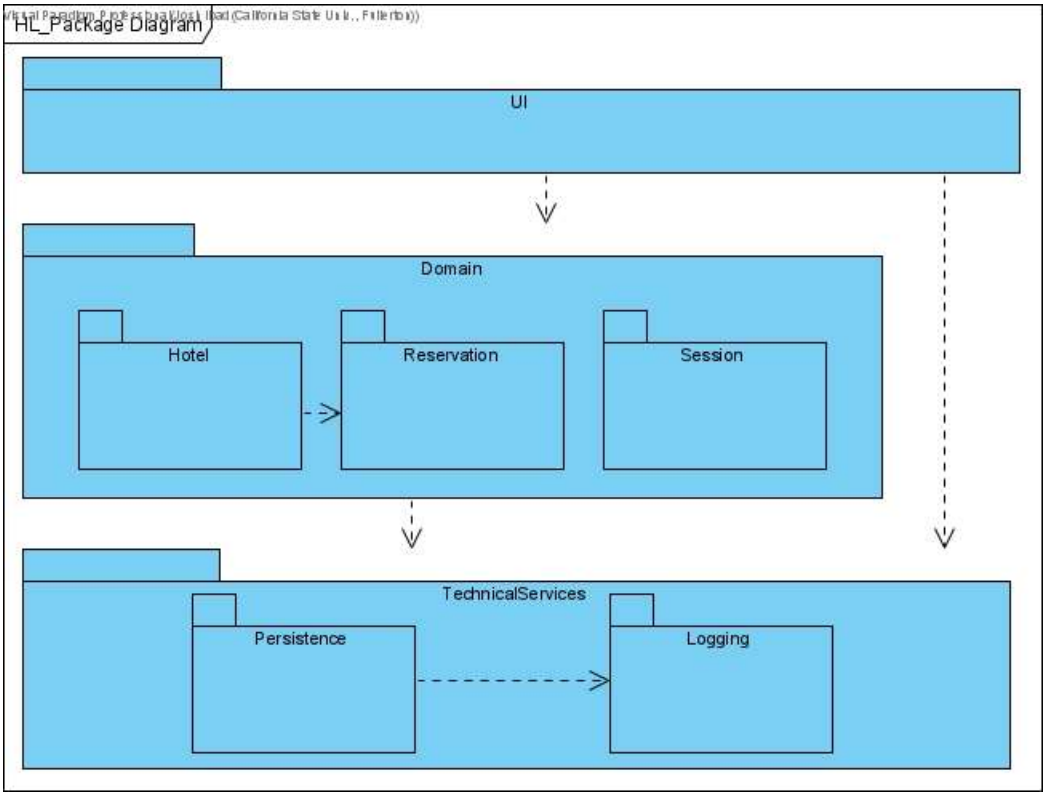
3 Logical View

3.1 Package Diagrams



3.1.1 Presentation (UI) Layer Components

N/A

3.1.2 Domain (Application) Layer Components

3.1.2.1 Hotel

The *Hotel* domain component is the component responsible for the Hotel concept, as a container of Rooms. The primary corresponding use case that the *Hotel* component is responsible for is the *Manage Hotel Rooms* use case. The *Hotel* component holds in it, classes such as Hotel and Room as well as the interface HotelHandler. The *Hotel* component's responsibility is to keep track of a Hotel and it's Rooms, along with the description of said Rooms. These Rooms have a price, descriptions, bed count, along with a BedType and RoomType. Any functionality, class, enumeration, or interface to support the *Manage Hotel Rooms* use case are packaged into the *Hotel* component.

3.1.2.2 Reservation

The *Reservation* domain component is the component responsible for the concept of Reservations and payment related concepts such as BillingMethods, Transactions, and Accounts. The primary corresponding use case that the *Reservation* component is responsible for is the *Manage Reservations* use case. The *Reservation* holds in it, classes such as Reservation, BillingMethod, Transaction, Account as well as the interface ReservationHandler. The *Reservation* component's responsibility is to keep track of reservations and payments for these Reservations, along with the times of reservation, time of payment, billing information such as card number, expiration, cvv codes, card holder name, address and contact, card types, payment dates, etc. Any functionality, class, enumeration or interface to support the *Manage Reservation* use case that deals explicitly with the Reservation rather than the Room, is packaged into the *Reservation* component.

3.1.2.3 Session

The *Session* domain component is the component responsible for serving as an actor's user Session in the system. The *Session* component facilitates the authentication functionality, which is not an explicit use case mentioned earlier, but is an essential supporting functionality that allows all other use cases to occur. The *Session* component takes care of user roles and permissions, making sure that they are both able to reach the functionalities that they are intended to have, and ONLY the functionalities and interfaces that they are intended to have. The Session draws greatly from the TechnicalServices layer's *Persistence* component to store user credentials and roles.

3.1.3 Technical Services Layer Components

3.1.3.1 Persistence

The *Persistence* technical services component is the component responsible for managing persistent data, that is, data that persists throughout system usage and system shutdowns. Ideally, a *Persistence* component would perform all CRUD operations to store and manage information, but temporarily, it solely performs Reads from regular text files in order to read the configuration of the UI and Logging.

3.1.3.2 Logging

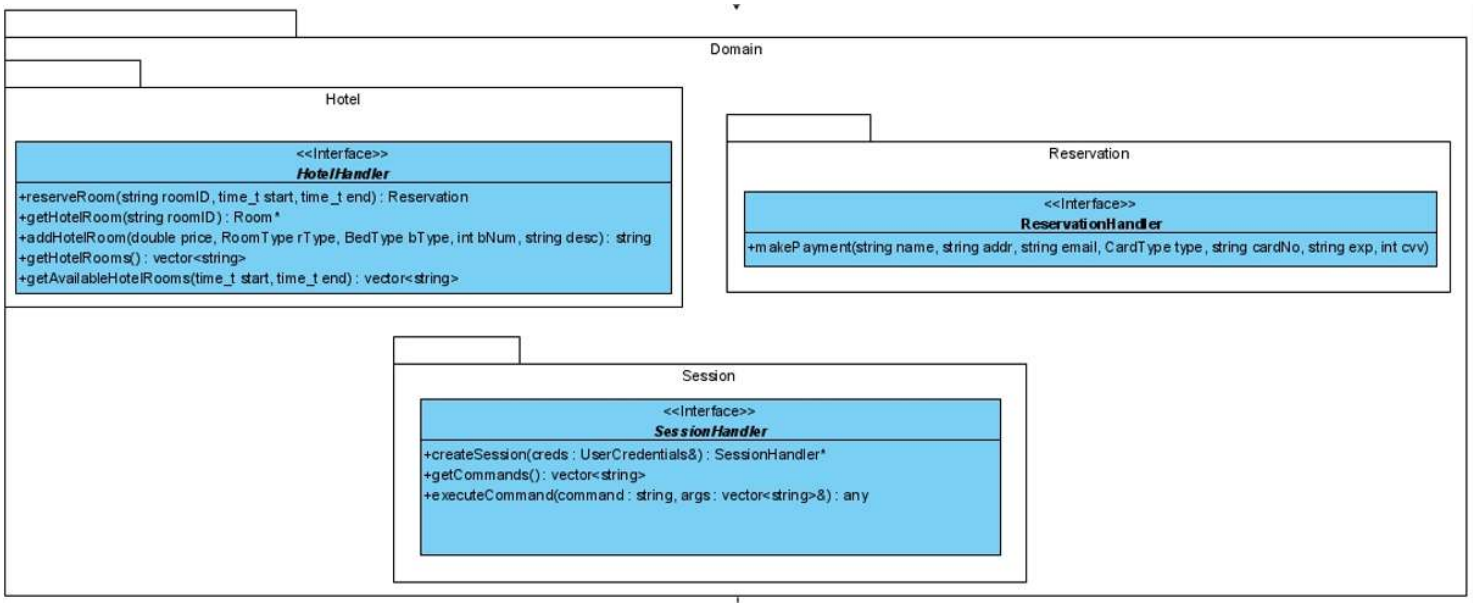
The *Logging* technical services component is the component responsible for logging system events and interactions. It takes care of printing messages into the command line in a well formatted manner, printing out the date and time of an event, followed by messages from the corresponding event. Logging provides insight to the functionality of technical services components such as the Logging and Persistence along with the UI layer. It also has the potential use of logging system events occuring the Domain layer if found necessary. With the first iterations of the Hotel Reservation system being a console application, the *Logging* component is essential for giving the user information on the system.

3.2 Interface Diagrams

3.2.1 Presentation (UI) Layer Interface Diagrams

N/A

3.2.2 Domain Layer Interface Diagrams



3.2.3 Technical Services Interface Diagrams