# Analiza systemu wspomagającego zarządzanie zdjęciami

Grupa nr.4

|  |  |
| --- | --- |
| **Produkt etapu** | **Autor** |
| Przestawienie koncepcji systemu | Emilian Bochenek |
| Specyfikacja funkcjonalna i niefunkcjonala |  |
| Model danych | Tomasz Jarnutowski |
| Model architektury systemu | Emilian Bochenek |

Spis treści

[Analiza systemu wspomagającego zarządzanie zdjęciami 1](#_Toc121069809)

[Przedstawienie koncepcji systemu 2](#_Toc121069810)

[Opis 2](#_Toc121069811)

[Założenia systemu 3](#_Toc121069812)

[Cele biznesowe 3](#_Toc121069813)

[Plan Złoty 3](#_Toc121069814)

[Plan Srebrny 3](#_Toc121069815)

[Plan Brązowy 3](#_Toc121069816)

[Plan darmowy 3](#_Toc121069817)

[Model danych 4](#_Toc121069818)

[Tabela User 4](#_Toc121069819)

[Tabela Photo 5](#_Toc121069820)

[Tabela Album 7](#_Toc121069821)

[Tabela AlbumPhoto 7](#_Toc121069822)

[Tabela Tag 7](#_Toc121069823)

[Tabela TagPhoto 8](#_Toc121069824)

[Tabela Favourite 8](#_Toc121069825)

# 

# Przedstawienie koncepcji systemu

## Opis

Celem systemu jest zarzadzanie biblioteka zdjęć oraz uporządkowanie ich w odpowiednich wyznaczonych dla nich tagach, czy folderach. Ma to za zadanie ułatwić znalezienie oraz zapanowanie nad naszym zbiorem zdjęć.

## Założenia systemu

Głównym założeniem systemu jest przechowywanie zdjęć, dodatkowymi elementami które chcemy wprowadzić aby ułatwić zarządzanie swoją biblioteką zdjęć jest dodawanie tagów do poszczególnych fotografii oraz porządkowanie ich w folderach. Dodatkowymi elementami będzie zbieranie metadanych zdjęcia do automatycznego porządkowania zdjęć wedle lokalizacji, co planujemy zrobić w późniejszych etapach. W pierwszy etapie skupimy się na przechowywaniu zdjęć użytkownika oraz nadawaniu tagów w celu łatwiejszego sortowania.

## Cele biznesowe

Naszym głównym celem biznesowym jest wprowadzenie 3 planów pojemnościowych,

Każdy z planów będzie posiadać większa ilość GB na zdjęcia uzytkownika:

## Plan Złoty

* 1Tb na zdjęcia
* Nielimitowana ilość tagów
* Nielimitowana ilość folderów

## Plan Srebrny

* 500Gb na zdjęcia
* 50 tagów
* 50 folderów

## Plan Brązowy

* 100Gb na zdjęcia
* 25 tagów
* 15 folderów

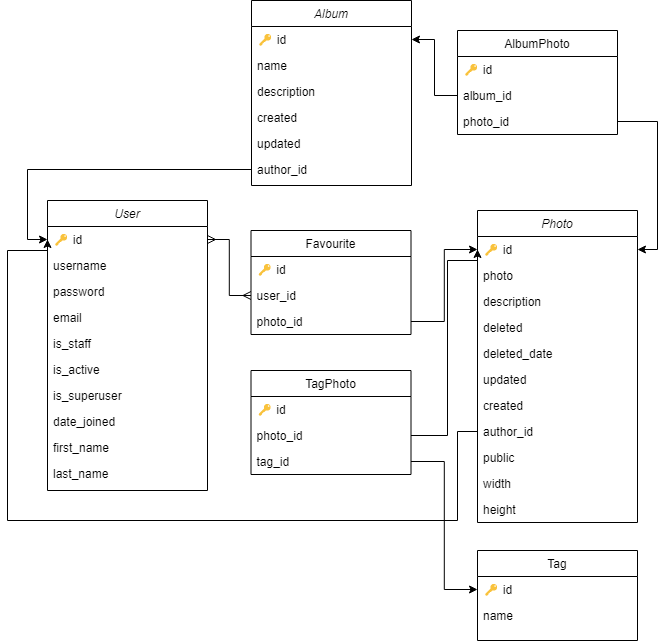
## Plan darmowy

* 15Gb na zdjęcia
* 10 predefiniowanych tagów
* 1 główny folder

Jak widzimy wyżej, każdy z planów ma ustaloną ilość dostępnej przestrzeni na zdjęcia oraz odpowiednią ilość tagów czy folderów.

Znacząco podnosząc ilość przestrzeni oraz dostępności do większej ilości tagów chcemy zachęcić użytkowników do wykupowania subskrypcji. Dodatkowo kupując subskrypcje roczną zaoferujemy 20% zniżki co pozwoli zyskać przyrost środków w początkowej fazie projektu która będziemy mogli przeznaczyć na kolejne udogodnienia.

# Model danych



## Tabela User

Tabela zawierająca dane o użytkowniku. Jest to domyślna tabela dla użytkownika w frameworku Django

|  |  |  |
| --- | --- | --- |
| **Nazwa** | **Typ** | **Opis** |
| id | Klucz główny, integer | Identyfikator |
| username | varchar, unikalne | Nazwa użytkownika |
| password | varchar | Hasło |
| email | varchar, unikalne | Email |
| is\_staff | bool | Czy jest moderatorem |
| is\_active | bool | Czy konto jest aktywne |
| is\_superuser | boll | Czy jest administratorem |
| date\_joined | datetime | Data utworzenia konta |
| first\_name | varchar | Imię |
| last\_name | varchar | Nazwisko |

## Tabela Photo

Tabela zawierająca informacje o zdjęciu. Posiada klucz obcy author\_id odnoszący się do tabeli User (Relacja typu OneToMany)

|  |  |  |
| --- | --- | --- |
| **Nazwa** | **Typ** | **Opis** |
| id | Klucz główny, integer | Identyfikator |
| photo | varchar | Ścieżka do pliku |
| description | text | Opis |
| public | boolean | Czy zdjęcie jest publiczne |
| deleted | boolean | Czy plik znajduje sie w koszu |
| deleted\_date | datetime | Data umieszczenia pliku w koszu |
| updated | datetime | Data ostatniej aktualizacji |
| created | datetime | Data utworzenia |
| author\_id | Klucz obcy, integer | Id autora zdjęcia |
| width | integer | Wysokość zdjęcia |
| height | integer | Szerokość zdjęcia |

## Tabela Album

Tabela zawierająca informacje o albumie. Posiada klucz obcy author\_id odnoszący się do jedynego autora albumu z tabeli User (Relacja typu OneToMany)

|  |  |  |
| --- | --- | --- |
| **Nazwa** | **Typ** | **Opis** |
| id | Klucz główny, integer | Identyfikator |
| name | varchar | Nazwa albumu |
| description | text | Opis |
| created | datetime | Data utworzenia |
| updated | datetime | Data ostatniej aktualizacji |
| author\_id | Klucz obcy, integer | Id autora albumu |

## Tabela AlbumPhoto

Tabiera zawierająca informację o zdjęciach przypisanych do albumu. Jest to relacja typu ManyToMany ze względu na fakt że każde zdjęcie może mieć wiele albumów

|  |  |  |
| --- | --- | --- |
| **Nazwa** | **Typ** | **Opis** |
| id | Klucz główny, integer | Identyfikator |
| album\_id | Klucz obcy, integer | ID albumu |
| photo\_id | Klucz obcy, integer | ID zdjęcia |

## Tabela Tag

Tabela słownikowa zawierające unikalne nazwy Tagów.

|  |  |  |
| --- | --- | --- |
| **Nazwa** | **Typ** | **Opis** |
| id | Klucz główny, integer | Identyfikator |
| name | varchar, unikalne | Nazwa tagu |

## Tabela TagPhoto

Tabiera zawierająca informację o tagach przypisanych do zdjęć. Jest to relacja typu ManyToMany ze względu na fakt że każde zdjęcie może mieć wiele tagów

|  |  |  |
| --- | --- | --- |
| **Nazwa** | **Typ** | **Opis** |
| id | Klucz główny, integer | Identyfikator |
| tag\_id | Klucz obcy, integer | ID tagu |
| photo\_id | Klucz obcy, integer | ID zdjęcia |

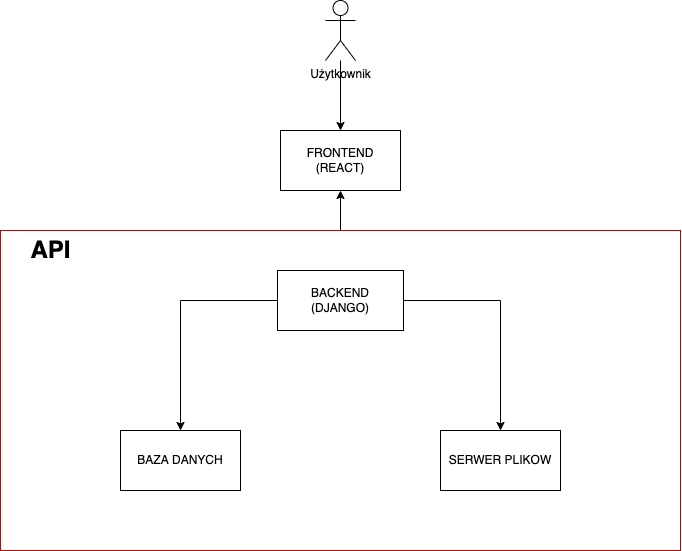
## Tabela Favourite

Tabiera zawierająca informację o zdjęciach dodanych do ulubionych. Jest to relacja typu ManyToMany ze względu na fakt że różni użytkownicy mogą dodać jedne zdjęcie do ulubionych

|  |  |  |
| --- | --- | --- |
| **Nazwa** | **Typ** | **Opis** |
| id | Klucz główny, integer | Identyfikator |
| photo\_id | Klucz obcy, integer | ID zdjęcia |
| user\_id | Klucz obcy, integer | ID użytkownika |

# Model architektury systemu

## Diagram



## Opis

### Frontend

Zaprojektowany będzie w React, będziemy dostarczać użytkownikowi wygenerowaną wcześniej stronę bez potrzeby przeprowadzania obliczeń za pomocą serwera frontendowego, zredukuje to czas załadowania strony i pozwoli nam na utrzymywanie większej ilości użytkowników.

### Backend

Napisany w django, będą tutaj wykonywane wszystkie obliczenia oraz komunikacja z baza danych oraz serwerem plików do przechowywania zdjęć.

Backend będzie tworzyć role API w naszej strukturze.