Технічне завдання на створення застосунку “**PhotoShare**” (REST API)

**Основний функціонал для REST API виконаний на FastAPI**

**Аутентифікація**

1. Створюємо механізм аутентифікації. Використовуємо JWT токени
2. Користувачі мають три ролі. Звичайний користувач, модератор, та адмінстратор. Перший користувач в системі завжди адміністратор
3. Для реалізації різних рівнів доступу (звичайний користувач, модератор і адміністратор) ми можемо використовувати декоратори FastAPI для перевірки токена і ролі користувача.

**Робота с світлинами**

1. Користувачі можуть завантажувати світлини з описом (POST).
2. Користувачі можуть видаляти світлини (DELETE).
3. Користувачі можуть редагувати опис світлини (PUT).
4. Користувачі можуть отримувати світлину за унікальним посиланням (GET).
5. Можливість додавати до 5 тегів під світлину. Додавання тегу не обов'язкове при завантаженні світлини.
6. Теги унікальні для всього застосунку. Тег передається на сервер по імені. Якщо такого тега не існує, то він створюється, якщо існує, то для світлини береться існуючий тег з такою назвою.
7. Користувачі можуть виконувати базові операції над світлинами, які дозволяє сервіс Cloudinary (<https://cloudinary.com/documentation/image_transformations>). Можливо вибрати обмежений набір трансформацій над світлинами для свого застосунку з Cloudinary.
8. Користувачі можуть створювати посилання на трансформоване зображення для перегляду світлини в вигляді URL та QR-code (<https://pypi.org/project/qrcode/>). Операція POST, оскільки створюється окреме посилання на трансформоване зображення, яке зберігається в базі даних
9. Створені посилання зберігаються на сервері і через мобільний телефон ми можемо відсканувати QR-code та побачити зображення
10. Адміністратори можуть робить всі CRUD операції зі світлинами користувачів

**Коментування**

1. Під кожною світлиною, є блок з коментарями. Користувачі можуть коментувати світлину один одного
2. Користувач може редагувати свій коментар, але не видаляти
3. Адміністратори та модератори можуть видаляти коментарі.
4. Для коментарів обов'язково зберігати час створення та час редагування коментаря в базі даних. Для реалізації функціональності коментарів, ми можемо використовувати відношення "один до багатьох" між світлинами і коментарями в базі даних. Для тимчасового маркування коментарів, використовувати стовпці "**created\_at**" і "**updated\_at**" у таблиці коментарів.

**Додатковий функціонал**

1. Створити маршрут для профіля користувача за його унікальним юзернеймом. Повинна повертатися вся інформація про користувача. Імя, коли зарєсттрований, кількість завантажених фото тощо
2. Користувач може редагувати інформацію про себе, та бачити інформацію про себе. Це мають бути різні маршрути з профілем користувача. Профіль для всіх користувачів, а інформація для себе - це те що можно редагувати
3. Адміністратор може робити користувачів неактивними (банити). Неактивні користувачі не можуть заходити в застосунок

**Додатково по можливості реалізувати наступні задачі, якщо дозволяє час.**

1. Реалізувати механізм виходу користувача з застосунку через logout. Access token повинен бути добавлений на час його існування в чорний список.
2. Рейтинг

* Користувачі можуть виставляти рейтинг світлині від 1 до 5 зірок. Рейтинг обчислюється як середнє значення оцінок всіх користувачів.
* Можна тільки раз виставляти оцінку світлині для користувача.
* Не можливо оцінювати свої світлини.
* Модератори та адміністратори можуть переглядати та видаляти оцінки користувачів.

1. Пошук та фільтрація

* Користувач може здійснювати пошук світлин за ключовим словом або тегом. Після пошуку користувач може відфільтрувати результати за рейтингом або датою додавання.
* Модератори та адміністратори можуть виконувати пошук та фільтрацію за користувачами, які додали світлини.

Після виконання основного функціоналу

1. Покрити застосунок модульними тестами, добитись покриття більш ніж на 90 %
2. Виконайте деплой застосунку для якогось хмарного сервісу на ваш вибір. Рекомендація Koyeb (<https://app.koyeb.com/auth/signin>) , Fly.io (<https://fly.io/app/sign-in>)

**Критерії прийому**

1. Web-застосунок реалізований на фреймворку FastAPI.
2. Проєкт має бути збережений в окремому репозиторії та бути загальнодоступним (GitHub, GitLab або BitBucket).
3. Для зберігання інформації про користувачів, світлини та коментарі використовувати PostgreSQL. Для взаємодії з базою даних, використовувати бібліотеку SQLAlchemy, яка надає ORM-функціональність для роботи з базою даних.
4. Проєкт містить докладну інструкцію щодо встановлення та використання.
5. Проєкт повністю реалізує вимоги, описані в завданні.
6. Проєкт має повну Swagger документацію

P.S.: Ви можете розширити функціонал проєкту на свій розсуд, обов'язково проконсультувавшись з ментором перед цим. Розглядайте цей проєкт, як частину вашого портфоліо і корисний вам інструмент. З цієї причини ініціатива у розширенні та доповненні вимог до проєкту вітається. Наприклад ви можете додати файл Dockerfile, щоб програма могла бути розміщена в контейнері Docker та образ завантажений на [dockerhub](https://hub.docker.com/).