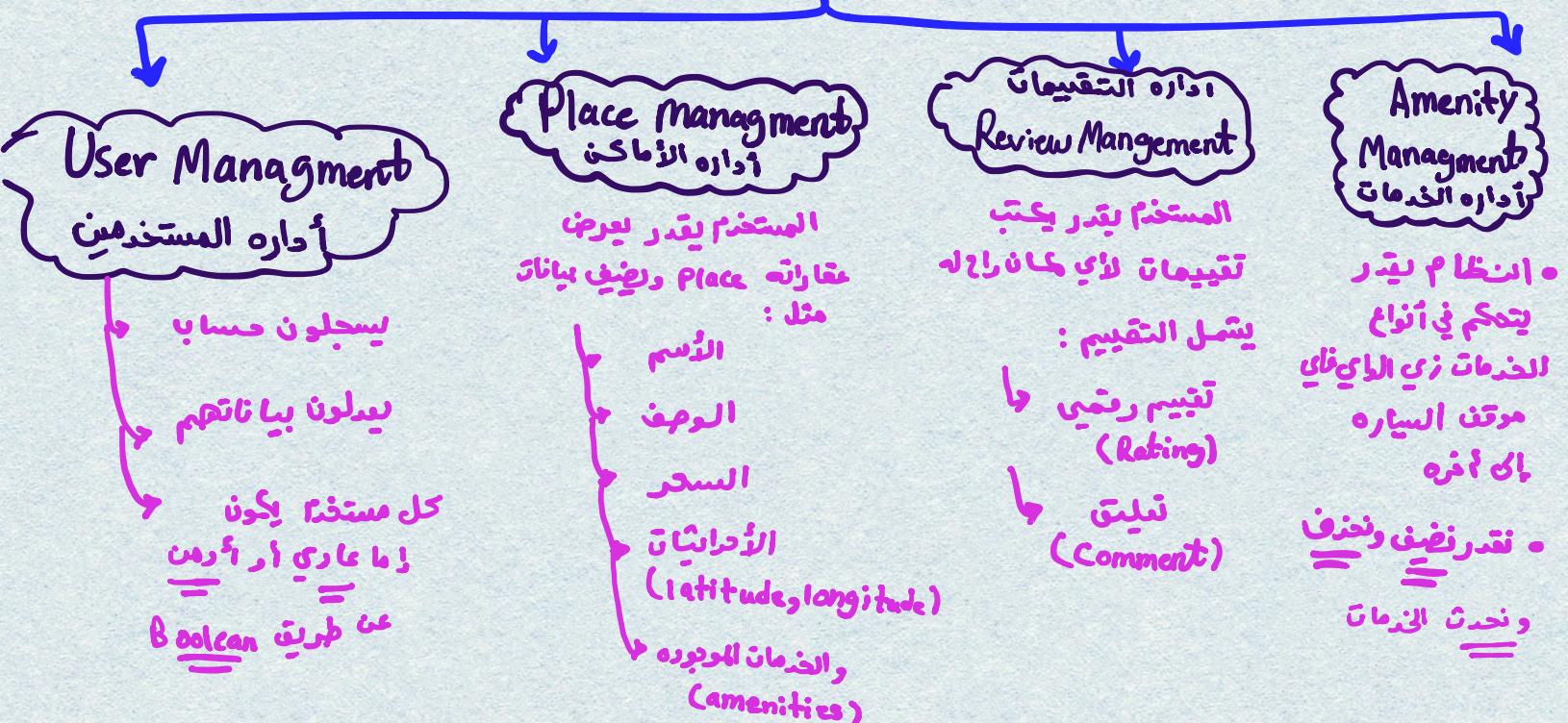
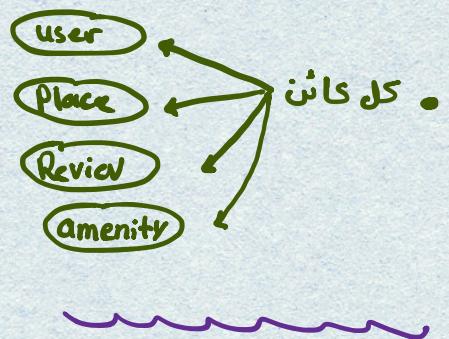


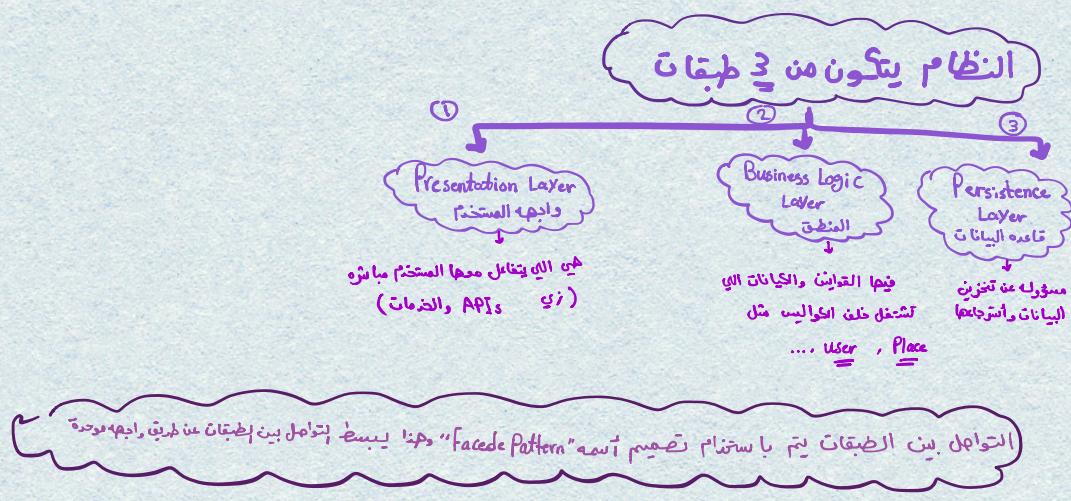
التطبيق يسع للمستخدمين إنهم ليسوون عده أثياد

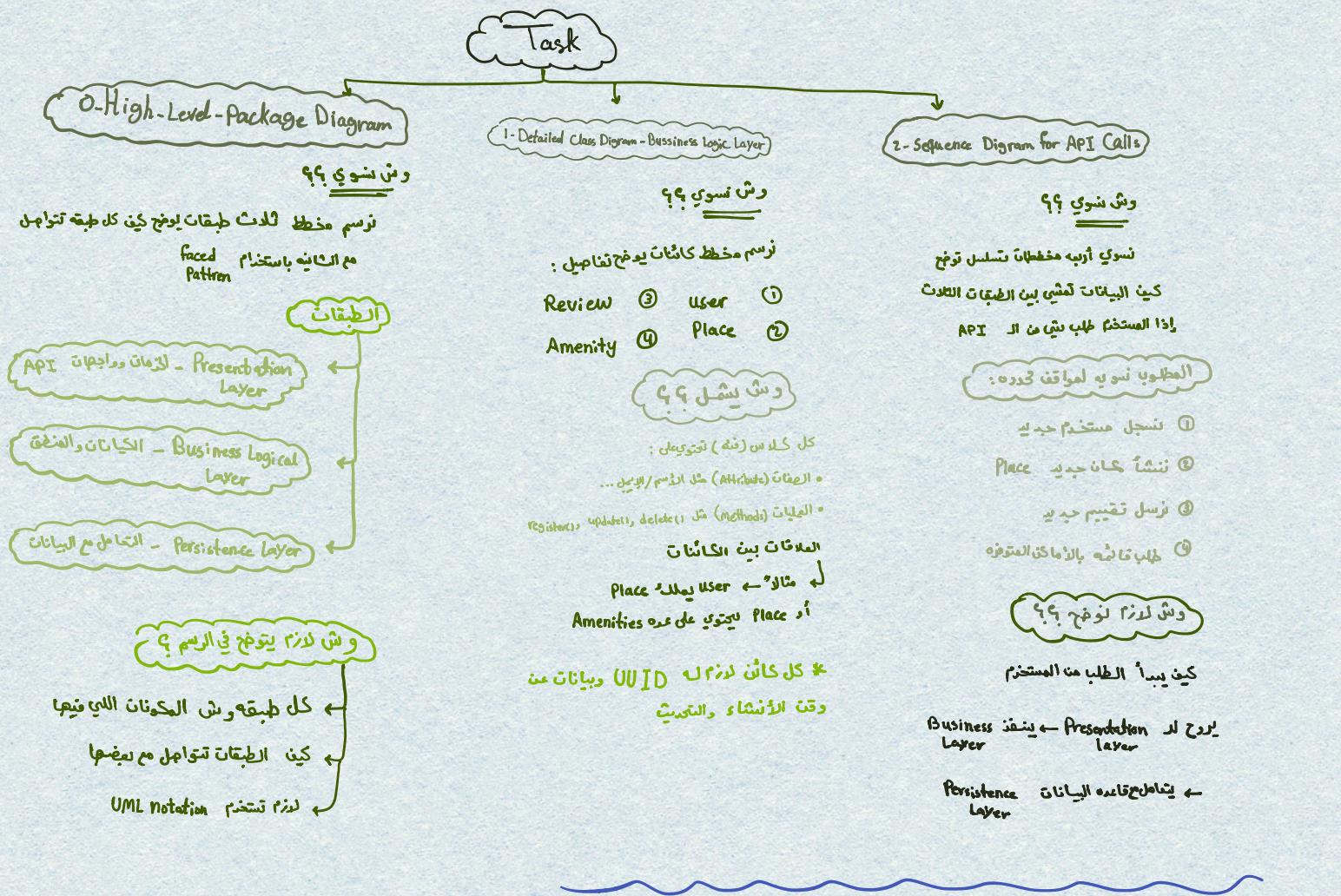


كل كائن يكون عنده ID مميز وكل كائن يتم تسجيل رقت لنشأوه وتحديه



هيكله النظام (Architecture)





Class Diagram

رسو مخطط هفضل يوضح الكلاسات وأضلاعه والروابط لكل

هن :

User ①

Place ②

Review ③

Amenity ④

(Sequence Diagrams)

رسو ٤ مخططات تتبع ومن يصيروا المستخدم

يتفاعل مع النظم في :

١ - تسجيل مستخدم جديد

٢ - إنشاء مكان جديد

٣ - كتابة تقييم

٤ - جلب قائمة الآماكن

هذى هي الواجهه اللي يتعامل مدها المستخدم
متاد :
• واجهات API
• الخدماں اللي تحمل أوامر المستخدم

هنا يتم تنفيذ القواليت لعنی :

ومن يصيروا إذا المستخدم ضغط زر ٩

• كين تتحقق من مجا البيانات ؟

• انكيداسات زي user و place يتم تنفيذ رملائهم

API - الزهان وواجهات
Presentation Layer

Business Logical Layer - الكيانات والمشفق

Persistence Layer - التعامل مع البيانات

هنا يتم حفظ البيانات :
• إنشاء مستخدم جديد
• تحديث بيانات مكان
• حدد التقييم

ما هي فايت

هو أسلوب تصميم Design Pattern يعطي واجهة موحدة وبيطه للمواد بين الطبقات

يعني بالداخل طريقة تكلم الثانية بشكل مباين ومحقد، شوي واجهة موحدة facade ترسل المدخل هنا وتبسط له

مثال: لوالمستخدم فقط "أتفق تقييم"

Layer Layer Presentation Layer

Persistence Business Logic

لـ ترجمة النتيجة إلى المستخدم

لـ كان facade هو الرئيسي بين الواجهة والم讐طق/البيانات

0. High-Level Package Diagram

نرسم مخطط عام High-level يوضح بنية النظام، يعني كيف التطبيق مقسم إلى طبقات (Packages)

وكيف تتعامل مع بعضه باستخدام Facade Pattern



① نفهم المشروع أنه يشبه Airbnb فيه أمثلاء مثل:

لـ مستخدمين (user) له أماكن (Place)

لـ خدمات [خواص] (Amenity) له تقييمات (Review)

② نفهم الطبقات (Packages)

لـ Persistence Layer (البيانات)

ـ تتعامل مع قاعدة البيانات (MongoDB, MySQL, الخ)
ـ فيها Repositories: UserRepository, PlaceRepository

Business Logic Layer (الم讐طق)

ـ فيها الكلاسات الذكيه مثل UserManger, PlaceManger
ـ تحدد المنطق، مثل إضافة مستخدم، التأكد من البيانات، الخ.

Presentation Layer (الواجهة)

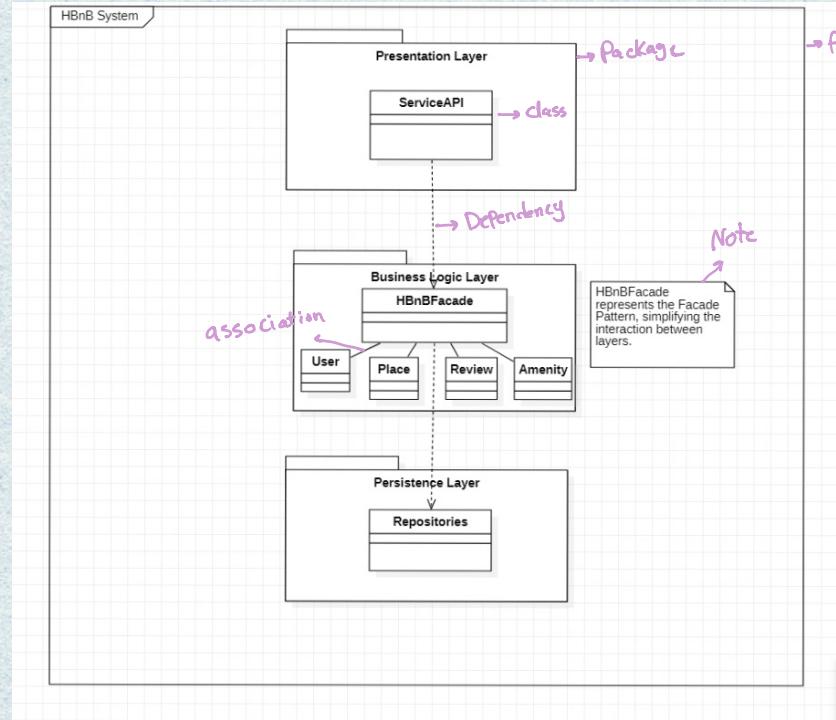
ـ المسؤول عن استقبال الطلبات من المستخدم (User API)
ـ ماتتفق شيء معقد فقط تنسق البيانات

ـ تتحمل بـ Facade

③. وين يدخل فايت بين Business Logic و Presentation

ـ يعني واجهة المستخدم Presentation ما تتحمل مباين على Business Logic
ـ تتحمل facade

ـ أو facade ينسى رسابو الكلاسات المنسنة من Business logic



رسمنا طبقات النظام (Layers)

ServiceAPI ← Presentation layer

وهذا الأجزء الذي يتعامل مع المستخدم أو مع التطبيق الآلي
ومنه هناك API أو واجهة لبرمجيات المستخدم
في الرسمة خطينا كلاس ServiceAPI فيها

HBnBFacade ← Business logic layer

- هو قلب النظام فيه كل القرائن أو المفاهيم التي تعنى عليه التطبيق
- بيان ما كل الطبقات تتواجد معاً فيه نمط كلاس HBnBFacade عتاد يكون واجهة موحدة

Repositories ← Persistence layer

خطينا ملحوظة تدفع أن هذا الكلاس يمثل الفئتين
وهي العلاقة التي يمثلها

وأخفينا كلاسات لأن كل واجهة بيان مستعمل

ما يفتح نقطتها كـ Attributes لبيانها موفرة رغبة

وبيان طلب تزويدهم ببياناتهم association

وبيان association يعني أنهما تقول أنا أنتظم

وأتفاصل معي لكنك بيان مستقل

وأرتكب معي بين الكلاسات يعني ليقصد على الثاني

عندي ثلاث طبقات والواجهة الوحيدة التي تتعامل معها هي

عندي الواجهة ServiceAPI تتكلم مع HBnBFacade وهو يتعامل مع

باقي الأجزاء (User/Place/Review/Amenity) ويرسل منهم معلومات

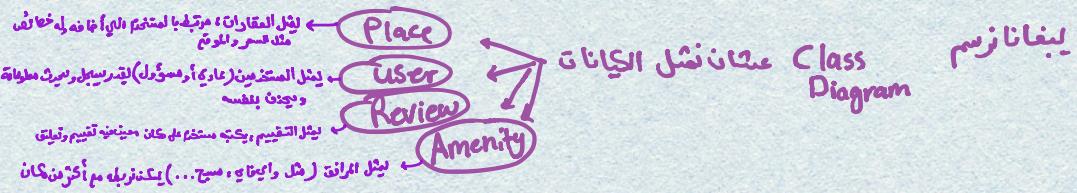
أو يستند عليهما ، إذا احتاج يقرأ أو يكتب بيانات تتعامل مع Repositories في طبقة البيانات

Repositories ← Persistence layer

هذى مسؤولة عن تخزين وأرجاع البيانات من قاعدة البيانات DataBase

في الرسمة فيه Repositories وهي المسؤولة عن التعامل مع البيانات

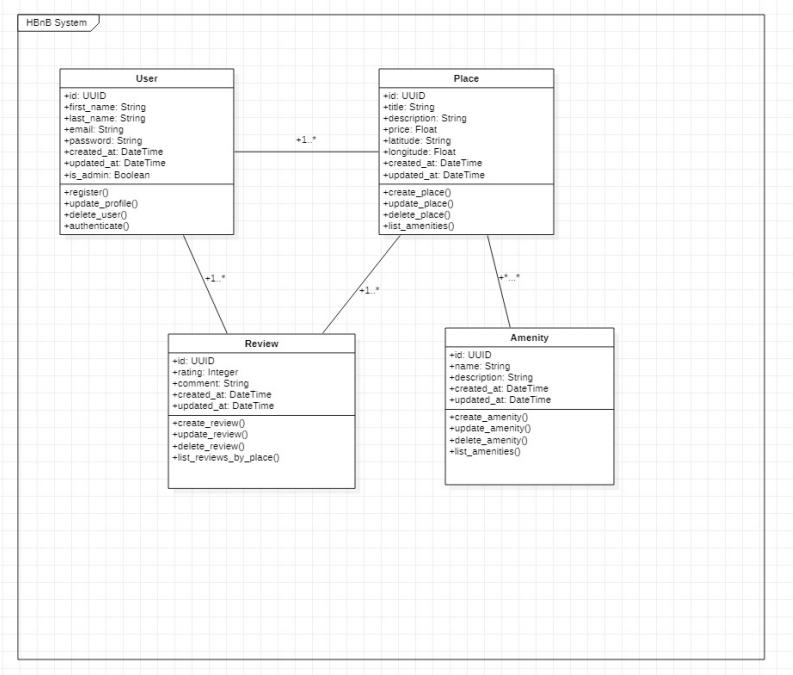
1. Detailed Class Diagram for Business Logic Layer



دلوخ الخواص (attribute) والوظائف (method)

والعلاقة (One to one / many to many one to many)

عرفنا الكلاسات الأساسية ٤



- User يمثل المستخدم فيه معلوماته مثل الاسم / البريد / كلمة السر ...

- Place يمثل المكان المزور فيه معلوماته مثل العنوان / السعر / الموقع ..

- Review يمثل التقييم فيه معلومات تعلق بقسم مكان ..

- Amenity يمثل وسائل راحة فيه معلومات اسم الرسالة، صنف ..

أخفنا خواص كل كلاس attribute

كل كلاس عنده Id خاص فيه (UUID)

و وقت الـ insert و التعديل Created_at, updated_at

وبقيه التفاصيل تختلف حسب نوع الكائن

مثلاً longitude, price, latitude are Place, email are user

أخفنا الوظائف Operation or method

مثال User يقدر يسجل register ليحل بياناته بنفسه

Place يقدر لينتني، يعدل، يحذف المكان

رسن السير Review

....

Amenity

أخفنا العلاقات بين الكلاسات

