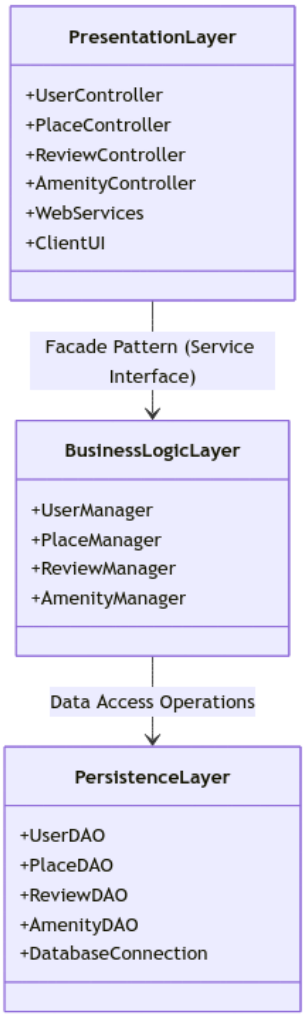
HBnB

Introduction:

A comprehensive web application that replicates the core functionalities of AirBnB. This project demonstrates a full-stack implementation using a layered architecture approach. This document presents the project to develop a housing location platform inspired by the Airbnb site. The goal is to design an application that allows users to publish, search and book accommodation intuitively and securely.

Architecture:

The project works in 3 layers



1: Presentation Layer; User interface and API for users to interact with the application.

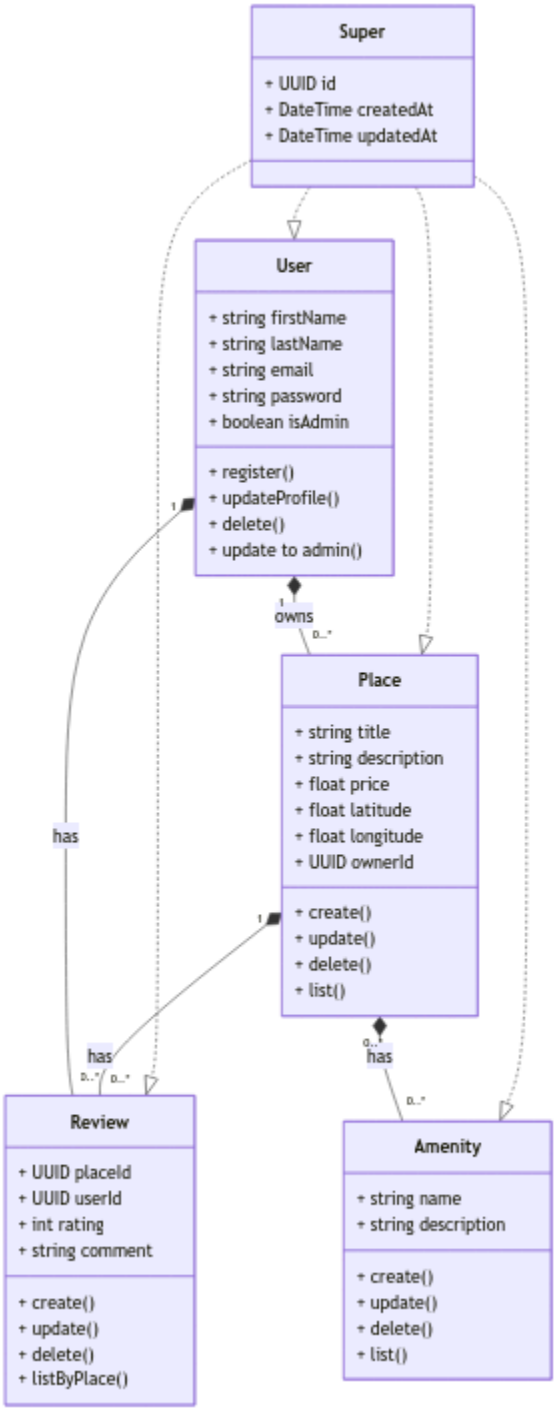
2: Business Logic Layer; Contains models and business logic (management of users, locations, reviews, etc.).

3: Persistence Layer; Responsible for storing and retrieving data in the database.

The pattern facade is an interface which is used to group functions to organize them and hide functions and subsystems, which simplifies it by allowing you to act on everything in one line.

Database Operations:
Data storage: Manage the saving and retrieval of information from the database.
Abstraction: Isolate the details of data storage so that business logic does not have to interact directly with the database.
Security: Protect sensitive data and manage access.
Performance: Optimize queries to ensure fast and efficient interactions with data.

Logic:



User is the class that contains the user attributes and methods.

Place is the class that contains the attributes of an announcement.

Review is the class that contains the attributes of the comments that all users can create.

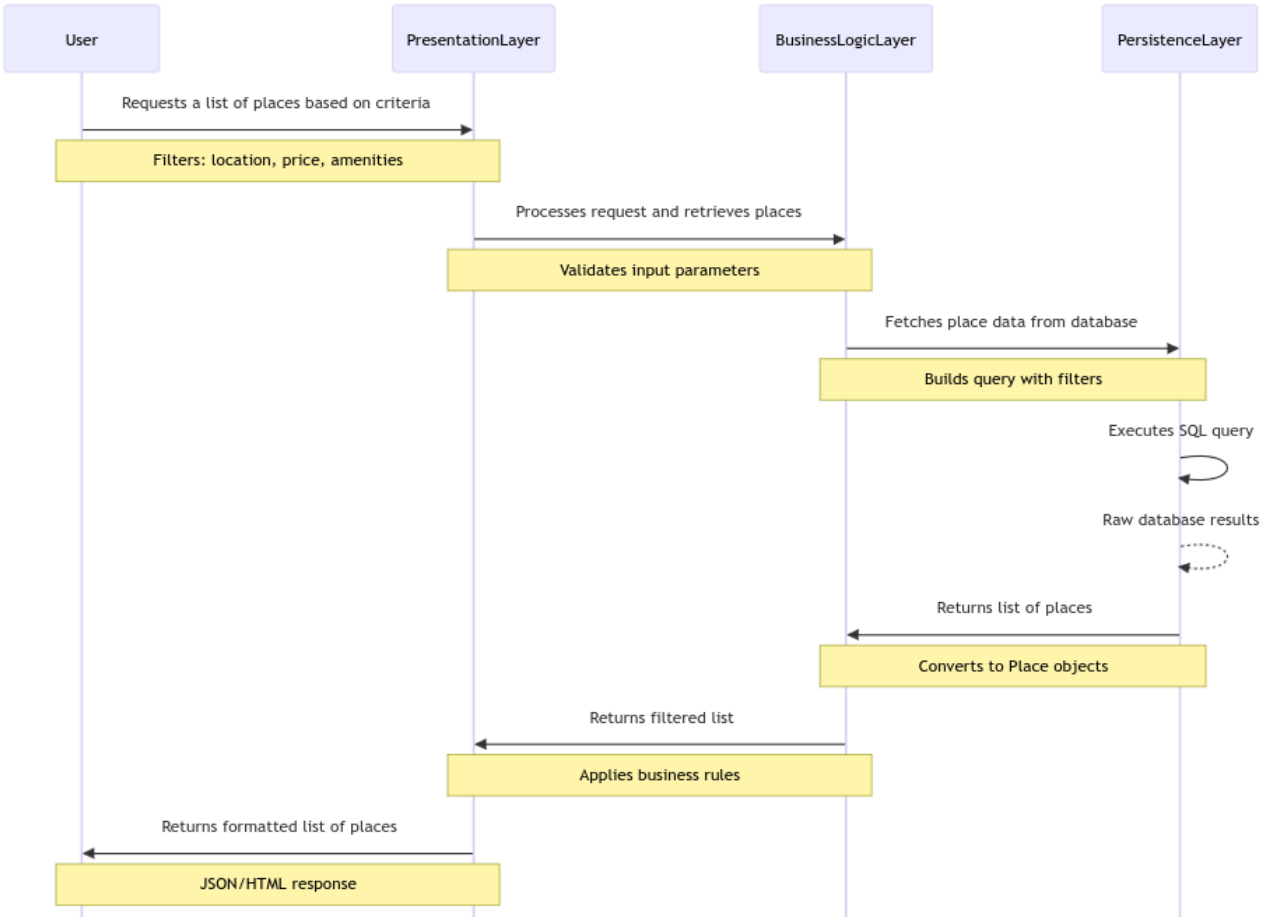
The amenity class contains the place option attributes.

The super class contains the mandatory attributes that all classes will inherit.

The review and amenity classes cannot exist without the place class, the place class cannot exist without the class user.

Interaction flow:

Fetchplace



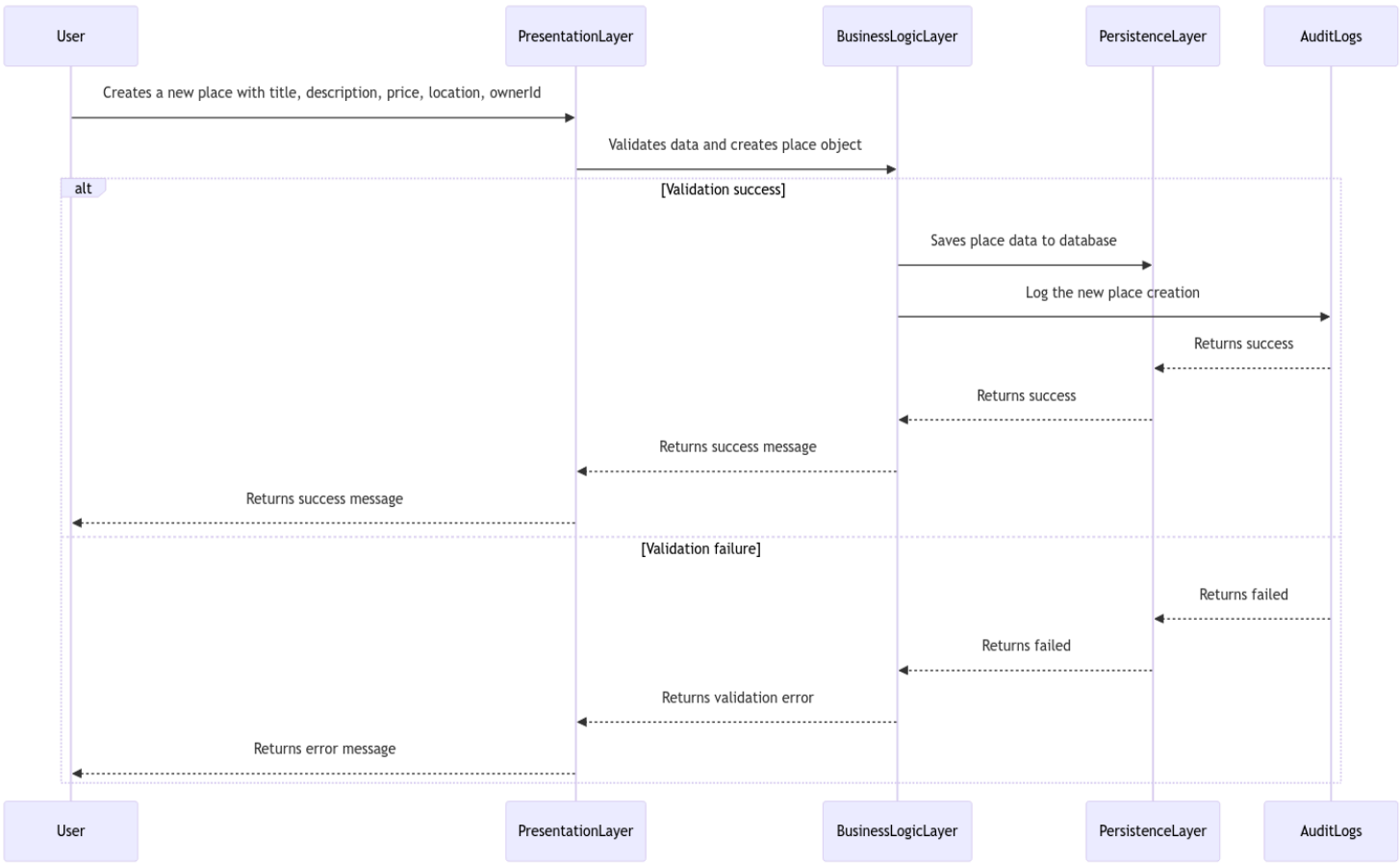
SOON

The user sends a query to the PresentationLayer indicating the requested criteria,
then the PresentationLayer receives the query, validates the input parameters and transmits it to the BusinessLogicLayer which receives the request, constructs the filtered query and finally sends it to the PersistenceLayer which executes the SQL query and retrieves the data.

END

The return path goes like this: PersistenceLayer: Returns the results as objects to the BusinessLogicLayer which applies additional business rules and returns them to the PresentationLayer which formats the response (JSON/HTML) and sends it to the user.

Place Creation

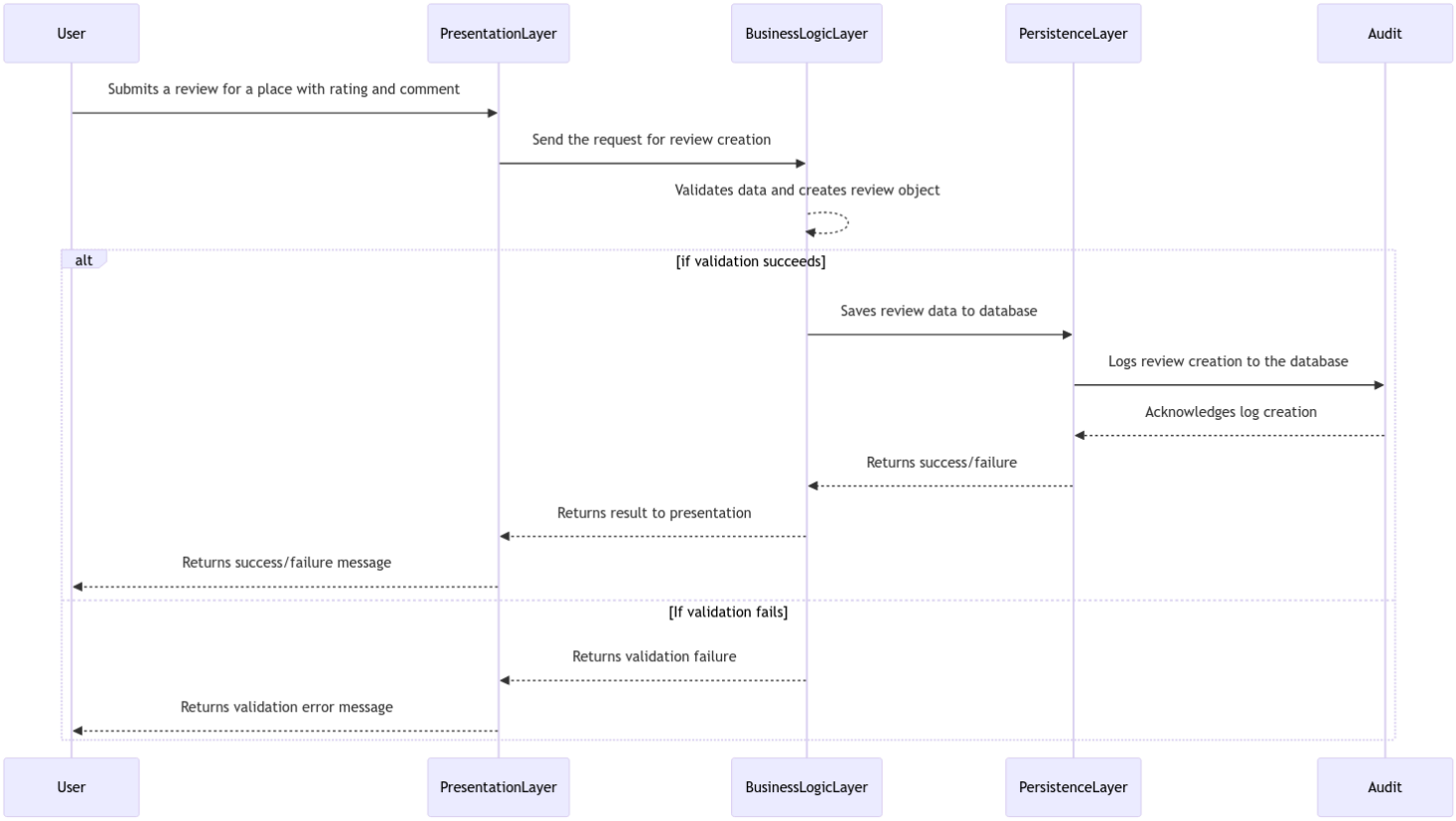


This diagram illustrates the process of creating a place with full feedback to the user:

User: Submits a creation request with location details (title, description, price, location, etc.).
Presentation Layer: Validates the initial data and passes it to the business logic layer.
Business Logic Layer: Validates the additional data.
If the validation succeeds, it sends the data to the Persistence Layer.
If validation fails, it returns an error.
Persistence Layer: Saves data in the database and returns a result (success or failure).
Audit: Logs the action for traceability.

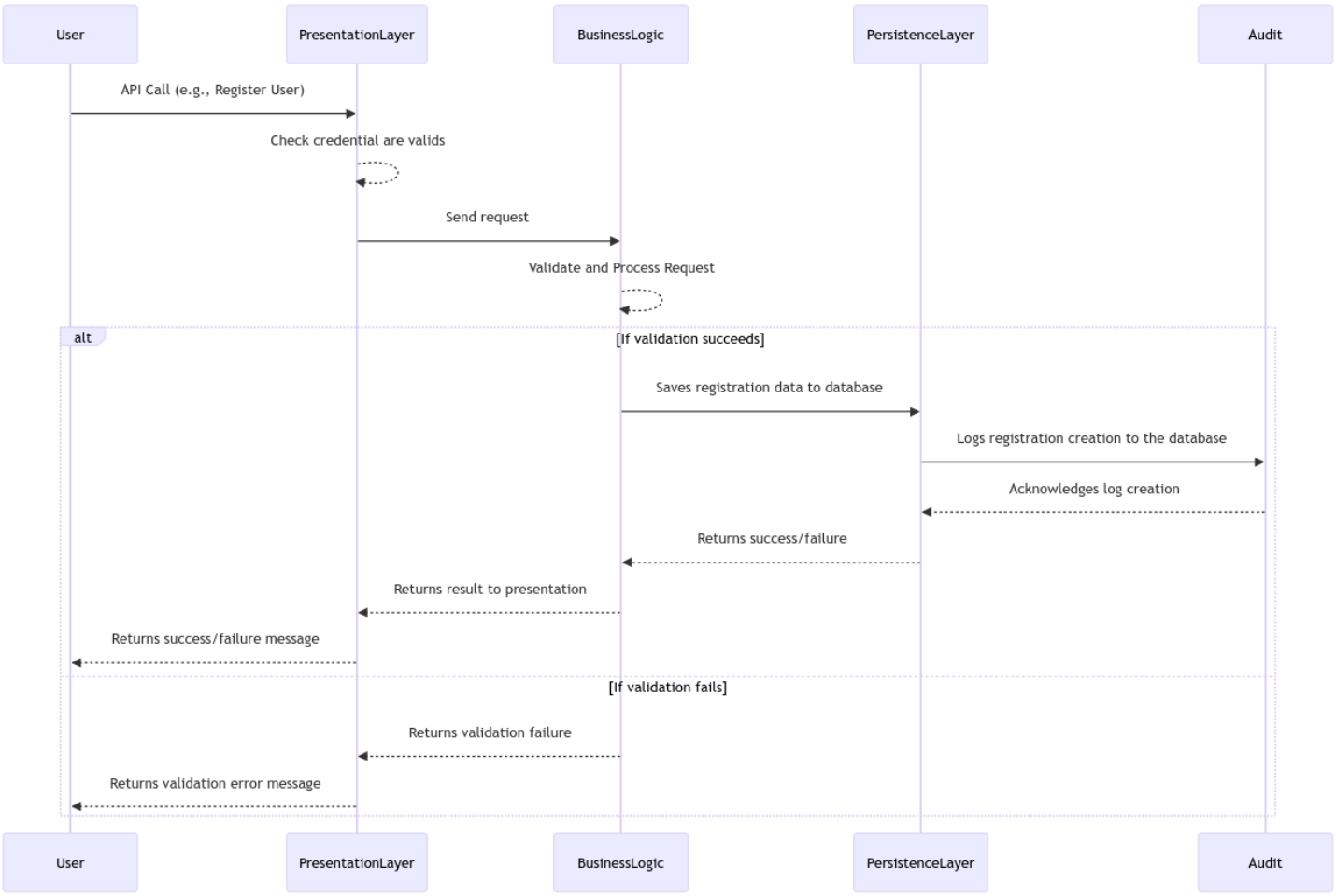
Return to the user: The presentation layer sends a message back to the user depending on the result of the backup (success or error).

Review Submission



The user fills out a comment with a note, the PresentationLayer validates and creates a new review object, the BusinessLogicLayer sends a request to the database to save the comment, the PersistenceLayer sends a request to the audit for traceability.
And a feedback message is sent to the user to confirm whether or not their comment has been taken into account.

User Registration



This sequence diagram illustrates the user registration process via an API. The user sends a request to the presentation layer, which first validates the credentials before forwarding the request to the business logic layer. The business logic performs a thorough validation and processes the request. If validation is successful, the data is sent to the persistence layer for storage, and an audit log is created and confirmed. The database then returns a success or failure message to the business logic, which passes it to the presentation layer to inform the user of the outcome. If validation fails, the business logic returns an error to the presentation layer, which sends an appropriate error message to the user.

Mano Delcourt
Esteban Cratere
Herve Le Guennec