Dokumentacja Projektu: OrganizeMe

Letycja Niemiec, Dominika Pucyk, Paweł Sala 8 czerwca 2024

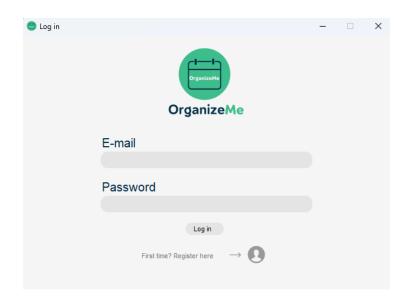
OrganizeMe - Twoja codzienna organizacja w jednym miejscu!

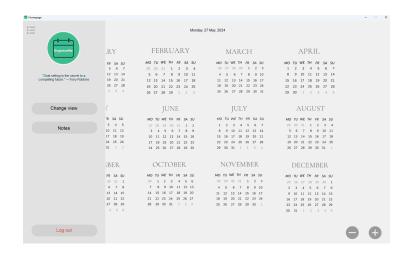


Dlaczego warto wybrać OrganizeMe?

- Intuicyjny interfejs: Planowanie codziennych zadań nigdy nie było łatwiejsze. Przejrzysty design pozwala na szybkie i łatwe dodawanie oraz zarządzanie wydarzeniami. Prosty i przejrzysty interfejs sprawia, że zarządzanie czasem staje się przyjemnością.
- Notatki w zasięgu ręki: Zapisuj ważne informacje i pomysły bezpośrednio w kalendarzu, aby zawsze mieć je pod ręką!
- Wszechstronność: Idealny zarówno do użytku osobistego, jak i zawodowego. Organizuj spotkania, terminy, ważne daty i notatki w jednym miejscu.
- Bezpieczne logowanie: Twoje dane są zawsze bezpieczne dzięki funkcji logowania. Wydarzenia, spotkania i notatki są przechowywane na Twoim osobistym koncie, co oznacza, że nigdy ich nie stracisz.

Pobierz teraz i zyskaj pełną kontrolę nad swoim harmonogramem!





1 Wprowadzenie

Projekt Kalendarz z Notatkami jest aplikacją stworzoną w języku Python z wykorzystaniem biblioteki Tkinter do interfejsu użytkownika oraz Firebase jako backendu do przechowywania danych. Aplikacja umożliwia użytkownikom logowanie się, rejestrowanie, dodawanie, pobieranie oraz usuwanie notatek.

2 Technologie

- Python główny język programowania użyty do stworzenia aplikacji.
- Tkinter biblioteka Pythona do tworzenia interfejsów graficznych.
- Firebase platforma do tworzenia aplikacji mobilnych i webowych, użyta tutaj jako backend do przechowywania danych.
- Pyrebase biblioteka Python umożliwiająca łatwą integrację z Firebase.
- python-dotenv biblioteka do zarządzania zmiennymi środowiskowymi.

3 Konfiguracja

3.1 Zmienna Środowiskowa

Projekt wykorzystuje plik .env do przechowywania danych konfiguracyjnych Firebase, takich jak apiKey, authDomain, databaseURL, projectId, storageBucket, messagingSenderId, i appId. Poniżej znajduje się przykładowa zawartość pliku .env:

```
apiKey=your_api_key
authDomain=your_project_id.firebaseapp.com
databaseURL=https://your_project_id.firebaseio.com
projectId=your_project_id
storageBucket=your_project_id.appspot.com
messagingSenderId=your_messaging_sender_id
appId=your_app_id
```

4 Działanie aplikacji

Aplikacja zawiera kilka kluczowych funkcji umożliwiających interakcję z Firebase oraz zarządzanie notatkami użytkowników. Poniżej opisane są dwie wybrane funkcje oraz działanie biblioteki doteny.

4.1 Inicjacja Firebase

```
def firebase_init():
    global auth, firebase
    try:
        firebase = pyrebase.initialize_app(firebase_config)
        auth = firebase.auth()
        return True
    except Exception as e:
        return False
```

Funkcja firebase_init() inicjalizuje połączenie z Firebase za pomocą konfiguracji zawartej w firebase_config. W przypadku pomyślnego połączenia zwraca True, w przeciwnym razie False.

4.2 Dodawanie notatki do Firebase

```
def add_note_to_firebase(title, content):
    try:
        db = firebase.database()
        db.child(user_id).push({"title": title, "content": content})
        print("Notatka_zostala_zapisana_w_firebase.")
        return True
    except Exception as e:
        print("Wystapil_blad_podczas_dodawania_notatki:", str(e))
        return False
```

Funkcja add_note_to_firebase(title, content) dodaje nową notatkę do Firebase. Wykorzystuje identyfikator użytkownika (user_id) do zorganizowania notatek w bazie danych. W przypadku sukcesu zwraca True, a w przypadku błędu False oraz wyświetla komunikat o błędzie.

4.3 firebase.database()

Funkcja firebase.database() w bibliotece Pyrebase służy do uzyskania referencji do bazy danych Firebase Realtime Database. Firebase Realtime Database to baza danych NoSQL, która przechowuje dane w formie drzewa JSON. Dzięki firebase.database() możemy uzyskać dostęp do tej bazy danych i wykonywać operacje takie jak zapisywanie, pobieranie i usuwanie danych.

4.4 db.child(user id).push()

Metoda db.child(user_id).push() służy do dodawania nowych danych do określonego miejsca w bazie danych. W tym przypadku:

- db.child(user_id) child() ustawia odniesienie do węzła w bazie danych, który odpowiada identyfikatorowi użytkownika (user_id). W Firebase Realtime Database dane są przechowywane w strukturze drzewa, a child() umożliwia nawigację do określonego węzła tego drzewa.
- push() push() tworzy nowy, unikalny klucz w węźle określonym przez child(user_id) i zapisuje pod nim przekazane dane ({"title": title, "content": content}). Nowy klucz jest generowany przez Firebase i jest unikalny w ramach tego wężła, co umożliwia przechowywanie wielu notatek dla jednego użytkownika bez nadpisywania istniejacych danych.

4.5 Działanie biblioteki doteny

Biblioteka dotenv służy do ładowania zmiennych środowiskowych z pliku .env do środowiska uruchomieniowego aplikacji. Umożliwia to przechowywanie danych konfiguracyjnych w oddzielnym pliku, co zwiększa bezpieczeństwo oraz ułatwia zarządzanie konfiguracją. W projekcie, zmienne środowiskowe są ładowane za pomocą polecenia:

```
from dotenv import dotenv_values
env_vars = dotenv_values('.env')
  Nastepnie zmienne te sa używane do konfiguracji Firebase:
firebase_config = {
    "apiKey": env_vars["apiKey"],
    "authDomain": env_vars["authDomain"],
    "databaseURL": env_vars["databaseURL"],
    "projectId": env_vars["projectId"],
    "storageBucket": env_vars["storageBucket"],
    "messagingSenderId": env_vars["messagingSenderId"],
    "appId": env_vars["appId"]
}
```

5 Podsumowanie

Projekt OrganizeMe łączy interfejs graficzny oparty na Tkinter z backendem opartym na Firebase, umożliwiając użytkownikom zarządzanie swoimi notatkami w prosty i efektywny sposób. Opisane funkcje oraz wykorzystane biblioteki pokazują, jak można bezpiecznie i efektywnie zarządzać konfiguracją aplikacji oraz integracją z zewnętrznymi usługami.