Institute Name: - *IT Vedant Education Pvt. Ltd*

Name: - *Siddhi Shripad Naik*

Email Address: - *shrisiddhi8406@gmail.com*

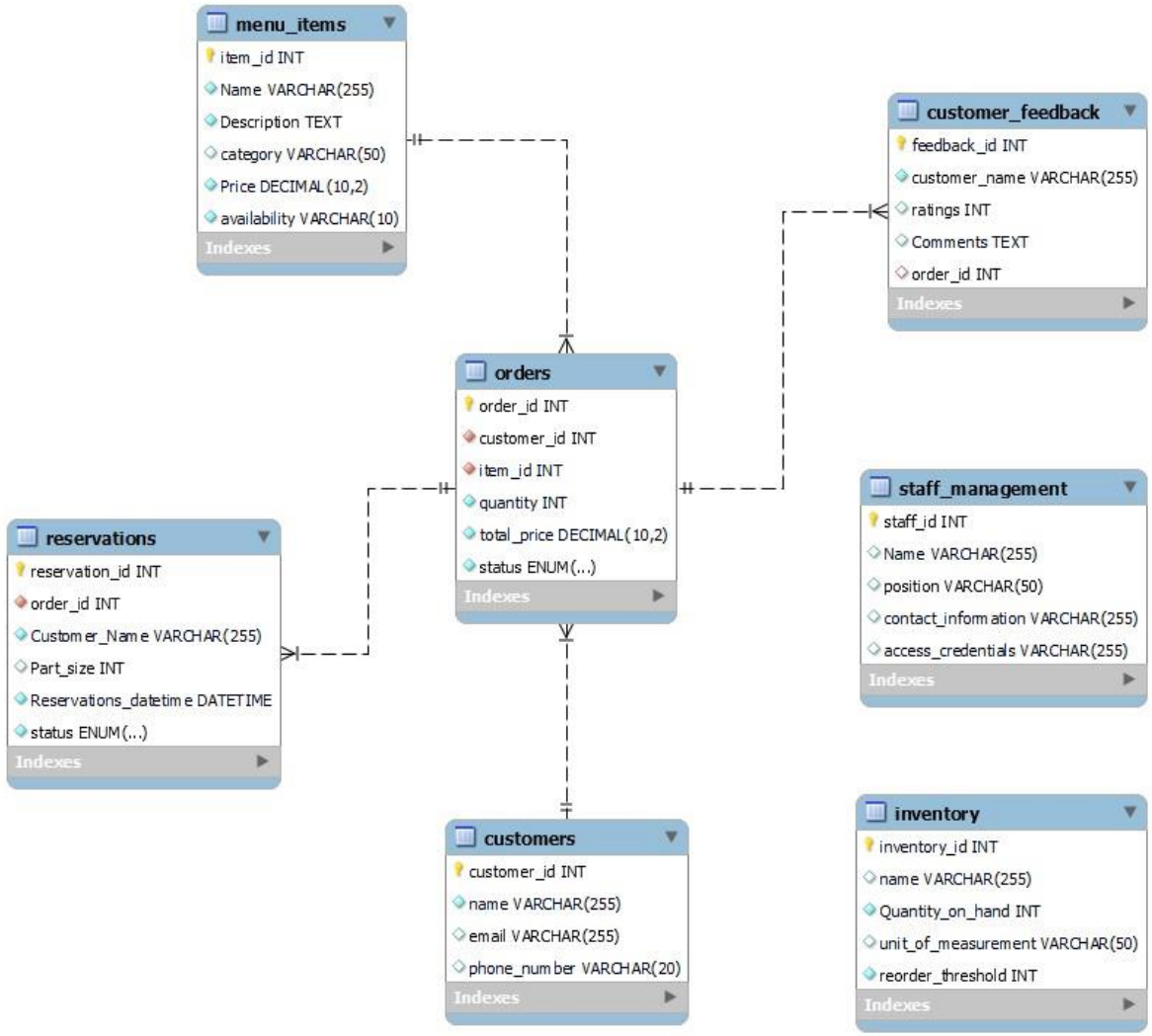
Project Name: -

Restaurant Management System

Project Objective:

The objective of the project seems to be creating a comprehensive system for managing a restaurant. The project aims to create a comprehensive management system that covers various aspects of running a restaurant, including menu, customers, orders, reservations, feedback, and staff. This system can streamline restaurant operations, enhance customer experience, and improve overall efficiency and management.

ER diagram:



Each rectangle represents an entity. Lines between the entities represent the relationships between them. The lines between the entities represent primary keys and foreign keys connecting the tables. The ER diagram showcases the relationships between the different tables, such as menu items, orders, reservations, customer feedback, staff management, and inventory as described in the project description.

Tables:

1. Menu_items:

The system allows management of the restaurant's menu items, including their names, description, categories, prices, and availability status.
Also adding item_id to identify unique menu items, allowing for easy tracking of individual dishes, their descriptions, prices, and availability.

2. Orders:

Table to track customer orders, including order ID, items ID, quantity, total price, customer ID and status (example- pending, completed).
order_id is used for each customer order, enabling efficient order management, tracking, and retrieval of order-related information such as items purchased, quantity, total price, and status.

3. Customers:

Table to manage the details of customers like there Id, name, email, phone-number.

4. Reservations:

Table to manage restaurant reservations, including reservation ID, customer name, party size, reservation date/time, and status.

5. Customer Feedback:

Table to collect feedback from customers, including feedback ID, customer name, ratings, comments, and order_ID.
order_id is used to display the dish id or names which they ordered and giving the ratings on that item.

6. Staff Management:

Tables for staff information, including staff ID, name, position, contact information, and access credentials.

7. Inventory:

Table to track inventory levels of ingredients and supplies used in the restaurant, including item ID, name, quantity on hand, unit of measurement, and reorder threshold.

Table description:

Menu items

Field	Type	Null	Key	Default	Extra
item_id	int	NO	PRI	NULL	
Name	varchar(255)	NO		NULL	
Description	text	NO		NULL	
category	varchar(50)	YES		NULL	
Price	decimal(10,2)	NO		NULL	
availability	varchar(10)	NO		NULL	

Orders

Field	Type	Null	Key	Default	Extra
order_id	int	NO	PRI	NULL	auto_increment
customer_id	int	NO	MUL	NULL	
item_id	int	NO		NULL	
quantity	int	NO		NULL	
total_price	decimal(10,2)	NO		NULL	
status	enum('Pending','Completed')	NO		NULL	

Customers

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	auto_increment
name	varchar(255)	NO		NULL	
email	varchar(255)	YES		NULL	
phone_number	varchar(20)	YES		NULL	

Reservations

Field	Type	Null	Key	Default	Extra
reservation_id	int	NO	PRI	NULL	auto_increment
order_id	int	NO	MUL	NULL	
Customer_Name	varchar(255)	NO		NULL	
Part_size	int	YES		NULL	
Reservations_datetime	datetime	NO		NULL	
status	enum('pending','confirmed','canceled')	NO		pending	

Customer feedback

Field	Type	Null	Key	Default	Extra
feedback_id	int	NO	PRI	NULL	auto_increment
customer_name	varchar(255)	NO		NULL	
ratings	int	YES		NULL	
Comments	text	YES		NULL	
order_id	int	YES	MUL	NULL	

Staff management

Field	Type	Null	Key	Default	Extra
staff_id	int	NO	PRI	NULL	auto_increment
Name	varchar(255)	YES		NULL	
position	varchar(50)	YES		NULL	
contact_information	varchar(255)	YES		NULL	
access_credentials	varchar(255)	YES		NULL	

Inventory

Field	Type	Null	Key	Default	Extra
inventory_id	int	NO	PRI	NULL	auto_increment
name	varchar(255)	YES		NULL	
Quantity_on_hand	int	NO		NULL	
unit_of_measurement	varchar(50)	YES		NULL	
reorder_threshold	int	NO		NULL	

Commands:

```
create database Restaurant_Management;

use Restaurant_Management;

create table Menu_items(item_id integer PRIMARY KEY, Name Varchar(255) Not Null,

Description text not null,

category varchar(50),

Price decimal(10,2) not null ,

availability varchar(10) Not Null);


INSERT INTO Menu_items VALUES

(1,'Chocolate Brownie', 'Warm chocolate brownie served with vanilla ice cream and chocolate sauce', 'Desserts', '60.99', 'yes');

INSERT INTO Menu_items VALUES

(2,'Hot Chocolate', 'Rich and creamy hot chocolate topped with whipped cream', 'Beverages', '30.49', 'yes');

INSERT INTO Menu_items VALUES

(3,'Iced Tea', 'Refreshing iced tea with lemon slices', 'Beverages', '20.49', 'yes');

INSERT INTO Menu_items VALUES

(4,'Cappuccino', 'Classic Italian coffee drink with espresso, steamed milk, and foamed milk', 'Beverages', '35.99', 'yes');

INSERT INTO Menu_items VALUES

(5,'Mango Smoothie', 'Blended mangoes with yogurt and honey', 'Beverages', '74.49', 'yes');

INSERT INTO Menu_items VALUES

(6,'Mushroom Risotto', 'Creamy risotto rice cooked with mushrooms, garlic, and Parmesan cheese', 'Vegetarian', '120.99', 'yes');

INSERT INTO Menu_items VALUES

(7,'Cheeseburger', 'Classic beef burger with cheese, lettuce, tomato, and pickles', 'Burgers', '99.99', 'yes');

INSERT INTO Menu_items VALUES

(8,'Margherita Pizza', 'Traditional Italian pizza with tomato sauce, mozzarella cheese, and basil', 'Pizza', '112.99', 'yes');

INSERT INTO Menu_items VALUES
```

```
(9,'Chicken Alfredo', 'Creamy fettuccine pasta with grilled chicken and Alfredo sauce', 'Pasta', '144.99', 'yes');

INSERT INTO Menu_items VALUES

(10,'Club Sandwich', 'Triple-decker sandwich with turkey, bacon, lettuce, tomato, and mayonnaise', 'Sandwiches', '100.99', 'yes');

INSERT INTO Menu_items VALUES

(11,'Caesar Salad', 'Fresh romaine lettuce with Caesar dressing, croutons, and Parmesan cheese', 'Salads', '80.99', 'yes');

INSERT INTO Menu_items VALUES

(12,'Espresso', 'Strong black coffee brewed by forcing hot water through finely-ground coffee beans', 'Beverages', '25.99', 'yes');

INSERT INTO Menu_items VALUES

(13,'Chicken Caesar Wrap', 'Grilled chicken, romaine lettuce, Parmesan cheese, and Caesar dressing in a tortilla wrap', 'Sandwiches', '90.99', 'yes');

INSERT INTO Menu_items VALUES

(14,'Fruit Salad', 'Assorted fresh fruits served with honey yogurt dressing', 'Salads', '50.99', 'yes');

INSERT INTO Menu_items VALUES

(15,'Apple Pie', 'Traditional American pie made with apples, cinnamon, and pastry crust', 'Desserts', '65.99', 'yes');


CREATE TABLE Customers (
    customer_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255),
    phone_number VARCHAR(20)
);


INSERT INTO Customers VALUES

(101,'John Doe', 'john@example.com', '123-456-7890'),

(102,'Jane Smith', 'jane@example.com', '456-789-0123'),

(103,'Michael Johnson', 'michael@example.com', '789-012-3456'),

(104,'Emily Wilson', 'emily@example.com', '012-345-6789'),
```

```
(105,'David Brown', 'david@example.com', '234-567-8901'),
(106,'Sarah Davis', 'sarah@example.com', '567-890-1234'),
(107,'Matthew Taylor', 'matthew@example.com', '890-123-4567'),
(108,'Jessica Martinez', 'jessica@example.com', '123-456-7890'),
(109,'Christopher Anderson', 'christopher@example.com', '456-789-0123'),
(110,'Amanda Thomas', 'amanda@example.com', '789-012-3456'),
(111,'James Hernandez', 'james@example.com', '012-345-6789'),
(112,'Jennifer Young', 'jennifer@example.com', '234-567-8901'),
(113,'Robert Lee', 'robert@example.com', '567-890-1234'),
(114,'Maria Gonzalez', 'maria@example.com', '890-123-4567'),
(115,'Daniel White', 'daniel@example.com', '123-456-7890');

CREATE TABLE Orders (
    order_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    customer_id INTEGER NOT NULL,
    item_id INTEGER NOT NULL,
    quantity INTEGER NOT NULL,
    total_price DECIMAL(10,2) NOT NULL,
    status ENUM('Pending', 'Completed') NOT NULL,
    FOREIGN KEY (item_id) REFERENCES Menu_items(item_id)
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);

INSERT INTO Orders (order_id,customer_id, item_id, quantity, total_price, status) VALUES
(1,101, 1, 2, 121.98, 'Completed'),
(2,102, 2, 1, 30.49, 'Pending'),
(3,103, 3, 3, 61.47, 'Completed'),
(4,104, 4, 1, 35.99, 'Completed'),
(5,105, 5, 2, 149.98, 'Pending'),
(6,106, 7, 1, 99.99, 'Completed'),
```

```
(7,107, 8, 3, 338.97, 'Pending'),
(8,108, 9, 1, 144.99, 'Completed'),
(9,109, 10, 2, 201.98, 'Pending'),
(10,110, 11, 1, 80.99, 'Completed'),
(11,111, 12, 3, 77.97, 'Pending'),
(12,112, 13, 1, 90.99, 'Completed'),
(13,113, 14, 2, 101.98, 'Pending'),
(14,114, 15, 1, 65.99, 'Completed'),
(15,115, 1, 3, 182.97, 'Pending');

create table Reservations (reservation_id integer auto_increment Primary Key,
order_id integer not null,
Customer_Name varchar(255) Not null,
Part_size integer,
Reservations_datetime datetime Not null,
status enum('pending','confirmed','canceled') Not null default 'pending',
Foreign key (order_id) references orders(order_id)
);

INSERT INTO Reservations (order_id,reservation_id,Customer_Name, Part_size,
Reservations_datetime, status) VALUES
(1,121,'John Doe', 4, '2024-05-01 18:00:00', 'confirmed'),
(2,122,'Jane Smith', 2, '2024-05-02 19:30:00', 'pending'),
(3,123,'Michael Johnson', 6, '2024-05-03 20:00:00', 'confirmed'),
(4,124,'Emily Wilson', 3, '2024-05-04 19:00:00', 'canceled'),
(5,125,'David Brown', 5, '2024-05-05 17:30:00', 'pending'),
(6,126,'Sarah Davis', 2, '2024-05-06 18:30:00', 'confirmed'),
(7,127,'Matthew Taylor', 4, '2024-05-07 19:00:00', 'pending'),
(8,128,'Jessica Martinez', 3, '2024-05-08 20:30:00', 'canceled'),
(9,129,'Christopher Anderson', 6, '2024-05-09 19:45:00', 'confirmed'),
(10,130,'Amanda Thomas', 2, '2024-05-10 18:15:00', 'pending'),
```

(11,131,'James Hernandez', 5, '2024-05-11 19:00:00', 'confirmed'),

(12,132,'Jennifer Young', 4, '2024-05-12 20:00:00', 'pending'),

(13,133,'Robert Lee', 3, '2024-05-13 18:30:00', 'canceled'),

(14,134,'Maria Gonzalez', 6, '2024-05-14 19:30:00', 'confirmed'),

(15,135,'Daniel White', 2, '2024-05-15 17:45:00', 'pending');

create table Customer_feedback(feedback_id integer auto_increment primary key,

customer_name varchar(255) Not null, ratings integer, Comments text,

order_id integer,

foreign key (order_id) references orders(order_id));

INSERT INTO Customer_feedback (customer_name, ratings, Comments, order_id) VALUES

('John Doe', 5, 'Great service and delicious food!', 1),

('Jane Smith', 4, 'The beverage was a bit too sweet for my taste, but overall good experience.', 2),

('Michael Johnson', 5, 'Excellent food quality and prompt service.', 3),

('Emily Wilson', 3, 'Service was slow and the coffee was lukewarm.', 4),

('David Brown', 4, 'Smoothie was refreshing, but the portion size could be bigger.', 5),

('Sarah Davis', 5, 'Burger was amazing! Would definitely come back for more.', 6),

('Matthew Taylor', 4, 'Pizza was delicious, but it arrived a bit cold.', 7),

('Jessica Martinez', 2, 'Pasta was overcooked and tasted bland.', 8),

('Christopher Anderson', 5, 'Great sandwich and friendly staff!', 9),

('Amanda Thomas', 3, 'Salad was fresh, but the dressing was too oily.', 10),

('James Hernandez', 4, 'Espresso was strong and flavorful.', 11),

('Jennifer Young', 5, 'Wrap was tasty and satisfying.', 12),

('Robert Lee', 2, 'Fruit salad was not fresh.', 13),

('Maria Gonzalez', 5, 'Apple pie was heavenly!', 14),

('Daniel White', 4, 'Brownie was delicious, but a bit too sweet for my liking.', 15);

```
create table Staff_management(staff_id integer auto_increment PRIMARY KEY,
Name varchar(255),
position varchar(50),
contact_information VARCHAR(255),
access_credentials VARCHAR(255));

INSERT INTO Staff_management(Name,position,contact_information,access_credentials)
Values
('Peter Young', 'Manager', 'peter@example.com', 'managerpass'),
('Bob Smith', 'Chef', 'bob@example.com', 'chefpass'),
('Samuel Thomas', 'Server', 'samuel@example.com', 'waiterpass'),
('David Wilson', 'Server', 'david@example.com', 'waiterpass'),
('Emma Brown', 'Chef', 'emma@example.com', 'chefpass'),
('Taylor Martinez', 'Bartender', 'taylor@example.com', 'bartenderpass'),
('Victoria Wilson', 'Server', 'victoria@example.com', 'waiterpass'),
('Henry White', 'Chef', 'henry@example.com', 'chefpass'),
('Isabella Clark', 'Server', 'isabella@example.com', 'waiterpass'),
('Jack Lee', 'Bartender', 'jack@example.com', 'bartenderpass'),
('Karen Gonzalez', 'Server', 'karen@example.com', 'waiterpass'),
('Luke Harris', 'Chef', 'luke@example.com', 'chefpass'),
('Megan King', 'Server', 'megan@example.com', 'waiterpass'),
('Nathan Lewis', 'Chef', 'nathan@example.com', 'chefpass'),
('Olivia Hall', 'Server', 'olivia@example.com', 'waiterpass');

create table Inventory(inventory_id integer auto_increment primary key,
name varchar(255), Quantity_on_hand INT NOT NULL,
unit_of_measurement VARCHAR(50),
reorder_threshold INT NOT NULL);

INSERT INTO Inventory (name, quantity_on_hand, unit_of_measurement,
reorder_threshold) VALUES
('Tomatoes', 50, 'pieces', 20),
('Onions', 50, 'pieces', 20),
```

('Garlic', 30, 'bulbs', 15),

('Carrots', 40, 'lbs', 20),

('Potatoes', 50, 'lbs', 20),

('Mozzarella Cheese', 20, 'ounces', 10),

('Pasta', 30, 'lbs', 15),

('Chicken', 25, 'lbs', 10),

('Rice', 50, 'lbs', 20),

('Chocolate Sauce', 15, 'liters', 5),

('Eggs', 60, 'pieces', 30),

('Bread', 40, 'loaves', 20),

('Coffee Beans', 20, 'lbs', 10),

('Milk', 60, 'liters', 30),

('Apples', 50, 'pieces', 20),

('Salmon', 25, 'lbs', 10);

Queries

- ***Join Queries***

To display the names of the items, customer names, and the items they ordered along with their prices

```
SELECT Customers.name AS Customer_Name, Menu_items.Name AS items_Name,
Orders.quantity AS Quantity,
Menu_items.Price AS Price_Per_Item, (Orders.quantity * Menu_items.Price) AS Total_Price
FROM Orders
INNER JOIN Menu_items ON Orders.item_id = Menu_items.item_id
INNER JOIN Customers ON Orders.customer_id = Customers.customer_id;
```

Retrieve customer names and their orders along with the total price for each order:

```
SELECT Customers.name AS Customer_Name, Orders.order_id, (Menu_items.Price *
Orders.quantity) AS Total_Price
FROM Orders
INNER JOIN Menu_items ON Orders.item_id = Menu_items.item_id
INNER JOIN Customers ON Orders.customer_id = Customers.customer_id
GROUP BY Orders.order_id;
```

Retrive customer names, their party size,reservations datetime and their phonenumber

```
select reservations.Customer_name, Part_size,
reservations_datetime,customers.phone_number
from reservations
join orders
on reservations.order_id = orders.order_id
join customers
on customers.customer_id = orders.customer_id;
```

Join Orders with Menu Items

```
SELECT * FROM Orders
INNER JOIN Menu_items ON Orders.item_id = Menu_items.item_id;
```

Retrieve all data from reservations along with orders and customer details.

```
SELECT * FROM Reservations
INNER join orders
on reservations.order_id = orders.order_id
join customers
on customers.customer_id = orders.customer_id;
```

Retrieve the names of customers along with the names of the dishes they ordered and the status of each order.

```
SELECT Customers.name AS Customer_Name, Menu_items.Name AS Dish_Name,
Orders.status
FROM Orders
INNER JOIN Menu_items ON Orders.item_id = Menu_items.item_id
INNER JOIN Customers ON Orders.customer_id = Customers.customer_id;
```

- ***Subqueries***

Retrieve the name of the customer 'John Doe', along with the order ID and quantity of each order placed by that customer.

```
SELECT name, customer_id or (SELECT order_id FROM Orders WHERE customer_id =
Customers.customer_id) AS order_id,
(SELECT quantity FROM Orders WHERE customer_id = Customers.customer_id) AS
quantity
FROM Customers
WHERE name = 'John Doe';
```

Retrieve the customer who made the highest number of reservations.

```
SELECT name
FROM Customers
WHERE customer_id = (SELECT customer_id FROM Reservations GROUP BY
customer_id ORDER BY COUNT(*) DESC LIMIT 1);
```

Retrieve the order IDs of the orders with the lowest and highest total prices from the Orders table.

```
SELECT order_id
FROM Orders
WHERE total_price = (SELECT MIN(total_price) FROM Orders);

SELECT order_id
FROM Orders
WHERE total_price = (SELECT MAX(total_price) FROM Orders);
```

Retrieve the number of customers who have made reservations.

```
SELECT COUNT(DISTINCT customer_id) AS Total_Customers
FROM customers;
```

Select all the data from staff management where share the same position as bob smith works.

select * from staff_management where position = (select position from staff_management where Name = "Bob Smith");

Select the names of the customers and there item/dish names which they ordered.

select (SELECT name FROM Customers WHERE customer_id = Orders.customer_id) AS Customer_Name,
(SELECT Name FROM Menu_items WHERE item_id = Orders.item_id) AS item_Name from orders;

- ***Normal Queries***

Sum of all quantity's in inventory
select sum(quantity_on_hand) from inventory;

maximum quantity on hand for each item in the inventory
select name, max(quantity_on_hand) from inventory group by name;

substring of their name starting from the second character and with a length of four characters in staff management table.
select substring(Name,2,4) from staff_management;

length of all names from customer table
select length(name) from customers;

reverse of the customer name for each entry in the customer_feedback table.
select reverse(customer_name) from customer_feedback;

names of all items in uppercase
select upper(name) from inventory;

Display the details of staff members starting from the third record and show the next five records after that
select * from staff_management limit 2,5;

display the name and their position where in there name t alphabet is present
select Name,position from staff_management where Name like "%T%";
Retrieve the count of items in each category from the Menu_items table.
SELECT category, COUNT(*) AS total_items
FROM Menu_items
GROUP BY category;

Retrieve the details of items from the 'inventory' table where the quantity on hand is either 50 or 40.
select * from inventory where quantity_on_hand in (50,40);

Retrieve the count of items in each category from the 'Menu_items' table, but only show categories with more than 3 items.

```
SELECT category, COUNT(*) AS total_items
FROM Menu_items
GROUP BY category
HAVING total_items > 3;
```

Retrieve the details of items from the inventory table where the reorder threshold is between 10 and 20.

```
select * from inventory where reorder_threshold between 10 and 20;
```

Conclusion:

In conclusion, developing a Restaurant Management System (RMS) presents an exciting opportunity to streamline restaurant operations and enhance customer experience.

By organizing menu items, orders, reservations, customer feedback, staff management, and inventory tracking into a comprehensive database, we can effectively manage various aspects of restaurant operations.

With features like menu management, order processing, reservation handling, and customer feedback collection, the Restaurant Management System not only simplifies day-to-day tasks for restaurant staff but also improves overall efficiency and customer satisfaction.

In essence, the Restaurant Management System is a vital tool for modernizing restaurant management and driving continuous improvement in the ever-evolving hospitality industry.

Thank You