HBnB testing report

1. Testing the `users.py` Endpoints:

a. Create a User (POST)

```
curl -X POST "http://127.0.0.1:5000/api/v1/users/" -H "Content-Type:
application/json" -d '{
    "first_name": "John",
    "last_name": "Doe",
    "email": "john.doe@example.com"
}'
```

Expected Response:

```
{
    "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "first_name": "John",
    "last_name": "Doe",
    "email": "john.doe@example.com"

}
```

Actual Response:

```
{

    "id": "0833bc01-fcee-429d-b456-ab3978f71c29",

    "first_name": "John",

    "last_name": "Doe",

    "email": "john.doe@example.com"

}
```

b. Test Invalid Data for User Creation (POST)

```
curl -X POST "http://127.0.0.1:5000/api/v1/users/" -H "Content-Type:
application/json" -d '{
    "first_name": "",
    "last_name": "",
    "email": "invalid-email"
}'
```

Expected Response:

```
{
    "error": "Invalid input data"

}
```

Actual Response:

```
{
    "error": "Invalid input data"
}
```

c. Get All Users (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/users/"
```

Expected Response:

```
[
    {
        "id": "some-id",
        "first_name": "John",
        "last_name": "Doe",
        "email": "john.doe@example.com"
    }

]
```

```
Actual Response:

[

    {

        "id": "627d8f59-ed31-45e4-8f0a-13093b107006",

        "first_name": "John",

        "last_name": "Doe",

        "email": "john.doe@example.com"

    }

]
```

d. Get User by ID (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/users/<user_id>"


Expected Response:

{

    "id": "some-id",

    "first_name": "John",

    "last_name": "Doe",

    "email": "john.doe@example.com"

}
```

Actual Response:

```
{

    "id": "627d8f59-ed31-45e4-8f0a-13093b107006",

    "first_name": "John",

    "last_name": "Doe",

    "email": "john.doe@example.com"

}
```

2. Testing the places.py Endpoints:

a. Create a Place (POST)

```
curl -X POST "http://127.0.0.1:5000/api/v1/places/" -H "Content-Type:
application/json" -d '{

    "title": "Beachfront Villa",

    "description": "A beautiful beachfront property",

    "price": 200.50,

    "latitude": 34.0522,

    "longitude": -118.2437,

    "owner": 3fa85f64-5717-4562-b3fc-2c963f66afa6

}'
```

Expected Response:

```
{
    "id": "some-place-id",
    "title": "Beachfront Villa",
    "description": "A beautiful beachfront property",
    "price": 200.50,
    "latitude": 34.0522,
    "longitude": -118.2437,
    "owner_id": "some-user-id"
}
```

Actual Response:

```
{
    "id": "bd8fb46e-1ae0-4191-815b-19f9ded48e03",
    "title": "Cozy Apartment",
    "description": "A nice place to stay",
    "price": 100.0,
    "latitude": 37.7749,
    "longitude": -122.4194,
    "owner_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6"
}
```

b. Get All Places (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/places/"
```

Expected Response:

```
[
    {
        "id": "some-place-id",
        "title": "Beachfront Villa",
        "latitude": 34.0522,
        "longitude": -118.2437
    }
]
```

Actual Response:

```
[
    {
        "id": "bd8fb46e-1ae0-4191-815b-19f9ded48e03",
        "title": "Cozy Apartment",
        "latitude": 37.7749,
        "longitude": -122.4194
    }
]
```

c. Get Place by ID (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/places/<place_id>"
```

Expected Response:

```
{

    "id": "some-place-id",

    "title": "Beachfront Villa",

    "description": "A beautiful beachfront property",

    "latitude": 34.0522,

    "longitude": -118.2437,

    "owner": {

        "id": "some-user-id",

        "first_name": "Jane",

        "last_name": "Smith",

        "email": "jane.smith@example.com"

    },

    "amenities": [

        {

            "id": "some-amenity-id",

            "name": "Pool"

        }

    ]

}
```

```
Actual Response:

{

    "id": "38cada6d-6ec6-4fd3-8e62-521e88aa4138",

    "title": "Cozy Apartment",

    "description": "A nice place to stay",

    "latitude": 37.7749,

    "longitude": -122.4194,

    "owner": {

        "id": "627d8f59-ed31-45e4-8f0a-13093b107006",

        "first_name": "John",

        "last_name": "Doe",

        "email": "john.doe@example.com"

    },

    "amenities": []
```

3. Testing the `amenities.py` Endpoints:

a. Create an Amenity (POST)

```
curl -X POST "http://127.0.0.1:5000/api/v1/amenities/" -H
"Content-Type: application/json" -d '{

    "name": "Swimming Pool"

}'
```

Expected Response:

```
{
    "id": "some-amenity-id",
    "name": "Swimming Pool"
}
```

Actual Response:

```
{
    "id": "3df3d598-6f60-4641-9d92-d2d86dc1c179",
    "name": "Swimming Pool"
}
```

b. Get All Amenities (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/amenities/"
```

Expected Response:

```
[
    {
        "id": "some-amenity-id",
        "name": "Swimming Pool"
    }
]
```

Actual Response:

```
[
    {
        "id": "3df3d598-6f60-4641-9d92-d2d86dc1c179",
        "name": "Swimming Pool"
    }
]
```

c. Get Amenity by ID (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/amenities/<amenity_id>"
```

Expected Response:

```
{
    "id": "some-amenity-id",
    "name": "Swimming Pool"
}
```

Actual Response:

```
{
    "id": "3df3d598-6f60-4641-9d92-d2d86dc1c179",
    "name": "Swimming Pool"
}
```

4. Testing the reviews.py Endpoints:

a. Create a Review (POST)

```
curl -X POST "http://127.0.0.1:5000/api/v1/reviews/" -H "Content-Type:
application/json" -d '{

    "text": "Amazing place!",

    "rating": 5,

    "user_id": baef0e99-deb8-4f71-99ba-48ff706b5fc9,

    "place_id": 38cada6d-6ec6-4fd3-8e62-521e88aa4138

}'
```

Expected Response:

```
{

    "text": "Amazing place!",

    "rating": 5,

    "user_id": "some-user-id",

    "place_id": "some-place-id"

}
```

Actual Response:

```
{

    "text": "Amazing place!",

    "rating": 5,

    "user_id": "baef0e99-deb8-4f71-99ba-48ff706b5fc9",

    "place_id": "38cada6d-6ec6-4fd3-8e62-521e88aa4138"

}
```

b. Get All Reviews (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/reviews/"
```

Expected Response:

```
[
    {
        "id": "some-review-id",
        "text": "Amazing place!",
        "rating": 5,
        "user_id": "some-user-id",
        "place_id": "some-place-id"
    }
]
```

Actual Response:

```
[
    {
        "id": "2bccac58-413d-4106-841d-edc9c708eca0",
        "text": "Amazing place!",
        "rating": 5
    }
]
```

c. Get Review by ID (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/reviews/<review_id>"
```

Expected Response:

```
{
    "id": "some-review-id",
    "text": "Amazing place!",
    "rating": 5,
    "user_id": "some-user-id",
    "place_id": "some-place-id"
}
```

Actual Response:

```
{
    "id": "2bccac58-413d-4106-841d-edc9c708eca0",
    "text": "Amazing place!",
    "rating": 5,
    "user_id": "baef0e99-deb8-4f71-99ba-48ff706b5fc9",
    "place_id": "38cada6d-6ec6-4fd3-8e62-521e88aa4138"
}
```

d. Update Review (PUT)

```
curl -X PUT "http://127.0.0.1:5000/api/v1/reviews/<review_id>" -H
"Content-Type: application/json" -d '{

    "text": "Updated review text",

    "rating": 4

}'
```

Expected Response:

```
{

    "message": "Review updated successfully"

}
```

Actual Response:

```
{

    "message": "Review updated successfully"

}
```

c. Get Review by place_ID (GET)

```
curl -X GET "http://127.0.0.1:5000/api/v1/places/<place_id>/reviews"
```

Expected Response:

```
{
    "id": "some-review-id",
    "text": "Amazing place!",
    "rating": 5,
    "user_id": "some-user-id",
    "place_id": "some-place-id"
}
```

Actual Response:

```
[
    {
        "id": "2bccac58-413d-4106-841d-edc9c708eca0",
        "text": "Updated review text",
        "rating": 4
    }
]
```

e. Delete Review (DELETE)

```
curl -X DELETE "http://127.0.0.1:5000/api/v1/reviews/<review_id>"
```

Expected Response:

```
{
    "message": "Review deleted successfully"
}
```

Actual Response:

```
{
    "message": "Review deleted successfully"
}
```