

# Dokumentacja, Design Projektu bazodanowego

## Aplikacja webowa do przeglądania i zarządzania zdjęciami użytkowników

Jakub Kogut

31 stycznia 2025

## Spis treści

<b>I Wstęp</b>	<b>3</b>
1 Opis i cel projektu . . . . .	3
2 Rodzaje użytkowników . . . . .	3
3 Wymagania Funkcjonalne dla każdego rodzaju użytkownika . . . . .	3
4 Użyte Technologie . . . . .	4
<b>II Logiczny Model Bazy Danych</b>	<b>4</b>
1 Identyfikacja encji i związków . . . . .	4
2 Opis związków . . . . .	4
3 Diagram ERD . . . . .	5
4 Opis tabel . . . . .	5
a Tabela: User . . . . .	5
b Tabela: Device . . . . .	5
c Tabela: Photo . . . . .	5
d Tabela: Tag . . . . .	6
e Tabela: PhotoTag . . . . .	6
f Tabela: Album . . . . .	6
g Tabela: AlbumPhoto . . . . .	6
5 Funkcje, Procedury, Triggery . . . . .	6
<b>III Dowód spełnienia postaci 3NF</b>	<b>6</b>
1 Definicja 3NF . . . . .	7
2 Struktura tabel i zależności funkcyjne . . . . .	7
a Tabela: User . . . . .	7
b Tabela: Device . . . . .	7
c Tabela: Photo . . . . .	8
d Tabela: Tag . . . . .	8
e Tabela: PhotoTag . . . . .	8
f Tabela: Album . . . . .	8
3 Wniosek końcowy . . . . .	9
<b>IV Zabezpieczenia i Uprawnienia</b>	<b>9</b>
1 Administrator . . . . .	9
2 Użytkownik . . . . .	9



# I Wstęp

## 1 Opis i cel projektu

Celem projektu jest stworzenie aplikacji webowej, która umożliwi użytkownikom przeglądanie i zarządzanie zdjęciami. Użytkownicy będą mogli dodawać zdjęcia, przeglądać zdjęcia innych użytkowników, dodawać komentarze. Aplikacja będzie posiadać interfejs webowy, który umożliwi użytkownikom korzystanie z aplikacji za pomocą przeglądarki internetowej. Aplikacja będzie korzystać z bazy danych do przechowywania informacji o użytkownikach, zdjęciach, komentarzach, ocenach oraz albumach zdjęć.

## 2 Rodzaje użytkowników

Ze względu na potrzeby aplikacji: różne poziomy dostępu, planuję wyodrębnić dwa rodzaje użytkowników:

- **Administrator (Admin):**

- Zarządza użytkownikami (dodawanie, usuwanie, edycja)
- Prawa odczytu, zapisu do wszystkich tabel bazy danych
- Generalne zarządzanie aplikacją: jej konfiguracja, ustawienia, itp.

- **Użytkownik (User):**

- Dodawanie własnych zdjęć
- Oglądanie zdjęć innych użytkowników (jeżeli są udostępnione)
- Dodawanie komentarzy do zdjęć

## 3 Wymagania Funkcjonalne dla każdego rodzaju użytkownika

### Administrator

1. Tworzenie, edycja, usuwanie użytkowników
2. Przeprowadzanie backupów i utrzymanie bazy danych
3. Zarządzanie ustawieniami aplikacji

### Użytkownik

1. Rejestracja, logowanie, wylogowanie
2. Dodawanie zdjęć
3. Usuwanie zdjęć
4. Zmienianie metadat zdjęć (tytuł, opis, tagi)
5. Przeglądanie zdjęć swoich oraz innych użytkowników (jeżeli są udostępnione)

## 4 Użyte Technologie

- **Frontend:** HTML, CSS, JavaScript, React.js
- **Backend:**
- **Baza Danych:** MySQL

Chcąc użyć nowych technologii, frontend strony planuje zbudować w React.js, natomiast stronę czysto bazodanową przy użyciu MySQL. Do komunikacji między frontendem a bazą danych użyję Axios.

## II Logiczny Model Bazy Danych

### 1 Identyfikacja encji i związków

- **Użytkownik (User):** Przechowuje informacje o zarejestrowanych użytkownikach aplikacji.
- **Urządzenie (Device):** Reprezentuje urządzenie użytkownika, z którego ujęto zdjęcie.
- **Zdjęcie (Photo):** Zawiera metadane zdjęcia oraz ścieżkę do pliku z obrazem.
- **Tag (Tag):** Przedstawia kategorie lub słowa kluczowe, które można przypisać do wielu zdjęć.
- **ZdjęcieTag (PhotoTag):** Tabela łącząca, która obsługuje relację wiele do wielu między Photo i Tag.
- **Album (Album):** Logiczna grupa zdjęć. Jeden użytkownik może mieć wiele albumów, a jeden album może zawierać wiele zdjęć.

### 2 Opis związków

- **Użytkownik – (1 do wielu) – Urządzenie:** Jeden użytkownik może mieć wiele urządzeń, ale każde urządzenie należy do dokładnie jednego użytkownika.
- **Użytkownik – (1 do wielu) – Zdjęcie:** Jeden użytkownik może przesłać wiele zdjęć, ale każde zdjęcie jest przesłane przez dokładnie jednego użytkownika.
- **Zdjęcie – (wiele do wielu) – Tag:** Zdjęcie może mieć wiele tagów, a jeden tag może być przypisany do wielu zdjęć. Relacja ta jest rozwiązana przez tabelę ZdjęcieTag.
- **Użytkownik – (1 do wielu) – Album:** Użytkownik może stworzyć wiele albumów, każdy album należy do jednego użytkownika.
- **Album – (wiele do wielu) – Zdjęcie:** Jeden album może zawierać wiele zdjęć, a jedno zdjęcie może się znajdować w wielu albumach.

### 3 Diagram ERD



### 4 Opis tabel

Poniżej przedstawiam opis tabel w projektowanej bazie danych.

#### a Tabela: User

- `user_id` (PK) – Klucz główny, identyfikator użytkownika.
- `username` (VARCHAR(30)) – Nazwa użytkownika.
- `password_hash` (VARCHAR(255)) – Hasz hasła użytkownika.
- `email` (VARCHAR(255)) – Adres email użytkownika.
- `role` (ENUM('admin', 'user')) – Rola użytkownika.

#### b Tabela: Device

- `device_id` (PK) – Klucz główny, identyfikator urządzenia.
- `user_id` (FK) – Klucz obcy, identyfikator użytkownika.
- `device_name` (VARCHAR(50)) – Nazwa urządzenia.
- `device_type` (ENUM('kamera', 'aparat', 'smartfon', ...)) – Typ urządzenia.

#### c Tabela: Photo

- `photo_id` (PK) – Klucz główny, identyfikator zdjęcia.
- `user_id` (FK) – Klucz obcy, identyfikator użytkownika.
- `device_id` (FK) – Klucz obcy, identyfikator urządzenia.
- `photo_path` (VARCHAR(255)) – Ścieżka do pliku z obrazem.

- `upload_time` (TIMESTAMP) – Czas przesłania zdjęcia.
- `description` (TEXT) – Opis zdjęcia.

**d Tabela: Tag**

- `tag_id` (PK) – Klucz główny, identyfikator tagu.
- `tag_name` (VARCHAR(50)) – Nazwa tagu.

**e Tabela: PhotoTag**

- `photo_id` (FK) – Klucz obcy, identyfikator zdjęcia.
- `tag_id` (FK) – Klucz obcy, identyfikator tagu.

**f Tabela: Album**

- `album_id` (PK) – Klucz główny, identyfikator albumu.
- `user_id` (FK) – Klucz obcy, identyfikator użytkownika.
- `album_name` (VARCHAR(50)) – Nazwa albumu.

**g Tabela: AlbumPhoto**

- `album_id` (FK) – Klucz obcy, identyfikator albumu.
- `photo_id` (FK) – Klucz obcy, identyfikator zdjęcia.

## 5 Funkcje, Procedury, Triggery

- **Procedura składowana do przesyłania zdjęcia:** Waliduje metadane pliku, sprawdza limity użytkownika oraz wstawia nowy rekord do tabeli `Photo`.
- **Funkcja zwracająca statystyki użytkownika:** Zwraca ilość zdjęć, komentarzy, tagów, itp. dla danego użytkownika.
- **Trigger do usuwania zdjęć:** Po usunięciu zdjęcia, usuwa również wszystkie komentarze, tagi oraz relacje z albumami.
- **Trigger do usuwania użytkownika:** Po usunięciu użytkownika, usuwa również wszystkie jego zdjęcia, komentarze, tagi oraz relacje z albumami.

## III Dowód spełnienia postaci 3NF

Aby zapewnić integralność danych oraz uniknąć redundancji, baza danych spełnia założenia 3NF.

W poniższej sekcji przedstawiam szczegółowy dowód, że projekt bazy danych opisany powyżej spełnia założenia **Trzeciej Postaci Normalnej (3NF)**. Warto pamiętać, że 3NF koncentruje się na wyeliminowaniu tzw. zależności przechodnich (transitive dependencies) oraz zapewnieniu, że atrybuty niekluczowe zależą bezpośrednio od klucza głównego (lub innego klucza kandydującego), a nie od innych atrybutów niekluczowych.

## 1 Definicja 3NF

Relacja (tabela)  $R$  jest w *Trzeciej Postaci Normalnej (3NF)* wtedy i tylko wtedy, gdy dla każdej nietrywialnej zależności funkcyjnej

$$X \rightarrow A$$

spełniony jest co najmniej jeden z warunków:

1.  $X$  jest *superkluczem* w  $R$ , czyli wyznacza wszystkie atrybuty w relacji,
2.  $A$  jest *atrybutem pierwotnym* (prime attribute), czyli należy do żadnego klucza kandydującego w  $R$ .

## 2 Struktura tabel i zależności funkcyjne

Poniżej przywołujemy najważniejsze tabele z określonymi kluczami głównymi oraz kluczami kandydackimi (gdzie dotyczy).

### a Tabela: User

- `user_id` (PK)
- Inne atrybuty: `username`, `password_hash`, `email`, `role`

Zależności funkcyjne:

1.  $\text{user\_id} \rightarrow (\text{username}, \text{password\_hash}, \text{email}, \text{role})$ .  
Klucz główny (`user_id`) wyznacza wszystkie pozostałe atrybuty, więc `user_id` jest superkluczem.
2.  $\text{username} \rightarrow \text{user\_id}$  (jeżeli `username` jest unikatowe).  
W takim wypadku `username` staje się kluczem kandydującym, a więc superkluczem.
3.  $\text{email} \rightarrow \text{user\_id}$  (jeżeli `email` jest unikatowe).  
Również może pełnić rolę klucza kandydującego, a więc superklucza.

**Wniosek (3NF):** Wszystkie nietrywialne zależności mają po lewej stronie klucz główny lub atrybut kluczowy (superklucz). Tabela `User` jest w 3NF.

### b Tabela: Device

- `device_id` (PK)
- Inne atrybuty: `user_id`, `device_name`, `device_type`

Zależność funkcyjna:

$$\text{device\_id} \rightarrow (\text{user\_id}, \text{device\_name}, \text{device\_type})$$

`device_id` jest kluczem głównym, czyli superkluczem, więc spełnia wymogi 3NF.

**c Tabela: Photo**

- `photo_id` (PK)
- Inne atrybuty: `user_id, device_id, photo_path, upload_time, description`

**Zależność funkcyjna:**

$$\text{photo\_id} \rightarrow (\text{user\_id}, \text{device\_id}, \text{photo\_path}, \text{upload\_time}, \text{description})$$

Klucz główny `photo_id` jest superkluczem. Zatem warunek 3NF jest spełniony.

**d Tabela: Tag**

- `tag_id` (PK)
- `tag_name`

**Zależności funkcyjne:**

$$\text{tag\_id} \rightarrow \text{tag\_name}$$

$$\text{tag\_name} \rightarrow \text{tag\_id} \quad (\text{jeżeli } \text{tag\_name} \text{ jest unikatowe})$$

Jeżeli `tag_name` jest unikatowe, może ono również stanowić klucz kandydujący. W każdym przypadku lewa strona (determinant) jest superkluczem. Tabela Tag jest w 3NF.

**e Tabela: PhotoTag**

- Klucz główny (złożony): `(photo_id, tag_id)`

**Zależności funkcyjne:**

$$(\text{photo\_id}, \text{tag\_id}) \rightarrow \dots$$

Nie ma dodatkowych atrybutów poza kluczami obcymi, a klucz złożony `(photo_id, tag_id)` jest superkluczem. Zatem tabela PhotoTag jest w 3NF.

**f Tabela: Album**

- `album_id` (PK)
- Inne atrybuty: `user_id, album_name`

**Zależności funkcyjne:**

$$\text{album\_id} \rightarrow (\text{user\_id}, \text{album\_name})$$

Klucz główny `album_id` to superklucz. Dodatkowo, jeśli w danym modelu `(user_id, album_name)` jest unikatowe (użytkownik nie może mieć dwóch albumów o tej samej nazwie), to także jest to klucz kandydujący. W obu przypadkach determinant jest superkluczem. Tabela Album jest zatem w 3NF.

### 3 Wniosek końcowy

We wszystkich przedstawionych tabelach:

- Każda nietrywialna zależność funkcyjna ( $X \rightarrow A$ ) ma determinanta  $X$  będącego superkluczem, lub atrybut  $A$  jest atrybutem pierwotnym (należy do jakiegoś klucza kandydującego).
- Oznacza to, że nie występują zależności przechodnie, w których atrybut niekluczowy zależałby pośrednio od innego atrybutu niekluczowego.

**Konkluzja:** Wszystkie tabele w projekcie spełniają założenia **Trzeciej Postaci Normalnej (3NF)**, gwarantując tym samym zminimalizowaną redundancję danych i zwiększoną integralność informacyjną.

## IV Zabezpieczenia i Uprawnienia

Zdefiniowano dwa poziomy dostępu do bazy danych: **Administrator** oraz **Użytkownik**. Każdy z tych poziomów ma określone uprawnienia do poszczególnych tabel.

### 1 Administrator

- Pełne prawa: SELECT, INSERT, UPDATE, DELETE na wszystkich tabelach.
- Prawo do tworzenia, dropowania tabel, procedur, funkcji, triggerów.
- Prawo do zarządzania użytkownikami, ustawieniami aplikacji.

### 2 Użytkownik

- Prawo do SELECT, INSERT, UPDATE, DELETE na tabelach: User, Device, Photo, Album.
- Brak dostępu do tabel: Tag, PhotoTag.

## V Podsumowanie

W powyższym dokumencie przedstawiono projekt bazy danych dla aplikacji webowej do przeglądania i zarządzania zdjęciami użytkowników. Zdefiniowano encje, związki między nimi, a także funkcje, procedury oraz triggers. Przedstawiono dowód spełnienia postaci 3NF oraz zabezpieczenia i uprawnienia dla poszczególnych rodzajów użytkowników.