Technical Documentation for high-level package diagram :

This diagram provides a high-level overview of the three-layer architecture implemented in our HBnB project for Holberton School.
It highlights the main components of each layer and their respective responsibilities.

The architecture is structured around the following three layers:

1. Presentation Layer

This package groups all client-facing services, including:

- The user interface
- API endpoints that facilitate communication between the client and the server

Responsibilities:

- Handle HTTP requests
- Validate input data
- Call business logic functions
- Return structured HTTP responses (typically in JSON format)

2. Business Logic Layer

This package defines the core functional rules of the application.

Responsibilities:

- Define and manipulate domain models (User, Place, etc.)
- Enforce business rules (e.g., a Place must have an associated host)
- Coordinate data flow between the presentation and persistence layers

3. Persistence Layer

This package manages the interaction with the data storage system (e.g., database or file-based storage).

Responsibilities:

- Abstract and encapsulate data access logic
- Read and write data
- Hide the details of the storage implementation
- Provide classes to persist and retrieve model instances

Use of the Facade Pattern

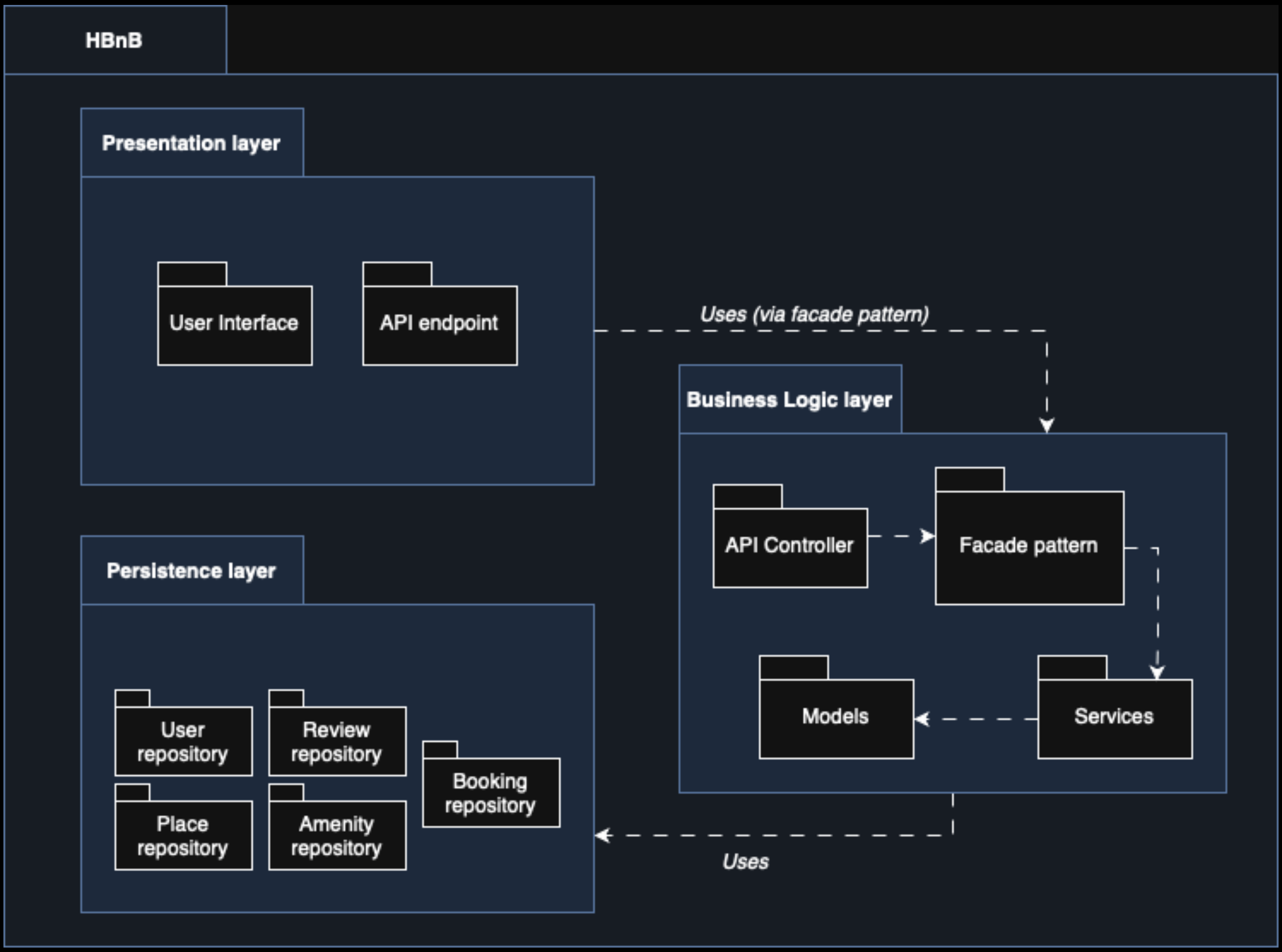
In this project, we implemented a Facade pattern within the business logic layer.

Purpose:
The facade serves as a unified interface for a group of classes or subsystems. It encapsulates the complexity of multiple interactions behind a single method call.

Example Use Case:
When an API endpoint (e.g., POST /places) is called, it invokes a facade method such as create_place(). This method:

- Validates the input data
- Instantiates the necessary model objects (Place, User, etc.)
- Persists them in the database
- Returns a well-structured result to the presentation layer

This pattern allows the API layer to remain clean and simple while centralizing complex logic within the business layer.


HBnB
Presentation layer
User Interface
API endpoint
Uses (via facade pattern)
Business Logic layer
API Controller
Facade pattern
Models
Services
Persistence layer
User repository
Review repository
Booking repository
Place repository
Amenity repository
Uses