**System Architecture and Component Diagram**

**Project Topic: Hotel Booking and Management Platform**

**Group Members**

|  |  |  |
| --- | --- | --- |
| **Member name** | **Member surname** | **Index number** |
| **Nigar** | **Alkhasova** | **54803** |
| **Burak** | **Ozek** | **55930** |
| **Nurkyz** | **Bolotbekova** | **56434** |
| **Patrik** | **Strzelczyk** | **53120** |

**Architectural Style**

**Chosen Architecture: Layered Architecture**

**Justification:**

1. **Separation of Concerns**: Each layer (UI, Business Logic, Data Access) has a distinct role, making it easier to develop, test, and maintain.
2. **Alignment with Team Structure and Django Framework**: The layered architecture fits well with Django's MVC pattern and supports organized development by a small team.

**Alignment with Requirements:**

* **Performance and Usability**: Clear separation allows frontend optimization for quick hotel searches and streamlined 3-step bookings.
* **Security**: A dedicated backend service layer manages secure operations like authentication and encrypted transactions.

**Scalability/Future Growth:**

* The layered model allows replacing or upgrading individual layers (e.g., swapping the database, redesigning UI) without impacting the whole system.

**Component Identification**

**Component 1 – User Interface (UI)**

* **Role**: Displays hotel listings, handles user input, and presents booking summaries and account management.
* **Addressed Requirements**: Room search, booking process, reviews, hotel photos, form validation, user-friendliness.

**Component 2 – Authentication and User Management**

* **Role**: Manages secure login, registration, password recovery, and session tracking.
* **Addressed Requirements**: Secure login, password recovery, encrypted passwords (security), GDPR compliance (regulatory).

**Component 3 – Booking Engine**

* **Role**: Handles booking logic, availability checking, date validation, cancellations, and confirmation.
* **Addressed Requirements**: Online booking, cancellations, seasonal pricing, booking summary, confirmation emails.

**Component 4 – Hotel Management Module**

* **Role**: Used by staff and admins to update room availability, assign rooms, and manage reports.
* **Addressed Requirements**: Update availability, analytics reports, check-ins/check-outs.

**Component 5 – Payment Gateway Integration**

* **Role**: Processes secure payments via third-party APIs like Stripe or PayPal.
* **Addressed Requirements**: Multiple payment methods, data protection compliance, booking completion.

**Component 6 – Database & Data Storage**

* **Role**: Stores user accounts, room data, booking history, hotel details, and reviews.
* **Addressed Requirements**: Customer booking history, data storage, analytics reports, session consistency.

**Traceability to Requirements & Scalability**

* **Traceability**: Each component was designed based on functional and non-functional requirements outlined in the specification. For example, the “Booking Engine” maps directly to booking functionality, while “Authentication” addresses GDPR and security constraints.
* **Scalability**: The layered architecture allows horizontal scaling (e.g., adding more booking servers during peak seasons) and flexible upgrades (e.g., UI redesign or switching payment APIs).

UML Component Diagram

