Задание #2

1 часть задания — работа с БД

Необходимо написать методы для получения/удаления данных.

Создать классы для работы с отелями и номерами.
Классы `HotelDAO`/`HotelRepository`/`HotelService` и `RoomDAO`/`RoomRepository`/`RoomService` должны содержать метод для поиска по отелям и номерам:

```
class HotelDAO(BaseDAO):
    model = Hotels

    @classmethod
    async def find_all(...):
        ...


class RoomDAO(BaseDAO):
    model = Rooms

    @classmethod
    async def find_all(...):
        ...
```

`HotelDAO` и `RoomDAO` — классы для работы с моделями (классами) Hotels и Rooms. Они содержат методы для получения, добавления, изменения и удаления записей из соответствующих таблиц в БД.

Помимо работы с отелями и номерами вам также следует дописать класс по работе с бронированиями, который мы начали писать вместе. Вам необходимо добавить два метода: на получение бронирований и удаление бронирования. Подумайте, стоит ли прописывать методы `find_all(...)` (это пример названия, вы можете придумать свое) и `delete(...)` в классе по работе с бронированиями `BookingDAO` или стоит воспользоваться методами из класса `BaseDAO`. Приступайте к выполнению первой части задания только после того, как прочитаете вторую часть — они сильно связаны друг с другом.

2 часть задания — написание эндпоинтов

Необходимо написать ряд эндпоинтов для работы с отелями, номерами и бронированиями.

Получение списка отелей

Пример эндпоинта: `/hotels/Алтай`.
HTTP метод: GET.
HTTP код ответа: 200.
Описание: возвращает список отелей по заданным параметрам, причем в отеле должен быть минимум 1 свободный номер.
Нужно быть авторизованным: нет.
Параметры: параметр пути `location` и параметры запроса `date_from, date_to`.
Ответ пользователю: для каждого отеля должно быть указано: `id`, `name`, `location`, `services`, `rooms_quantity`, `image_id, rooms_left` (количество оставшихся номеров).

Примечание к этому эндпоинту: в разных источниках можно встретить разную реализацию поиска по местоположению (location): где-то это параметр пути (как у нас), а где-то параметр запроса. Вы можете самостоятельно принять решение относительно этого параметра и поместить его либо в сам URL либо поместить рядом с параметрами `date_from` и `date_to`.

Получение списка комнат

Пример эндпоинта: `/hotels/1/rooms`.
HTTP метод: GET.
HTTP код ответа: 200.
Описание: возвращает список всех номеров определенного отеля.
Нужно быть авторизованным: нет.
Параметры: параметр пути `hotel_id` и параметры запроса `date_from, date_to`.
Ответ пользователю: для каждого номера должно быть указано: `id`, `hotel_id`, `name`, `description`, `services`, `price, quantity`, `image_id`, `total_cost` (стоимость бронирования номера за весь период), `rooms_left` (количество оставшихся номеров).

Получение списка бронирований

Пример эндпоинта: `/bookings`.
HTTP метод: GET.
HTTP код ответа: 200.
Описание: возвращает список всех бронирований пользователя.
Нужно быть авторизованным: да.
Параметры: отсутствуют.
Ответ пользователю: для каждого бронирования должно быть указано: `room_id`, `user_id`, `date_from`, `date_to`, `price`, `total_cost`, `total_days`, `image_id`(для номера), `name`(для номера), `description`, `services`(для номера).

Удаление бронирования

Пример эндпоинта: `/bookings/1`.
HTTP метод: DELETE.
HTTP код ответа: 204.
Описание: удаляет бронь пользователя.

Нужно быть авторизованным: да.
Параметры: параметр пути `booking_id`.
Ответ пользователю: отсутствует.

Получение конкретного отеля (опционально, может пригодиться для фронтенда)

Пример эндпоинта: `/hotels/id/1`.
HTTP метод: GET.
HTTP код ответа: 200.
Описание: возвращает все данные по одному отелю.
Нужно быть авторизованным: нет.
Параметры: параметр пути `hotel_id`.
Ответ пользователю: для отеля должно быть указано: `id`, `name`, `location`, `services`, `rooms_quantity`, `image_id`.