WACHEMO UNIVERSITY

COLLAGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF SOFTWARE ENGINERING

Project title: RESTAURANT MANAGEMENT SYSTEM

GROUP: 6

SECTION: A

Student name	**ID**
1. Wuleta Abebe	1600384
2. Gedamo Tesema	1600179
3. Melese Kifile	1600250
4. Mihiret Seid	1602312
5. Feven Beyene	1601078

Submission date: 30/4/2017 EC

Submitted to: Mr Mekuwannt

Contents

1. Introduction

The restaurant management system database project aims to develop a comprehensive database system for managing various aspects of a restaurant business, such as customer information, orders, inventory, and employees. Restaurant Management System (RMS) Database is designed to enhance operations within a restaurant, to support multiple view of data, to support the sharing of data among multiple user, enhancing efficiency in managing orders, inventory, staff, and customer relationships and mainly to reduce data redundancy. This document outlines the conceptual design, functionality, and logical design (implementation) strategies of restaurant management system database.

1.1. Background

Agape Restaurant is located in Hossaena, Ethiopia, and is popular for fine Ethiopian cuisine and traditional Ethiopian hospitality. The restaurant currently operates its order management, reservations, and inventory on a manual basis. Such a traditional system, though it keeps intact the cultural essence of the place, often results in inefficiency, errors, and increased inability to handle advanced customer demands. Agape Restaurant is a managerial organizational structure, whereby the restaurant has a manager to oversee the activities, an administrator responsible for records and coordination, and a total of four waiters who offer quality service to all customers. While the team members are very committed and putting in great effort, the lack of an automated system hurts their productivity and that of the customer experience. This Project has been performed in identifying the requirement for a modern, integrated database system that could address such operational challenges better while ensuring high efficiency and scalability, together with increased customer satisfaction

1.2. Purpose of the System

The purpose of this system is to provide a structured approach to designing and implementing a restaurant management system database. It covers the entire process, from analyzing the existing system to designing the database, implementing the system, and providing user training. The primary purpose is to create a RMS database that the restaurant management system uses to automate and optimize restaurant operations, enabling better management of orders, tables, staff schedules, and inventory. The system will provide real-time data access for decision-making a

1.3 Scope of the system

The scope of the Agape Restaurant Management System database includes creating, setting up, and combining a few parts to help manage the main tasks of the restaurant. In simple terms, the system uses basic tools to automate processes, aiming to make things run smoothly, accurately, and keep customers happy. For this purpose, the scope of this restaurant management system database include: -

1. All the entity, their attribute and their relationship.

2. describe how the database used in the restaurant to control the data redundancy and all the limitation in the manual system.

1.4. Objectives of the System

1.4.1 General Objective

The primary objective is to develop database for a restaurant management system.

1.4.2 Specific Objectives

- Create a normalized database schema.

- Implement database that can support multiple user and thereby role.

- To create a database that can be scalable and that can be handle an increase in data.

-To create insulation between programing and data.

- Develop database that can efficiently store and retrieve data that can be related to the restaurant core operation

- To support sharing of data and multiple user transaction.

1.5. Methodology

1.5.1 Requirement Elicitation Methodology

The Following are some of the methodologies that were used in capturing the requirements for defining the Agape Restaurant Management System database:

- Interviews: Through structured interviews with the manager, administrator, and waiters about their workflow, problems faced and the functionality desired.
- Observation: Observed day-to-day operations at Agape Restaurant in relation to inefficiencies in their current manual processes of handling orders, billing, and inventories.
- Document Analysis: Analyzed available records, such as order logs, inventory sheets, and financial records, to understand the flow of information and how things are presently done.
- Focus Groups: Discussion with the staff to identify pain points and gather insights into features that could improve their efficiency and reduce workload.
- Questionnaires: Surveys were given to the regular customers to understand the feelings and expectations of customers from the better restaurant system.
- Brainstorming Sessions: Consolidated findings with the project team and refined the system requirements into feasibility and priority.

1.5.2 Implementation methodology

- Structured query language(SQL) is used for the implementation purpose.

1.5.3 designing

Conceptual design - conceptual design includes entity relation diagram (ER -DIAGRAM)

Logical design - logical design includes table.

Physical design - is an implementation or direct creation of database using sql.

2. The Existing System

2.1 Over view

The existing system is a manual system that relies on paper-based records to manage customer information, orders, inventory, and employee data. Currently agape restaurant handles its data using manual approach. this approach causes many limitations in the restaurant. this causes the need of the development of database that used by the restaurant management system.

problem of the existing system

manual process is

- prone to error.

- Difficult to update, retrieve and integrate

- there is data but difficult to compile the information.

- cross referencing is difficult.

2.2. New Opportunities

The restaurant management system database

- providing multiple interface to different class of user.
- providing backup and recovery service.
- representing complex relationship among data
- controlling redundancy in data storing and development maintenance effort.
- support sharing data among multiple user.
- ensure data consistence

2.3. Functional Requirements

- register all actor have to register

- Login all actor has to login extend logout

- View menu customer and manager view menu extend manager edit menu

- Pace order customer place order

- View order customer and waiter view order extend customer edit order

- Receive notification all customer receive notification when their order is not available

- Clear bill

- manager clear bill

- Generate notification- waiter and admin generate notification

- View customer recorded- manager and admin view customer record

- Selling report :-admin produce selling report

- Log out

2.4. Non-Functional Requirements

- Performance
    - The system must handle multiple concurrent users without performance degradation.
    - Fast Response time
- Reliability
    - The system must remain operational and available for the majority of the time during business hours, with minimal downtime.
    - Data integrity should be preserved during system crashes or power failures.

- Scalability
    - The system should support future enhancements, such as adding new features or modules.
    - It must handle an increase in the number of users or transactions without significant redesign.
- Usability
    - The user interface must be intuitive and easy to navigate for all users, including non technical staff.
    - Provide multilingual support, if necessary, to cater to diverse user needs.
- Security
    - Ensure secure login with role-based access control for admin and staff.
    - Data encryption should be used for sensitive information, such as billing details.
- Maintainability
    - The system must be easy to update, with clear documentation for developers.
    - Allow admins to configure settings (e.g., menu updates or staff changes) without technical expertise.
- Compatibility
    - The system must work flawlessly on Windows-based machines and common web browsers.
    - Ensure compatibility with both local and cloud-based hosting environments.
- Accessibility
    - Ensure the system is accessible to users with basic technology skills.

3. The Proposed System

3.1 Overview

The restaurant management system (RMS) will be a web-based application accessible via desktops and tablets. It will consist of a user-friendly front-end interface for staff and a robust back-end database for data management. The system will be built using a three-tier architecture: presentation layer (front-end), application layer (business logic), and data layer (database).

3.2. Features of the Proposed System

Our Restaurant Management System Database includes various features that enhancing operations, customer experience, and improve overall efficiency.

Here are some key features of Our Restaurant Management System Database:

1. Customer Management

- Customer Profiles: Store customer information such as name, contact details, preferences, and order history.

2. Order Management

- Order Placement: Facilitate easy order entry by staff.
- Order Tracking: Real-time tracking of order status from placement to delivery.
- Order History: Access to past orders for easy reordering.

3. Menu Management

- Dynamic Menu: Ability to add, edit, or remove menu items, including descriptions and prices.

4. Inventory Management

- Stock Tracking: Monitor inventory levels of ingredients and supplies in real-time.
- Reorder Alerts: Automatic notifications for low stock items to prevent shortages.

- Supplier Management: Maintain records of suppliers and their contact information.

5. Staff Management

- Employee Profiles: Store information about staff members, including roles, schedules, and contact details.
- Scheduling: Create and manage staff schedules, including shift assignments.
- Payroll Management: Track hours worked and calculate payroll.

6. Table Management

- Reservation System: Allow customers to make reservations online or through the app.
- Table Status Tracking: Monitor which tables are occupied, reserved, or available in real-time.

7. Payment Processing

- Multiple Payment Options: Support various payment methods (credit/debit cards, mobile payments, cash).
- Transaction History: Record all financial transactions for auditing purposes.

8. Reporting and Analytics

- Sales Reports: Generate reports on daily, weekly, or monthly sales performance.
- Inventory Reports: Analyze inventory usage and wastage.
- Customer Insights: Gain insights into customer preferences and behaviors.

9. Security Features

- User Authentication: Secure login for staff with role-based access control.
- Data Encryption: Protect sensitive customer and payment information.

10. Integration Capabilities

- Third-Party Integrations: Connect with other systems like accounting software, delivery services, or marketing tools.

3.3. Feasibility Analysis (including Budget and Schedule)

3.3.1 Technical *feasibility*

The technical feasibility of the Agape Restaurant Management System involves assessing the resources and technology needed to develop and operate it efficiently.

Hardware Requirements

- Server: Quad-core processor, 16 GB RAM, 500 GB SSD
- Staff Devices: Desktop/laptop with 4 GB RAM; tablets or mobile devices for POS

Software Requirements

- DBMS: MySQL for data management
- Development Tools: PHP, Apache/Nginx, Visual Studio Code
- Operating Systems: Linux (Ubuntu/CentOS) for servers, Windows/macOS/iOS/Android for clients

Network Needs

- Secure local network; internet for online orders or remote management

Technical Skills Needed

- Database Design: Managing tables for menus, orders, customer data
- Web Development: PHP, HTML, CSS, JavaScript
- System Management: Server setup, data security, reliability

Development and Testing Setup

- Development: Local machines, Git for version control

- Testing: Dedicated server, performance/scalability tools

3.3.2. Budget (Cost) Feasibility

- Labor Costs:
  - Database Design/Analyst: 15000 birr
  - Database Developer: 1200birr
  - QA Tester: 10000birr
  - Project Management 18000
  - Total Estimated Labor Costs: 55000 birr
  - Software and Tool Costs:
  - RDBMS (Open Source): 0 birr (freely available)
  - Database Modeling Tools: 0 birr (using freely available tools or community licenses)
  - Total Estimated Software Costs: 0 birr
- **Infrastructure Costs:**
  - Cloud server 9000 birr
  - Total Estimated Infrastructure Cost: 9000 birr
  - Contingency: 40000 birr
  - Total cost (budget)=104000 birr
- **Cost Analysis:**
  - The estimated cost falls within a reasonable range for a project of this scope.
  - By using an open source DBMS system we are able to substantially reduce costs.

- There may be further opportunities to reduce infrastructure costs by using free cloud-based development environments.

5. Schedule Feasibility

- Phase 1: Planning and Requirement Gathering: 2 weeks.
- Phase 2: Database Design and Modeling: 4 weeks.
- Phase 3: Implementation and Setup: 4 weeks.
- Phase 4: Testing: 2 weeks.
- Phase 5: Documentation: 2 weeks (overlapping with other phases).
- Phase 6: Deployment and Data Migration: 2 weeks.
- Total Estimated Schedule: 16 weeks (4 month)
- Schedule Analysis:
- The project can reasonably be completed within 4 months.

4. Database Design

4.1 Conceptual Database Design of the New system

4.1.1 ***Entities identification and Description***

1. ADMIN: Manages the system, oversees operations, and handles user roles and system settings.
2. RESTAURANT: Represents the restaurant where orders are managed and operations are conducted.
3. MANAGER: Oversees restaurant operations, employee management, and customer satisfaction.
4. CUSTOMER: Represents customers who place orders at the restaurant.
5. WAITER: Responsible for taking orders, serving food, and providing customer service.
6. ORDER: Represents an order placed by a customer, containing general order information.
7. ORDER_DETAIL: Contains detailed information about items in an order.

8. BILL: Represents the bill issued for an order, detailing payment information.
9. MENU: Contains information about food items available for ordering.
10. CHEF: Represents chefs responsible for preparing meals.
11. INVENTORY: Tracks stock items and manages supply levels for the restaurant.

4.1.2 Entity Attributes identification and description

1. ADMIN

- Attributes:
    - Name
    - Admin_ID(primary key)
    - Address

2. RESTAURANT

- Attributes:

    - Restaurant_ID(primary key)
    - Name
    - Location

3. MANAGER

- Attributes:
    - Manager_ID (primary key)
    - Name
    - Phone

4. CUSTOMER

- **Attributes**:
    - Customer_ID(primary key)
    - Name
    - Contact_information

5. WAITER

- **Attributes**:
    - Waiter_ID (primary key)
    - Name
    - Salary

6. ORDER

- **Attributes**:
    - Order_ID(primary key)
    - Customer_ID (**FK**)
    - Total_Amount
    - Order_Date

7. ORDER_DETAIL

- **Attributes**:
    - Order_Detail_ID)
    - Menu_ID (**FK**)
    - Order ID (FK)
    - Quantity
    - Sub_Total

8. BILL

- **Attributes**:
    - Bill_ID(primary key)
    - Order_ID (**FK**)
    - Payment_Date
    - Amount_Paid
    - Payment_Method

9. MENU

- Attributes:
    - **PK**: Menu_ID
    - Name
    - Price
    - Available

10. CHEF

- **Attributes**:
    - Chef_ID(primary key)
    - Name
    - Phone
    - Salary

11. INVENTORY

- **Attributes**:

- Inventory_ID(primary key)
- Name
- Quantity_In_Stock
- Reorder_Level

4.1.3 Relationship Identification and Description

1. Restaurant to Manager

Manager MANAGES Restaurant (1:1)

2. Manager to Menu

Manager UPDATE (add or delete) multiple menu item(1:M)

3.. Waiter to Chef

Waiter FORWARD ORDER TO Chef(M:N)

4. Customer to Order

Customer PLACE Order(1:M)

5.Waiter to Customer

Waiter SERVE ORDER to customer(M:N)

6. Order to Waiter

Order TAKEN BY Waiter(M:N)

7. Order to Order detail

Order DETRMINED BY many detail(1:M)

8.Customer to Bill

Customer PAYS Bill (1:1)

9. Manager to Bill

Manager CLEAR Payment(1:M)

10. Admin to Restaurant

Admin MONITOR Restaurant(1:M)

11. Admin to Customer

Admin TAKE FEEDBAK from Customer

12. Customer to Menu

Customer VIWE Menu(M:1)

13. Inventory to Menu

each Inventory item provides menu

4.1.4 ER Diagram

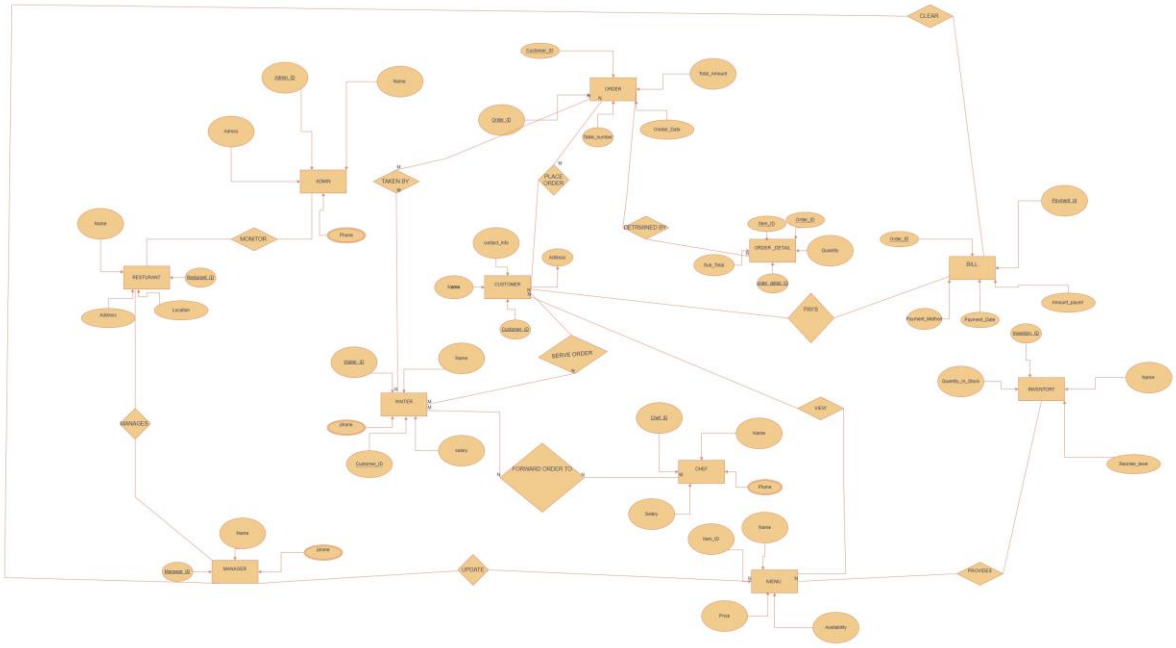

Figure 1 ER Diagram

4.2 Logical Database Design of the New System

4.2.1 ER to table mapping

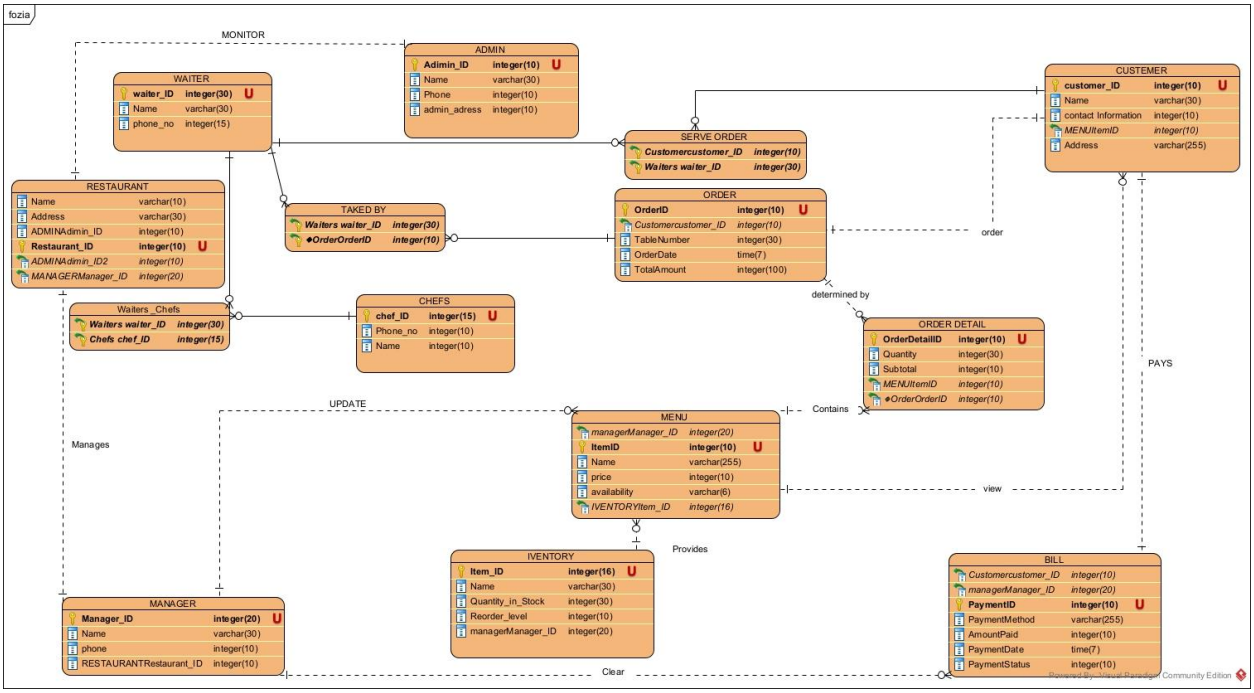

Figure 2 ER to table mapping

4.2.2 Validate Model using Normalization

Customer table

Customer_Id	Customer_Name	Phone Number	Address
1001	Endrias Dejene	0932144435, 0916553986	123 Main Street, Hossaina, Central Ethiopia
1002	Tesma Kebede	0913222188	456 High Street, Wolayita, South Ethiopia
1003	Johann Degsewu	0972117111, 0945162615	782 Avenue, Hossaina, Central Ethiopia
1004	Kaleb Aschalewu	0711626361	926 Pin Ln, Adama, Oromia

4.2.2.1 First Normal Form (1NF)

To normalize the above table into the first normal form:

1. **Eliminate Multivalued Attribute (Phone Number)**: We will create a new table for the phone numbers, as each customer can have multiple phone numbers.

Customer phone table

Customer_ID	Phone
1001	0932144435
1001	0916553986
1002	091322188
1003	0972117111
1003	0945162615
1004	0711626361

2. **Eliminate Composite Attribute (Address)**: We will split the "Address" column into separate columns for "Street", "City", and "Region".Customer Table

Customer _ID	Customer name	Street	City	Region
1001	Endrias Dejene	123 Main Street	Hossaina	Central Ethiopia
1002	Tesma Kebede	456 High Street	Wolayita Sodo	South Ethiopia
1003	Johann Degsewu	782 Avenue	Hossaina	Central Ethiopia
1004	Kaleb Aschalewu	926 Pin Ln	Adama	Oromia

The customer table normalized in to first normal form.

4.2.2.2 Second Normal Form (2NF)

The tables are already in **First Normal Form (1NF)**, and they are now in **Second Normal Form (2NF)** because:

- The "Customer Table" has a single attribute primary key (`Customer_ID`), and all non-key attributes depend fully on it.
- The "Customer Phone Table" is also in **1NF** and meets the requirements of **2NF** since the composite key (Customer_ID, Phone) is used, and all non-key attributes depend on the full key.

4.2.2.3 Third Normal Form (3NF)

To normalize the tables into **Third Normal Form (3NF)**, we must resolve transitive dependencies.

- In the **Customer Table**, the "Region" column is transitively dependent on `Customer_ID` through the non-key attribute "City". To resolve this, we will create a new table for "City" and "Region" and link it to the "Customer" table.

Customer table

Customer_ID	Customer_Name	Street	city(FK)
1001	Endrias Dejene	123MainStreet	Hossaina
1002	Tesma Kebede	456High street	Wolayita sodo
1003	Johann Degsewu	782 Avenu	Hossaina
1004	Kaleb Aschalewu	926 pin ln	Addis Abeba

City table

City(PK)	Region
Hossaina	Central Ethiopia
Wolayita sodo	South Ethiopia
Addis Abeba	Oromia

Customer phone table

Customer_ID	Customer Phone
1001	0932144435
1001	0916553986
1002	0913222188
1003	0972117111

1003	0945162615
1004	0711626361

The table normalized in to third normal form(3NF)

4.2.3. Relational schema with referential integrity after normalization

```
-- Customers Table

CREATE TABLE Customers (

   CustomerID VARCHAR(15) PRIMARY KEY,

   CustomerName VARCHAR(100),

   Street VARCHAR(100),

   City VARCHAR(50),

   FOREIGN KEY (City) REFERENCES Cities(City)

);


-- Cities Table

CREATE TABLE Cities (

   City VARCHAR(50) PRIMARY KEY,

   Region VARCHAR(50)

);


-- CustomerPhones Table
```

```
CREATE TABLE CustomerPhones (
    CustomerID VARCHAR(15
    Phone VARCHAR(30
    PRIMARY KEY (CustomerID, Phone),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);
```

4.3 Physical database design of the new system

4.3.1. Physical Design Strategy

- Goal: To design a database architecture that is performant, scalable, secure, and reliable.
- Key Considerations:

❖ Performance

- Indexing strategy to optimize queries
- Data types and lengths selection to reduce space usage and enhance retrieval efficiency.
- Appropriate database server selection.
- Connection pooling to manage connection requests to the database.

❖ Scalability:

- Database partitioning for large tables.
- Choose a database server to meet current and expected traffic.
- Consider cloud-based database solutions for easier scalability.

❖ Security:

- Role-based access controls to limit access to sensitive information.

- Encryption for sensitive data in transit and at rest.
- Regular data backups for data safety.

❖ Reliability:

- Proper server and hardware implementation to allow for optimal performance.
- Data backups and recovery procedures.
- Transaction logs to enable data restoration.

• Specific Strategies:

Indexing: Create indexes on columns frequently used in queries (e.g., OrderID, CustomerID, MenuItemID, OrderDate).

Data Types: Choose appropriate data types to minimize storage.

Partitioning: Consider partitioning large tables like Orders or Customers if they become too large.

Storage: Optimize storage allocation based on data access patterns, using SSDs for frequently accessed data.

Backups: Establish a backup strategy including full backups and transaction logs.

4.3.2. Hardware Implementation

• Database Server:

CPU: Multi-core processor (e.g., Intel Xeon or AMD EPYC) for good performance.

RAM: Sufficient RAM (e.g., 32 GB to 128+ GB, depending on the expected database size and workload) to keep frequently used data in memory.

Storage: Fast storage subsystem (e.g., SSDs) for storing data and indexes; RAID configuration for data protection.

Network: High-speed network connection to serve clients

Operating System: A stable server OS (e.g., Linux, Windows Server)

- Client Machines (Front End)

CPU: hp Core i5

RAM: 16GB

Storage: SSD is preferable to speed up local operations.

Network: Stable network connection.

- Other Hardware

Backup Storage: Off-site backup storage for data integrity.

Network Router: To provide an interface to clients and servers.

4.3.3. Front-End Design Outline

For context since this is a database project

The front-end design is an important part of an RMS, as this is what users will interface with.

- User Interface (UI) Principles:

Intuitive and Easy to Use: Simple and clear navigation, with clear call-to-action elements.

Responsive Design: Compatible with multiple screen sizes (desktop, tablets, smartphones).

Accessibility: Adherence to accessibility standards for all users.

- Functional Modules:

Point of Sale (POS): For order taking, payments, and table management.

Menu Management: For updating menu items and categories.

Inventory Management: For tracking stock levels, reporting, and reordering.

Customer Management: For customer data maintenance and loyalty programs

5.Implementation and support

5.1 Codding and testing (practical work)

```
CREATE DATABASE RestaurantAgape;

USE RestaurantAgape;

CREATE TABLE Admin (

    Admin_ID INT PRIMARY KEY,

    Name VARCHAR(30) NOT NULL,

    Admin_Address INT,

    Phone_No BIGINT

);

CREATE TABLE Manager (

    Manager_ID INT PRIMARY KEY,

    Name VARCHAR(30) NOT NULL,

    Phone BIGINT

);

CREATE TABLE Inventory (

    Item_ID INT PRIMARY KEY,

    Name VARCHAR(30) NOT NULL,

    Quantity_In_Stock INT,

    Reorder_Level INT,

    Manager_ID INT,
```

```
    FOREIGN KEY (Manager_ID) REFERENCES Manager(Manager_ID)
);
CREATE TABLE Menu (
    Item_ID INT PRIMARY KEY,
    Name VARCHAR(30) NOT NULL,
    Price INT,
    Availability VARCHAR(16),
    Manager_ID INT,
    Inventory_Item_ID INT,
    FOREIGN KEY (Manager_ID) REFERENCES Manager(Manager_ID),
    FOREIGN KEY (Inventory_Item_ID) REFERENCES Inventory(Item_ID)
);
CREATE TABLE Customer (
    Customer_ID INT PRIMARY KEY,
    Name VARCHAR(30) NOT NULL,
    Contact_Information BIGINT,
    Menu_Item_ID INT,
    Admin_ID INT,
    FOREIGN KEY (Menu_Item_ID) REFERENCES Menu(Item_ID),
    FOREIGN KEY (Admin_ID) REFERENCES Admin(Admin_ID)
);
```

```
CREATE TABLE Restaurant (
    Restaurant_ID INT PRIMARY KEY,
    Name VARCHAR(30) NOT NULL,
    Address VARCHAR(30),
    Admin_ID INT,
    FOREIGN KEY (Admin_ID) REFERENCES Admin(Admin_ID)
);
CREATE TABLE Waiter (
    Waiter_ID INT PRIMARY KEY,
    Name VARCHAR(30) NOT NULL,
    Phone_No BIGINT
);
CREATE TABLE Chef (
    Chef_ID INT PRIMARY KEY,
    Name VARCHAR(30) NOT NULL,
    Phone_No BIGINT
);
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    Customer_ID INT,
    Table_Number INT,
```

```
    Order_Date TIME,
    Total_Amount INT,
    FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID)
);
CREATE TABLE Order_Detail (
    OrderDetailID INT PRIMARY KEY,
    Order_ID INT,
    Menu_Item_ID INT,
    Quantity INT,
  Subtotal INT,
    FOREIGN KEY (Order_ID) REFERENCES Orders(OrderID),
    FOREIGN KEY (Menu_Item_ID) REFERENCES Menu(Item_ID)
);
CREATE TABLE Bill (
    PaymentID INT PRIMARY KEY,
    Customer_ID INT,
    Manager_ID INT,
    Payment_Method VARCHAR(255),
    Amount_Paid INT,
    Payment_Date TIME,
    Payment_Status VARCHAR(10),
```

```
    FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
    FOREIGN KEY (Manager_ID) REFERENCES Manager(Manager_ID)
);
CREATE TABLE Taken_By (
    Waiter_ID INT,
    Order_ID INT,
    PRIMARY KEY (Waiter_ID, Order_ID),
    FOREIGN KEY (Waiter_ID) REFERENCES Waiter(Waiter_ID),
    FOREIGN KEY (Order_ID) REFERENCES Orders(OrderID)
);
CREATE TABLE Waiters_Chefs (
    Waiter_ID INT,
    Chef_ID INT,
    PRIMARY KEY (Waiter_ID, Chef_ID),
    FOREIGN KEY (Waiter_ID) REFERENCES Waiter(Waiter_ID),
    FOREIGN KEY (Chef_ID) REFERENCES Chef(Chef_ID)
);
```

5.2 User manual

This section provides a guide for end-users on how to interact with the Agape Restaurant Management System (RMS) Database. It covers the main functions of the database, how to access and manage data, and how to interact with various features from a user perspective.

- Logging into the System
    - Launch the Agape RMS application on your computer.
    - Enter your username and password provided by the system administrator.
    - Click "Login" to access the system's main dashboard.
- Navigating the Interface
    - Menu Bar: Located at the top, it allows access to various sections such as Orders, Menu Management, Reports, and User Settings.
    - Search Bar: Positioned on the left-hand side for quickly finding menu items, orders, or other data.
    - Dashboard: Displays key metrics such as daily sales, customer visits, and order statuses.
- Managing Menu Items
    - Add a New Item:
        - Go to the "Menu Management" section.
        - Click the 'Add New Item' button.
        - Fill in the item name, description, price, and category.
        - Click 'Save' to add the item to the menu.
    - Edit an Existing Item:
        - Use the search bar to find the item.
        - Select the item and click 'Edit'.
        - Update the information and click 'Save'.
- Creating and Managing Orders
    - Create a New Order:
        - From the Dashboard, click on 'New Order'.
        - Select items from the menu and add them to the order.
        - Specify customer details

        - Click 'Submit Order' to confirm the order.
    - View and Manage Existing Orders:
        - Go to the 'Orders' section from the Menu Bar.
        - Search for the order using the order ID or customer name.
        - Click on the order to view details and update its status (e.g., 'In Progress', 'Completed').
- Generating Reports
    - Access the Reports Section: Click on the 'Reports' tab in the Menu Bar.
    - Select Report Type: Choose from available report types such as Sales, Inventory, or Employee Performance.
    - Set Report Parameters: Customize the report by specifying date ranges, categories, or other filters.
    - Generate and Export: Click 'Generate Report' to view the data, and use the 'Export' button to download the report in formats like PDF or Excel.
- Managing User Accounts
    - Access User Settings: Click on 'User Settings' from the Menu Bar.
    - Add a New User:
        - Click on the 'Add New User' button.
        - Fill in user details like name, role, and access privileges.
        - Click 'Save' to create the user account.
    - Edit or Remove Users:
        - Search for the user using the search bar.
        - Click 'Edit' to modify their details or 'Delete' to remove the user.
- Backing Up Data
    - Access Backup Settings: Click on 'Settings' and then 'Backup'.
    - Create Backup: Click 'Create Backup' to back up the database.
    - Restore Backup: Use 'Restore Backup' to revert to a previous database state.
- Logging Out
    - Click on your profile icon in the top right corner.
    - Select 'Logout' from the dropdown menu to end your session.

5.3. User Training

This section outlines the training requirements for users of the Agape Restaurant Management System (RMS). Proper training ensures that users are equipped to efficiently operate the system, manage data, and leverage its full potential.

- **Training Overview**
    - User training is designed to familiarize staff with the features and functionality of the Agape RMS. Training ensures that users can navigate the system, perform tasks efficiently, and troubleshoot basic issues.
    - Training sessions will cover various modules, including Menu Management, Order Management, Reports, User Settings, and Backup Procedures.
- **Target Audience**
    - The training is intended for restaurant managers, servers, kitchen staff, and administrators who will interact with the RMS regularly.
    - Specific training sessions may be tailored to the different roles within the restaurant to ensure relevance and effectiveness.
- **Training Methods**
    - **In-person Workshops**: Hands-on training sessions conducted by an experienced trainer. These workshops allow users to work directly with the system in a controlled environment.
    - **Online Tutorials**: A series of video tutorials that demonstrate key tasks and features of the RMS. These tutorials can be accessed at any time for reference.
    - **User Manuals and Documentation**: Comprehensive guides that provide step-by-step instructions for using the system. Users can consult these manuals as a reference when they need help with specific tasks.

- **Interactive Simulations**: Virtual simulations that mimic real-life usage of the RMS, allowing users to practice performing tasks without affecting the actual database.

- **Training Content**

  - **System Introduction**: Overview of the RMS system, including its purpose, modules, and user interface.

  - **Menu Management**: Training on how to add, edit, and manage menu items, including categories, pricing, and descriptions.

  - **Order Management**: Instruction on how to create, manage, and process customer orders. Includes workflows for order tracking, updates, and completion.

  - **Generating Reports**: Training on creating custom reports, filtering data, and exporting report formats.

  - **User Management**: Understanding how to create, edit, and manage user roles, permissions, and access control.

  - **Backup and Recovery**: Training on creating backups of the database, restoring backups, and ensuring data safety.

- **Training Schedule**

  - A detailed training schedule will be provided to all users. The schedule will be designed to accommodate various shifts and roles in the restaurant, ensuring maximum participation.

  - Follow-up sessions may be scheduled to address specific questions or areas of difficulty.

- **Support and Ongoing Learning**

  - **Help Desk Support**: A dedicated help desk will be available to assist users with technical or operational questions after training.

- **Continuous Learning**: Access to additional resources such as advanced training, FAQ sections, and system updates will be provided to help users stay informed and proficient in using the RMS.

- **Training Evaluation**
  - At the end of each training session, users will be asked to complete a brief evaluation to assess their understanding of the system and the effectiveness of the training.
  - Feedback will be reviewed to continuously improve future training sessions.

Collaboration among Group members

Database Responsibilities and Task Allocation

The database-related tasks for the Agape Restaurant Management System will be handled by various team members, each taking ownership of specific stages of the database design, implementation, and deployment process. Below is a breakdown of who will handle each task:

1. Entity-Relationship (ER) Diagram Creation

- Responsible Person(s): Wuleta Abebe (System Analyst & Assistant Developer), Gedamo Tesema (Project Manager & System Analyst), Melese Kifile (UI/UX Designer)
    - Task Description:
        - Wuleta Abebe and Gedamo Tesema will collaboratively create the ER diagram to represent the relationships between key entities in the system (such as Orders, Customers, Menu Items, Staff, etc.). This diagram will serve as the foundation for the database structure and is crucial for visualizing the flow of data within the system.
        - Melese Kifile will provide input on how the database structure might affect the user interface design, ensuring that the ER diagram takes into account the usability of the database from a user experience perspective.

2. Database Mapping

- Responsible Person(s): Feven Beyene (Software Developer), Wuleta Abebe (System Analyst & Assistant Developer)
    - Task Description:
        - Feven Beyene will take the ER diagram and translate it into actual database schema, including table creation, field types, primary/foreign key relationships, and constraints. This will be done in collaboration with

Wuleta Abebe to ensure that the database structure supports the system's requirements.

- Wuleta Abebe will assist with ensuring that the database design maps correctly to the system’s functional needs and requirements.

3. Database Documentation

- Responsible Person(s): Wuleta Abebe (System Analyst & Assistant Developer), Gedamo Tesema (Project Manager & System Analyst)
    - Task Description:
        - Wuleta Abebe will document the entire database structure, including table definitions, relationships, indexing strategies, and constraints. This documentation will be used by the development team for further system implementation and maintenance.
        - Gedamo Tesema will assist in ensuring that the documentation aligns with the overall system specifications and is clear for all stakeholders involved in the project.

4. Database Coding and Implementation

- Responsible Person(s): Feven Beyene (Software Developer), Wuleta Abebe (System Analyst & Assistant Developer)
    - Task Description:
        - Feven Beyene will implement the database using SQL (or a similar database management language). This includes the creation of tables, relationships, triggers, stored procedures, and functions necessary for managing the data.
        - Wuleta Abebe will collaborate to ensure the correct coding of queries, and stored procedures, ensuring performance and security standards are met.

        - Feven Beyene will also ensure proper data validation, integrity constraints, and efficient data handling for the application.

5. Database Testing and Quality Assurance

- Responsible Person(s): Mihret Seid (Quality Assurance and Deployment Specialist), Wuleta Abebe (System Analyst & Assistant Developer)
    - Task Description:
        - Mihret Seid will be responsible for testing the database and ensuring its functionality, performance, and security. This includes testing the queries, stored procedures, and overall database integrity under different scenarios.
        - Wuleta Abebe will work alongside Mihret Seid to define and document test cases, ensuring that all database functions are tested thoroughly and meet the system's requirements.

6. Database Deployment

- Responsible Person(s): Mihret Seid (Quality Assurance and Deployment Specialist)
    - Task Description:
        - Mihret Seid will lead the database deployment process, ensuring the database is properly migrated and configured in the production environment. This will include setting up database connections, importing data, and optimizing performance.
        - Mihret Seid will also ensure the database is ready for integration with the rest of the system and monitor any issues during deployment.

7. Database Training

- Responsible Person(s): Mihret Seid (Quality Assurance and Deployment Specialist), Feven Beyene (Software Developer), Melese Kifile (UI/UX Designer)
    - Task Description:

- Mihret Seid will conduct training for restaurant staff on the use of the database, focusing on the practical use of the system to manage orders, inventory, and billing.
- Feven Beyene will assist in training staff on how to use system features that interact with the database, ensuring that they understand the flow of data and how to input/update information within the system.
- Melese Kifile will assist with training related to the user interface, ensuring staff understand how to interact with the system through the UI and how the database connects to the interface.

Task Overview Summary

Task	Responsible Person(s)
ER Diagram Creation	Wuleta Abebe, Gedamo Tesema, Melese Kifile
Database Mapping	Feven Beyene, Wuleta Abebe
Database Documentation	Wuleta Abebe, Gedamo Tesema
Database Coding/Implementation	Feven Beyene, Wuleta Abebe
Database Testing	Mihret Seid, Wuleta Abebe
Database Deployment	Mihret Seid
Database Training	Mihret Seid, Feven Beyene, Melese Kifile

Collaboration Workflow

The team follows a structured workflow to ensure that all database-related tasks are completed efficiently:

1. Initial Planning and ER Diagram: Gedamo Tesema, Wuleta Abebe, and Melese Kifile work together to define the database structure, incorporating feedback on how the data will interact with the user interface.

2. Mapping and Coding: Feven Beyene takes the ER diagram and works on the database implementation, while Wuleta Abebe assists in ensuring all requirements are met.

3. Testing and Quality Assurance: Mihret Seid ensures the system works flawlessly by testing the database and all its functions, supported by Wuleta Abebe.

4. Deployment and Training: Mihret Seid ensures the system is ready for use, while Feven Beyene and Melese Kifile assist with the training on the practical use of the system.

.

..