Documentation Compilation

Franck SPADATTO, Hai-Tu NGUYEN, Adel MERJRISSI

For **HOLBERTON School**

Summary

INTRODUCTION

This document presents the technical foundation of the HBnB Evolution project, a software application designed to simplify the management of tourist accommodations, inspired by the AirBnB model.
This will help understanding the high-level architecture, the detailed design of the business logic and the interactions within the system.

The PlantUML and Mermaid Chart Editor were used to create all the diagrams.

1. High-level Architecture

High-level Package Diagram

This section presents the three-layer architecture of the HBnB application and explains how these layers communicate using the facade design pattern.
It provides an overview of the organization of the main components and their internal interactions.
This software architecture distributes responsibilities across three distinct levels: each layer has a dedicated role and only interacts with its adjacent layers.
The package diagram highlights how the application is structured around these three primary layers.

The package diagram represents the structure of the application with its three main layers :

- ➢ **Presentation Layer**
  - responsibilities: handles the interaction between the user and the application
  - includes : services and APIs exposed to the users
  - functions: display information

- ➢ **Business Logic Layer**
  - responsibilities: contains the core business logic and the models that represent the entities in the system
  - includes: user entity, place entity, review entity and amenity entity
  - functions: change entities

- ➢ **Persistence Layer**
  - responsibilities: handles database interaction for data storage and retrieval
  - includes: database storage
  - functions: read data, write data

Including two interfaces:

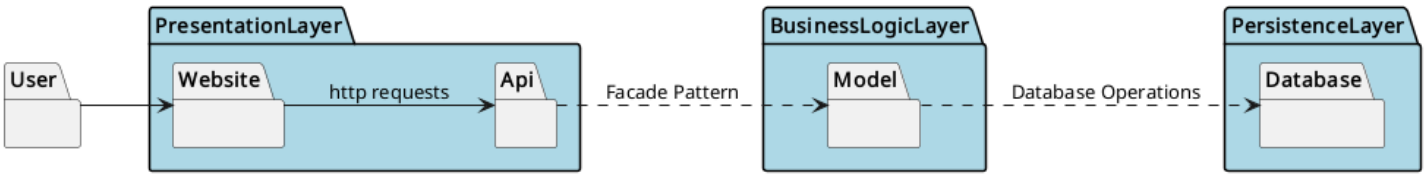
- ➢ **Facade Pattern** (between Presentation Layer and Business Logic Layer)

The facade pattern is used to simplify communication between components, conceal underlying complexity by providing a straightforward and limited interface, and strengthen layering in software architecture by minimizing direct dependencies between layers.

➤ **Database Operations** (between Business Logic Layer and Persistence Layer)

Whenever operations involving data—such as saving, fetching, or updating—are required, the business logic layer delegates these tasks to the persistence layer, which is responsible for handling them.

Package Diagram



2. Business Logic Layer

Detailed Class Diagram for Business Logic Layer

This section aims to develop a comprehensive class diagram representing the Business Logic Layer of the HBnB application. It covers the entities contained in this layer, outlining their properties, functions, and how they are interconnected.

The detailed class diagram represents the structure of the Business Logic Layer:

- **class Base** (parent class)
    - attributes:
        - Id: identification of the object
        - CreatedAt: datetime of the creation
        - UpdatedAt: datetime of the updating
    - methods:
        - create() to create data
        - update() to update data
        - delete() to delete data

- **class User**
    - attributes:
        - IdUser: identification of the user
        - FirstName: identity of the user
        - LastName: identity of the user
        - Email: connexion information
        - Password: connexion information
        - IsAdmin: administrator status (boolean)

- **class Place**
    - attributes:
        - IdUser: key of class User to join it
        - IdPlace: identification of the place

        - Title: title of the place
        - Description: description of the place
        - Price: cost of the place
        - Coordonates: latitude and longitude
        - Owner: owner of the place

- **class Review**
    - attributes:
        - IdUser: key of class User to join it
        - IdPlace: key of class Place to join it
        - IdReview: title of the review
        - Rating: rating chosen by the user
        - Comment: review of the user

- **class Amenity**
    - attributes:
        - IdPlace: key of class Place to join it
        - IdAmenity: identification of the amenity
        - Name: name of the amenity
        - Description: description of the amenity

Class Diagram

DIAGRAMME DE FRANCK ICI (je n'arrive pas à le charger ici)

3. API Interaction flow

Sequence Diagrams for API Calls

The purpose of this section is to create sequence diagrams for four distinct API calls, illustrating how the different layers interact. This approach helps clarify the order of operations involved, from the initial request through to the final response.

These sequence diagrams allows us to describe and visualize the way components of the system interact for four specific use cases:

- **User Registration** (user signs up for a new account)
  - fields:
    - FirstName
    - LastName
    - Email & Password
  - return: success or failure

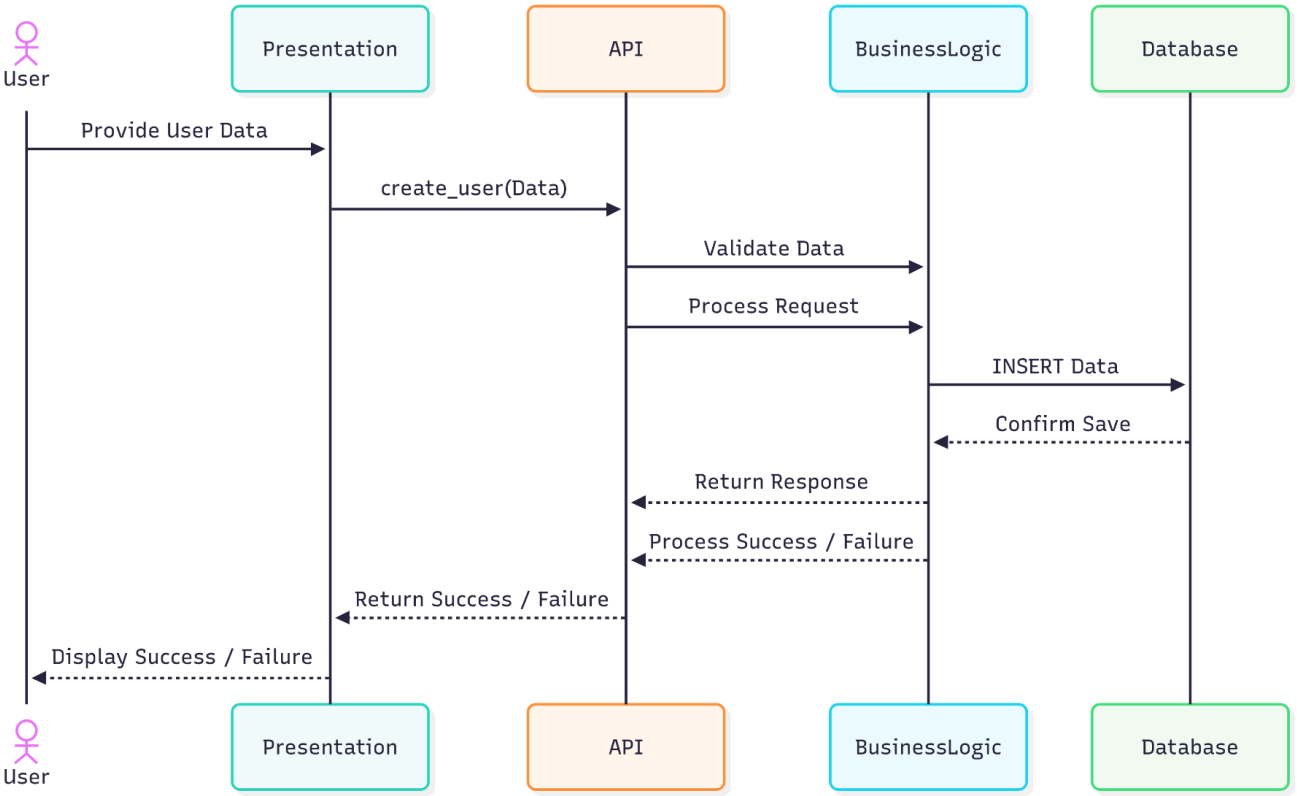
- **Place Creation** (user creates a new place listing)
  - fields:
    - Title
    - Address
    - Location
    - Price
    - Description
  - return: success or failure

- **Review Submission** (user submits a review for a place)
    - Title
    - Text
    - Rating
  - return: success or failure

- **Places List Fetching** (user requests a list of places based on certain criteria)
  - fields:
    - Place
    - Capacity
    - Price

○ return: list of results

Sequence Diagram : New Account



This diagram outlines how interactions progress when someone signs up on the HBnB Evolution platform:

1. **User → Presentation → API**:
   The user initiates the registration process by sending a request to the API via Presentation.

2. **API → BusinessLogic**:
   The API passes the registration details to the business logic layer for verification and handling.

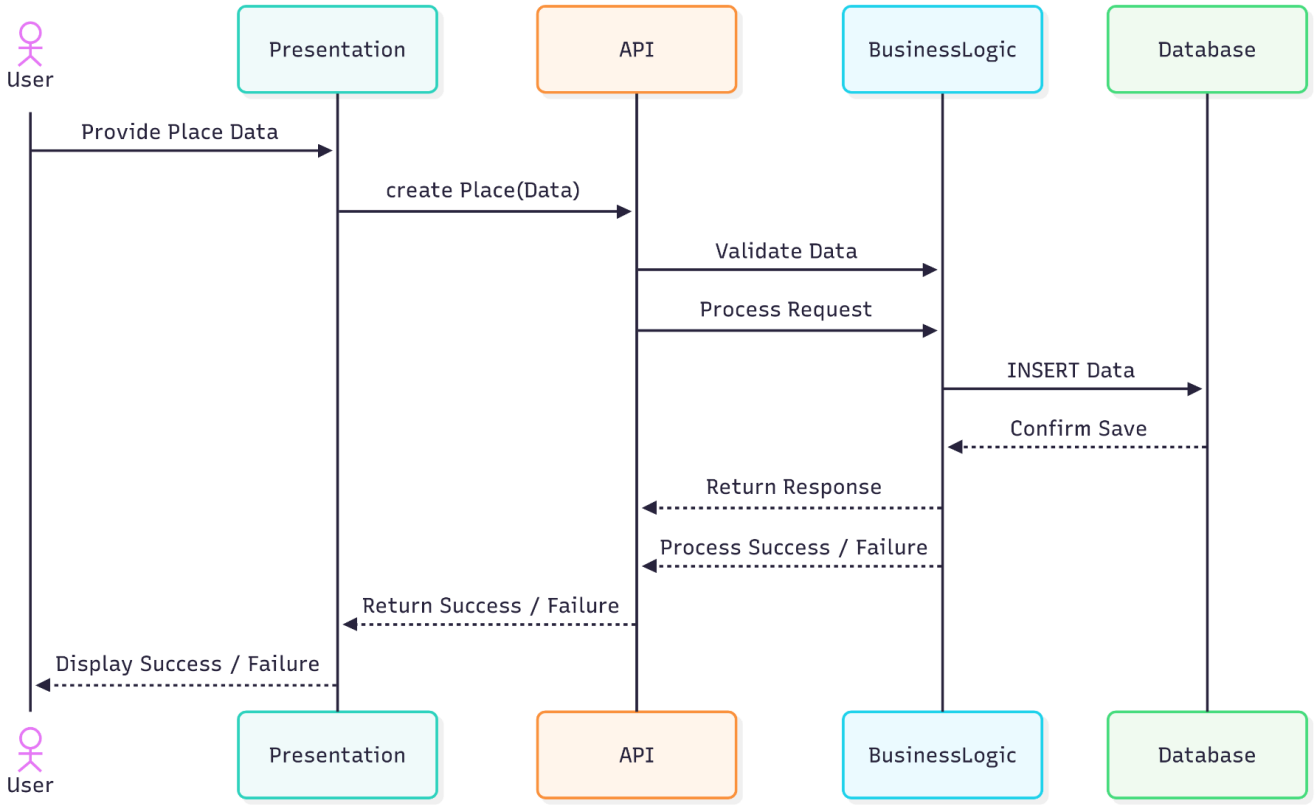
3. **BusinessLogic → Database**:
   After validation, the business logic instructs the database to store the user's information.

4. **Database → BusinessLogic**:
   The database confirms that the information has been recorded successfully.

5. **BusinessLogic → API**:
   The business logic responds to the API, indicating whether the registration was successful or not.

6. **API → User**:
   The API then communicates and displays the result—success or failure—back to the user.

Sequence Diagram : Place Creation



This sequence diagram demonstrates how a new place is created within the HBnB Evolution application:

1. **User → Presentation → API**:
   The user initiates a request to add a new place through the API.

2. **API → BusinessLogic**:
   The API passes this request along to the business logic layer, which checks and processes the data for the new place.

3. **BusinessLogic → Database**:
   Once validated, the business logic layer sends instructions to the database to save the place information.

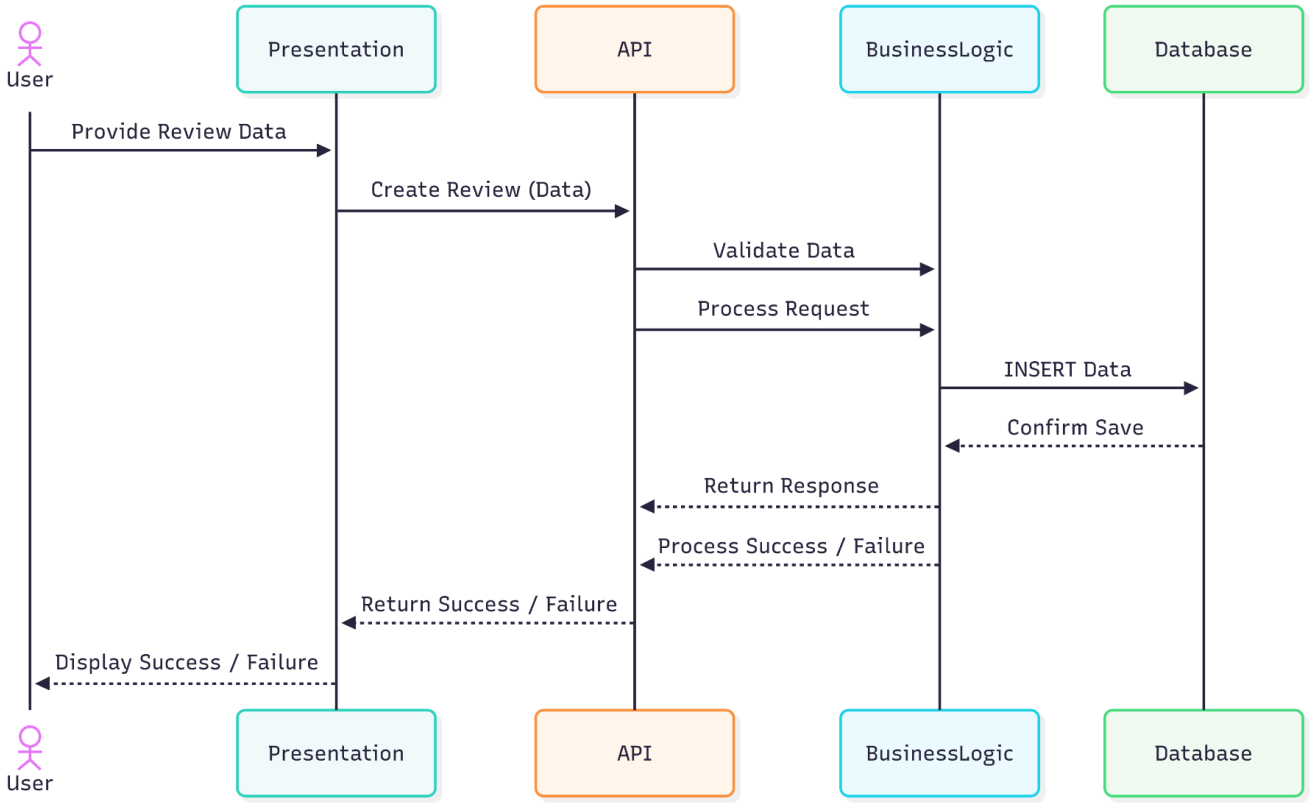
4. **Database → BusinessLogic**:

The database provides confirmation that the place has been recorded successfully.

5. **BusinessLogic → API**:
   The business logic layer informs the API about the result, indicating either success or failure.

6. **API → User**:
   Finally, the API via Presentation communicates the outcome to the user, confirming whether or not the creation of the place was successful.

Sequence Diagram : Review Submission



This diagram describes the steps taken when a user submits a review for a place in the HBnB Evolution app:

1. **User → Presentation → API**:
   The user provides a rating and comment and sends a review request to the API.

2. **API → BusinessLogic**:
   The API relays this information to the business logic layer for data validation and processing.

3. **BusinessLogic → Database**:
   After successful validation, the business logic layer instructs the database to store the review.

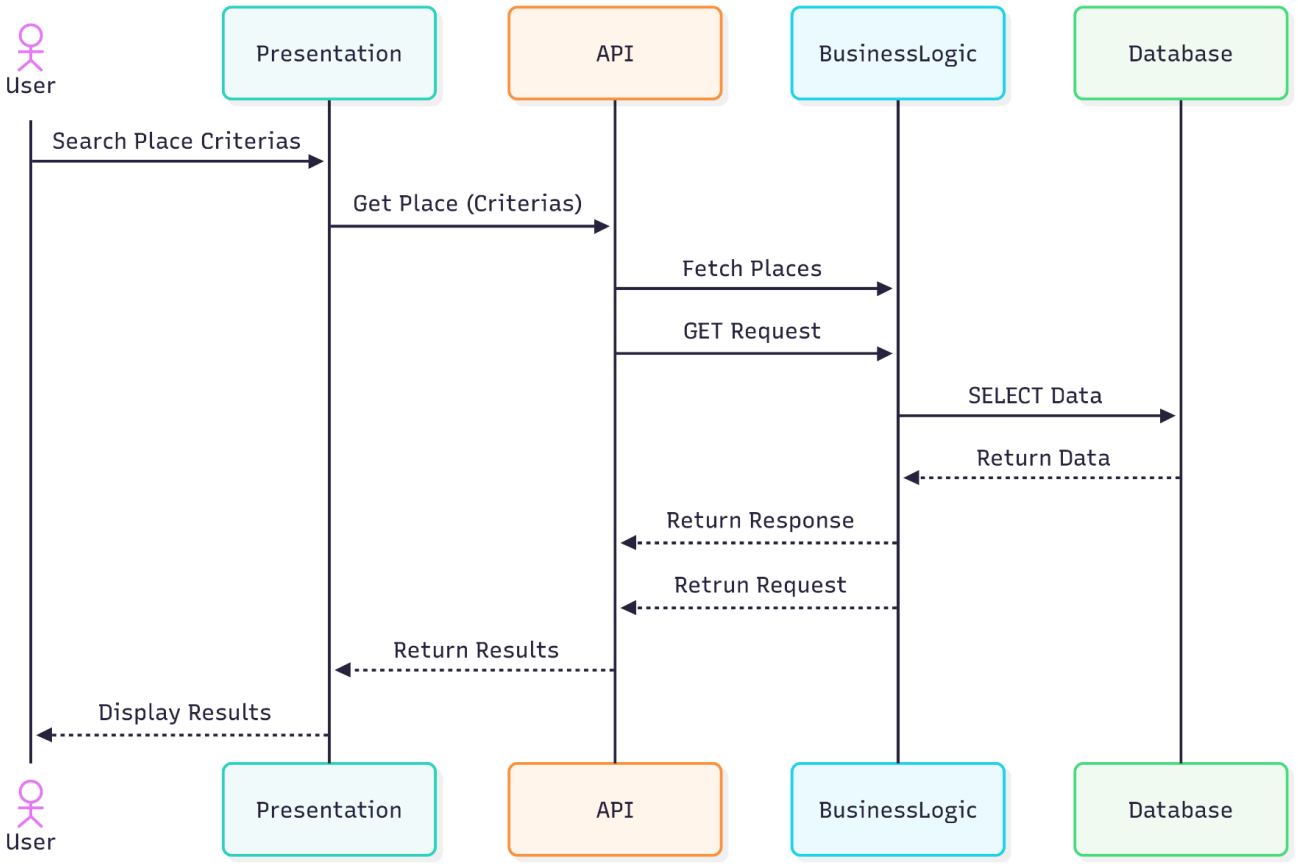
4. **Database → BusinessLogic**:

The database responds to confirm that the review has been stored.

5. **BusinessLogic → API**:
   The business logic notifies the API about whether the operation succeeded or failed.

6. **API → User**:
   The API communicates the result to the user, confirming if their review has been saved.

Sequence Diagram : Places and List Fetching



This diagram illustrates the process for retrieving a filtered list of places based on specific criteria:

1. **User → Presentation → API**:
   The user sends a search request with certain filters to the API.

2. **API → BusinessLogic**:
   The API relays this request to the business logic layer for validation and handling.

3. **BusinessLogic → Database**:
   The business logic layer queries the database to find places matching the provided criteria.

4. **Database → BusinessLogic**:
   The database returns all corresponding places to the business logic layer.

5. **BusinessLogic → API**:
   The business logic sends the result set back to the API.

6. **API → User**:
   Finally, the API provides the search results or an appropriate error message to the user.

Conclusion

At the end of this initial part, we have created a complete set of technical documentation.
This documentation ensures that we have a solid understanding of the application design and architecture. This whole document will be updated as needed during the development of the HBnB Evolution, in order to match any future change made in its architecture.