EAST WEST UNIVERSITY

PROJECT REPORT

Design and Implementation of a Restaurant and Dish Review with Booking System

COURSE DETAILS

PROJECT GROUP NO	03
COURSE TITLE	: Database Systems
COURSE CODE	: CSE302
SECTION	03
ACADEMIC SESSION	: Spring-2025

DATE OF SUBMISSION : 30th May, 2025

Contents

Page Number

INTRODUCTION

The Restaurant and Dish Review with Booking System represents an advanced, database-driven application developed to streamline the processes associated with restaurant discovery, dish recommendations, and table reservations. This system addresses the growing need for a centralized platform that allows users to explore various dining establishments, engage in the submission and evaluation of reviews, and facilitate real-time table bookings. By prioritizing data integrity, the application enhances the efficiency of querying mechanisms for tailored recommendations, ultimately contributing to an improved user experience in the culinary domain.

OBJECTIVES

- **Centralized Restaurant & Dish Management:** Develop and oversee dynamic profiles for a diverse array of restaurants, showcasing essential details such as their location, unique cuisine styles. Each restaurant will feature a meticulously crafted menu, complete with vivid dish descriptions, price points, and options catering to various dietary needs, ensuring every diner can find their perfect meal.

- **User-Centric Review System:** Empower users to share their culinary experiences by rating both restaurants and individual dishes on a dynamic 1–5 star scale. Accompanied by thoughtful comments, this system fosters a community of food lovers who can express their opinions and recommendations, guiding others on their gastronomic journeys.

- **Real-Time Booking System:** Create a seamless dining experience by offering users the ability to check table availability and make hassle-free reservations. With real-time updates on booking statuses—whether confirmed, pending, or cancelled—users can enjoy peace of mind while planning their outings.

- **Intelligent Recommendations:** Delight users with personalized suggestions for top-rated dishes that cater to their unique tastes and preferences. Whether it's a favorite cuisine, or dietary restrictions, our intelligent recommendation system ensures that users discover exceptional meals that suit their cravings.

- **Data-Driven Reports:** Deliver captivating reports that highlight culinary trends. These insights provide a snapshot of popular choices, helping food enthusiasts make informed decisions and explore the best that the dining scene has to offer.

Entity–Relationship (ER) Model

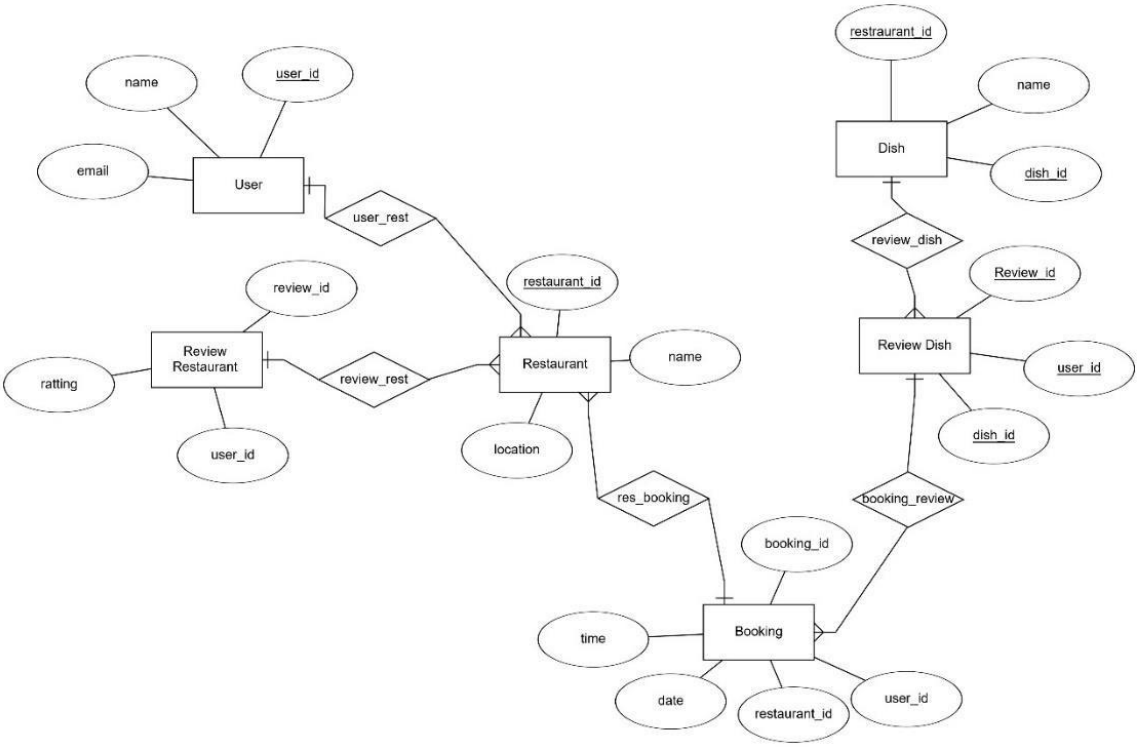

Figure: E-R Model of The Restaurant and Dish Review with Booking System

Schema Design of Project

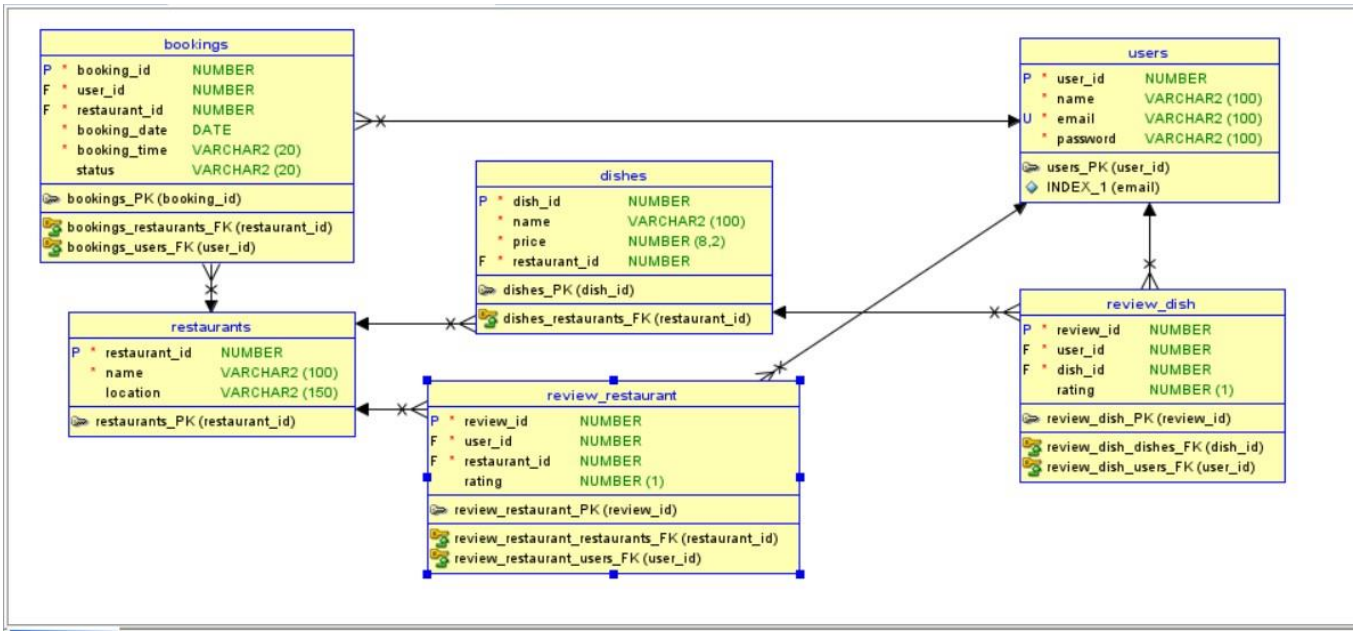

Figure: Schema Design of The Restaurant and Dish Review with Booking System

IMPLEMENTATION CHECKLIST

Description	Remarks
Web Application URL	https://apex.oracle.com/pls/apex/r/apex/app-builder/apps?session=113237323035294
Application ID	97386
Workspace Username	Sam1660_
Workspace Email	samigtxn@gmail.com
Workspace Password	Java890@
Authentication Type	Custom
Username and Password to Login as Admin	**Username:** 1 **password:** pass123
Username and Password to Login as User	**Username:** 2 **password:** pass123 **Username:** 3 **password:** pass123
Number of queries	1
Number of Reports with Forms	6

SQL, HTML CODE & PROJECT OUTPUT WITH SCRRENSHOTS

- **Insertion:**

```sql
-- Insert into USERS
INSERT INTO users (user_id, name, role, password) VALUES (1, 'Alice Ahmed', 'ADMIN', 'pass123');
INSERT INTO users (user_id, name, role, password) VALUES (2, 'Bob Karim', 'CUSTOMER', 'pass123');
INSERT INTO users (user_id, name, role, password) VALUES (3, 'Charlie Reza', 'CUSTOMER', 'pass123');

-- Insert into RESTAURANTS
INSERT INTO restaurants (restaurant_id, name, location) VALUES (1, 'Spice Garden', 'Banani');
INSERT INTO restaurants (restaurant_id, name, location) VALUES (2, 'Urban Eatery', 'Gulshan');
INSERT INTO restaurants (restaurant_id, name, location) VALUES (3, 'The Food Lounge', 'Dhanmondi');

-- Insert into DISHES
INSERT INTO dishes (dish_id, name, price, restaurant_id) VALUES (1, 'Chicken Biryani', 250.00, 1);
INSERT INTO dishes (dish_id, name, price, restaurant_id) VALUES (2, 'Beef Burger', 300.00, 2);
INSERT INTO dishes (dish_id, name, price, restaurant_id) VALUES (3, 'Pasta Alfredo', 280.00, 3);
INSERT INTO dishes (dish_id, name, price, restaurant_id) VALUES (4, 'Thai Soup', 220.00, 1);
INSERT INTO dishes (dish_id, name, price, restaurant_id) VALUES (5, 'Grilled Chicken', 320.00, 2);

-- Insert into REVIEW_RESTAURANT
INSERT INTO review_restaurant (review_id, user_id, restaurant_id, rating) VALUES (1, 1, 1, 5);
INSERT INTO review_restaurant (review_id, user_id, restaurant_id, rating) VALUES (2, 2, 2, 4);
INSERT INTO review_restaurant (review_id, user_id, restaurant_id, rating) VALUES (3, 3, 3, 3);

-- Insert into REVIEW_DISH
INSERT INTO review_dish (review_id, user_id, dish_id, rating) VALUES (1, 1, 1, 5);
INSERT INTO review_dish (review_id, user_id, dish_id, rating) VALUES (2, 2, 2, 4);
INSERT INTO review_dish (review_id, user_id, dish_id, rating) VALUES (3, 3, 3, 5);

-- Insert into BOOKINGS
INSERT INTO bookings (booking_id, user_id, restaurant_id, booking_date, booking_time, status)
VALUES (1, 1, 1, TO_DATE('2025-06-01', 'YYYY-MM-DD'), '7:00 PM', 'Confirmed');

INSERT INTO bookings (booking_id, user_id, restaurant_id, booking_date, booking_time, status)
VALUES (2, 2, 2, TO_DATE('2025-06-03', 'YYYY-MM-DD'), '8:30 PM', 'Pending');

INSERT INTO bookings (booking_id, user_id, restaurant_id, booking_date, booking_time, status)
VALUES (3, 3, 3, TO_DATE('2025-06-05', 'YYYY-MM-DD'), '6:00 PM', 'Confirmed');
```

Figure: SQL Command for data insertion

- **Global Homepage:**

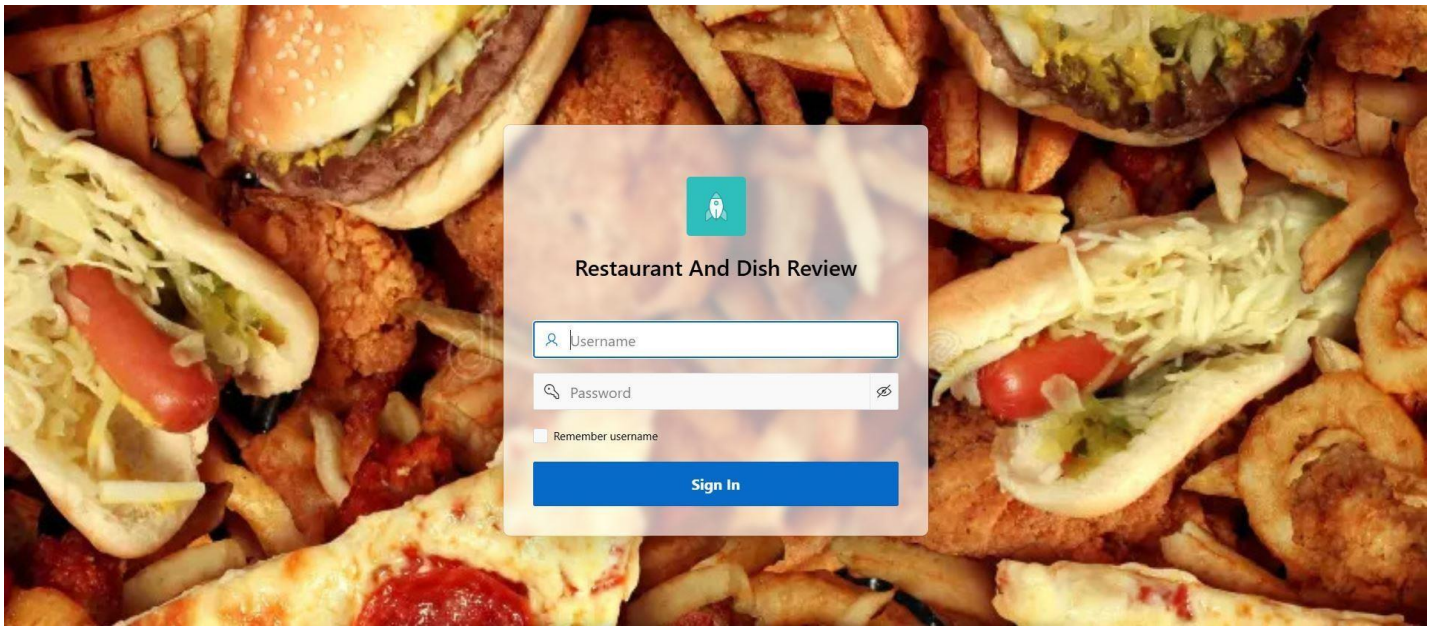

Figure: Global Homepage Interface

- **Homepage:**

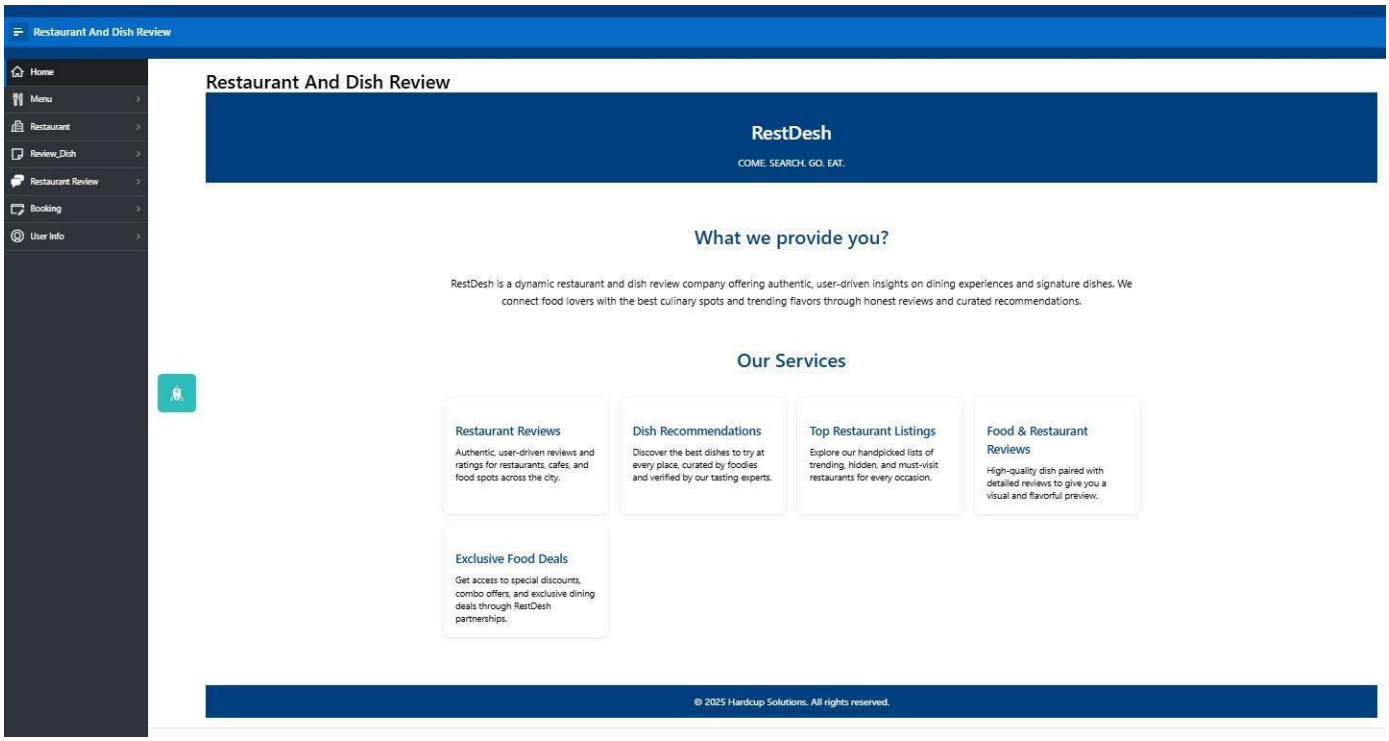

Figure: Homepage Interface

- **Homepage HTML Code:**

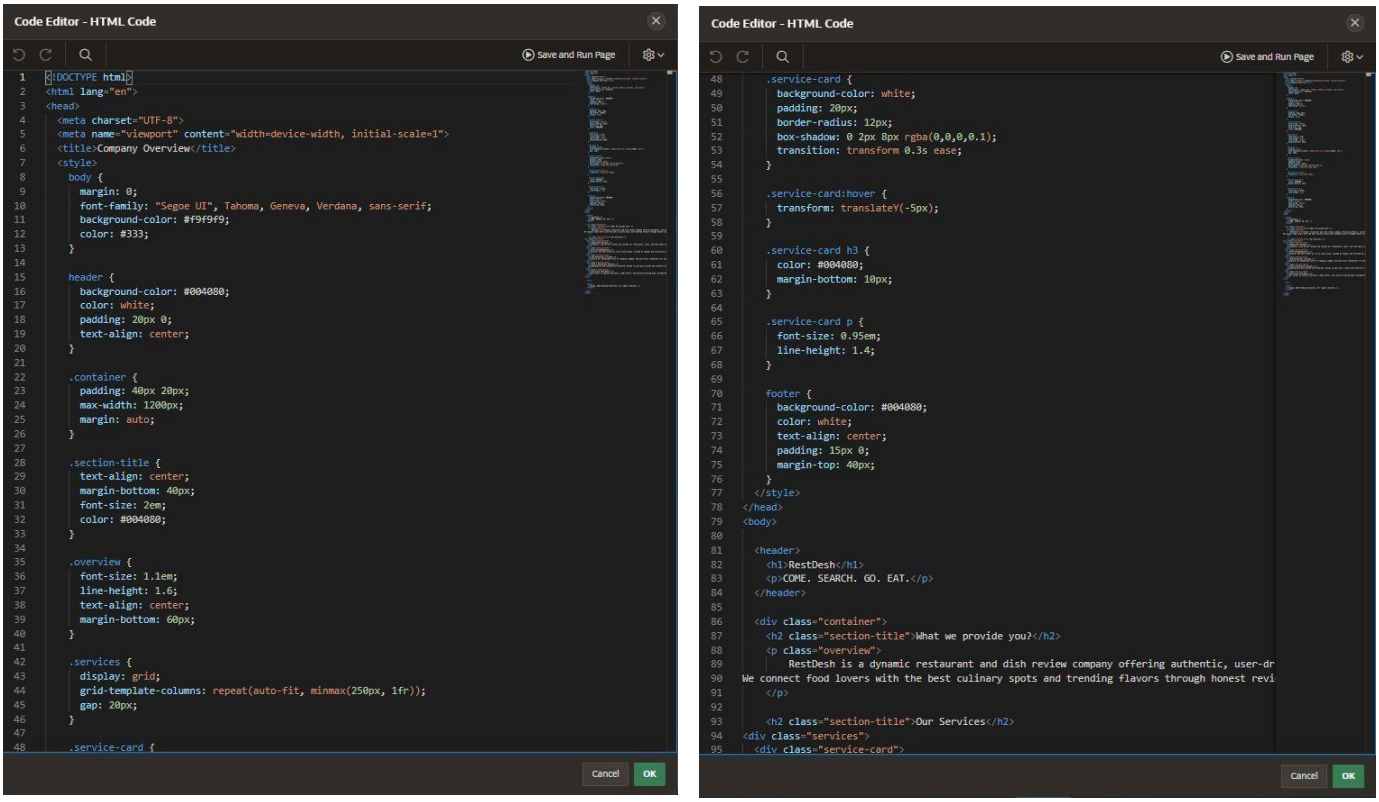

Figure: Homepage Interface HTML Code

- **Menu:**

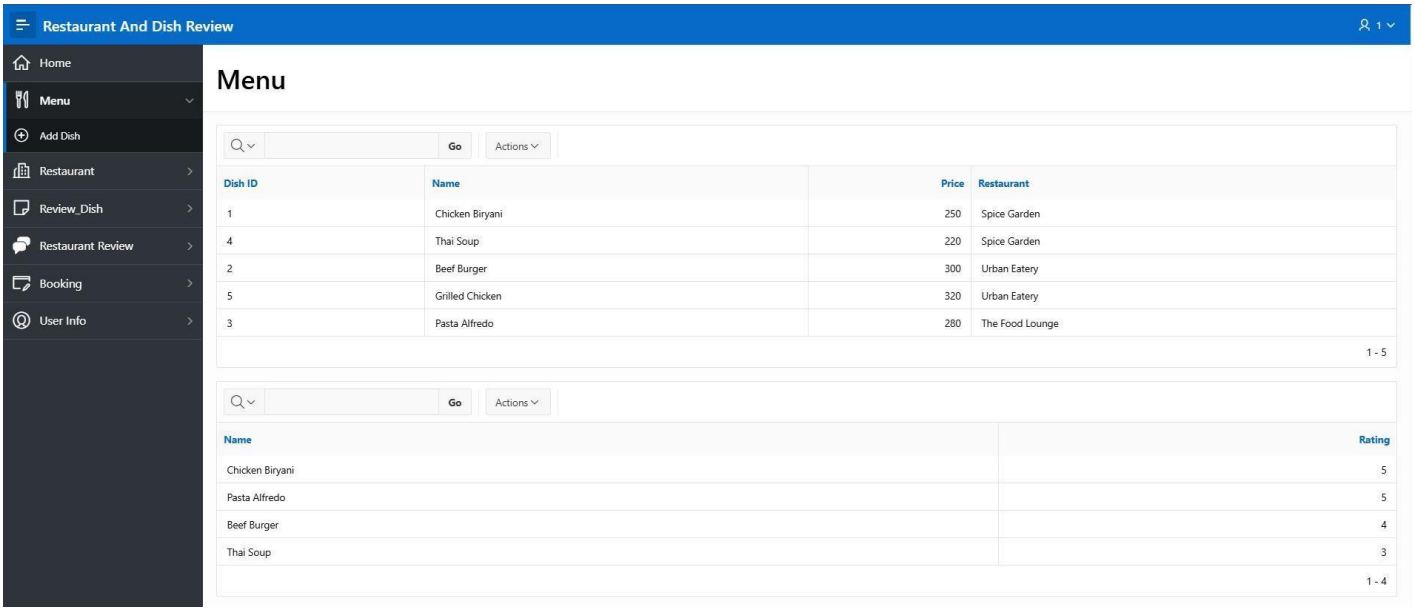

Figure: Menu Interface

- **Adding Dish info (For Admin Account):**

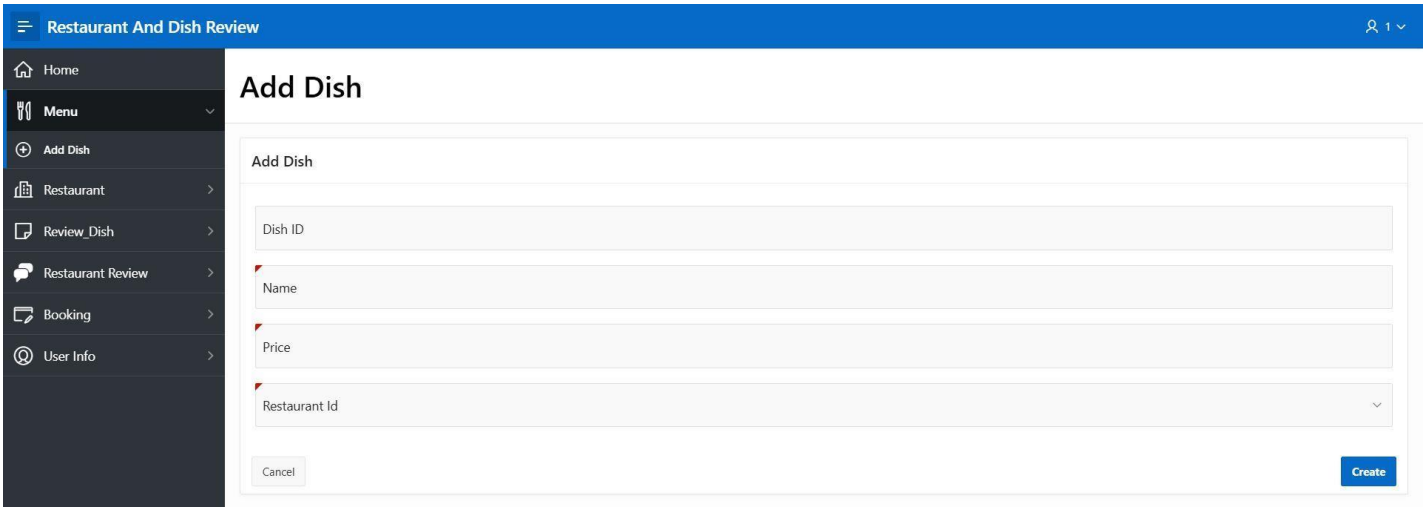

Figure: Adding Dish info interface from Admin account

- **Adding Dish & Restaurant info (For User Account):**

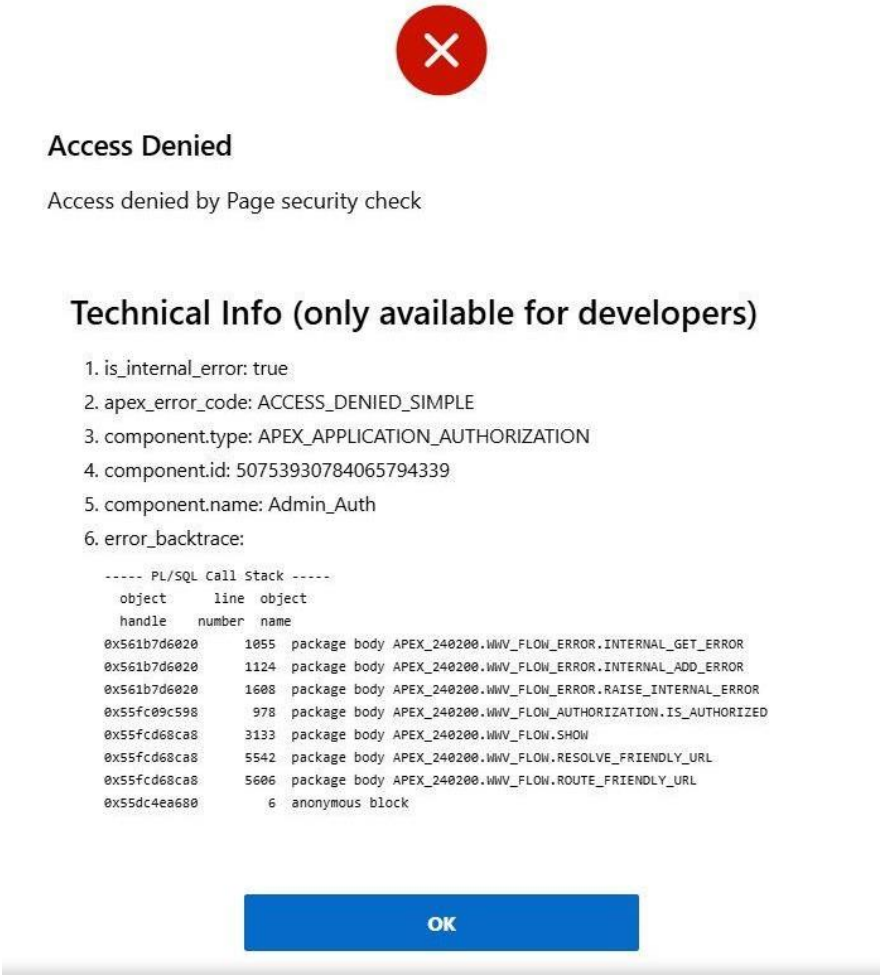

Figure: Adding Dish & Restaurant info interface from User account

- **SQL for Adding Dish info:**

```
-- DISHES Table
CREATE TABLE dishes (
    dish_id         NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    name            VARCHAR2(100) NOT NULL,
    price           NUMBER(8, 2) NOT NULL,
    restaurant_id   NUMBER NOT NULL,
    FOREIGN KEY (restaurant_id) REFERENCES restaurants(restaurant_id) ON DELETE CASCADE
);
```

Figure: SQL Command for Adding Dish info

- ## Restaurant:

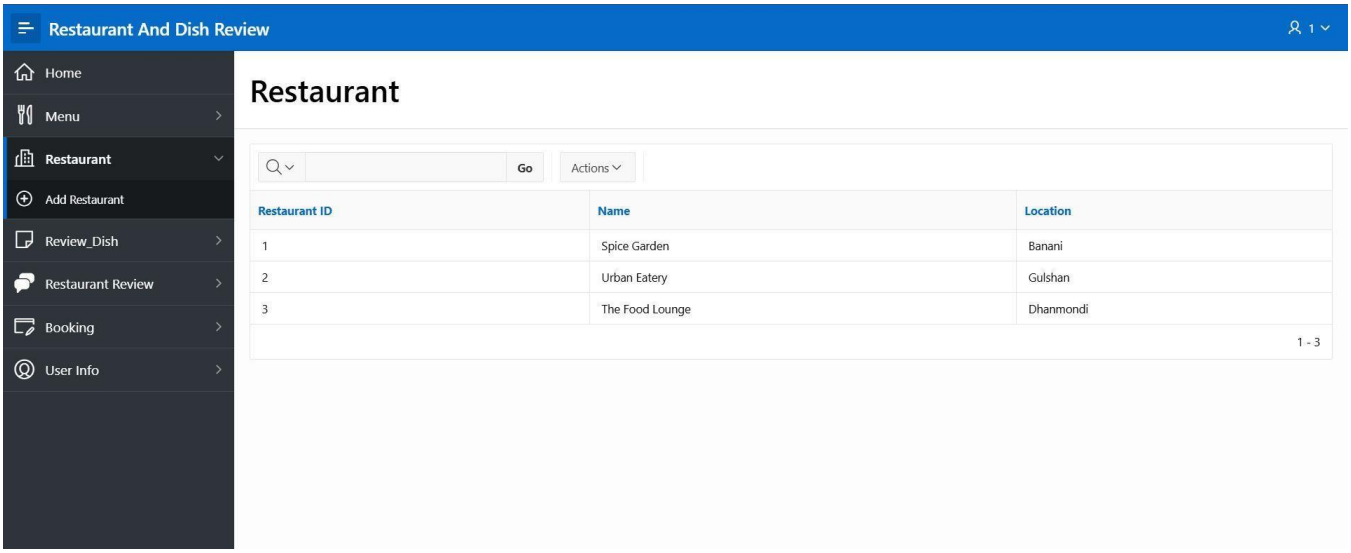

Figure: Restaurant Interface

- ## Adding Restaurant Info:

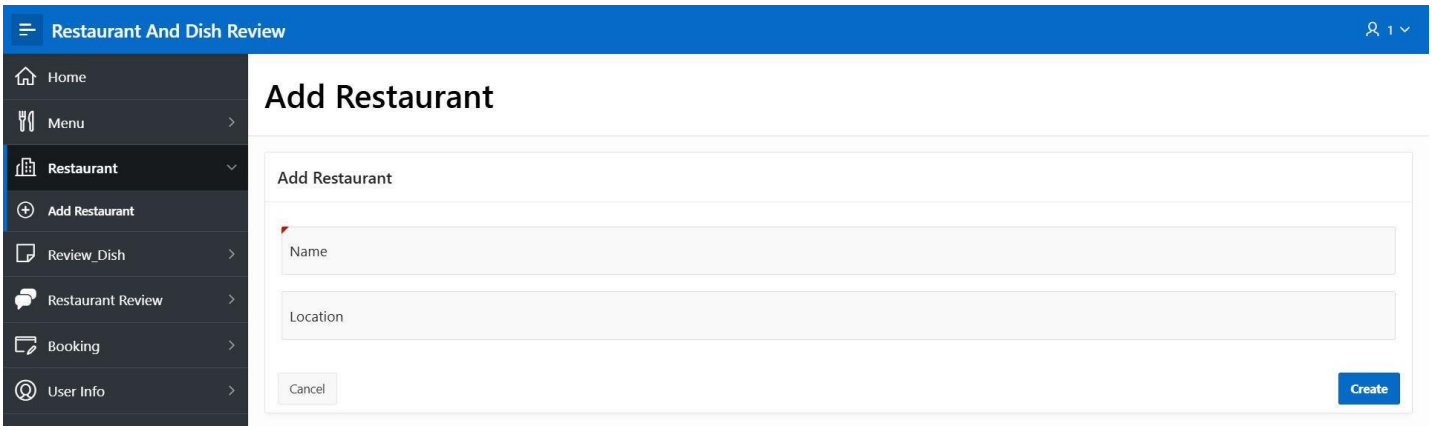

Figure: Adding Restaurant info interface

- ## SQL for Adding Restaurant Info:

```
-- RESTAURANTS Table
CREATE TABLE restaurants (
    restaurant_id   NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    name            VARCHAR2(100) NOT NULL,
    location        VARCHAR2(150)
);
```

Figure: SQL Command for Adding Restaurant info interface

- **Dish_Review:**

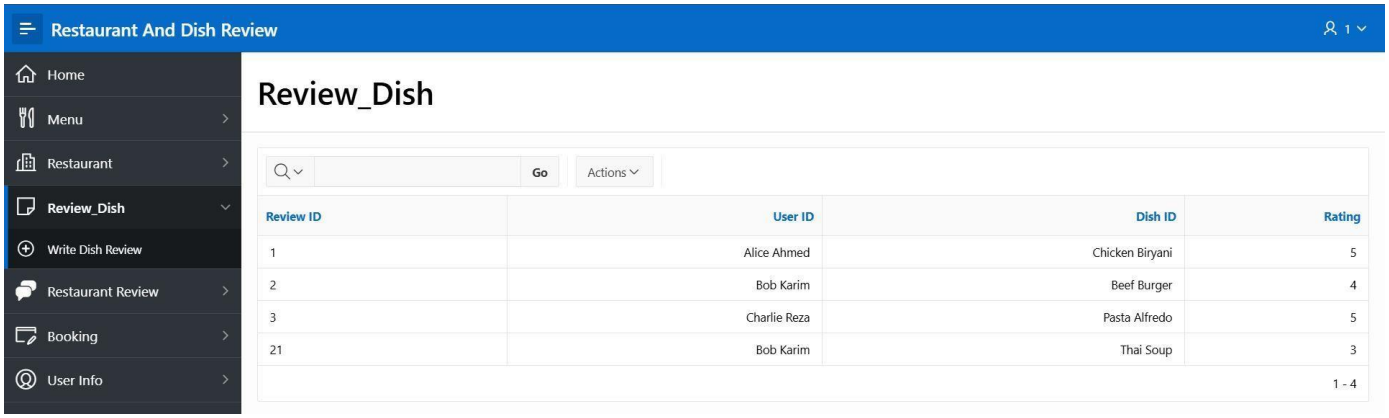

Figure: Dish Review

- **Writing Dish Review:**

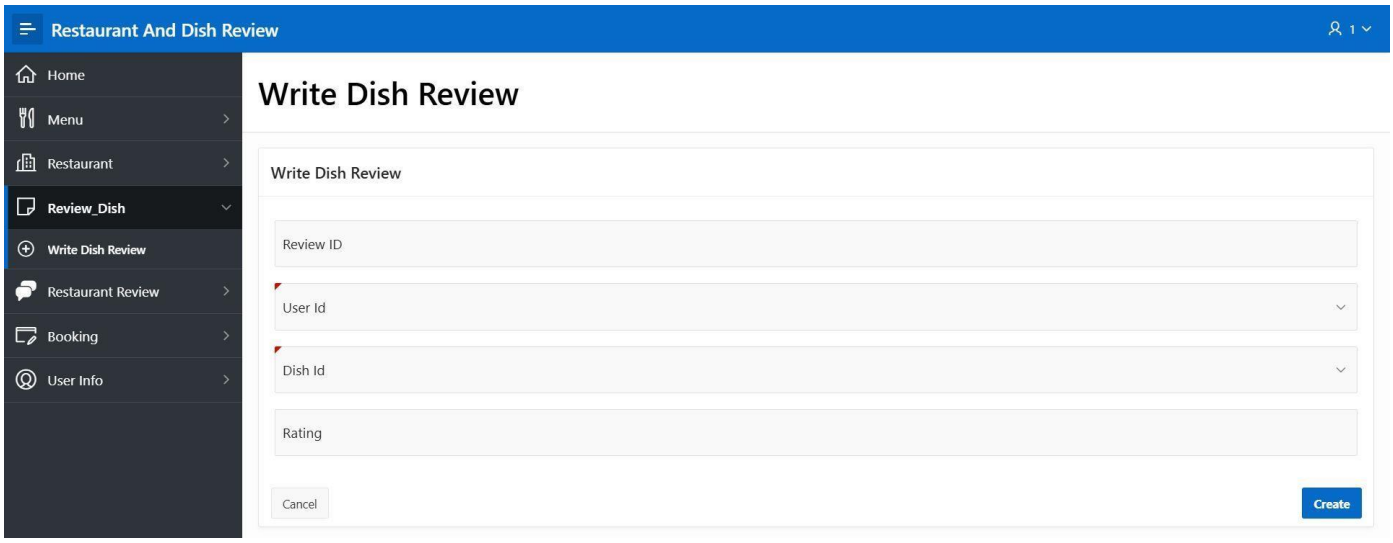

Figure: Adding Dish Review

- **SQL for Writing Dish Review:**

```
-- REVIEW_DISH Table
CREATE TABLE review_dish (
    review_id       NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    user_id         NUMBER NOT NULL,
    dish_id         NUMBER NOT NULL,
    rating          NUMBER(1) CHECK (rating BETWEEN 1 AND 5),
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (dish_id) REFERENCES dishes(dish_id) ON DELETE CASCADE
);
```

Figure: SQL Command for Adding Dish Review

- **Restaurant_Review:**

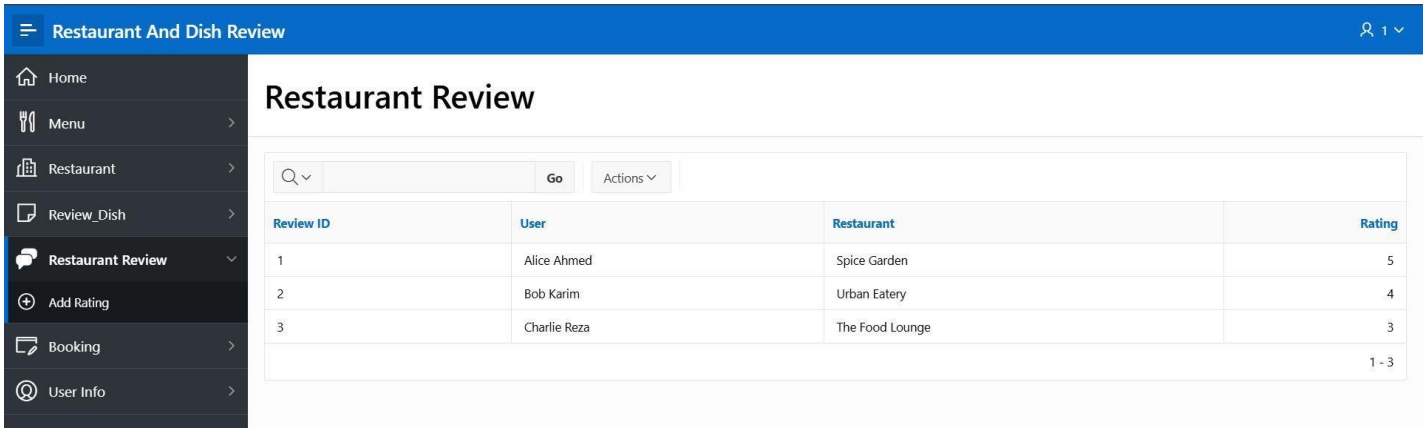

Figure: Restaurant_Review

- **Adding Restaurant Review:**

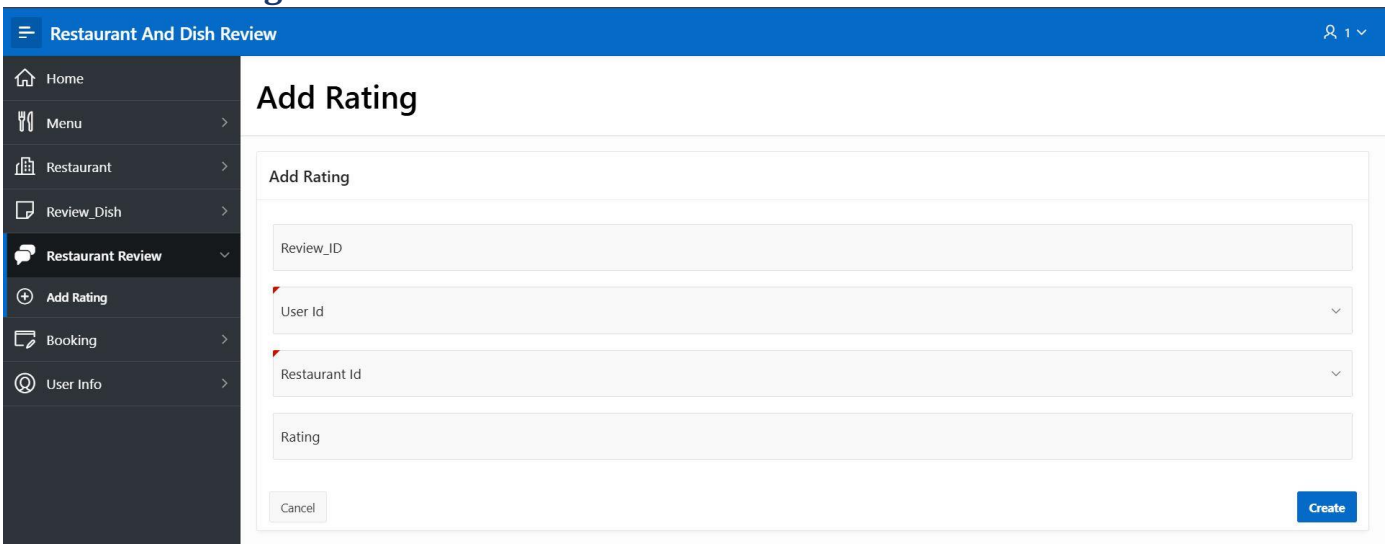

Figure: Adding Restaurant Review

- **SQL for Adding Restaurant Review:**

```
-- REVIEW_RESTAURANT Table
CREATE TABLE review_restaurant (
    review_id       NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    user_id         NUMBER NOT NULL,
    restaurant_id   NUMBER NOT NULL,
    rating          NUMBER(1) CHECK (rating BETWEEN 1 AND 5),
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (restaurant_id) REFERENCES restaurants(restaurant_id) ON DELETE CASCADE
);
```

Figure: SQL for Adding Restaurant Review

- **Booking:**

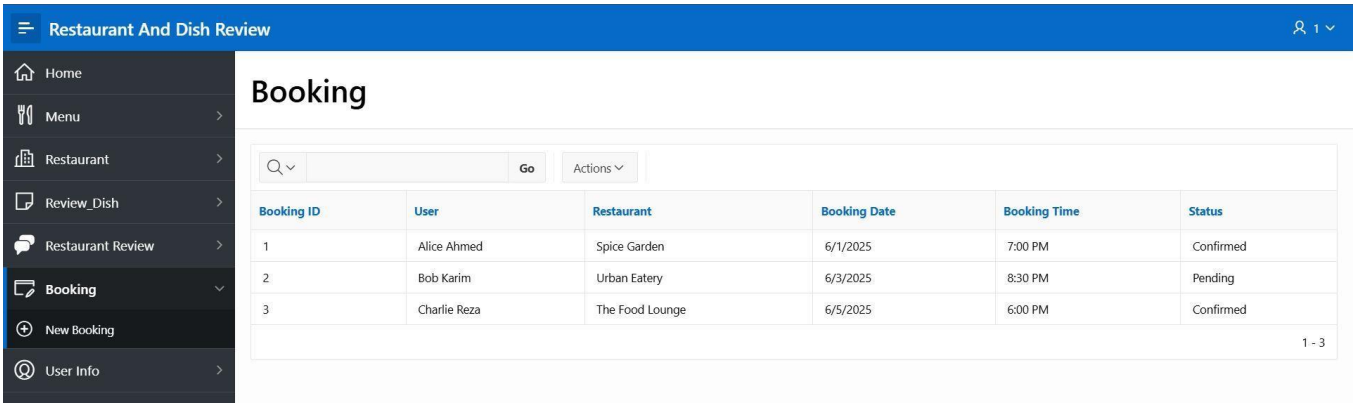

Figure: Booking Interface

- **Adding New Booking:**

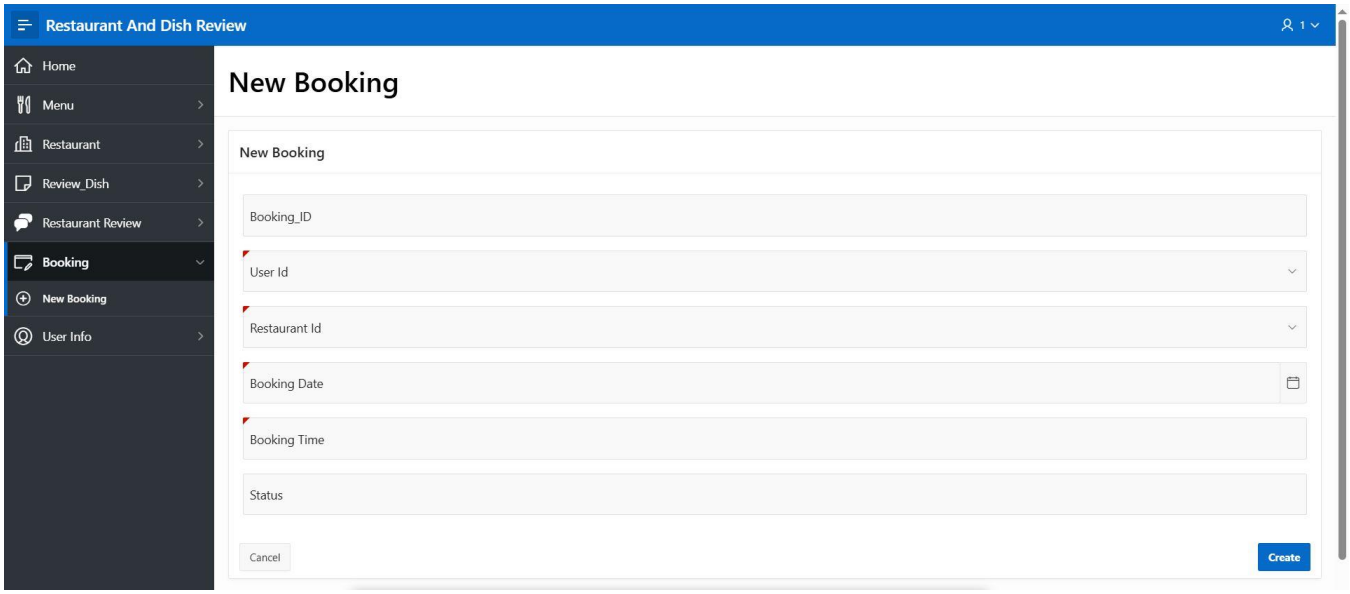

Figure: Adding New Booking

- **SQL for Adding New Booking:**

```
-- BOOKINGS Table
CREATE TABLE bookings (
    booking_id      NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    user_id         NUMBER NOT NULL,
    restaurant_id   NUMBER NOT NULL,
    booking_date    DATE NOT NULL,
    booking_time    VARCHAR2(20) NOT NULL,
    status          VARCHAR2(20) DEFAULT 'Pending',
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (restaurant_id) REFERENCES restaurants(restaurant_id) ON DELETE CASCADE
);
```

Figure: SQL for Adding New Booking

- ## User Info:

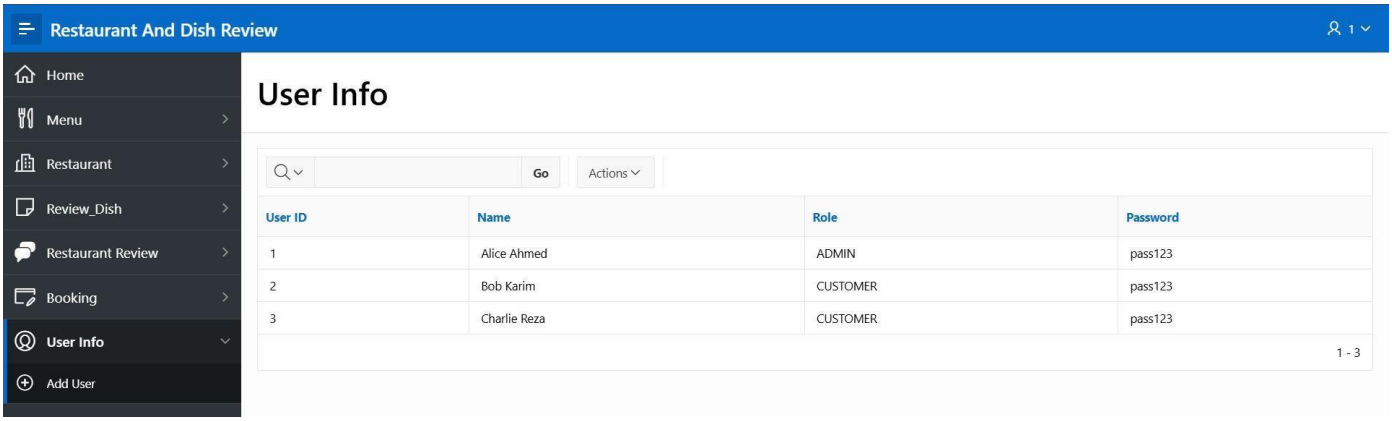

Figure: Menu Interface

- ## Adding User info:

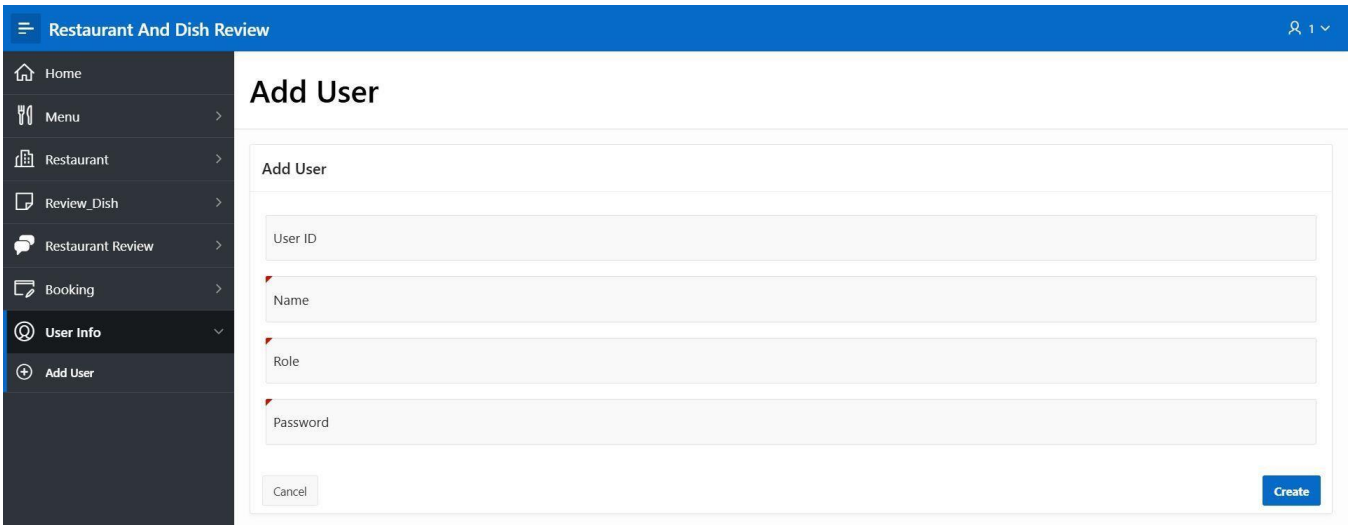

Figure: Adding user info interface

- ## SQL for Adding User info:

```
-- USERS Table
CREATE TABLE users (
    user_id       NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    name          VARCHAR2(100) NOT NULL,
    email         VARCHAR2(100) UNIQUE NOT NULL,
    password      VARCHAR2(100) NOT NULL
);
```

Figure: SQL Command for Adding user info

CONCLUSION

The design and implementation of the Restaurant and Dish Review with Booking System vividly illustrate the power of relational databases in transforming restaurant operations. This innovative system serves as a dynamic hub, consolidating vital information about restaurants, an array of delectable dishes, users, and bookings under one roof. It enriches the dining experience by providing real-time table availability, allowing patrons to effortlessly secure a seat at their favorite spots. User-generated reviews add an authentic touch, guiding diners through a sea of culinary choices based on the shared experiences of others.

A standout element of this system is its personalized dish recommendation engine, which tailors suggestions to individual preferences and tastes. By harnessing sophisticated techniques such as Entity-Relationship (ER) modeling and normalization, the system ensures that the vast amounts of data are neatly organized and easily accessible. Multi-table queries facilitate seamless data retrieval, maintaining consistency and reliability across the board. The intelligent dish recommendation feature shines brightly, offering diners insights into the best options available within their desired price range. Utilizing advanced aggregate functions, the system intelligently analyzes trending dishes, presenting users with engaging recommendations that marry popularity with affordability.

This project has provided rich, hands-on experience in various aspects of database design, backend logic, and user-centric development. With a keen emphasis on scalability and a modular architecture, the system is primed for future enhancements, ensuring it evolves alongside changing user needs and culinary trends. Ultimately, this venture harmoniously blends theoretical understanding with practical application, making a significant impact in the vibrant world of the restaurant industry.