

System Przypominajek

Dokumentacja

Wykonawcy:

Bartłomiej Jędrzejczyk

Jakub Brzegowski

Angelika Chwastek

Spis treści

Zagadnienie biznesowe	3
Wymagania systemowe i funkcjonalne	4
1. Logowanie i konto użytkownika	4
2. Tworzenie notatek i przypomnień	4
3. Wyświetlanie przypomnień	4
4. Baza danych i utrwalanie informacji	4
5. Model architektury	5
Harmonogram prac i zespół projektowy	6
Analiza zagadnienia i jego modelowanie	7
Implementacja i testowanie	8
1. Omówienie fragmentów kodu	8
2. Testy	9
Podsumowanie	11

Zagadnienie biznesowe

Naszym zagadnieniem biznesowym jest System Przypominajek, po angielsku RemiSystem, mający zaspokoić potrzebę organizacji czasu w coraz bardziej zabieganym świecie. Chcemy stworzyć prosty w obsłudze i lekki serwis, który pozwoli użytkownikowi na zapisywanie notatek oraz dodawanie do nich przypomnień, mających dotrzeć do niego w odpowiednim czasie. System powinien być w stanie osiągnąć użytkownika nawet jeżeli ten go aktywnie nie używa, co chcemy osiągnąć przy pomocy opcji powiadomień przez e-mail.

Proponowana aplikacja zaspokaja wspomnianą potrzebę, ale mogłaby też być wykorzystana w większym systemie jako jedna z funkcji, np. w elektronicznym kalendarzu.

Wymagania systemowe i funkcjonalne

1. Logowanie i konto użytkownika

Możliwość zalogowania się na podstawie nazwy użytkownika oraz hasła. Do każdego konta powinien być przypisany adres e-mail. Użytkownik ma możliwość zmiany hasła oraz ustawiania preferencji: czy chce otrzymywać powiadomienia mailowo czy wyłącznie jako pop-up.

2. Tworzenie notatek i przypomnień

Użytkownik zalogowany ma możliwość tworzenia notatek. Notatka powinna zawierać: tytuł oraz opcjonalnie treść i powinna być automatycznie przypisywana danemu użytkownikowi. Nie ma możliwości przypisywania notatek innym użytkownikom.

Po stworzeniu notatki użytkownik ma możliwość dodania do niej przypomnień. Przypomnień może być wiele i mogą być jednym z dwóch typów:

- **zwykłe** – zawiera datę oraz godzinę (z dokładnością do minuty) wyświetlenia przypomnienia.
- **cykliczne** – stworzone w oparciu o format CRON. Będą wyświetlane wielokrotnie w ustalonym przez użytkownika czasie, np. cotygodniowo.

Użytkownik zalogowany ma również możliwość edycji oraz usuwania zarówno swoich notatek jak i przypomnień z nimi powiązanych.

3. Wyświetlanie przypomnień

System wyświetla przypomnienia w formie pop-up w przeglądarce o odpowiednim czasie z dokładnością do minuty. Jeżeli użytkownik wybrał sobie taką opcję w preferencjach, system dodatkowo wysyła powiadomienie na adres e-mail użytkownika.

4. Baza danych i utrwalanie informacji

Firebase będzie przechowywać informacje o użytkownikach takie jak: login, hasło, email. Będzie zwracać id, co umożliwi odnoszenie się do użytkownika w bazie danych.

Baza danych będzie przechowywać informacje o notatkach i przypomnieniach.

Tabela NOTE:

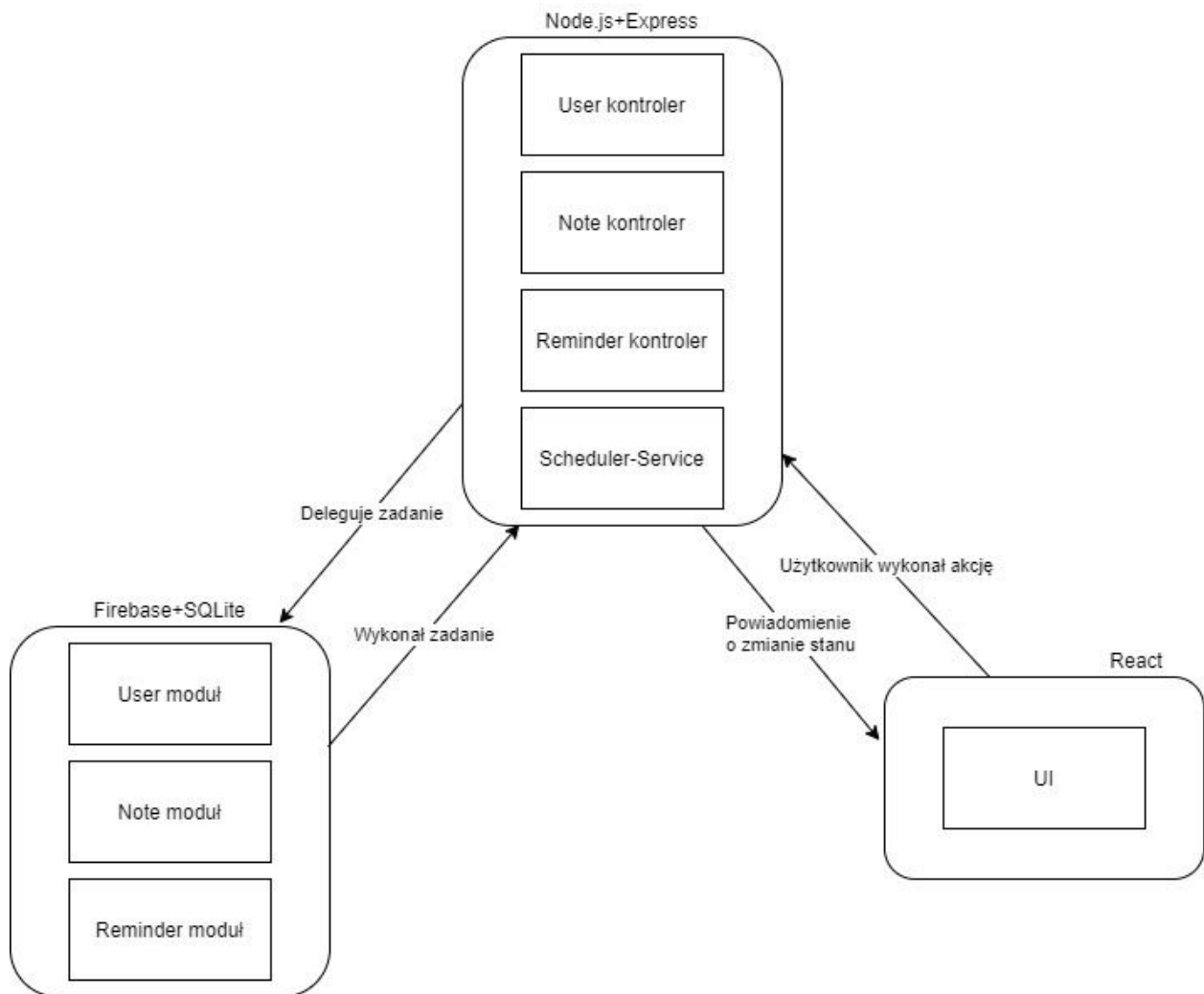
- id – varchar, primary key

- title – varchar, tytuł notatki
- content – varchar, zawartość notatki. Pole nieobowiązkowe.
- user_id – varchar, wskazuje na użytkownika, który stworzył notatkę

Tabela REMINDER:

- id – varchar, primary key
- time – varchar (cron w formie stringa), zawiera albo datę przypomnienia albo harmonogram przypominania
- note_id – int, foreign key; wskazuje na notatkę, której dotyczy przypomnienie

5. Model architektury

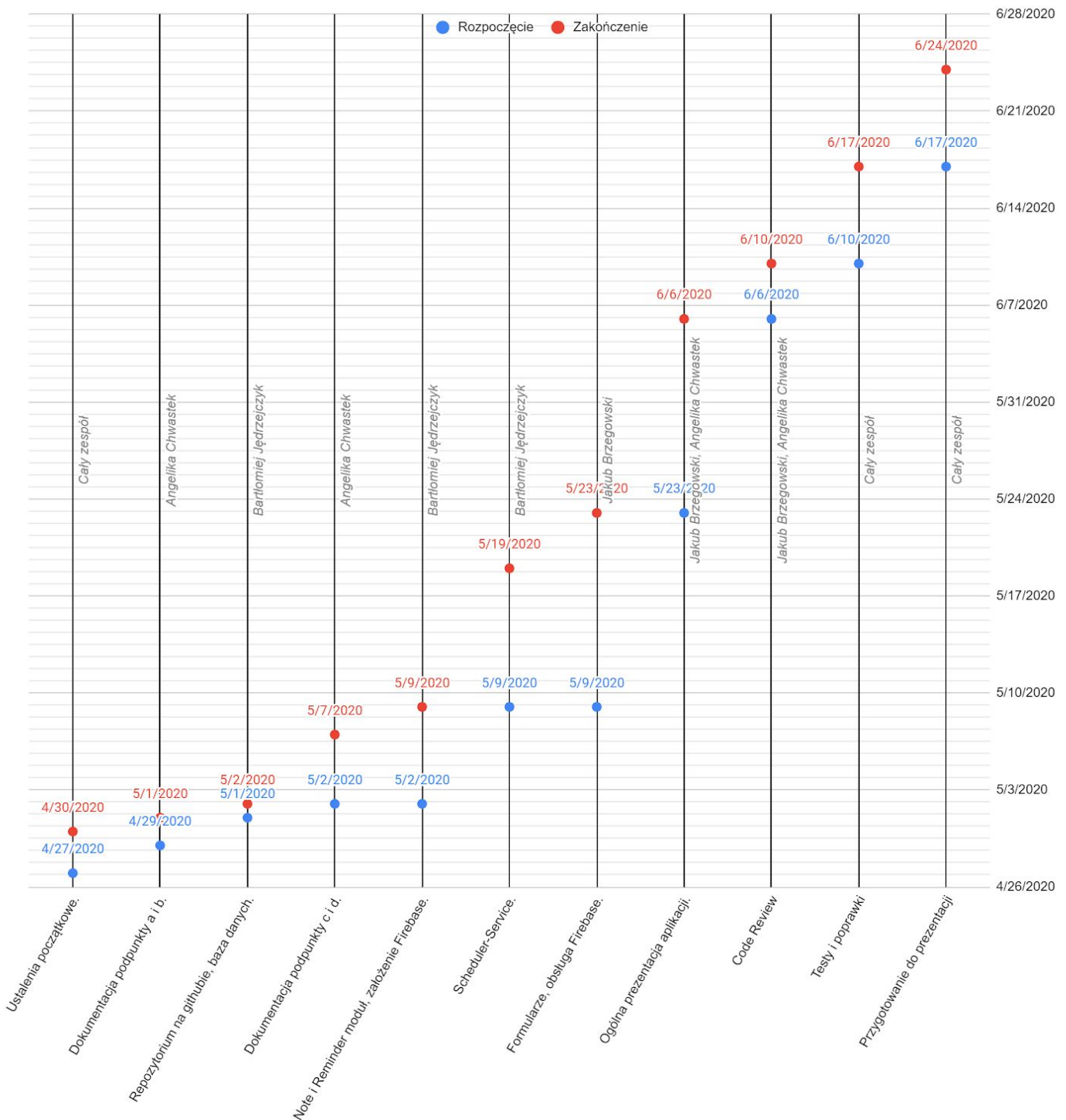


Harmonogram prac i zespół projektowy

Bartek Jędrzejczyk – Tech-Lead i backend.

Jakub Brzegowski – frontend.

Angelika Chwastek – fullstack i dokumentacja.



Analiza zagadnienia i jego modelowanie

System przypominajek posiada dwa własne moduły: Notatki i Przypomnienia.

Przypomnienia zawierają informacje o notatce, której dotyczą, oraz o czasie, kiedy powinny być wyświetlane, sformatowaną w cron, co pozwala na tworzenie przypomnień zarówno jednorazowych jak i cyklicznych.

Notatki zawierają tytuł oraz opis – który może być traktowany jako ich zawartość czy też rozwinięcie – a ponadto informację o tym, który użytkownik je stworzył.

System używa Firebase'a do obsługi trzeciego modułu – użytkowników. Przechowuje on informacje na temat adresu e-mail użytkownika, jego hasła, jego pseudonimu, a także o jego ustawieniach systemowych.

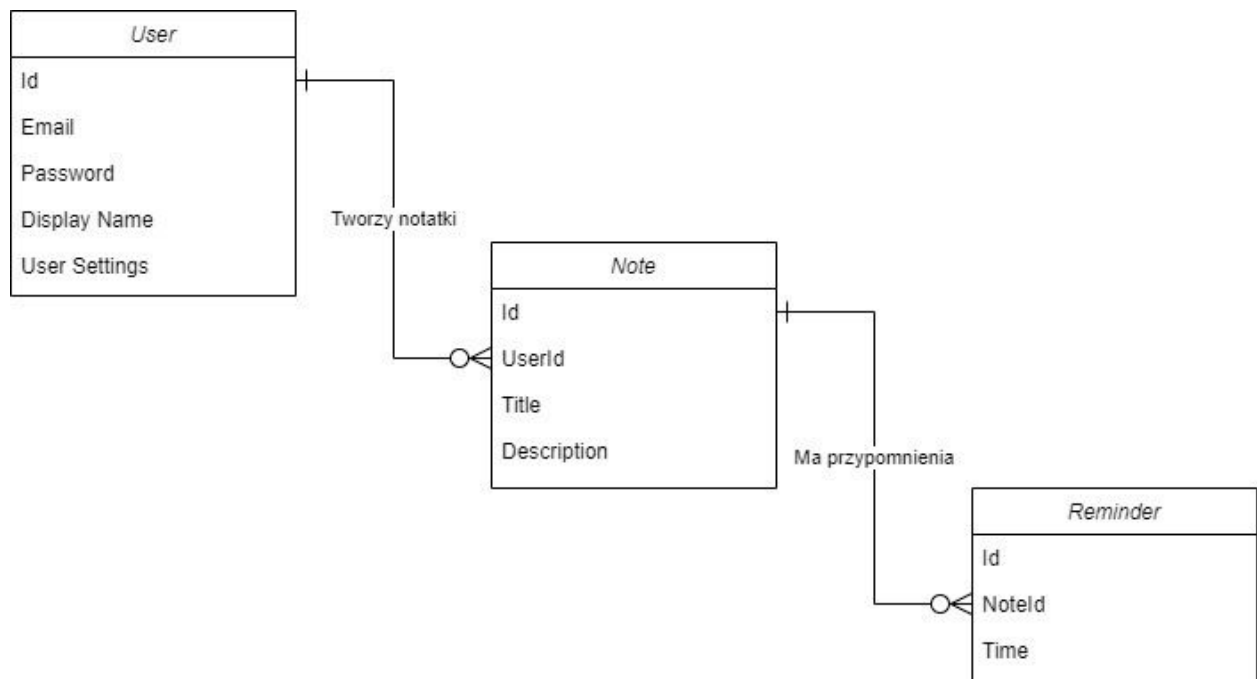


Diagram ilustrujący zależności pomiędzy modułami

Identyczna struktura Notatek i Przypomnień używana jest w bazie danych utworzonej przy pomocy SQLite. Notatki posiadają również pole **UserId**, które wskazuje na użytkownika przy pomocy id wygenerowanego przez Firebase.

Implementacja i testowanie

1. Omówienie fragmentów kodu

System Przypominajek znajduje się w całości w katalogu *remisystem*, gdzie podzielony jest na dwie części. Backend zawarty jest w katalogu *api*, natomiast frontend w pozostałych katalogach.

System Przypominajek korzysta z Firebase do tworzenia użytkowników:

```
const createUserWithEmailAndPasswordHandler = async (
  event,
  displayName,
  email,
  password
) => {
  event.preventDefault();
  setIsDuringProcessing(true);
  setError(null);
  try {
    const userCred = await auth.createUserWithEmailAndPassword(
      email,
      password
    );
    await userCred.user.updateProfile({
      displayName,
    });
    forceUserAuthUpdate();
  } catch (authError) {
    if (componentIsMounted.current) {
      setError(authError.message);
    }
  } finally {
    if (componentIsMounted.current) {
      setIsDuringProcessing(false);
    }
  }
};
```

(fragment kodu z *remisystem/src/components/SignUp.js*)

A także do logowania:


```
const signInHandler = (event, email, password) => {
  event.preventDefault();
  setIsDuringProcessing(true);
  setError(null);
  auth
    .signInWithEmailAndPassword(email, password)
    .catch((authError) => {
      setError(authError.message);
    })
    .finally(() => setIsDuringProcessing(false));
};
(fragment kodu z remisystem/src/components/SignIn.js)
```

Pozyskiwany jest wówczas token który przekazywany jest w nagłówku requestu z frontendu do backendu, gdzie następnie jest weryfikowany przy pomocy poniższego modułu:

```
const admin = require("firebase-admin");
require('dotenv').config();
module.exports = async (request, response, next) => {
  let idToken = request.headers.authorization;
  if (!idToken) {
    return response.status(401).send("Authorization required");
  }
  try {
    idToken = idToken.replace("Bearer ", "");
    if (idToken !== process.env.INTERNAL_BEARER) {
      request.user = await admin.auth().verifyIdToken(idToken);
    }
    return next();
  } catch (e) {
    return response.status(401).send("Authorization failed");
  }
};
(fragment kodu z remisystem/api/service/authService.js)
```

2. Testy

Do testowania Systemu Przypominajek wykorzystano testy funkcjonalne. Przykładowo, poniżej fragment testu sprawdzający endpoint zwracający wszystkie notatki danego użytkownika.

```
describe('GET /api/notes-by-user', () => {
```

```

it('When there is an unauthorized request the server returns 401.', async () => {
  // Act:
  const result = await request.get('/api/notes-by-user');
  // Result
  expect(result.status).toBe(401);
});
it('When there is an authorized request, the server returns the notes of the user.', async
() => {
  // Act:
  const result = await request.get('/api/notes-by-user')
    .set('Authorization', `Bearer ${accessToken}`);
  // Result
  expect(result.status).toBe(200);
  expect(result.body.length).toBe(2);
  expect(result.body.some(note => note.id === noteId && note.title === noteTitle &&
note.description === noteDescription)).toBe(true);
  expect(result.body.some(note => note.id === noteId2 && note.title === noteTitle2
&& note.description === noteDescription2)).toBe(true);
});
});
(fragment kodu z remisystem/api/tests/api.test.js)

```

Wykonano również testy jakościowe przy pomocy usługi SonarCloud. Link poniżej:

https://sonarcloud.io/dashboard?id=jb087_remisystem&fbclid=IwAR3G1z4OCC0-Jm1flc_hlz6VCgEpcsqvqpGUPwvR98QXvib6B6ePR66WZQRM

Oprócz tego wykonywano testy manualne.

Podsumowanie

1. Osiągnięty cel

W wyniku pracy powstał działający system udostępniony na heroku pod następującym adresem:

<https://remisystem.herokuapp.com/>

Posiada on funkcjonalności tak jak opisano w dokumentacji; wszystkie zamierzone cele zostały osiągnięte. Nie napotkano przy tym większych trudności.

2. Perspektywa rozwoju

System można rozwijać o kolejne funkcjonalności i opcje. Kilka propozycji:

- umożliwienie przypisania więcej niż jednego adresu e-mail do konta i wskazywania, na który z nich powinno przychodzić dane przypomnienie,
- stworzenie wersji mobilnej systemu na smartfony i tablety,
- dodanie opcji współdzielenia notatek z innymi użytkownikami.