

CHAPTER 1

INTRODUCTION

1.1 DBMS

A database is simply an organized collection of related data, typically stored on disk, and accessible by possibly many concurrent users. Databases are generally separated into application areas. For example, one database may contain Human Resource (employee and payroll) data; another may contain sales data; another may contain accounting data; and so on. Databases are managed by a DBMS.

A Database Management System (DBMS) is a set of programs that manages any number of databases.

DBMS is responsible for:

- Accessing data.
- Inserting, Updating, and Deleting data.
- Security.
- Integrity, Facilitated by:
 - Locking
 - Logging
 - Application-defined rules, including triggers
- Supporting batch and on-line programs.
- Facilitating backups and recoveries.
- Optimizing performance.
- Maximizing availability.
- Maintaining the catalog and directory of database objects.
- Managing the buffer pools.
- Acting as an interface to other systems programs.
- Supporting user interface packages, such as the popular SQL interface for relational database system.

Types of DBMS:

There are 3 traditional types of database management systems:

- i. Hierarchical
- ii. Relational
- iii. Network

Current popular database system includes Oracle, Sybase (same as Microsoft's SQL Server but on a different platform), IBM's DB2, IMS, and SQL/DS; Ingres; Informix; and smaller, but reasonably powerful off-shell products such as dBase, Access, Foxpro, Paradox, and dozens of other.

Factors influencing the choice of a database product:

- The computing platform (i.e., hardware, operating system)
- The volume of data to be managed
- The number of transactions required per second
- Existing applications or interfaces that an organization may have
- Support for heterogeneous and/or distributed computing
- Cost
- Vendor support

Object-oriented database systems are currently in development they allow us to model and manipulate complex data structures and objects, and hence support many new applications, including CAD/CAM. As of 1996, object-oriented databases represent a very small segment of the commercial market (perhaps 1%). It is interesting to note that some major DBMS vendors are starting to support complex objects (such as images) in their relational product.

1.1.1 Characteristics of DBMS:

1. Represent Some Aspects of real world applications

A database represents some features of real world applications. Any change in real world is reflected in the database. For example, let us take railway reservation system; we have in our mind some certain application of maintaining records of attendance, waiting list, train arrival and departure time, certain day etc. related to each train.

2. Self describing nature

A database is of self describing nature; it always describes and narrates itself. It contains the description of the whole data structure, the constraints and the variables. It makes its different from traditional file management system in which definition was not the part of application program. These definitions are used by the users and DBMS software when needed.

3. Logical relationship between records and data

A database gives a logical relationship between its records and data. So a user can access various records depending upon the logical conditions by a single query from the database.

4. Control Data Redundancy

DBMS follows the rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Normalization is a mathematically rich and scientific process that reduces data redundancy.

5. Query Language

DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and as different filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used.

6. Multiuser and Concurrent Access

DBMS supports multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when user attempt to handle the same data items, but users are always unaware of them.

7. Multiple views of database

Basically, a view is a subset of database. A view is defined and devoted for a particular user of the system. Different users of the system may have different views of the same system. Every view contains only the data of interest to a user or a group of users. It is the responsibility of users to be aware of how and where the data of their interest is stored.

8. Security

Features like multiple view offer security to some extend where users are unable to access data of other users and departments. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with different features. For example, a user in the Sales department cannot see the data that belongs to the purchase department. Additionally, it can also be managed how much data of the Sales department should be displayed to the user. Since a DBMS is not saved on the disk as traditional file systems, it is very hard for miscreants to break the code.

9. ACID Properties

DBMS follows the concepts of Atomicity, Consistency, Isolation, and Durability (normally shortened as ACID) in order to ensure accuracy, completeness, and data integrity. These concepts are applied on transactions, which manipulate data in a database.

1.2 PROBLEM DESCRIPTION:

Consider the schema for food ordering system

- CATEGORY(categoryid, catname)
- PRODUCT(productid, categoryid, productname, price, photo)
- PURCHASE(purchaseid, customer, total, date_purchase)
- PURCHASE_DETAIL(pdid, purchaseid, productid, quantity)

1.3 EXPLANATION ABOUT THE PROBLEM CONSIDERED:

The Project computerization of services in Food Ordering System is designed once and can be updated many times. So, the burden of application maintenance is shortlisted by all the general guidelines. This software will be capable of performing following tasks.

- i. Details about the food items
- ii. Orders to be placed of a user
- iii. Ordered details

The Database will be capable of doing following operations:

- Add or update or delete the details of the Category of food.
- Add or update or delete the details of the Food items
- Maintains the order records.

CHAPTER 2

REQUIREMENT SPECIFICATION

The requirements can be broken down into 2 major categories namely hardware and software requirements. The former specifies the minimal hardware facilities expected in a system in which the project has to be run. The latter specifies the essential software needed to build and run the project.

2.1 Hardware Requirements

The Hardware requirements are very minimal and the program can be run on most of the machines.

- Processor - Intel 486/Pentium processor or better
- Processor Speed - 500 MHz or above
- Hard Disk - 20GB(approx)
- RAM - 64MB or above
- Storage Space - Approx. 2MB

2.2 Software Requirements

- Technology Implemented : Apache Server, MySQL Server
- Language Used : PHP
- Database : My SQL
- User Interface Design : HTML, CSS, Bootstrap
- Web Browser : Google Chrome
- Software : XAMPP Version: 7.1.10

CHAPTER 3

DATABASE DESIGN

3.1 SET OF ENTITIES AND ATTRIBUTION

- An **entity** is an object that exists and is distinguishable from other objects. For instance, John Harris with S.I.N.890-12-3456 is an entity, as he can be uniquely identified as one particular person in the universe.
- An entity may be **concrete** (a person or a book, for example) or **abstract** (like a holiday or a concept).
- An **entity set** is a set of entities of the same type (e.g., all person having an account at a bank).
- Entity set **need not be disjoint**. For example, the entity set employee (all employee of bank) and the entity set customer (all customer of the bank) may have members in common.
- An entity is represented by a set. of **attributes**
 - E.g., name, S.I.N., street, city for “customer” entity.
 - The **domain** of the attribute is the set of permitted values (e.g., the telephone number must be seven positive integers).

Attributes decided the property of characteristic of an entity. attributes represented using eclipse.

- i. **Key attributes**: it represents the main characteristic of entity. it is used to represent the primary key.
- ii. **Composite attributes**: an attribute can also have same attribute.
- iii. **Single valued attributes**: attribute which have single value for a particular entity.
- iv. **Multivalued attributes**: attribute which have many values for a particular entity.

3.1.1 CONSTRAINTS AMONG ATTRIBUTES:

a. Domain constraint

Domain integrity means the definition of the valid set of values for an attribute we define datatype, length or size.

b. Foreign key integrity constraints

The constraints identifies any column referencing the primary key in another table. It establishes the relationship between 2 columns in the same table or between different tables.

c. Unique key constraints

It ensures that a column or group of columns in each row have distinct value.

d. Referential integrity constraints

The constraints is specified between 2 tables and is used to maintain consistency among rows between the 2 tables

e. SQL check constraints

It defines a bus rule on a column, all the rows must satisfy this rule it can be applied to a specific column or group of columns.

3.1.2 FUNCTIONAL DEPENDENCY:

In relational database theory, a functional dependency is a constraint between two sets of attributes in a relation from a database. In other words, functional dependency is a constraint that describes the relationship between attributes in a relation.

Given a relation R, a set of attributes X in R is said to functionally determine another set of attributes Y, also in R, (written $X \rightarrow Y$) if, and only if, each X value in R is associated with precisely one Y value in R; R is then said to satisfy the functional dependence $X \rightarrow Y$.

3.1.3 GUIDELINES TO DESIGN GOOD RELATION SCHEMA (G1 OR G2)

a) Guideline 1 (semantics of attributes)

- ❖ Design a relational schema so that it is easy to explain its meaning.
- ❖ Do not combine attribute from multiple entity types and relationship types into a single relation.
- ❖ Intuitively, if a relation schema corresponds to one entity type, or one relationship type, the meaning tends to clear.

b) Guideline 2 (redundant information in tuples and update anomalies)

- ❖ Design the basic relation schema so that no insertion, deletion, or modification anomalies are present in the relation.
- ❖ If any anomalies are present, note them clearly and make sure that the programs that update the database will operate correctly.
- ❖ Due to improper grouping of attribute into a relation schema, the following problems are encountered.
 - I. Storage wastage +
 - II. Insert anomalies +
 - III. Delete anomalies +
 - IV. Modification anomalies +

c) Guideline 3 (null value in tuples)

- If possible avoid placing the attribute in a base relation whose values may frequently be null. If nulls are unavoidable, make sure they apply in exceptional case only and not to majority of tuples in a relation.
- Problems with null values:
 - A. Waste of disk space
 - B. Problems of understanding the meaning of attribute
 - C. Problems in specifying JOIN operations
 - D. Problems in applying some aggregate functions
 - E. May have multiple interpretation (not applicable, unknow, unavailable)

d) Guideline 4 (generation of spurious tuples)

- When we apply join operations spurious tuple will be generated.
- Bad designs for a relational database may result in erroneous result for certain join operations
- The “lossless join” property is used to guarantee meaningful result for join operations

Note: the relation should be designed to satisfy the lossless join condition. Tuples should be generated by doing a natural-join of any relation.

3.2 SCHEMA DIAGRAM:

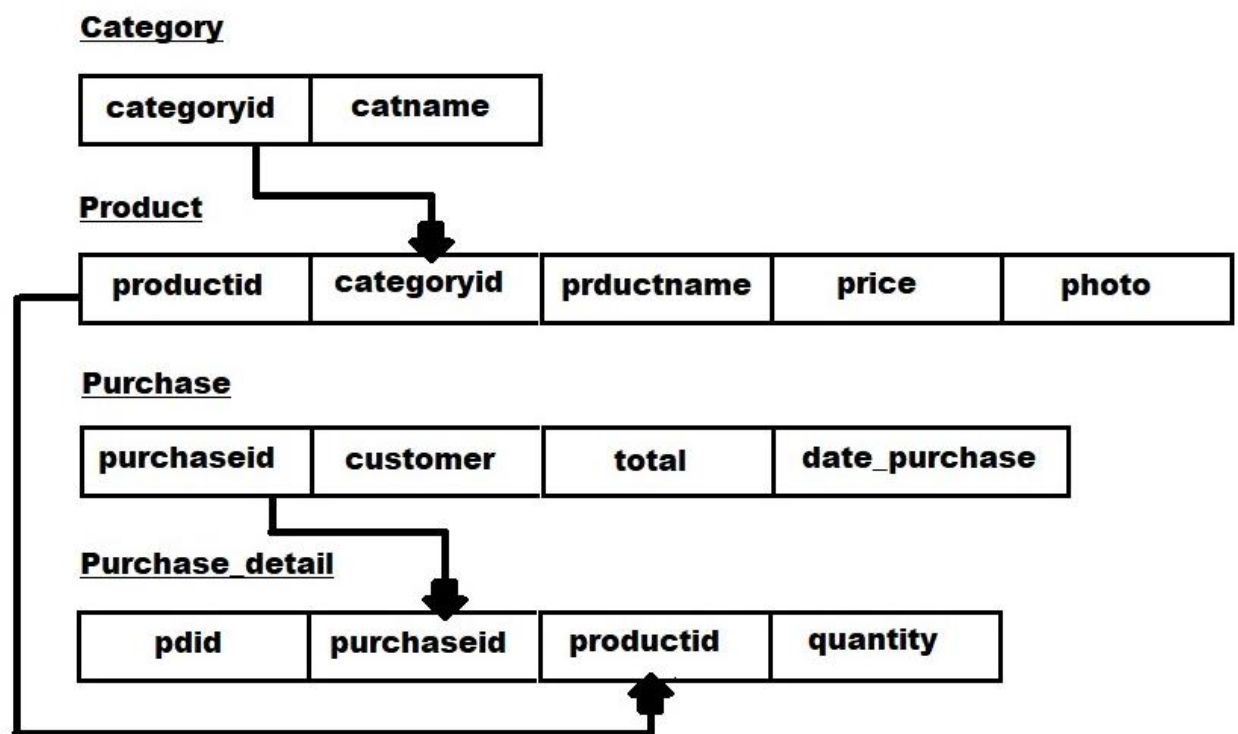


Figure 1 – Schema Diagram

3.3 ENTITY RELATIONSHIP DIAGRAM:

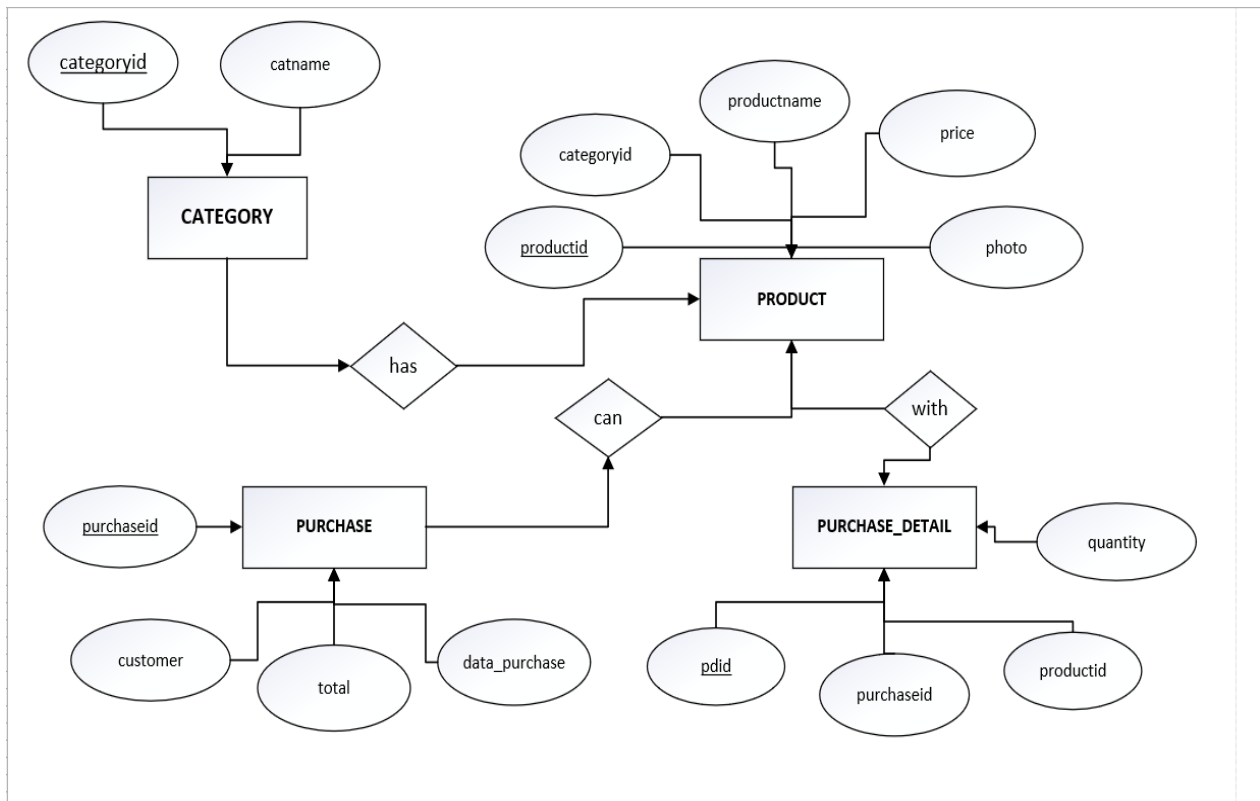


Figure 2 – E R Diagram

3.4 DATABASE SCHEMA

CATEGORY:

categoryid	catname
4	BREAKFAST
5	LUNCH
6	DINNER
7	BEVERAGES

Table 1 – Category Schema

PRODUCT:

productid	categoryid	productname	price	photo
14	4	Masala dosa	50	upload/masala_dosa_1573029668.jpeg
15	4	Benne Dosa	50	upload/benne_dosa_1573029757.jpeg
16	5	South Indian Meals	120	upload/south_indian_meals_1573029997.jpeg
17	6	Spaghetti	80	upload/spagetti_1539097965.png
18	7	Lassi	70	upload/lassi_1573030120.jpeg
19	7	Mango Lassi	80	upload/mango_lassi_1573030160.jpeg
20	6	Light Dinner	150	upload/light_dinner_1573030089.jpeg
21	4	Idli	40	upload/idli_1573029799.jpeg
22	4	Poori	45	upload/poori_1573029911.jpeg
23	7	Coca-Cola	80	upload/cocacola_1539097796.jpg

Table 2 – Product Schema**PURCHASE:**

purchaseid	customer	total	date_purchase
8	Sapna	600	2017-12-06 15:29:00
9	Kumari	450	2018-10-09 20:19:43

Table 3 – Purchase Schema**PURCHASE DETAIL:**

pdid	purchaseid	productid	quantity
21	12	15	2
22	12	18	2
23	13	14	2
24	13	19	2
25	14	21	2
26	14	23	2
27	15	22	2
28	15	18	2

Table 4 – Purchase Detail Schema

CHAPTER 4

SOURCE CODE

4.1 DATABASE CODE IMPLEMENTATION:

FILE NAME: foodsys.sql

```
CREATE TABLE IF NOT EXISTS `category` (  
  `categoryid` int(11) NOT NULL,  
  `catname` varchar(30) NOT NULL  
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=latin1;
```

```
INSERT INTO `category` (`categoryid`, `catname`) VALUES  
(4, 'BREAKFAST'),  
(5, 'LUNCH'),  
(6, 'DINNER'),  
(7, 'BEVERAGES');
```

```
CREATE TABLE IF NOT EXISTS `product` (  
  `productid` int(11) NOT NULL,  
  `categoryid` int(1) NOT NULL,  
  `productname` varchar(30) NOT NULL,  
  `price` double NOT NULL,  
  `photo` varchar(150) NOT NULL  
) ENGINE=InnoDB AUTO_INCREMENT=24 DEFAULT CHARSET=latin1;
```

```
INSERT INTO `product` (`productid`, `categoryid`, `productname`, `price`, `photo`)  
VALUES  
(14, 4, 'Masala Dosa ', 50, 'upload/masala_dosa_1573029668.jpeg'),  
(15, 4, 'Benne Dosa', 50, 'upload/benne_dosa_1573029757.jpeg'),  
(16, 5, 'South Indian Meals', 120, 'south_indian_meals_1573029997.jpeg'),  
(17, 6, 'Spaghetti', 80, 'upload/spagetti_1539097965.png'),
```

FOOD ORDERING SYSTEM

```
(18, 7, 'Lassi', 70, 'upload/lassi_1573030120.jpeg'),  
(19, 7, 'Mango Lassi', 80, 'upload/mango_lassi_1573030160.jpeg'),  
(20, 6, 'Light Dinner', 150, 'upload/light_dinner_1573030089.jpeg'),  
(21, 4, 'Idli', 40, 'upload/idli_1573029799.jpeg'),  
(22, 4, 'Poori', 45, 'upload/poori_1573029911.jpeg'),  
(23, 7, 'Coca-Cola', 80, 'upload/cocacola_1539097796.jpg');
```

```
CREATE TABLE IF NOT EXISTS `purchase` (  
  `purchaseid` int(11) NOT NULL,  
  `customer` varchar(50) NOT NULL,  
  `total` double NOT NULL,  
  `date_purchase` datetime NOT NULL  
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=latin1;
```

```
INSERT INTO `purchase` (`purchaseid`, `customer`, `total`, `date_purchase`)  
VALUES  
(8, 'Sapna', 600, '2017-12-06 15:29:00'),  
(9, 'Kumari', 450, '2018-10-09 20:19:43');
```

```
CREATE TABLE IF NOT EXISTS `purchase_detail` (  
  `pdid` int(11) NOT NULL,  
  `purchaseid` int(11) NOT NULL,  
  `productid` int(11) NOT NULL,  
  `quantity` int(11) NOT NULL  
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=latin1;
```

```
INSERT INTO `purchase_detail` (`pdid`, `purchaseid`, `productid`, `quantity`)  
VALUES  
(13, 8, 15, 2),  
(14, 8, 17, 2),  
(15, 8, 18, 2),  
(16, 9, 15, 3);
```

```
ALTER TABLE `category`  
ADD PRIMARY KEY (`categoryid`);
```

FOOD ORDERING SYSTEM

```
ALTER TABLE `product`  
ADD PRIMARY KEY (`productid`);
```

```
ALTER TABLE `purchase`  
ADD PRIMARY KEY (`purchaseid`);
```

```
ALTER TABLE `purchase_detail`  
ADD PRIMARY KEY (`pdid`);
```

```
ALTER TABLE `category`  
MODIFY          `categoryid`          int(11)          NOT          NULL  
AUTO_INCREMENT,AUTO_INCREMENT=9;
```

```
ALTER TABLE `product`  
MODIFY          `productid`          int(11)          NOT          NULL  
AUTO_INCREMENT,AUTO_INCREMENT=24;
```

```
ALTER TABLE `purchase`  
MODIFY          `purchaseid`          int(11)          NOT          NULL  
AUTO_INCREMENT,AUTO_INCREMENT=10;
```

```
ALTER TABLE `purchase_detail`  
MODIFY          `pdid`          int(11)          NOT          NULL  
AUTO_INCREMENT,AUTO_INCREMENT=17;
```

4.2 PHP CODES IMPLEMENTATION:**ADD CATEGORY****FILE NAME:** addcategory.php

```
<?php
    include('conn.php');

    $cname=$_POST['cname'];

    $sql="insert into category (catname) values ('$cname')";
    $conn->query($sql);

    header('location:category.php');

?>
```

ADD PRODUCT**FILE NAME:** addproduct.php

```
<?php
    include('conn.php');

    $pname=$_POST['pname'];
    $price=$_POST['price'];
    $category=$_POST['category'];

    $fileinfo=PATHINFO($_FILES["photo"]["name"]);

    if(empty($fileinfo['filename'])){
        $location="";
    }
    else{
```


FOOD ORDERING SYSTEM

```
        $newFilename=$fileinfo['filename'] . "_" . time() . "." .  
$fileinfo['extension'];  
        move_uploaded_file($_FILES["photo"]["tmp_name"],"upload/" .  
$newFilename);  
        $location="upload/" . $newFilename;  
    }  
  
    $sql="insert into product (productname, categoryid, price, photo) values  
($pname, '$category', '$price', '$location')";  
    $conn->query($sql);  
  
    header('location:product.php');  
  
?>
```

DELETE CATEGORY

FILE NAME: delete_category.php

```
<?php  
    include('conn.php');  
  
    $id = $_GET['category'];  
  
    $sql="delete from category where categoryid='$id'";  
    $conn->query($sql);  
  
    header('location:category.php');  
  
?>
```

DELETE PRODUCT

FILE NAME: delete_product.php

```
<?php
    include('conn.php');

    $id = $_GET['product'];

    $sql="delete from product where productid='$id'";
    $conn->query($sql);

    header('location:product.php');
?>
```

DBMS CONNECTION

FILE NAME: conn.php

```
<?php

$conn = new mysqli('localhost', 'root', '', 'foodsys');

if ($conn->connect_error) {

    die("Connection failed: " . $conn->connect_error);

}

?>
```

CHAPTER 5

SCREENSHOTS

MENU SCREEN:

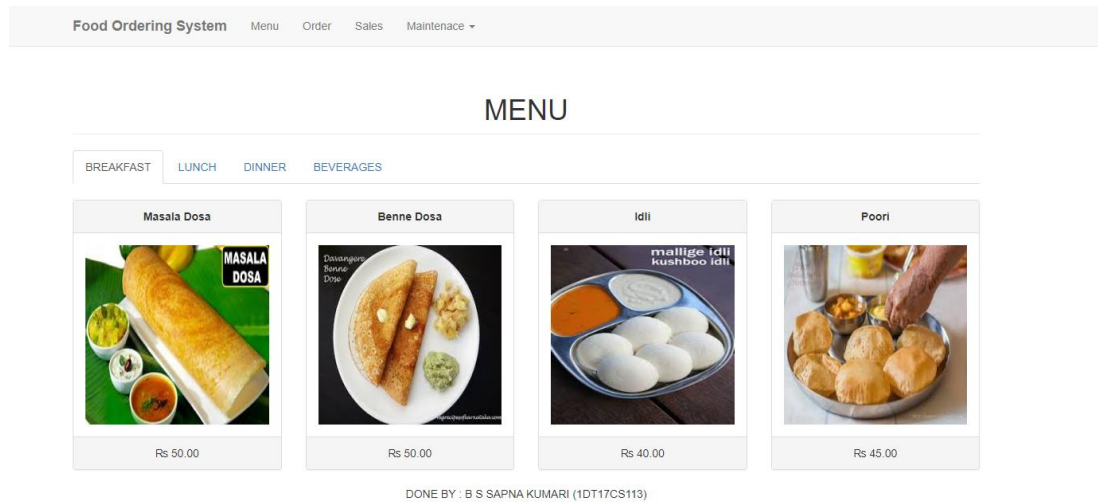


Figure 3 – Menu Screen

ORDER SCREEN:

Food Ordering System Menu Order Sales Maintenance

ORDER

<input type="checkbox"/>	Category	Product Name	Price	Quantity
<input type="checkbox"/>	BREAKFAST	Benne Dosa	Rs 50.00	<input type="text"/>
<input type="checkbox"/>	BREAKFAST	Idli	Rs 40.00	<input type="text"/>
<input type="checkbox"/>	BREAKFAST	Masala Dosa	Rs 50.00	<input type="text"/>
<input type="checkbox"/>	BREAKFAST	Poori	Rs 45.00	<input type="text"/>
<input type="checkbox"/>	LUNCH	South Indian Meals	Rs 120.00	<input type="text"/>
<input type="checkbox"/>	DINNER	Light Dinner	Rs 150.00	<input type="text"/>
<input type="checkbox"/>	DINNER	Spaghetti	Rs 80.00	<input type="text"/>
<input type="checkbox"/>	BEVERAGES	Coca-Cola	Rs 80.00	<input type="text"/>
<input type="checkbox"/>	BEVERAGES	Lassi	Rs 70.00	<input type="text"/>
<input type="checkbox"/>	BEVERAGES	Mango Lassi	Rs 80.00	<input type="text"/>

Customer Name

Figure 4 – Order Screen

SALES SCREEN:

Food Ordering System Menu Order Sales Maintenance ▾			
SALES			
Date	Customer	Total Sales	Details
Oct 09, 2018 08:19 PM	Kumari	Rs 450.00	View
Dec 06, 2017 03:29 PM	Sapna	Rs 600.00	View

Figure 5 – Sales Screen

FULL SALES DETAILS SCREENS:

Sales Full Details			
Customer: Sapna		Dec 06, 2017 03:29 PM	
Product Name	Price	Purchase Quantity	Subtotal
Benne Dosa	Rs 50.00	2	Rs 100.00
Spaghetti	Rs 80.00	2	Rs 160.00
Lassi	Rs 70.00	2	Rs 140.00
TOTAL			Rs 600.00
Close			

Figure 6 – Full Sales Details Screen

CATEGORY MAINTENANCE SCREEN:

Food Ordering System Menu Order Sales Maintenance ▾	
CATEGORY CRUD	
+ Category	
Category Name	Action
BREAKFAST	Edit Delete
LUNCH	Edit Delete
DINNER	Edit Delete
BEVERAGES	Edit Delete

Figure 7 – Category Maintenance Screen

PRODUCT MAINTENANCE SCREEN:











Food Ordering System Menu Order Sales Maintenance ▾			
PRODUCTS CRUD			
All Category ▾		+ Product	
Photo	Product Name	Price	Action
	Benne Dosa	Rs 50.00	Edit Delete
	Idli	Rs 40.00	Edit Delete
	Masala Dosa	Rs 50.00	Edit Delete
	Poori	Rs 45.00	Edit Delete
	South Indian Meals	Rs 120.00	Edit Delete
	Light Dinner	Rs 150.00	Edit Delete
	Spaghetti	Rs 80.00	Edit Delete
	Coca-Cola	Rs 80.00	Edit Delete
	Lassi	Rs 70.00	Edit Delete
	Mango Lassi	Rs 80.00	Edit Delete

Figure 8 – Product Maintenance Screen

CONCLUSION

The project is based on the food ordering system. This helps the restaurant to do all function in an effective manner and faster way. This reduces the manual work.

Every steps has been taken to make the working of project comfortable as possible for the users. The report have been created and can be called according to the requiriements.

all the data are stored in tables automatically. This application provides security from loss, disclosure, modification and destruction of data.

All the database is automatically updated. It also keeps the record of food items prepared and the sales of food and also the record of balance food.

All the food order details like order types (normal, home delivery, party order etc) are stored daily.

REFERENCES:

BOOK REFERENCES:

- Learn to Code HTML and CSS: Develop and Style Websites (Web Design Courses) 1st, Kindle Edition by Shay Howe
- *PHP 6 and MySQL 5* - Larry Ullman

WEBSITE REFERENCES:

HTML Learning:

- <https://www.codecademy.com/>
- <https://dash.generalassemb.ly/>
- <https://www.w3schools.com/>

PHP Learning:

- <http://www.tutorialspoint.com/php/>
- <https://killerphp.com>
- <https://www.w3schools.com/>

PERSONAL DETAILS:

NAME: B S SAPNA KUMARI

USN: 1DT17CS113

SEMISTER AND SECTION: 5th SEM & 'C' SEC

**COLLEGE: DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND
MANAGEMENT**

EMAIL ID: kumaribssapnakumari@gmail.com