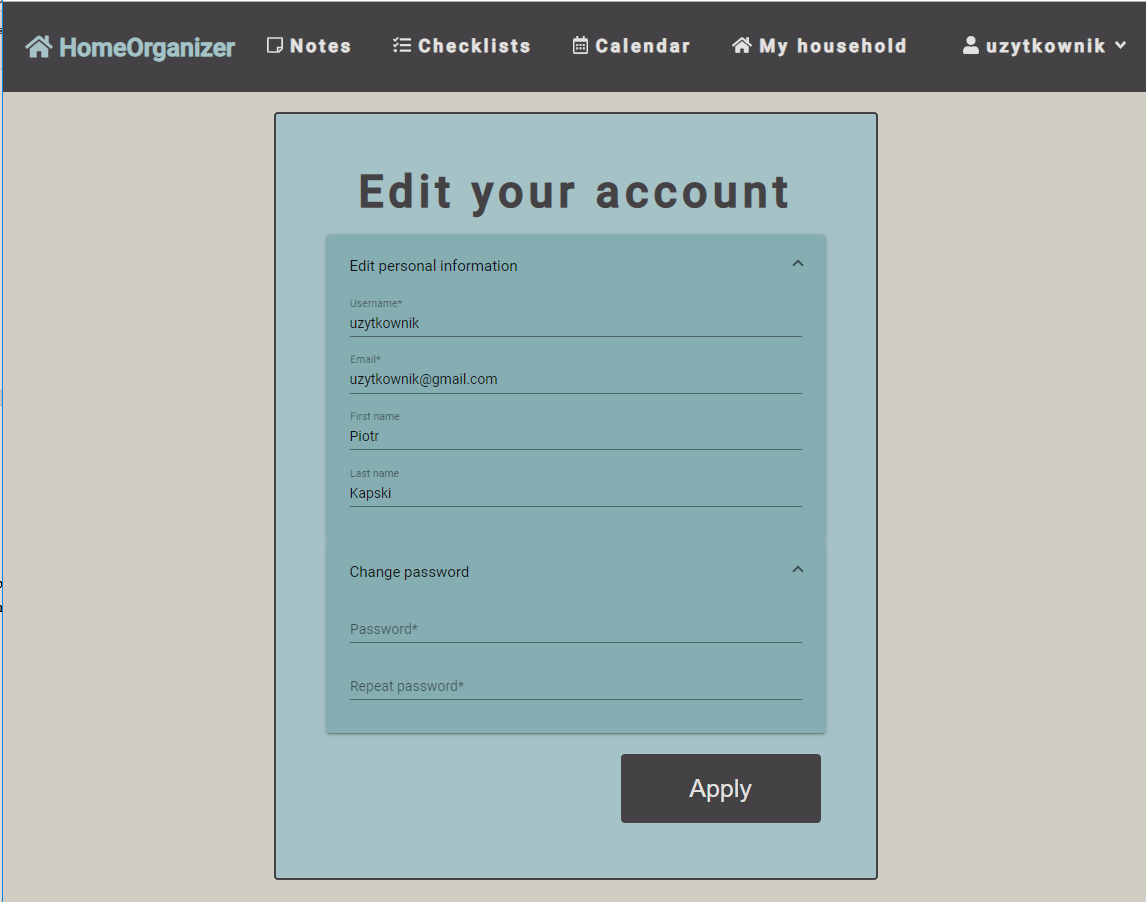
Rys.19 *Lista rozwijana do edycji i wylogowania się*

Po wybraniu opcji edycji pojawi się okno podzielone na 2 kategorie: edycje danych personalnych oraz zmiana hasła, jak na rysunku 20. Mimo iż są one oddzielone, możliwa jest jednoczesna zmiana danych w obu kategoriach.



Rys.20 *Okno edycji danych – zamnkięte*

Po rozwinięciu zakładek, możliwa jest edycja danych (rys.21). Należy pamiętać, że przy edycji trzeba zachować wszystkie reguły jak przy rejestracji, np. nazwa użytkownika musi mieć minimum 5 znaków i być unikatowa. Po uzupełnieniu wszystkich chcianych pól, konieczne jest potwierdzenie przyciskiem **Apply**. Jeśli wystąpił błąd przy edycji, zostanie wyświetlony odpowiedni błąd. W przeciwnym wypadku, na dole strony wyświetli się potwierdzenie sukcesu.



Rys.21 *Okno edycji danych – otwarte*

# Specyfikacja wewnętrzna

Rozdział ten zawiera takie elementy, jak:

* przedstawienie idei,
* architektura systemu,
* opis struktur danych (i organizacji bazy danych),
* komponenty, moduły, biblioteki, przegląd ważniejszych klas (jeśli występują),
* przegląd ważniejszych algorytmów (jeśli występują),
* szczegóły implementacji wybranych fragmentów, zastosowane wzorce projektowe
* diagramy UML.

Krótka wstawka kodu w linii tekstu jest możliwa, np. class Main. Dłuższe fragmenty należy zamieszczać zapisywać czcionką *Courier* lub *Courier New* o wielkości 10 umieszczać w ramkach (Listing 4.1) z odstępem między wiersza­mi o war­tości 1. Wszystkie linie kodu powinny być ponumerowane tak, aby można było do nich odwoływać się w treści dokumentu.

*Listing 1. Generowanie liczb losowych*

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | **package** polsl.iinf.lab;  **import** java.util.Random;  **public** **class** Main {  **public** **static** **void** main(String[] args) {  Random r = **new** Random();  // losowanie liczby z zakresu 1..10  **int** a = r.nextInt(10 + 1);  System.*out*.println(a);  // Losowanie liczby z zakresu -5..15  System.*out*.println(r.nextInt(21) - 5);  }  } |

# Weryfikacja i walidacja

Rozdział ten zawiera takie elementy, jak:

* sposób testowania w ramach pracy (np. odniesienie do modelu V),
* organizacja eksperymentów,
* przypadki testowe i zakres testowania (pełny/niepełny),
* wykryte i usunięte błędy,
* opcjonalne wyniki badań eksperymentalnych.

# Podsumowanie i wnioski

Rozdział ten obejmuje następujące elementy:

* uzyskane wyniki w świetle postawionych celów i zdefiniowanych wymagań,
* kierunki ewentualnych dalszych prac (rozbudowa funkcjonalna, …),
* problemy napotkane w trakcie pracy.

# Bibliografia

|  |  |
| --- | --- |
| [1] | Simon Kemp. *Digital 2019: Global Internet Use Accelerates – We Are Social*. https://wearesocial.com/blog/2019/01/digital-2019-global-internet-use-accelerates [data dostępu: 2019-11-20] |
| [2] | *Spring framework*. <https://spring.io/> [data dostępu: 2019-11-22] |
| [3] | *Angular* <https://angular.io/> [data dostępu: 2019-11-22] |
| [4] | *MongoDB* <https://www.mongodb.com/> [data dostępu: 2019-11-22] |
| [5] | *Robo 3T* <https://robomongo.org/> [data dostępu: 2019-11-23] |

# Spis skrótów i symboli

|  |  |
| --- | --- |
| *DNA* | kwas deoksyrybonukleinowy (ang. *deoxyribonucleic acid*) |
| *MVC* | model – widok – kontroler (ang. *model–view–controller*) |
| *N* | Liczebność zbioru danych |

# Zawartość dołączonej płyty

Na płycie DVD dołączonej do dokumentacji znajdują się następujące materiały:

* praca w formacie pdf,
* źródła programu,
* zbiory danych użyte w eksperymentach.

# Spis rysunków

# Spis tabel