HBnB Project

1. Introduction

This technical documentation aims to provide an overview of the architecture, design and interactions of the HBnB Evolution application, a simplified version of Airbnb. It will serve as a guide for development by ensuring a clear understanding of concepts and requirements.

The application enables the following operations:

- User management: registration, profile updates, distinction between regular users and administrators.
- Places management: Creating, updating, deleting and listing properties.
- Reviews management: Add, modify, delete and display reviews of places.
- Amenities management: Add, modify, delete and display amenities associated with places.

2. Application architecture and layers

HBnB Evolution follows a layered architecture divided into :

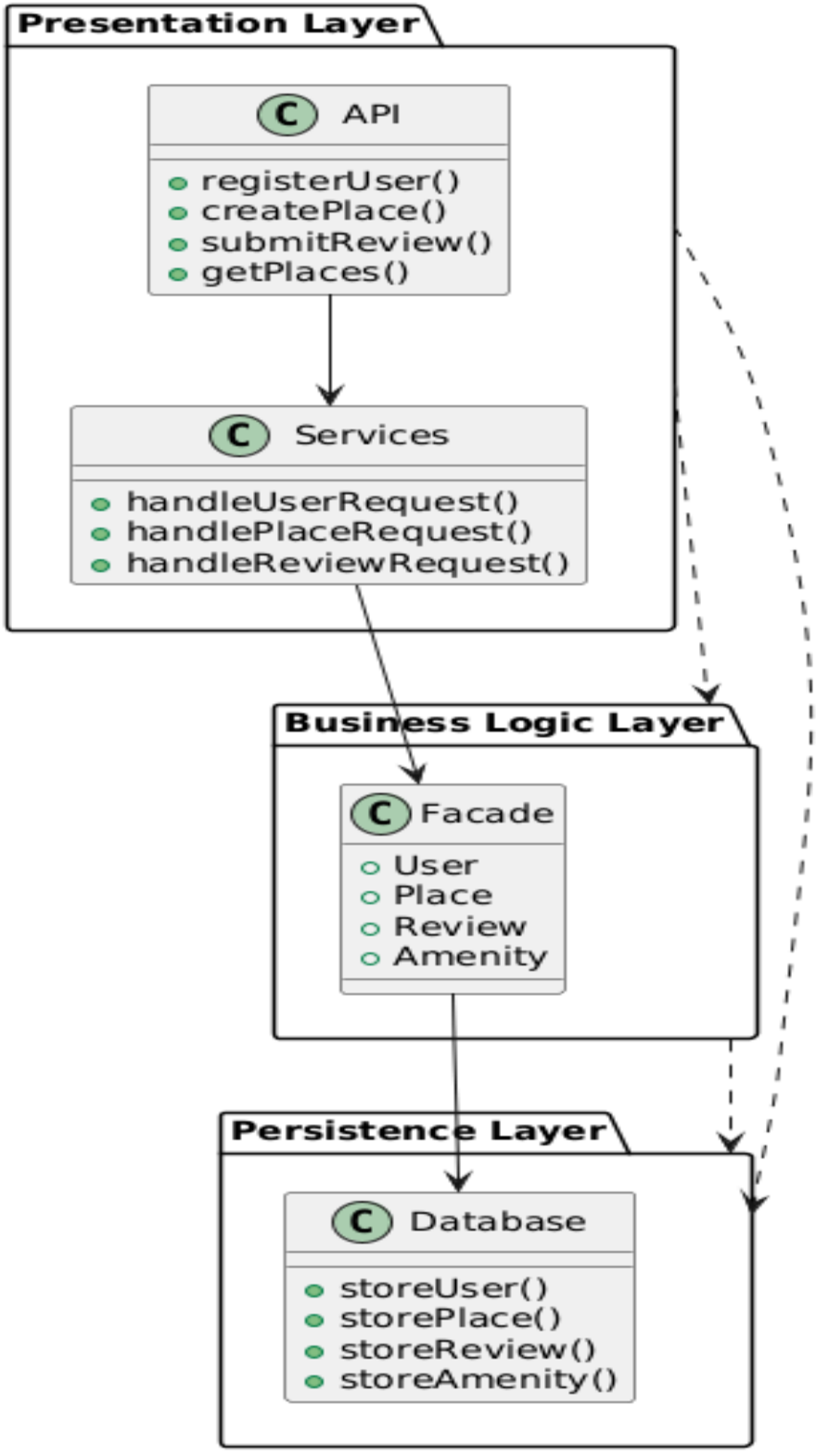
- **Presentation layer: user interaction services and APIs.**
- **Business logic layer: Management models and rules.**
- **Persistence layer: database and information storage.**

The interaction between these layers is based on the MVC model and the Facade design pattern to separate responsibilities and simplify integration.

3. UML diagrams

3.1. Package diagrams

The package diagram represents the overall structure of the application by illustrating the different layers and their communication.

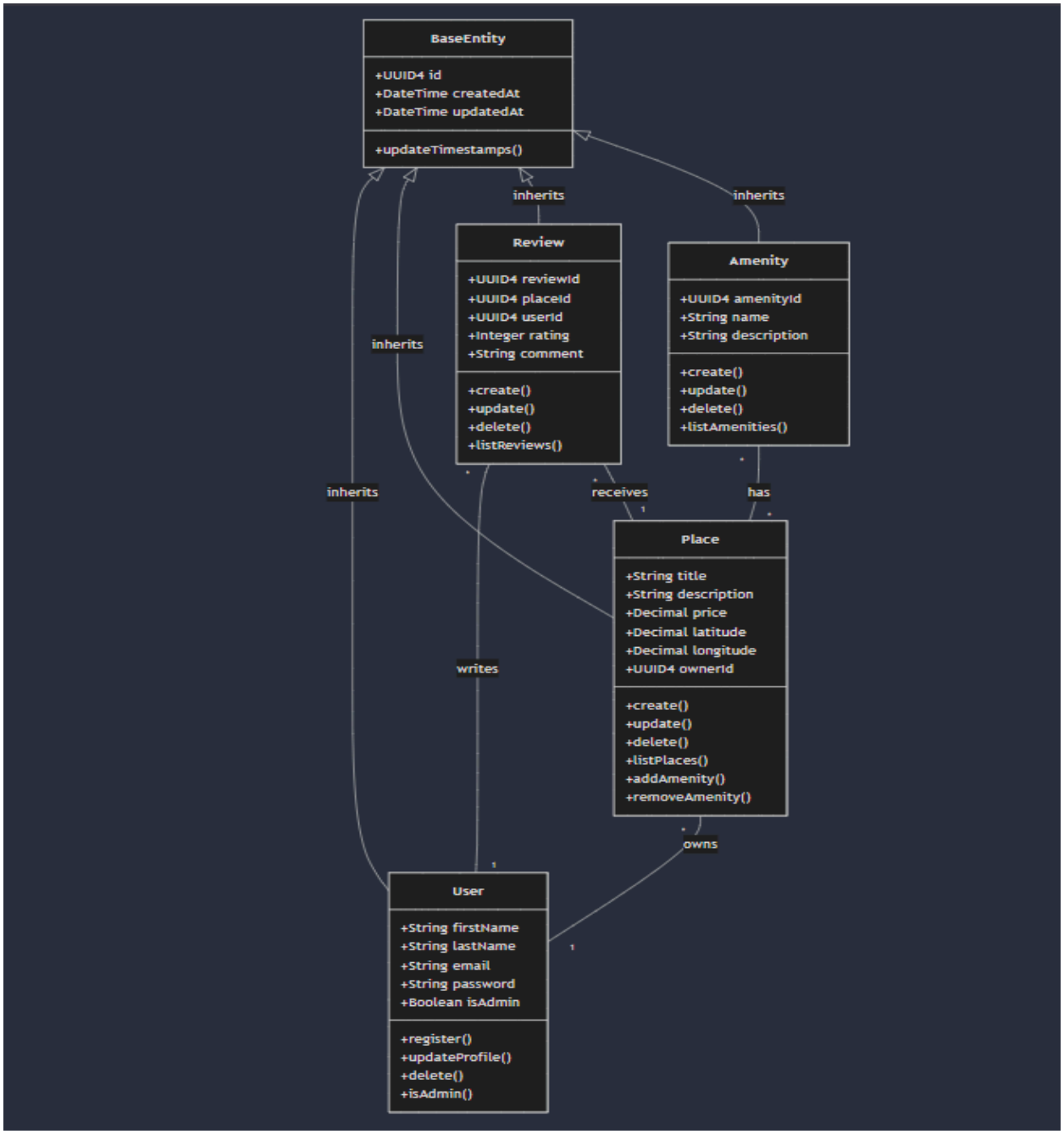

Presentation Layer
API
registerUser()
createPlace()
submitReview()
getPlaces()
Services
handleUserRequest()
handlePlaceRequest()
handleReviewRequest()
Business Logic Layer
Facade
User
Place
Review
Amenity
Persistence Layer
Database
storeUser()
storePlace()
storeReview()
storeAmenity()

Explanation:

- The presentation layer contains the API controllers.
- The business logic layer defines entities and rules.
- The persistence layer manages data access via a relational database.

3.2. Classes diagrams

The class diagram shows the main entities and their relationships.



Entities and their Attributes

- **BaseEntity**(id, created_at, updated_at)
- **User** (first_name, last_name, email, password, is_admin)
- **Place** (name, description, price, latitude, longitude, owner_id)
- **Review** (user_id, place_id, rating, comment)
- **Amenity** (name, description)

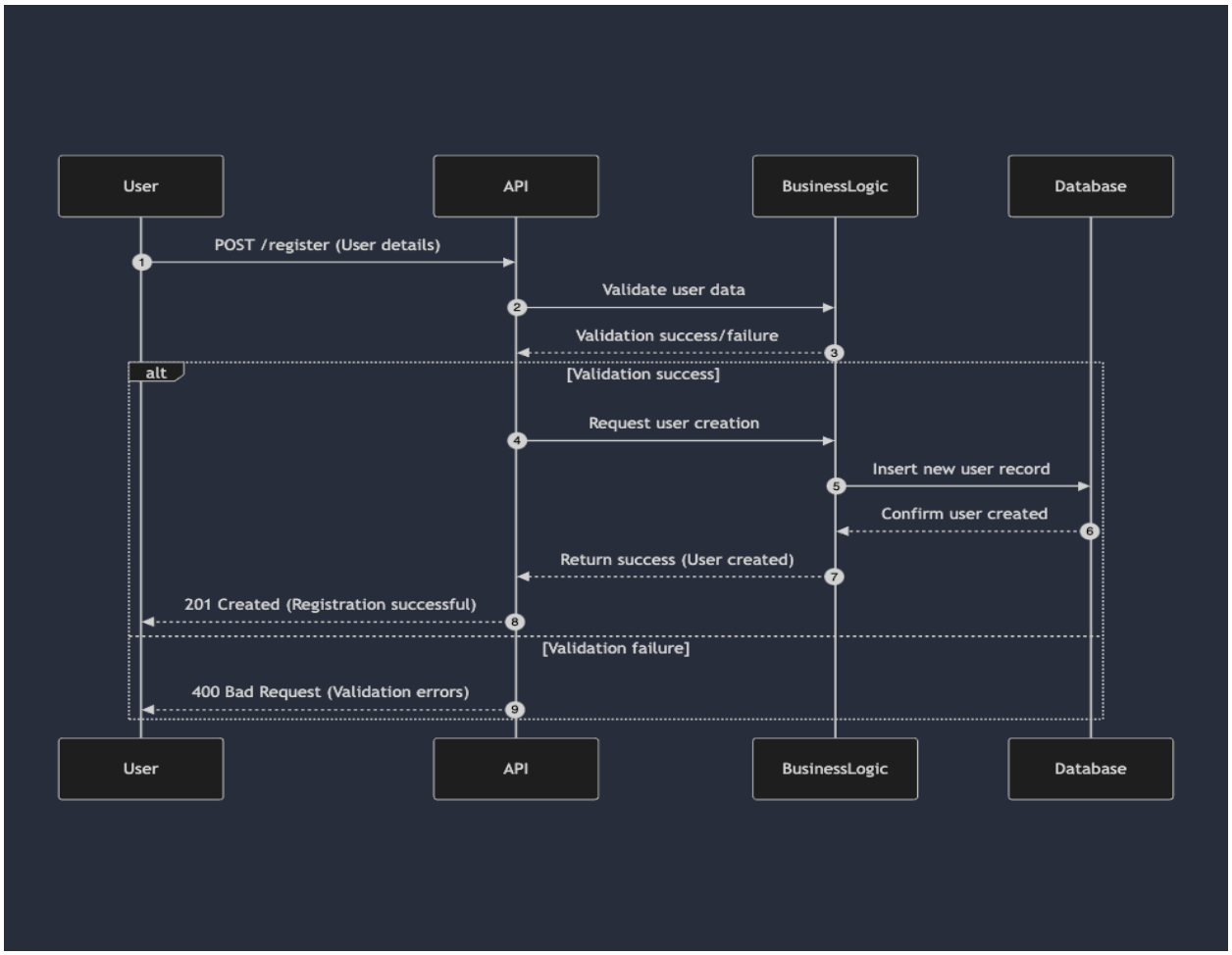
Explanation :

- A User can own several Places.
- A User can leave several Reviews on Places.
- A Place can have several Amenities.

3.3.Sequence Diagrams

Sequence diagrams describe how the main API interactions work.

Example: User registration



1. The user sends a POST request to the API.
2. The API controller validates the data and calls the business logic layer.
3. The business logic layer creates a new user.
4. The persistence layer registers the user in the database.
5. A confirmation response is sent back to the user.

4. Conclusion

This document provides a complete overview of HBnB Evolution, covering its architecture, entities and interactions. It will serve as a reference for the next phases of the project and will facilitate the implementation of the application.