Implementacja systemu i wykonanie testów sprawdzających

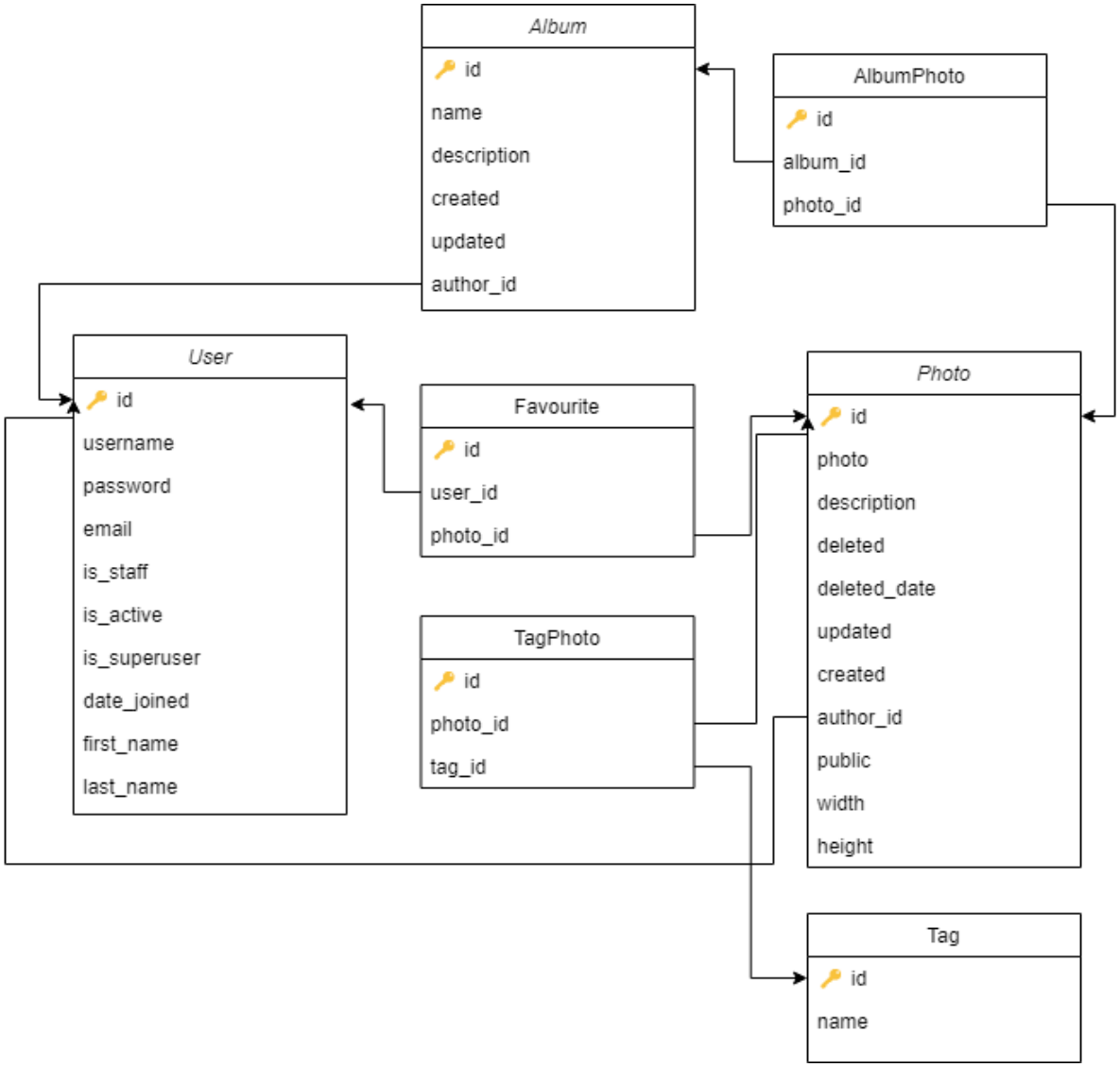
Etap IV | Grupa nr.3

Produkt etapu	**Autor**
Implemetacja bazy danych	Tomasz Jarnutowski
Implementacja warstwy logicznej	
Implementacja GUI	
Gotowy system informatyczny	
Wyniki przeprowadzonych testów	

Spis treści

1. Implementacja bazy danych

1.1. Diagram bazy danych



1.2. Sposób implementacji bazy danych

```python
from django.db import models
from django.contrib.auth.models import User

class Tag(models.Model):
    name = models.CharField(max_length=50)

    def __str__(self) -> str:
        return self.name

class Photo(models.Model):
    photo = models.ImageField()
    description = models.TextField(max_length=300, blank=True)
    tags = models.ForeignKey(Tag, on_delete=models.CASCADE)
    author = models.ForeignKey(
        User, on_delete=models.CASCADE, related_name="photo_author"
    )
    favourite = models.ManyToManyField(User)
    deleted = models.BooleanField(default=False)
    deleted_date = models.DateTimeField(null=True)
    updated = models.DateTimeField(auto_now=True)
    created = models.DateTimeField(auto_now_add=True)
    public = models.BooleanField(default=False)
    width = models.IntegerField(blank=True)
    height = models.IntegerField(blank=True)

    def __str__(self):
        return f"{self.author} at {self.created}"

class Album(models.Model):
    name = models.CharField(max_length=50)
    description = models.TextField(max_length=300, blank=True)
    favourite = models.ManyToManyField(User)
    created = models.DateTimeField(auto_now_add=True)
    updated = models.DateTimeField(auto_now=True)
    photos = models.ManyToManyField(Photo)
    author = models.ForeignKey(
        User, on_delete=models.CASCADE, related_name="album_author"
    )

    def __str__(self):
        return self.name
```

Baza została zaimplementowana z użyciem DjangoORM. Tabele są reprezentowana za pomocą klas, a zmienne odpowiadają nazw kolumn. Funkcje „__str__” pozwalają na łatwiejszy odczyt wpisów w przypadku użycia domyślnego panelu administracyjnego.

DjangoORM domyślnie sam dodaje kolumnę „ID”, dlatego nie ich w implementacji powyżej. Na powyższym podglądzie nie ma również tabeli User, jest to spowodowane faktem że w projekcie jest użyty domyślny model użytkownika który jest dostarczany przez framework. Tabele określające relacje wiele do wielu są tworzone z poziomu zmiennej klasy(models.ManyToManyField) i nie ma potrzeby tworzenia dodatkowej klasy. Relacja typu jeden do wielu jest reprezentowaną za pomocą models.ForeignKey. W przypadku pól typu dat, używam opcji „auto_now” – która aktualizację datę przy każdej edycji wpisu oraz „auto_now_add” która dodaję datę w momencie tworzenia wpisu.

2. Implementacja warstwy logicznej

3. Implementacja GUI

4. Gotowy system informatyczny

5. Wyniki przeprowadzonych testów