CSI2132[A]
Hotel Project Deliverable 2 Group 19
Mustafa Ahmed 300242013| Ashvin Ramanathan 300242541|
Connor States 300254333
March 31, 2023

DBMS Description:

We used a handful of technologies to design our DBMS. Firstly, our database for storing data was designed using PostgreSQL. The database was setup using PGAdmin4. Secondly the backend of our DBMS was programmed with Java using Apache Tomcat to communicate with the database. The full backend was designed using Intellij. The webpages themselves were designed with HTML. However the HTML files were converted to jsp files to bridge the connection between the database and the webpages.

Program Requirements:

To run the program you will need PGAdmin4 to build and run the PostgreSQL database. For the backend of the application, you will need to have java installed. You will also need Intellij installed in order to use Apache Tomcat. In Intellij the SmartTomCat plugin is required. No internet connection is required at this time because both our database and website are locally hosted. With more time, we would have hosted the app as a live service.

Changes Since Deliverable 1:

Our database has remained largely the same since deliverable 1. The key changes are to the *rents* table to include a customer's credit card so that it is possible for an employee to approve a booking. There is now also an extra column in the *works_at* table called position_title to keep a track of each employees job when searching for them.

Code to Build the Database Tables:

PLEASE NOTE: It is not ideal to copy the code from this file to build the tables due to inconsistencies when copy/pasting text from pdf files. Please refer to the SQL and txt files with the rest of our code to build the tables.

```
CREATE TABLE employees(
emp_num SERIAL PRIMARY KEY,
sin VARCHAR,
family_name VARCHAR,
given_name VARCHAR,
address VARCHAR,
manager_id SERIAL
);

CREATE TABLE hotel_chains(
chain_name VARCHAR PRIMARY KEY,
office_location VARCHAR,
num_hotels INTEGER,
phone_numbers VARCHAR
);

CREATE TABLE hotels(
contact_phone VARCHAR PRIMARY KEY,
contact_email VARCHAR,
rating DOUBLE PRECISION,
address VARCHAR,
num_of_rooms INTEGER,
manager_id INTEGER,
city VARCHAR,
hotel_id SERIAL
);

CREATE TABLE rooms(
room_id SERIAL PRIMARY KEY,
room_num INTEGER,
price DOUBLE PRECISION,
amenities VARCHAR,
capacity VARCHAR,
sea_view BOOLEAN,
is_extendable BOOLEAN,
problems VARCHAR
);


CREATE TABLE customers(
sin VARCHAR PRIMARY KEY,
family_name VARCHAR,
given_name VARCHAR,
address varchar,
```

```
email varchar,
registration_date DATE
);

CREATE TABLE works_at(
emp_num INTEGER REFERENCES employees(emp_num),
contact_phone VARCHAR REFERENCES hotels(contact_phone),
chain_name VARCHAR REFERENCES hotel_chains(chain_name),
position_title VARCHAR
);

CREATE TABLE belongs_to(
contact_phone VARCHAR REFERENCES hotels(contact_phone),
chain_name VARCHAR REFERENCES hotel_chains(chain_name)
);

CREATE TABLE has(
contact_phone VARCHAR REFERENCES hotels(contact_phone),
room_id INTEGER REFERENCES rooms(room_id)
);

CREATE TABLE rents(
sin VARCHAR REFERENCES customers(sin),
room_id INTEGER REFERENCES rooms(room_id),
approved BOOLEAN,
credit_card VARCHAR,
start_date DATE,
end_date DATE
);

CREATE FUNCTION remove_works_at()
RETURNS TRIGGER AS $$
BEGIN
   DELETE FROM works_at WHERE emp_num = OLD.emp_num;
   RETURN OLD;
END;
$$ Language plpgsql;

CREATE trigger terminate_employee BEFORE DELETE ON employees FOR EACH ROW
EXECUTE FUNCTION remove_works_at();

CREATE FUNCTION remove_rents()
RETURNS TRIGGER AS $$
BEGIN
   DELETE FROM rents WHERE sin = OLD.sin;
   RETURN OLD;
END;
$$ Language plpgsql;

CREATE trigger cancel_rents BEFORE DELETE ON customers FOR EACH ROW EXECUTE
FUNCTION remove_rents();
```

```
Create view hotel_rooms AS select contact_phone, Count(*) as num_rooms from has Group by contact_phone;
Create view room_by_city AS Select city, SUM(num_of_rooms) AS total_rooms from hotels group by city;

CREATE INDEX employees_sin ON employees(sin);
CREATE INDEX hotels_phone ON hotels(contact_phone);
CREATE INDEX has_phone_and_id ON has(contact_phone, room_id);
```