**1. UML Class Diagram Overview**

The **UML Class Diagram** for the AI Hotel Reservation System defines the structural design of the system, showcasing classes, their attributes, methods, and relationships. This diagram serves as a blueprint for implementing the system's functionality while ensuring modularity and scalability.

**2. Key Classes & Functionality**

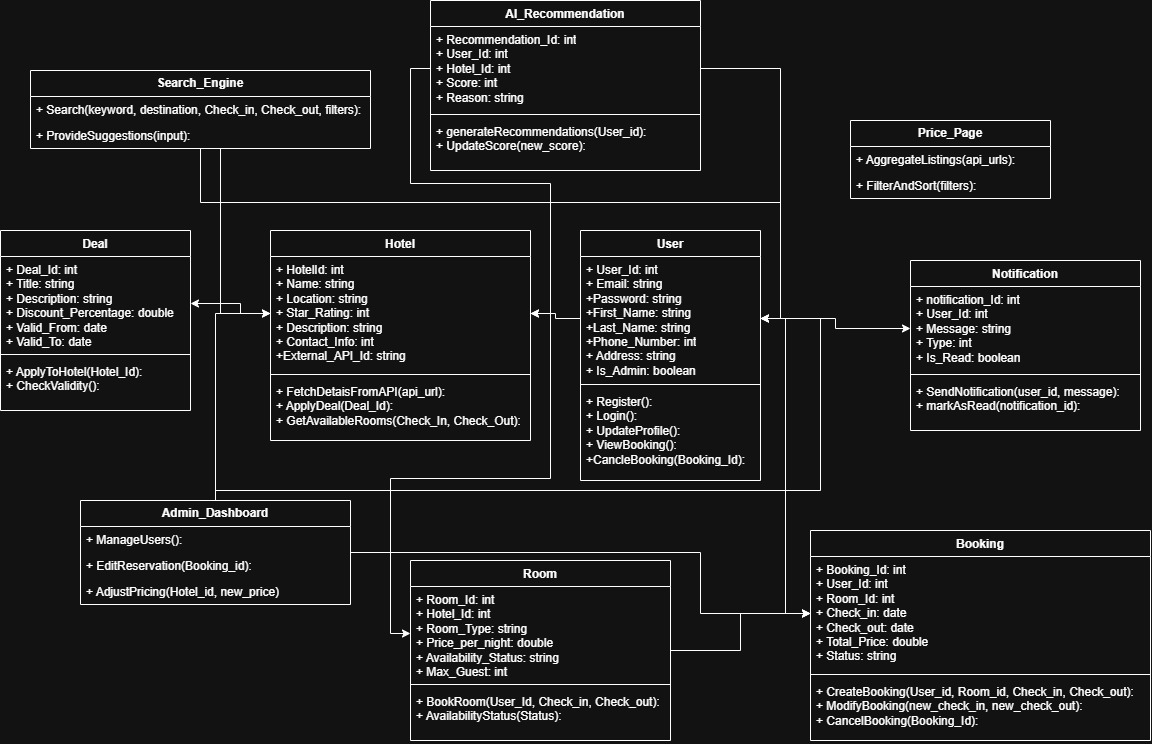
1. **Search\_Engine**
   * **Purpose:** Handles hotel searches and AI-driven recommendations.
   * **Key Methods:**
     + **Search(keyword, destination, dates, filters):** Retrieves hotel listings.
     + **ProvideSuggestions(input):** Generates AI-powered recommendations.
2. **Hotel & Deal**
   * **Hotel:** Stores hotel details (name, location, rating) and manages API data fetching(*FetchDetailsFromAPI*).
   * **Deal:** Manages promotional discounts and validity checks(*ApplyToHotel, CheckValidity*).
3. **User & Notification**
   * **User:** Manages user profiles, authentication (*Register, Login*), and bookings (*ViewBooking, CancelBooking*).
   * **Notification**: Handles alerts (*SendNotification, MarkAsRead*) for bookings and promotions.
4. **Admin\_Dashboard**
   * **Purpose**: Provides administrative controls for managing users, reservations, and pricing.
   * **Key Methods:**
     + **ManageUsers():** Edits user accounts.
     + **EditReservation(Booking\_id):** Modifies bookings.
5. **Booking & Room**
   * **Booking:** Tracks reservation details (*dates,* *price, status*) and supports modifications (*ModifyBooking*).
   * **Room:** Manages room availability and pricing (BookRoom, CheckAvailability).

**3. Critical Relationships**

* **Associations:**
  + *User ↔ Booking*: A user can have multiple bookings.
  + *Hotel ↔ Room*: A hotel offers multiple room types.
* **Dependencies:**
  + *Search\_Engine → Hotel*: Fetches hotel data for searches.
  + *Admin\_Dashboard → Booking*: Modifies reservation details.

**4. Design Advantages**

* **Modularity**: Isolates features (e.g., search, booking) for independent updates.
* **Extensibility**: Easily add new methods (e.g., payment processing) without disrupting existing classes.
* **AI Integration**: The Search\_Engine and AI Recommendation classes enable seamless personalization.

**Figure 1:** This UML design aligns with **Scrum’s iterative approach**, allowing incremental feature development while maintaining a clean, maintainable codebase.