

**FACULTY OF FOCIM**

**COURSE: DIPLOMA IN INFORMATION TECHNOLOGY**

**SOFTWARE DESIGN SPECIFICATION**

**REG\_NUMBER:**

**NAME:**

**SUPERVISOR:**

**DATE SUBMITTED: dd/mm/yyyy**

Table of Contents

[1 Introduction 3](#_Toc161864968)

[2 Business Rules 3](#_Toc161864969)

[3 Process Modeling 3](#_Toc161864970)

[3.1.1 Symbols 3](#_Toc161864971)

[3.1.2 Processes 3](#_Toc161864972)

[3.1.3 Context Diagram 4](#_Toc161864973)

[3.1.4 Data Flow Diagrams (DFDs) 4](#_Toc161864974)

[4 Data Modeling 5](#_Toc161864975)

[4.1.1 Entities and Attributes 5](#_Toc161864976)

[4.1.2 Normalization 6](#_Toc161864977)

[4.1.3 Data Dictionary 6](#_Toc161864978)

[4.1.4 Entity Relationship Diagram (ERD) 8](#_Toc161864979)

[5 Pseudocode 8](#_Toc161864980)

[5.1 Food Module 8](#_Toc161864981)

[5.2 Order Module 9](#_Toc161864982)

[5.3 Reports Module 9](#_Toc161864983)

[5.4 System flowchart 11](#_Toc161864984)

# Introduction

This document outlines the software design specifications (SDS) for the Restaurant Management System (RMS). The RMS is a standalone application designed to streamline daily operations in a small-scale restaurant. It caters to a single user who fulfills both cashier and manager roles. The system offers functionalities for:

* Point-of-Sale (POS): Taking orders, managing payments, and generating bills.
* Food Management: Adding, editing, and disabling menu items.
* Order Cancellation: Canceling orders, partially or completely, before payment.
* Bill Printing: Generating and printing bills for completed orders.
* Reporting: Generating reports on sales, revenue, and popular items.

This SDS serves as a blueprint for the system's development, ensuring it meets the functional and technical requirements outlined in the Software Requirements Specification (SRS) document.

# Business Rules

The RMS operates on a single user basis within the following business rules:

* + Can add, edit, and delete menu items.
  + Manage orders, including adding items, modifying quantities, and calculating totals.
  + Accept cash payments for completed orders.
  + Cancel orders before payment.
  + Print bills upon successful order completion.
  + Generate reports on various aspects:
    - Sales by transaction date (day, week, month, year)
    - Sales by food category

# Process Modeling

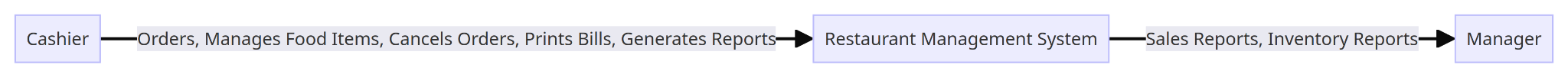
### Symbols

* **Process:** Rectangle
* **External Entity:** Square
* **Data Flow:** Arrow
* **Data Store:** Open-ended rectangle

### Processes

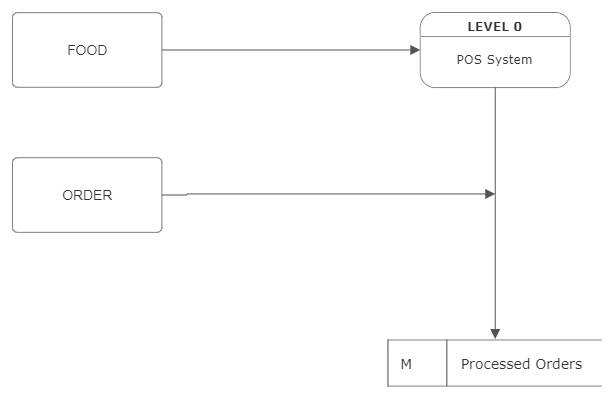
* Take Order
* Manage Food Items
* Process Payment (future integration)
* Cancel Order
* Print Bill
* Generate Report

### Context Diagram



### Data Flow Diagrams (DFDs)

#### Level 0 DFD



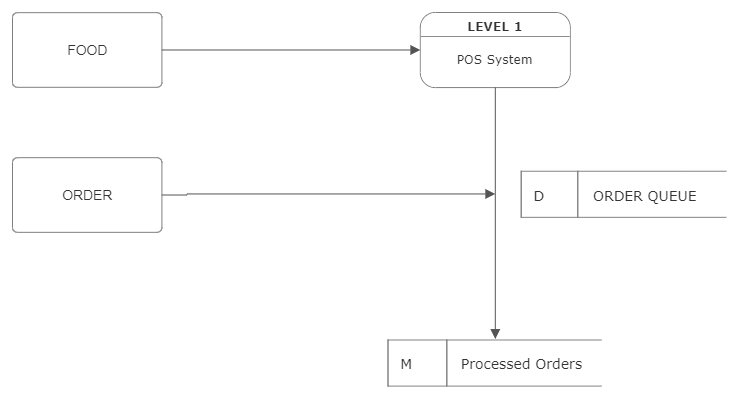
**Process**: POS System.

**Entities**: Food, Order.

**Data** **Flows**:

* Food data flow from the Food entity to the POS System.
* Order data flow from the Order entity to the POS System.
* Processed orders flow from the POS System to the Order entity.

#### Level 1 DFD



**\*Data Stores:**

* Order Queue (where orders are temporarily stored before processing).

**\*Data Flows:**

* Food data flow from the Food entity to the POS System.
* Order data flow from the Order entity to the POS System.
* Processed orders flow from the POS System to the Order entity.
* Order details flow from the POS System to the Order Queue.

# Data Modeling

### Entities and Attributes

**tbl\_food:**

* **Attributes:**
  + ID (INT): Unique identifier for each food item (Primary Key with Auto Increment)
  + foodcode (VARCHAR(100)): Unique code for each food item
  + foodname (VARCHAR(100)): Name of the food item
  + price (DECIMAL(10,2)): Price of the food item
  + img (BLOB): Image of the food item (optional)

**tbl\_pos:**

* **Attributes:**
  + ID (INT): Unique identifier for each transaction (Primary Key with Auto Increment)
  + transno (VARCHAR(100)): Transaction number
  + transdate (DATE): Date of the transaction
  + transmonth (VARCHAR(100)): Month of the transaction (redundant with transdate)
  + foodcode (VARCHAR(100)): Foreign key referencing foodcode in tbl\_food
  + foodname (VARCHAR(100)): Name of the food item (redundant with foodcode, likely populated for display purposes)
  + price (DECIMAL(10,2)): Price of the food item (redundant with price in tbl\_food, likely populated for calculation purposes)
  + qty (VARCHAR(100)): Quantity of the food item purchased
  + totalprice (DECIMAL(10,2)): Total price of the individual food item (price \* qty)
  + grandtotal (DECIMAL(10,2)): Total price of the entire transaction (sum of all total prices)
  + nooffoods (VARCHAR(100)): Total number of food items in the transaction

### Normalization

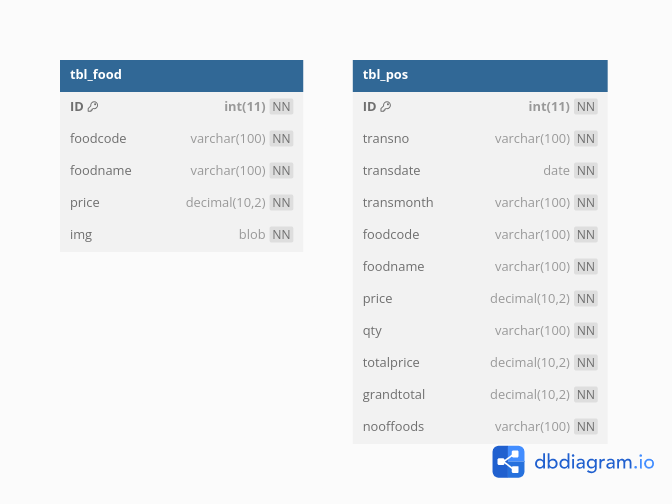
The data model is already in 3NF as it meets the following requirements:

* **1NF:** Each table has unique attribute names, and each attribute value is atomic (cannot be further decomposed).
* **2NF:** There are no partial dependencies on any non-candidate key attributes. In other words, any non-key attribute is fully dependent on the entire primary key, not just a part of it.
* **3NF:** There are no transitive dependencies. This means that a non-key attribute is not dependent on another non-key attribute; it solely relies on the primary key.

### Data Dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity | Attribute | Data Type | Description | Primary Key | Foreign Key |
| Order | order\_id | INT | Unique identifier for each order | Yes |  |
| Order | date | DATE | Date of the order |  |  |
| Order | time | TIME | Time of the order |  |  |
| Order | cashier\_id | INT | ID of the cashier who took the order |  | Foreign Key references cashier table (cashier\_id) |
| Order | customer\_name | VARCHAR(255) | Customer name (optional) |  |  |
| Order | total\_amount | DECIMAL(10,2) | Total amount of the order |  |  |
| Order | payment\_method | VARCHAR(255) | Payment method used (optional) |  |  |
| Food Item | food\_id | INT | Unique identifier for each food item | Yes |  |
| Food Item | food\_code | VARCHAR(100) | Unique code for each food item | Yes |  |
| Food Item | name | VARCHAR(255) | Name of the food item |  |  |
| Food Item | price | DECIMAL(10,2) | Price of the food item |  |  |
| Food Item | description | TEXT | Description of the food item (optional) |  |  |
| Food Item | image | BLOB | Image of the food item (optional) |  |  |
| Food Item | category | VARCHAR(255) | Category of the food item (optional) |  |  |

### Entity Relationship Diagram (ERD)



# Pseudocode

## Food Module

**Add Food Item**

* **Input:** food code, food name, price (optional: image)
* **Process:**
  + Validate food code uniqueness.
  + Create a new record in tbl\_food with provided information.

**Edit Food Item**

* **Input:** food ID, new food information (optional)
* **Process:**
  + Find the food item in tbl\_food by ID.
  + Update the corresponding record with the provided information.

**Delete Food Item**

* **Input:** food ID
* **Process:**
  + Find the food item in tbl\_food by ID.
  + Delete the record from tbl\_food.

**View Food List**

* **Process:**
  + Retrieve all records from tbl\_food.
  + Display the list of food items with their details (code, name, price).

## Order Module

**Create Order**

* **Input:** List of food items (code, quantity)
* **Process:**
  + For each food item:
    - Find the corresponding item in tbl\_food using the code.
    - Validate item availability (sufficient quantity in stock).
  + Calculate the total price of each item (price \* quantity).
  + Calculate the grand total of the order (sum of individual item prices).
  + Create a new record in tbl\_pos with transaction details (food code, name, price, quantity, total price, grand total, etc.).

**Cancel Order:**

* **Input:** Order ID
* **Process:**
  + Find the order in tbl\_pos by ID.
  + Verify the order status (not yet completed).
  + Delete the corresponding record from tbl\_pos.

## Reports Module

**Sales Report by Date**

* **Input:** Start date, end date (optional)
* **Process:**
  + Filter records in tbl\_pos based on the date range (or all dates if no dates provided).
  + Calculate total sales for the specified period.
  + Display the report with details like date, transaction number, total amount.

**Sales Report by Category (Optional):**

* **Input:** (Optional) Category of food items
* **Process:**
  + Filter records in tbl\_pos based on the category (or all categories if none provided).
  + Calculate total sales for the specified category (or all categories).
  + Display the report with details like category, food name, quantity sold, total amount.

## System flowchart

