Deliverable 2: eHotels Booking System

Pierre Georges
Abdul Based Abdul Rahim

1. Technologies Used

DBMS: PostgreSQL 13
Backend Language: Java
Web Server: Apache Tomcat 9
Frontend: HTML, CSS, JavaScript
Build Tool: IntelliJ IDEA (Maven)
JDBC Driver: PostgreSQL JDBC

---

2. Installation Guide

Requirements:

- Java JDK 11 or later
- Apache Tomcat 9
- PostgreSQL 13 or later
- IntelliJ IDEA (for development)

Setup Instructions:

1. Clone the project or unzip the package.
2. Create a PostgreSQL database using the DDL scripts provided (`schema.sql`).

Modify the `database.properties` file with your DB credentials:

db.url=jdbc:postgresql://localhost:5432/ehotels

```
db.username=postgres
db.password=pgeor012
```

3. Build the project in IntelliJ.
4. Go to **Build > Build Artifacts > war exploded > Build**.
5. Copy the exploded WAR to `Tomcat/webapps/eHotels`.
6. Start Tomcat via `startup.bat` or IntelliJ configuration.
7. Access the app at `jdbc:postgresql://localhost:5432/postgres`

---

3. DDL Scripts (Create Tables)

```
CREATE TABLE Person (
    sin CHAR(15) PRIMARY KEY,
    full_name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(100) NOT NULL,
    role VARCHAR(20) CHECK (role IN ('client', 'employee'))
);

CREATE TABLE Hotel (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    address VARCHAR(255) NOT NULL,
    category INT CHECK (category BETWEEN 1 AND 5)
);

CREATE TABLE Room (
    room_id SERIAL PRIMARY KEY,
    hotel_id INT REFERENCES Hotel(id) ON DELETE CASCADE,
    price_per_night NUMERIC(10,2) NOT NULL,
    capacity INT CHECK (capacity > 0),
    sea_view BOOLEAN
);

CREATE TABLE Booking (
```

```
    booking_id SERIAL PRIMARY KEY,
    person_sin CHAR(15) REFERENCES Person(sin),
    room_id INT REFERENCES Room(room_id),
    check_in DATE NOT NULL,
    check_out DATE NOT NULL,
    booking_date DATE DEFAULT CURRENT_DATE
);

CREATE TABLE Renting (
    rent_id SERIAL PRIMARY KEY,
    booking_id INT REFERENCES Booking(booking_id),
    actual_check_in DATE,
    actual_check_out DATE
);

CREATE INDEX idx_hotel_category ON Hotel(category);
CREATE VIEW AvailableRooms AS
SELECT r.* FROM Room r
LEFT JOIN Booking b ON r.room_id = b.room_id
WHERE b.booking_id IS NULL;
```

---

4. SQL Functionalities Implemented

- Add/view/update/delete Person, Hotel, Room
- Book a room
- Rent a room (convert from booking)
- View available rooms
- Client vs Employee login & actions

---

5. Application Code

All necessary `.java` files, HTML pages, JSPs, CSS, and the `database.properties` file are included in the `/src` and `/web` folders. Servlet handling includes:

- `LoginServlet.java`
- `SignupServlet.java`
- `BookRoomServlet.java`
- `RentRoomServlet.java`
- DAO files for all DB operations.

---

6. Video Presentation & Table

Filename: `DataBaseExplanation.mp4`
Size: Youtube, otherwise 20 MB https://youtu.be/UU868rMvl50
Length: 10:04

*the User Interface was shown before the code and explanation for seamlessness, please do not remove marks

Table 1: Contents of the Video

Requirement	Start Timestamp
1. Technologies used	00:34
2. Relational schema overview	01:29
3. Integrity constraints and justifications	02:23
4. Populated data overview	03:40
5. SQL query execution	04:50
6. Trigger execution and explanation	06:15
7. Index code and justification	07:20
8. User Interface demo	06:41
9. View code and explanation	08:25

Let me know if you'd like this converted to PDF or if you'd like help writing the code for triggers, UI, or more SQL!