SYNOPSIS :

HONEY

INTRODUCTION:

" **The Honey Bee-Keepers And Village Industries Co-Operative Society Ltd** ", the apiary farm which contains 3 types of honey bees and this bees count is in lakhs. During a single collection trip, a honey bee will visit anywhere from 50 to 100 flowers and farm produces 150 to 200 kgs of honey per day, honey is supplied daily or weekly to local markets and neighborhood states as well. Farm takes Raw materials like Jar, dippers and papers etc.,. from the surrounding farmers. Bee wax has high usage this is found in many of our everyday products including medicines, cosmetics , polishes and soaps etc.,.

OBJECTIVE:

Web application for the client HONEY to maintain the

Database .Objective is to establish cost effective and hygenic and its value-added products, to overcome the existing problems faced by the HONEY production formers.

- We are providing web base application
- To providing detailed information about HONEY and HONEY production, sale production.

INPUT:

- Customer login details
- Customer details
- Purchase details
- Former details
- Employee details

- Employee salary details
- Production details
- Production sales
- Product details

Output:

- Customer report
- Purchase report
- Employee report
- Production report
- Product report

Project Modules:

Admin module :

Admin must have privileges to this application for managing category details of different type products, maintaining sales,products,stock.

Production Module:

In this complete production details will be there

Employee Module:

It consists of employees information and salary payment

ExpensesModule :

In this complete maintenance of farm information will be present

Customers Module:

Here the customer can access all the products and customer can order the products. Each users has his/her user ID and password.

Sales Module:

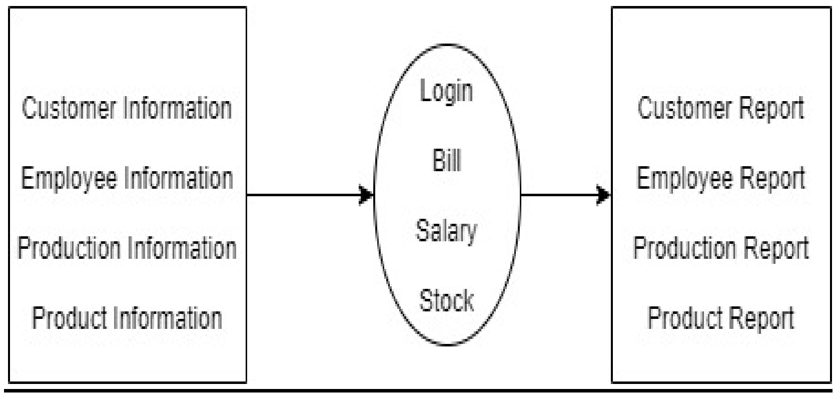
It consistent of complete sales information which is done with per day or per week and also per month.

Purchase Module:

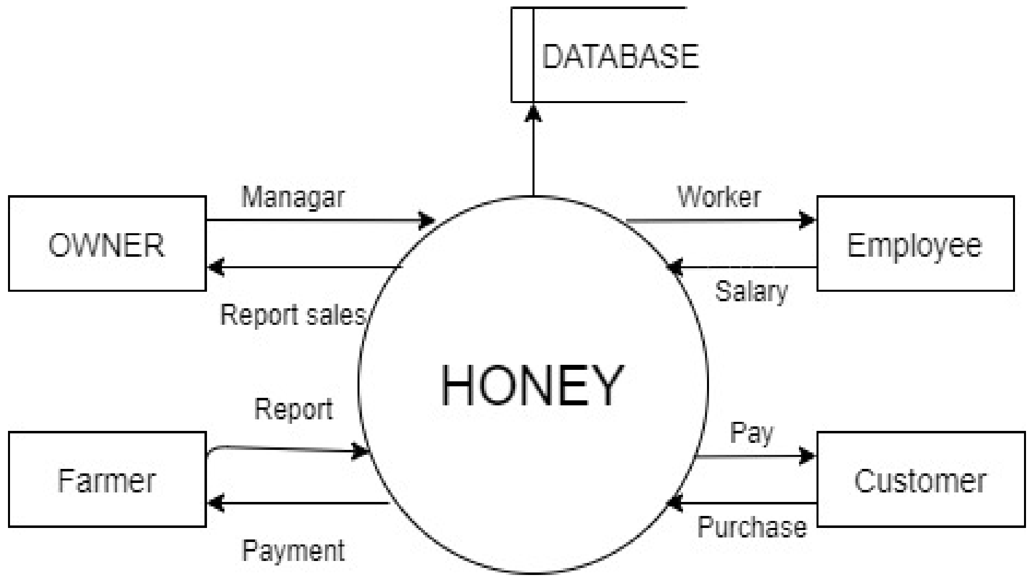
It contains the purchase details. Like quantity quality, cost.

Transportation Module: Product tracking (from where we hv purchased rowmaterial, and where we are delivering our products) .

PROCESS LOGIC :



ZERO LEVEL DFD:



Present System:

For an accounting purpose they use day book and ledgersbin that all recordsshould be maintained.It beome burden for them so we introduced computerized manner fortheir maintenace.

Our project aims at creating an application which will automate the processes in a educational trust and website design. It is also desirable to maintain every transactions.

proposed system:

The project can be modified in future according to the need of employee,administrator and user.

This project can be used not only as the management system also only by including some more management features and modifications.

TOOL/PLATFORM OR LANGUAGE TO BE USED

HARDWARE REQUIREMENTS

Operating System:	Windows 2007 & above
Processor	: i3 and Higher
RAM	: 256MB and more
Hard Disk	: 4GB and higher

SOFTWARE REQUIREMENT

Front End	: HTML, CSS,JAVA SCRIPT...ETC
Middle Ware:	PHP,JAVA
Back End	: MYSQL...ETC
Server	: Wamp server
Operating System:	Window xp

DURATION OF PROJECT

3 months

<u>MEMBER OF PROJECT:</u>

<u>1) Anagha Hegde</u>

<u>2)Madhu Prasanna Undavalli</u>

<u>CLIENT OF THE PROJECT:</u>

<u>Shreedhar Hegde</u>

ARE YOU DOING THIS PROJECT FOR ANY INDUSTRY OR CLIENT?

IF YES ACCEPTANCE OF THE PROJECT:

- YES

SCOPE OR FUTURE APPLICATION

- Online payments can be done.
- Application for mobile can be done.
- One centrelized format can be done to all the formers

2.FRAME WORK

2.1 HTML

HTML means Hypertext Markup Language. HTML is a method of describing the format of document, which allows them to be viewed on computer screen. Web browsers display HTML documents, program which can navigate across networks and display a wide variety of types of information. HTML pages can be developed to be simple text or to be complex multimedia extra advantages containing, moving images, virtual reality, and java applets.

Hypertext Markup language (HTML) is used to creating the web page either of static or dynamic and used to develop the user friendly web pages.

HTML is used for developing web pages .HTML is popularly used in World Wide Web (WWW). It uses ASCII characters for both the main text and formatting instructions the main text is data and the whole information is used by the browser to format the data. A HTML document is simply a text file, which contains certain information you would like to publish.

A set of instruction embedded in a document is called Markup Language. These instructions describe what the document text means and how it should look in a display. The language also tells you how to make a document with other document on your local systems. The World Wide Web and other inter resources such as FTP.

The global publishing format of the Internet is HTML. It allows authors to use not only text but also format that text with headings, list and tables, and also includes still images videos, and sound within text. Readers can access pages information from any where in the world at the click of mouse button information can be downloaded to readers own PC or workstations HTML pages can also be used for entering a data and as a front end for commercial transaction.

2.2 Dreamweaver:

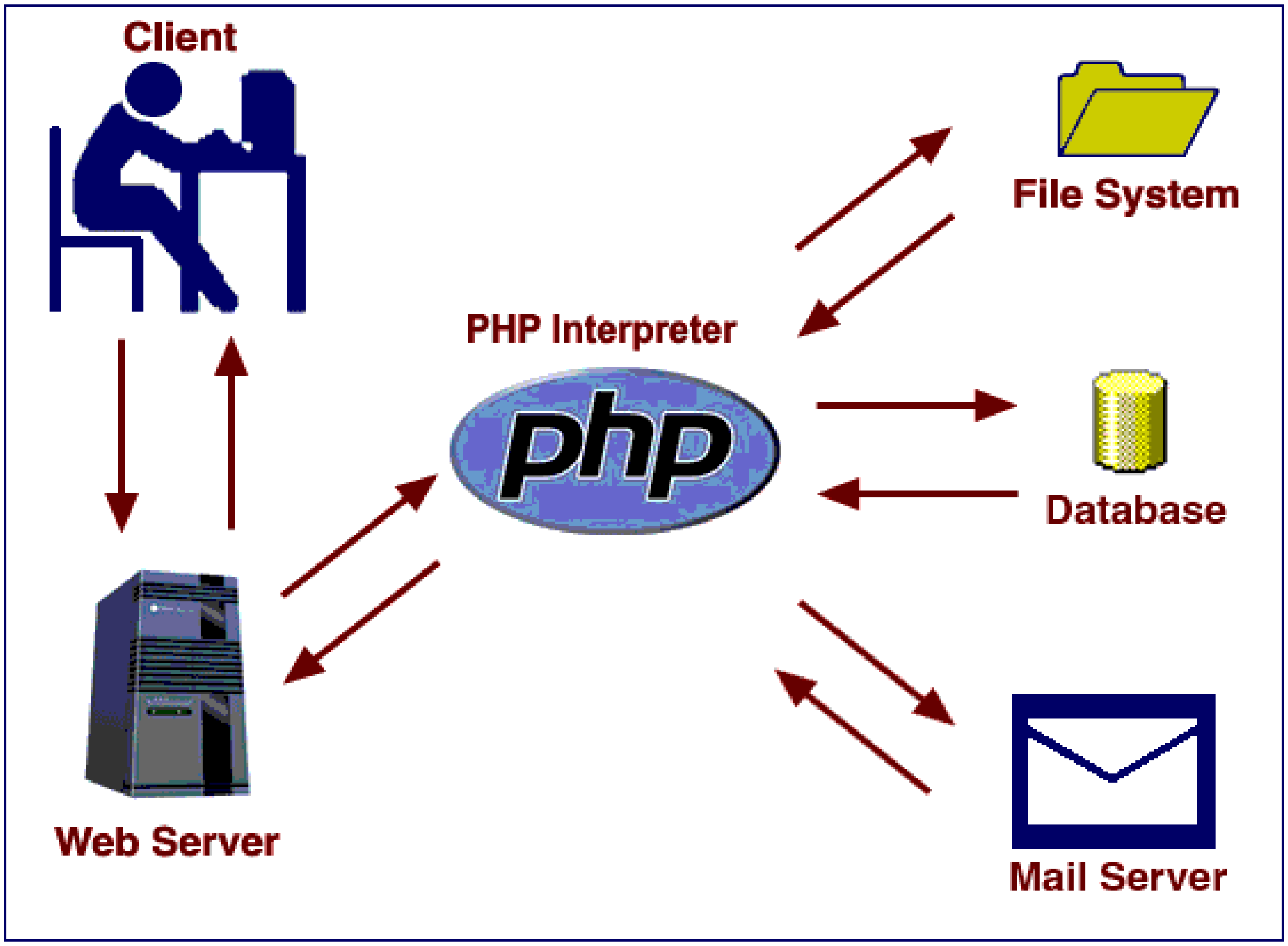
A website authoring program originally developed by Macromedia. It does many things including allowing designer to move back and forth between visual and HTML modes.

2.3 PHP:

PHP is a scripting language originally designed for producing dynamic webpages. It has evolved to include a command line interface capability and can be used in standalone graphical application. While PHP was originally created by RasmusLerdorf in 1995, the main implementation of PHP is now produced by the PHP Groups and serves as the de facto standard for PHP as there is no formal specification.

PHP is a scripting language under the PHP License; however it is incompatible with the GNU General Public License (GPL). Due to restrictions on the usage of the term PHP. It is widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It generally runs on a web server, taking PHP code as its input, I am creating web pages as out puts. It can be deployed on web servers and on almost every operating system and platform free of charge. PHP in installed on more the twenty million web sites and one million web servers.

PHP Architecture:



Usage:

PHP primarily acts as a filter, taking input from a file or stream containing text and/or PHP instructions and outputs another stream of data; most commonly the output will be HTML. It can automatically detect the language of the user. From PHP 4, the PHP parser compiles input to produce byte code for processing by the Zend Engine, giving improved performance over its interpreter predecessor. Originally designed to create dynamic web pages, PHP's principal focus is server side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client, such as Microsoft's Active Server Pages, Sun Microsystems' JavaServer Pages, and mod_perl. PHP has also attracted the development of many frameworks that provide building blocks and a design structure to promote rapid application development (RAD). Some of these include CakePHP, Symfony, CodeIgniter, and Zend Framework, offering features similar to other web application frameworks.

The WAMP architecture has become popular in the web industry as a way of deploying web applications. PHP is commonly used as the PHP in this bundle alongside Linux, Apache and MySQL, although they may also refer to Python or Perl. As of April 2007, over 20 million Internet domains were hosted on servers with PHP installed, and PHP was recorded as the most popular Apache module. Significant websites are written in PHP including the user-facing portion of Facebook, Wikipedia (MediaWiki), Yahoo!, MyYearbook, Wordpress.

In addition to server-side scripting, PHP can be used to create stand-alone, compiled applications and libraries, it can be used for shell scripting.

2.4 WAMP:

Stands for "Windows Apache MySQL, and PHP". WAMP is a variation of LAMP for windows system and is often installed as a software bundle (Apache, MySQL, and PHP). It is often used for wed development and internal testing, but may also be used to serve live wed site.

The important part of the WAMP is Apache (or "Apache HTTP Server") which is used to run the web server within the windows. By running the locate Apache Web Server on a Windows machine, a web developer can test web pages in a web browser with out-publishing live on the internet.

WAMP also includes MySQL and PHP, which are two of the most common technologies used for creating dynamic web sites. MySQL is a high speed database while PHP is a scripting language that can be used to access data from data base. by installing these two components locally a developer can build and test a dynamic web site before publishing it to a public web server.

While Apache, MySQL and PHP are open source components that can be installed individually, they are usually installed together. One popular package is called "WAMP Sever", which provides a user friendly way to install and configure the "AMP" components on windows.

2.5 MY SQL:

What is Database?

Quite simply, Its an organised collection of data. A Database management system (DBMS) such as access file maker Pro, Oracle or SQL server provides you with the software tools you need to organize that data in a flexible manner. It includes facilities to add modify or delete data from database, ask questions (or queries) about the data stored in the database and produce reports summarizing selected contents.

My SQL is a multithreaded, multi-user SQL database management system (DBMS). The basic program runs as a server providing multi-user access to a number of databases. My SQL was owned and sponsored by a single for-profit firm the Sidish company. My SQL now a subsidiary of Sun Micro System, which hold the copy write to most of the database. The data in My SQL is stored in database objects called Tables. A table is a collection of related data entries and consist of columns and rows. Databases are useful when storing information categorically.

Hardware required for the project:

• Processor	**:**	Intel -III based system.
• RAM	**:**	2GB and more
• Hard Disk	**:**	Minimum 10GB free space.

Software required for the project:

• Operating System	**:**	Windows XP or Above
• Front End	**:**	HTML, CSS.
• Back End	**:**	My SQL
• Middleware	**:**	PHP
• Server	**:**	Wamp server
• Designig tool	**:**	Dreamweaver

3.PROJECT SUBJECT

Now a days it is really hard to manage the work of different activities without the help of software. Each and every product that is produces takes time to manage outlet, manage categories, products, users, outlet and distributors. This software helps us to manage the distribution of alloy products. It helps to make bills and maintain the stock of all type of alloy products.

3.1 NUMBER OF MODULES:

1. Admin privileges Module

- Admin is responsible for Creating Users of the System.
- Handles all the details of the industry.

2. Employee Module

- His contact and service details management.
- Monthly reports of his attendance.
- Every employee's salary report.

3. Customer Module

- Customer details and customer order details are maintained.
- Orderd product information will bw maintained

4. SOFTWARE REQUIREMENTS SPECIFICATION

4.1 INTRODUCTION:

SRS is the official statement of what is required by the system developers; it includes both user requirements for the system and detailed specification of the system requirements. This document is used while designing the proposed system and can also be used in the future if the system is to be enhanced.

4.1.1 Purpose:

The purpose of this Requirements Elicitation document is to provide a clear understanding as to what actually the Industry Management Syatem is and to identify the critical requirements essential for the project's successful completion.

This document explains our team architecture, our teams's initial understanding of the user needs.

4.1.2 Document Conventions:

Main Section Title: Font: Times New Roman, Bold: Size 16

Sub Section Title: Font: Times New Roman, Bold: Size 14

Other Text Matter: Font: Times New Roman, Bold: Size 12

4.1.3 Scope:

This document is intended for providing an abstract overview of the system and general overview of the entire project. The scope of the document:

- Team Architecture,
- System Functional and Non-Functional Requirements
- Prototype of the System,

4.1.4 References:

https://www.w3schools.com/

https://getootstrap.com/

https://www.youtube.com/

https://stackoverflow.com/

https://docs.djangoproject.com/en/2.1/

4.2 General Description

This section will give an overview of the whole application. The explanation of the application will be in its context to show the application interacts with other systems and introduce the basic functionality of it. It will also describe all the constraints and assumptions for the application.

4.2.1 Product Perspective:

As per the requirements of the client, to build a customized application that facilitates to maintain the entire details of the alloy industry to get through the demerits of existing manual system with new implementations such as:

- Graphical User Interface: Interface which allows the client to interact with the system.
- Maintenance of product and user history that provide details separately in database.
- Provides ordering of products and bill generation.
- Providing OTP service to the user through mail in case of forgot password.
- Captcha facilities to provide security services.
- Payments are done to the employees.
- Stock report generation.

Existing System:

- In the present system there is no user system intervention.
- Retrieval of data takes lot of time.
- Any ordering should be done manually.

Proposed System:

- Gives a platform to communicate with the system and perform a transaction.
- It is very economic as compared to present system.
- Sends notification to the user Email-id.
- Provides data security and huge maintenance of data.

4.2.2 Product Fucctionalities:

Industry Management Project should support the following functionalities:

- Login: Sign in into the website.
- Sign up: Registers into the website.
- Gmail Service: Provides mailing service via Gmail from to the registered users.
- Ordering: Distributors can order the products and can be confirmed through the invoice generated.

4.2.3 User Characteristics:

End Users

- No specific knowledge or skills are required from the end user.
- End user should have basic idea about computer operations.

Administrator

- Administrator must be having good knowledge of database management system.
- Administrator must be capable to manage user rights.

4.3 Specific Requirements:

4.3.1 Functional Requirements:

a) Login

- Start the application.
- User enters the username and password.
- System does authentication and main screen is displayed.

Authorization Fails

- Prompt the user that he typed the wrong password.
- Allow him to re-enter the password.

b) Change Password

- User initiates the change password command.

- User is prompted for to enter old password, new password and confirm password.
- System does authentication.
- New password is registered with the system.

Authorization Fails

- Prompt the user that he typed wrong password and allow him to re-enter. Give him 3 chances.

c) Forgot Password

- User initiates the forgot password command.
- User is prompted to enter email and prompted with a link to recover password.
- System authenticaton.

Authorization Fails

- Allow the user to re-enter the password.
- New password and confirm password do not match. Allow the user to re-enter the
- **Order Placing** password. Give him 3 chances.

d)SignUp

- After successfully logging into the system, the distributor is redirected to the respective page, where he can browse through the menu and place his orders.
- The only requirement here is that the distributor needs to log into the system.

e) Administrative log-in

- Administrator involves managing of the orders placed by the customers. Also, adding and updating the products in the database.

4.3.2 Other Non-Functional Requirements:

a) Performance Requirements

- Should run on 500MHz, 64mb machine.

- 90% of responses should be within 3 second, except for downloading for which more time is acceptable.

b) Security Requirements

This application requires a Google email address with password is required to verify the identification of the user

4.3.3 External Interface Requirements:

a) User Interfaces:

Most user-friendly interface has been designed, Login pages for the Customer, Employee and admin. The main interfaces used in the system are the forms and menus.

b) Hardware Interfaces:

The system does not require any additional hardware interfaces, so the user need not focus on the hardware apart from the standard hardware.

c) Software Interfaces: WAMP server, internet browser installed on the server machine

5.DESIGN OF THE SYSTEM

5.1 Introduction

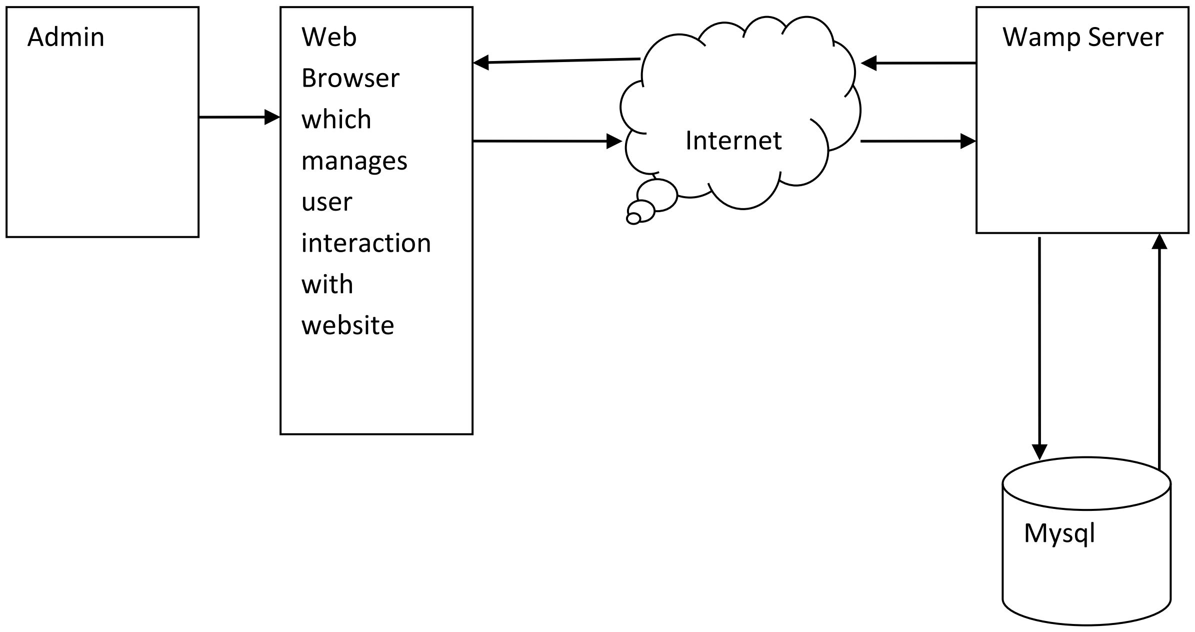
The purpose of the decision phase is to plan a solution of the problem specified by the requirements document. This phase is the first step in moving the problem domain to the solution domain. It involves the process, in which conceiving, planning and carrying out the plan generating the necessary report, In other words, the design phase act as a bridge between SRS and implementation phase. The design of the system is perhaps the most critical factor affecting the quality of the software, and as a major impact on the later phase, particularly the testing and maintenance.

Software Design

Design is the key phase of any project. It is the first step in moving from the problem domain to the solution domain. The input to the design phase is the specifications of the system to be designed. The output of the top-level design is the architectural design, or the system design for the software system to be built. A design should be very clear, verifiable, complete, traceable, efficient and simple.

Architecture Design

The architecture design defines the relationship among major structural element of the program. Architecture diagram shows the relationship between different components of system. This diagram helps to understand the overall concept of system.



Logical design

The graphical representation of system data and how the process transforms the data is known as Data Flow Diagram. It shows the logical flow of the data.

The logical design describes the detailed specification for the system, describing its features, an effective communication and the user interface requirements. The logical design of proposed system should include the following.

1. External system structure.
2. Relationship between all the activities.
3. The physical construction and all the activities.
4. Global data.
5. Control flow.
6. Derived program structure.

Design Principles

Basic design principles that enable the software engineer to navigate the design process are:

- The design process should not suffer from "Tunnel vision".
- The design should be traceable to analysis model.
- The design should not reinvent the wheel.
- The design should minimize the intellectual distance between the software and the problem, as it exists in the real world.
- The design should exhibit uniformity and integrity.
- The design should be structured to accommodate changes.
- The design is not coding and coding is not design.
- The design should be reviewed to minimize the conceptual errors.

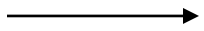
5.2 Data Flow Diagram

The data flow diagram (DFD) is one of the important modeling tools. It shows the user of the data pictorially. DFD represents the flow of the data between different transformations and processes in the systems. The data flow diagram shows logical flow of the data. It represents the functional dependencies within a system. It shows output values in a computation or derived from input values. It is a simple pictorial representation or model for system behavior. It specifies, "What is to be done but not how is to be done". It describes the

logical structure of the system. It relates data information to various processess of the system. It follows top-down approach.

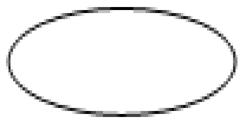
Data Flow Diagram Notations:

Data Flow:



It may be from file-to-file or file-to-process or process-to-process. It is generally in terms of attributes. There may be either an input data flow or output data flow.

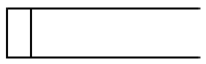
Functional processing:



The process is nothing but the transformation of data. It starts

With the subject and has the verb followed by the subject.

Data store:



It includes file, data base and repository.

Actor/source/sink:

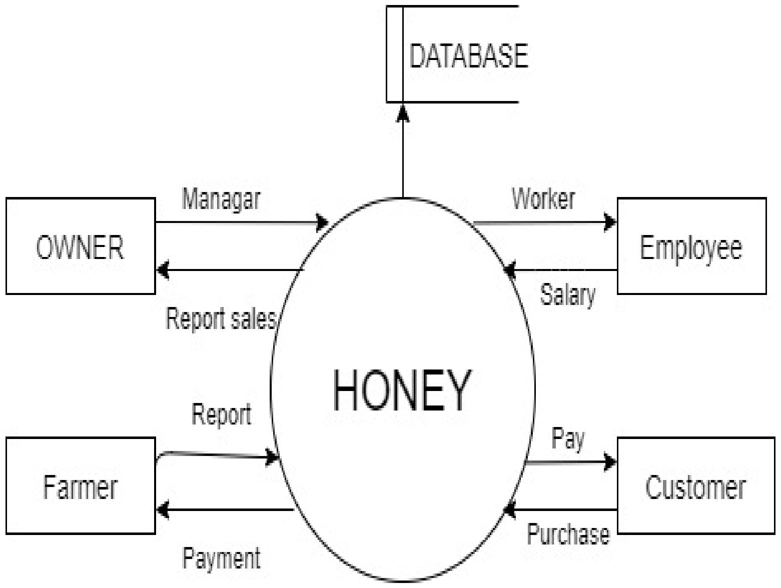
The files which are outside the system and used by the process or processes of the system.

Objectives:

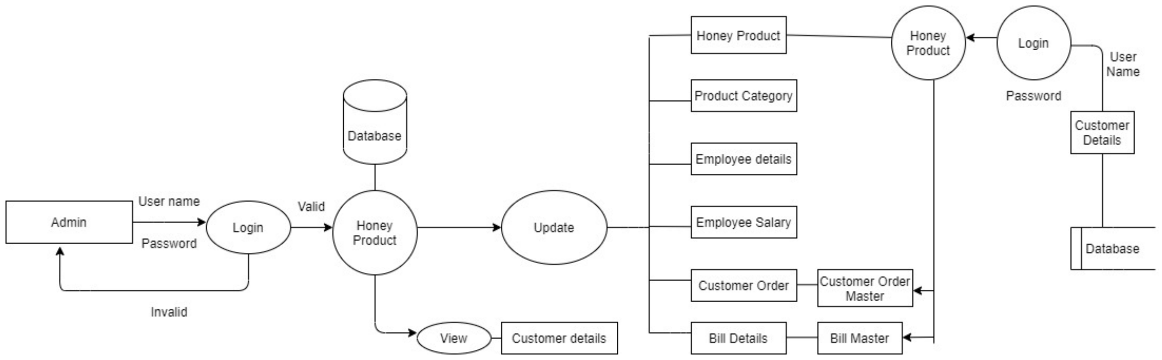
- To graphically document boundaries of a system.
- To provide hierarchy breakdown of the system.
- To show movement of information between a system and its environment.
- To document information flows within the system.
- To aid communication between users and developers.

1.5 DATA FLOW DIAGRAM:

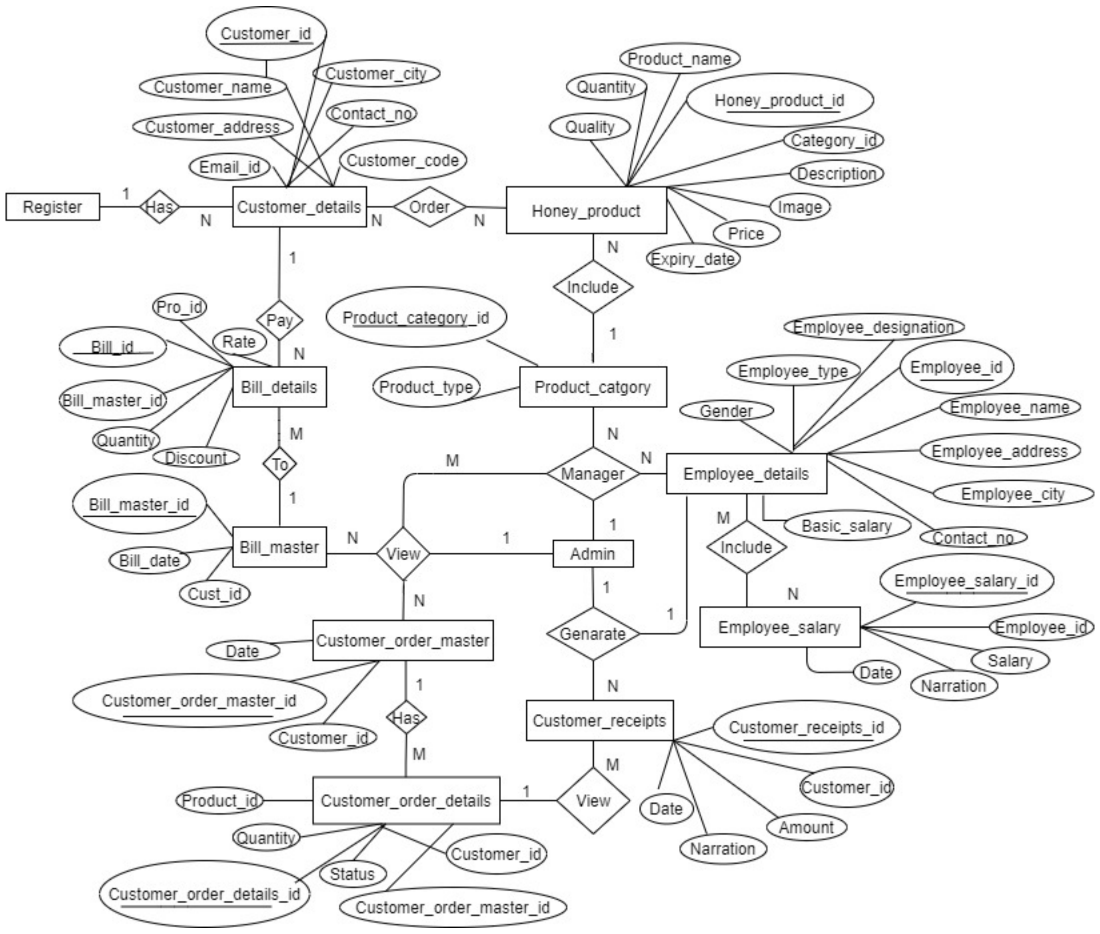
Context level diagram (zero level DFD):



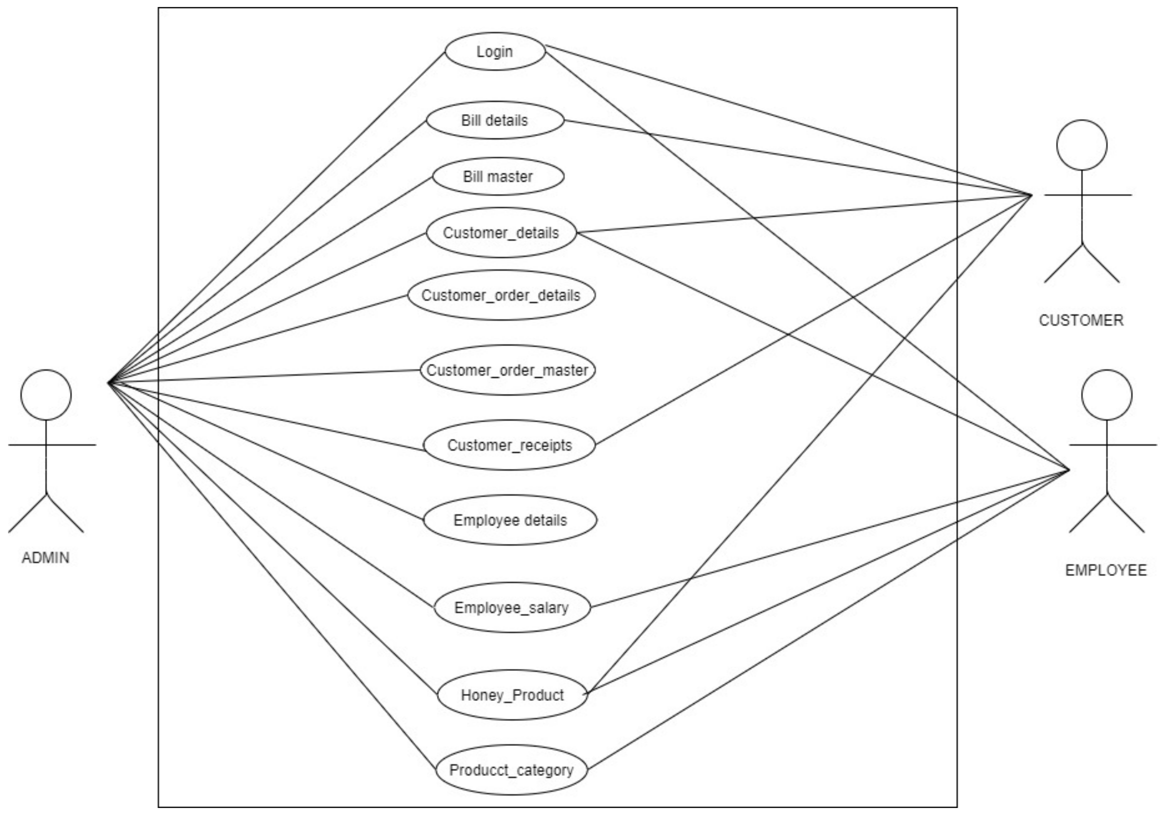
Context level diagram (one level DFD):



5.3 ER Diagram



5.4 USE-CASE DIAGRAM



6. IMPLEMENTATION

6.1 Introduction:

Implementation is the process of converting a new revised system design into operation. The objective is to put the new revised system, which has been tested into operation while holding costs, risks and personal irritation to the minimum. A critical aspect of the implementation process is to ensure that there will be no description in the function of the organization. The best methods for going control while implementation is that, any new system would be to use well planned test files for testing all new programs. Another factor to be convinced in the implementation phase in the acquisition of the hardware and software. Once the software is developed for the system and testing is carried out, it is the process of making the newly designed system fully operational and consistent in performance

Example:

```
<?php
Echo “WELCOME TO OUR PROJECT”?>
```

Speed optimization:

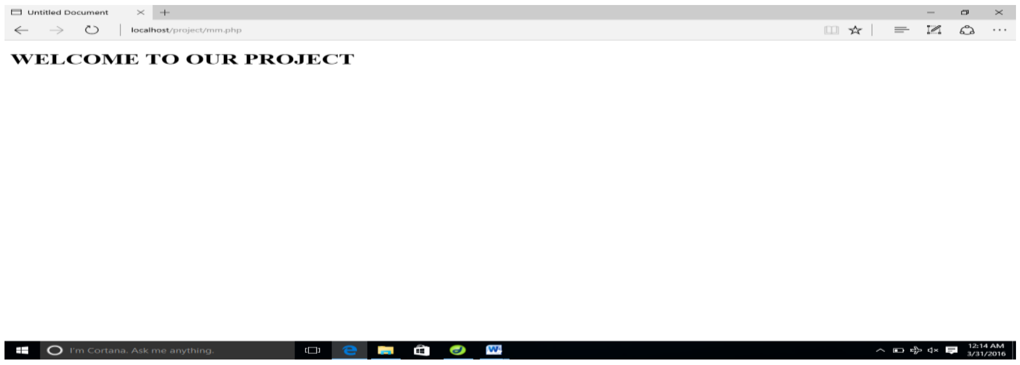
As with many scripting languages, PHP scripts are normally kept as human-readable source code, even on production web servers. In this case, PHP scripts will be compiled at runtime by the PHP engine, which increases their execution time. PHP scripts are able to be compiled before runtime using PHP compilers as with other programming languages such as C (the language PHP and its extensions are written in). Code optimizers aim to reduce the computational complexity of the compiled code by reducing its size and making other changes that can reduce the execution time with the overall goal of improving performance. The nature of the PHP compilers such that there are often opportunities for code optimization, and an example of a code optimizer is the Zend Optimizer PHP extension.

Another approach for reducing overhead for high load PHP servers is using PHP accelerators. These can offer significant performance gains by caching the compiled form of a PHP script in shared memory to avoid the overhead of parsing and compiling the code every time the script runs.

Example to display message using HTML page:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">

<title>Untitled Document</title>

</head>

<body>

<h1>WELCOME TO OUR PROJECT</h1>

</body>

</html>
```

Output:



WELCOME TO OUR PROJECT

6.2 Database Evolution

SQL was invented in the year 1960's by E. F. Cod of IBM in order to increase data integrity and reduce repetitive data. RDBMS did not appear until the late 70's when Sybase and Oracle introduced systems.

SQL server was originally a Sybase product. Microsoft bought the product outright from Sybase and by version 7.0, the version prior to 2000 all the code had been rewritten by Microsoft's programming.

Features of SQL:

- The entire SQL has been divided into 4 major categories
    1. Data Manipulation Language.
    2. Data Definition Language.
    3. Transaction control language.
    4. Data Control Lnaguage.
- It is simple English like language and uses simple commands such as SELECT, CREATE, DROP etc.
- It is not having conditional loops, variables and most of the commands are single line commands.
- To implement application logics, SQL has got extension language popularly called as PL/SQL (Procedural language of SQL).
- One of the key features of sql server is the XML support. XML has grown to be standard technology for organizations that share data on the web.

Security:

Views are basically used as a part of security, means in many organizations end user will never be given original tables and all data entry will be done with the help of views only. But the database administrator will be able to see everything because all the operations done by the different users will come to the same table.

Queries:

A query is a question or a request. With MySql, we can query a database for specific information and have a record set returned.

Create a connection to a database:

Before you can access data in a database, you must create a connection to the database. In PHP, this is done with the mysql_connect () function.

Syntax:

Mysql connect (server name, username, password);

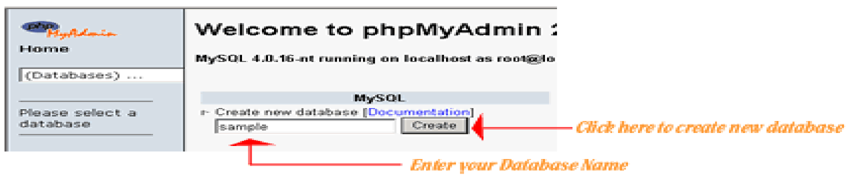
Server name: Optional Specifies the Server to connect.

Default values is localhost: 3306

```
<?php
$con=mysql_connect('localhost','root','');
mysql_select_db ('project', $con);     ?>
```

Steps to create a database in PHPMyAdmin

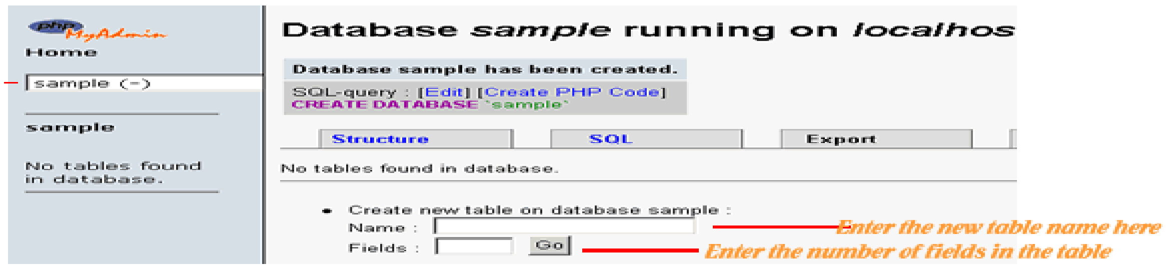
1) The following figure shows your PHPMyAdmin interface, just enter your database name and click the 'Create' button to create your database.



Welcome to phpMyAdmin 2

MySQL 4.0.16-nt running on localhost as root@lo

2) Now to create a new table enter your table name and the number of fields in the table, then click the 'Go' Button.



Database *sample* running on *localhos*

Database sample has been created.

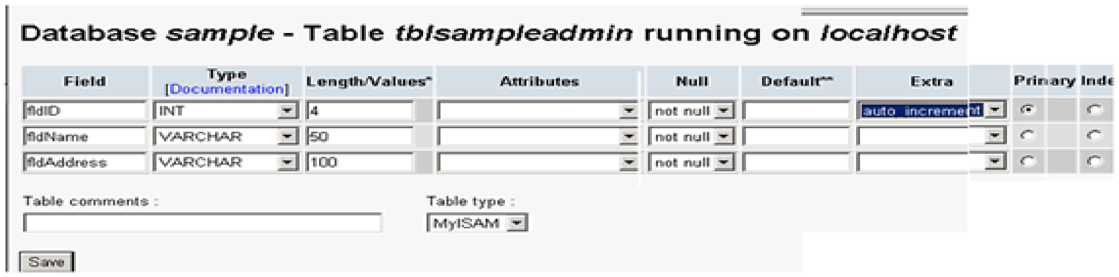
SQL-query : [Edit] [Create PHP Code]
CREATE DATABASE `sample`

No tables found in database.

• Create new table on database sample :
Name :
Fields :

phpMyAdmin
Home
sample (-)
sample
No tables found in database.
Structure
SQL
Export
Go
Enter the new table name here
Enter the number of fields in the table

3)The next step is to create the fields, just enter values for each field name, type, length of the field, null option and mention whether it is a primary key or not. Then click the 'Save' button to complete your table creation.



Database *sample* - Table *tblsampleadmin* running on *localhost*

Field	Type [Documentation]	Length/Values*	Attributes	Null	Default**	Extra	Primary	Inde
fldID	INT	4		not null		auto_increment	●	○
fldName	VARCHAR	50		not null			○	○
fldAddress	VARCHAR	100		not null			○	○

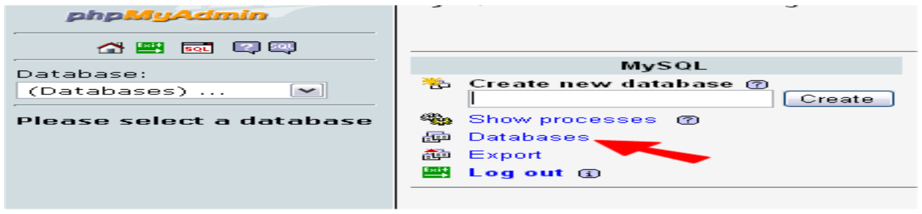
Table comments :

Table type : MyISAM

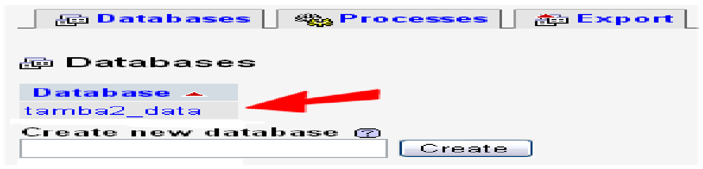
4) The following figure is displayed upon successful creation of your table

Steps to Drop Table in PHPMyAdmin:

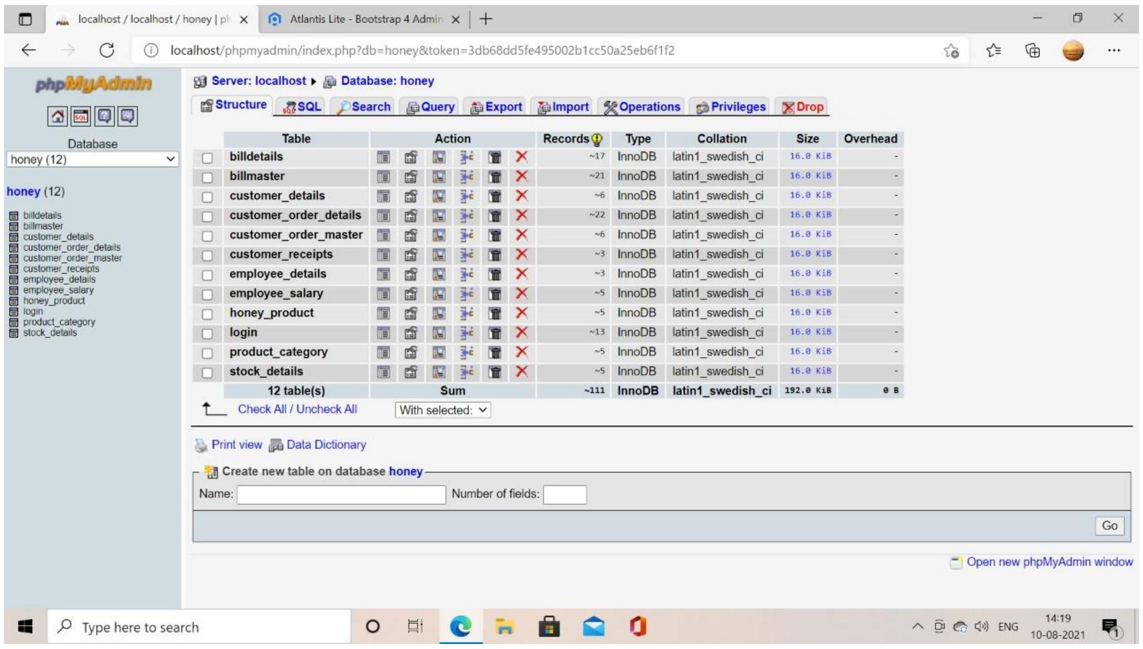
Login to phpmyadmin. Click 'databases'



List of your databases will appear. Click the one that is your WordPress database.



Note the size of the 'wp_bad_behaviour_log' table - this is one to be emptied in this example.



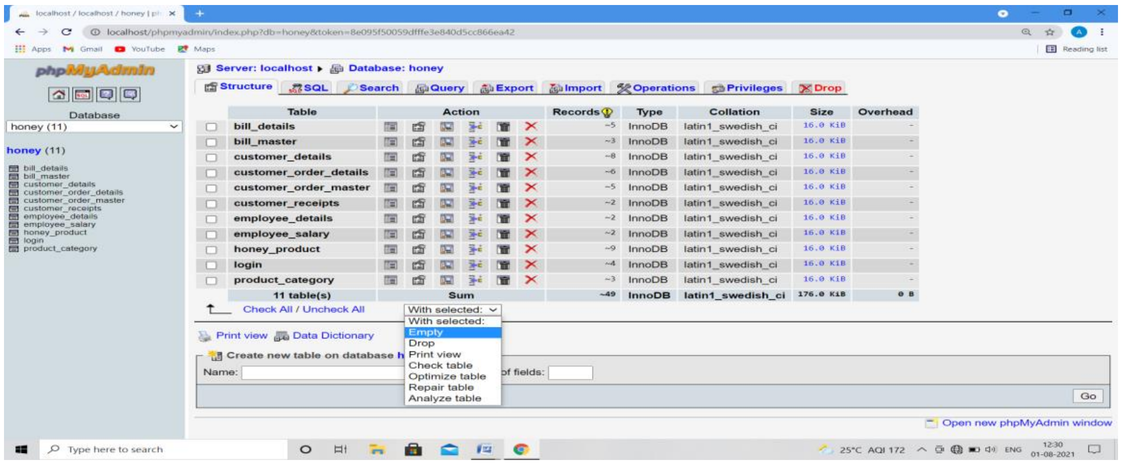
localhost/phpmyadmin/index.php?db=honey&token=3db68dd5fe495002b1cc50a25eb6f1f2

Server: localhost ▸ Database: honey

	Table	Action	Records	Type	Collation	Size	Overhead
☐	billdetails		~17	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	billmaster		~21	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_details		~6	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_order_details		~22	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_order_master		~6	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_receipts		~3	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	employee_details		~3	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	employee_salary		~5	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	honey_product		~5	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	login		~13	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	product_category		~5	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	stock_details		~5	InnoDB	latin1_swedish_ci	16.0 KiB	-
	12 table(s)	Sum	~111	InnoDB	latin1_swedish_ci	192.0 KiB	0 B

Now select the box to the left of the table you wish to empty.

Note: Your table may have a different name, and unless you have been told, do NOT empty a table that is used by the Word Press core.



localhost/phpmyadmin/index.php?db=honey&token=8e095f50059dfffe3e840d5cc866ea42

Server: localhost ▸ Database: honey

	Table	Action	Records	Type	Collation	Size	Overhead
☐	bill_details		~5	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	bill_master		~3	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_details		~8	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_order_details		~6	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_order_master		~5	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	customer_receipts		~2	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	employee_details		~2	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	employee_salary		~2	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	honey_product		~9	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	login		~4	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	product_category		~3	InnoDB	latin1_swedish_ci	16.0 KiB	-
	11 table(s)	Sum	~49	InnoDB	latin1_swedish_ci	176.0 KiB	0 B

From the drop-down menu, highlight and click the 'Empty' option.

You will now get a confirmation screen.

This is your last chance to check - there is no 'UNDO' function here !

You will now get a confirmation screen.

This is your last chance to check - there is no 'UNDO' function here !

Click 'Yes' and you will be returned to viewing all the tables in your install.

And your table has been cleared out.

If you needed to Drop a table, follow exactly the same, but select 'Drop' from the menu.

7.DATABASE TABLES

7.1 Tables Used In Our Project:

Table structure for BILL DETAILS table:

Primary Key: Bill_details_ID

Column	Type	Null	Constraint
Bill_id	int(11)	No	Primary Key
Bill_master_id	int(11)	No	
Pro_id	int(11)	No	
Quantity	int(11)	No	
Rate	double	No	
Discount	double	No	

Table structure for BILL MASTER table:

Primary Key: Bill_Master_ID

Column	**Type**	**Null**	**Constraint**
Bill_master_id	int(11)	No	Primary Key
Bill_date	date	No	
Cust_id	int(11)	No	

Table structure for CUSTOMER DETAILS table:

Primary Key: Customer_**ID**

Column	**Type**	**Null**	**Constraint**
Customer_**ID**	int(11)	No	Primary Key
Customer_name	varchar(50)	No	
Customer _Address	varchar(200)	No	
Customer_ City	varchar(50)	No	
Contact_no	varchar(12)	No	
Email_ID	varchar(100)	No	
Customer _code	varchar(50)	No	

Table structure for CUSTOMER_ORDER_ DETAILS table:

Primary Key: Customer_order_details_**ID**

Column	**Type**	**Null**	**Constraint**
Customer_order_details_**ID**	int(11)	No	Primary Key
Customer ID	int(11)	No	
Customer_order_master_details_**ID**	int(11)	No	
Product_id	int(11)	Yes	
Quantity	int(11)	No	
Status	int(11)	No	

Table structure for CUSTOMER_ORDER_MASTER table:

Primary Key: Customer_order_master_**ID**

Column	**Type**	**Null**	**Constraint**
Customer_order_master_**ID**	int(11)	No	Primary Key
Date	varchar(20)	No	
Customer_id	Int(11)	No	

Table structure for CUSTOMER_RECEIPTS table:

Primary Key: Customer_receipts_id

Column	**Type**	**Null**	**Constraint**
Customer_receipts_id	int(11)	No	Primary Key
Customer_id	int(11)	No	

Amount	int(11)	No	
Narration	int(11)	No	
Rate	double	No	

Table structure for EMPLOYEE DETAILS table:

Primary Key: Employee_ID

Column	**Type**	**Null**	**Constraint**
Employee_ID	int(11)	No	Primary Key
Employee _Name	varchar(100)	No	
Employee_address	varchar(500)	No	
Employee_city	varchar(100)	No	
Gender	enum(‘male’,’female)	No	
Contact_No	varchar(12)	No	
Employee_type	varchar(100)	No	
Employee_designation	varchar(200)	No	
Basic_salary	Double	Yes	

Table structure for EMPLOYEE_SALARY table:

Primary Key: Employee_salary_id

Column	**Type**	**Null**	**Constraint**
Employee_salary_id	int(11)	No	Primary Key

Employee_id	varchar(200)	No	
Salary	int(200)	No	
Date	varchar(200)	No	
Naration	varchar (200)	No	

Table structure for HONEY PRODUCT table:

Primary Key: honey_product_id

Column	**Type**	**Null**	**Constraint**
honey_product_id	int(11)	No	Primary Key
Category_ID	varchar(200)	No	Foreign Key
Product_name	varchar(200)	No	
Description	varchar(200)	No	
Image	varchar(200)	No	
Price	varchar(200)	No	
Quantity	varchar(200)	No	
Quality	varchar(200)	No	
Expiry_date	varchar(200)	No	

Table structure for PRODUCT CATEGORY table:

Primary Key: product_category_id

Column	Type	Null	Constraint
product_category_id	int(20)	No	Primary Key
Product_type	varchar(200)	No	

8.SCREEN SHOT

Fig 1. A VIEW OF HOMEPAGE

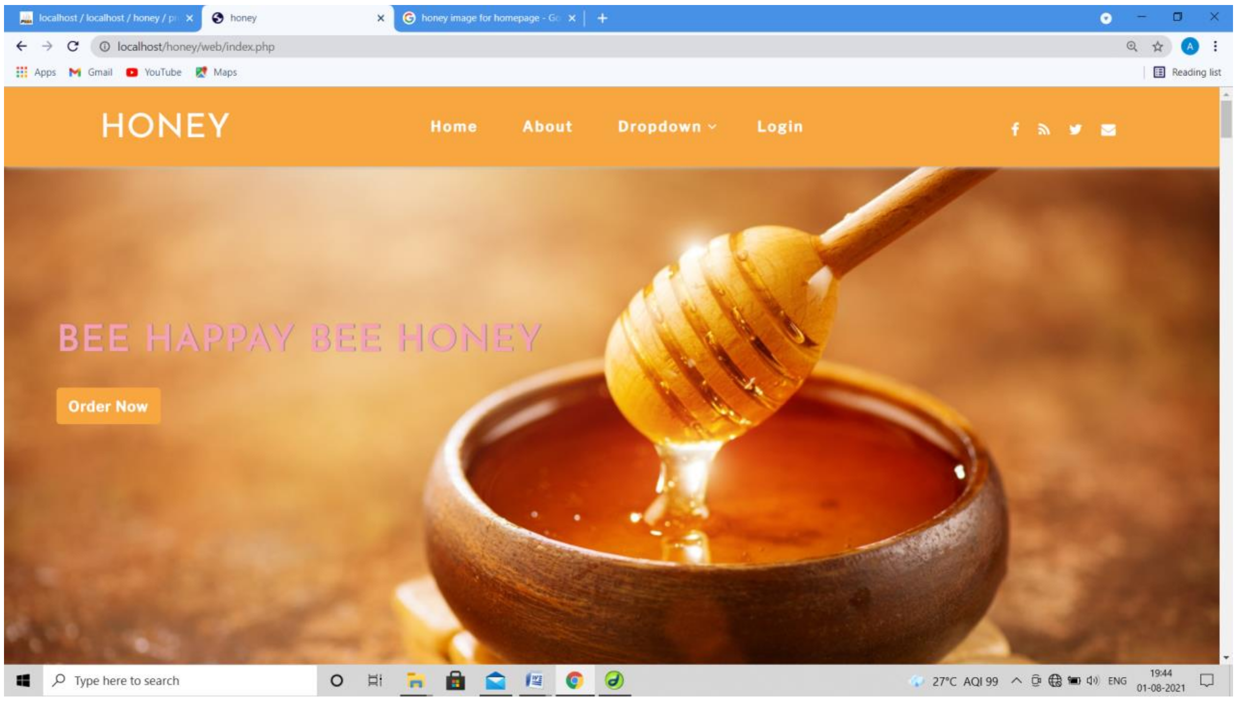


Fig 2. A VIEW OF LOGIN PAGE

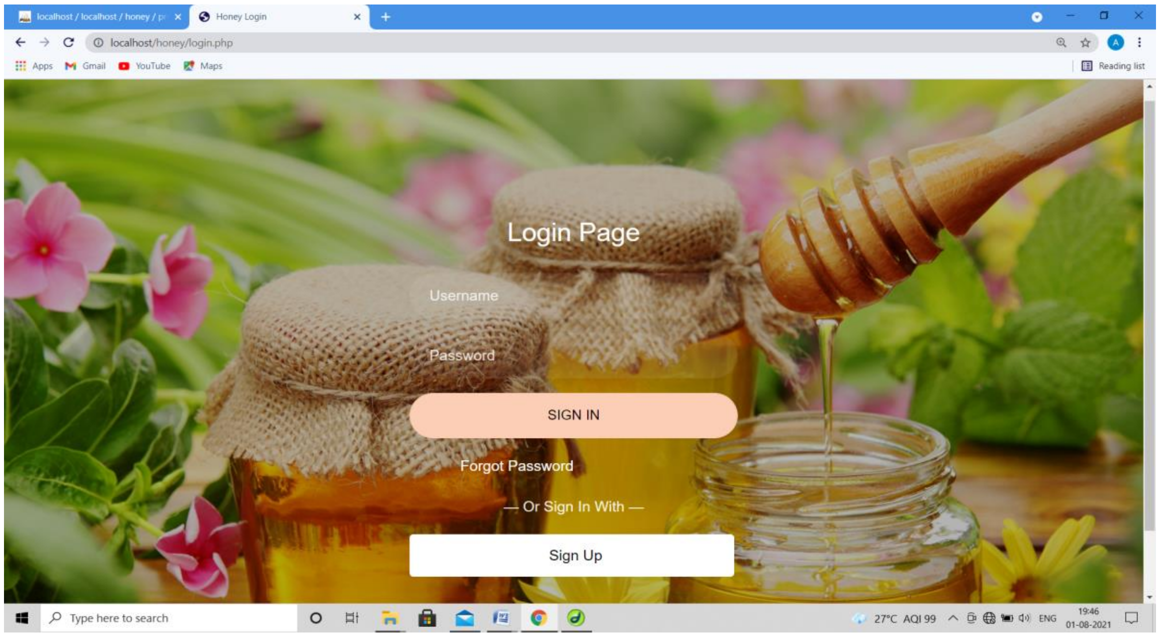


Fig 3. A VIEW OF REGISTRATION PAGE

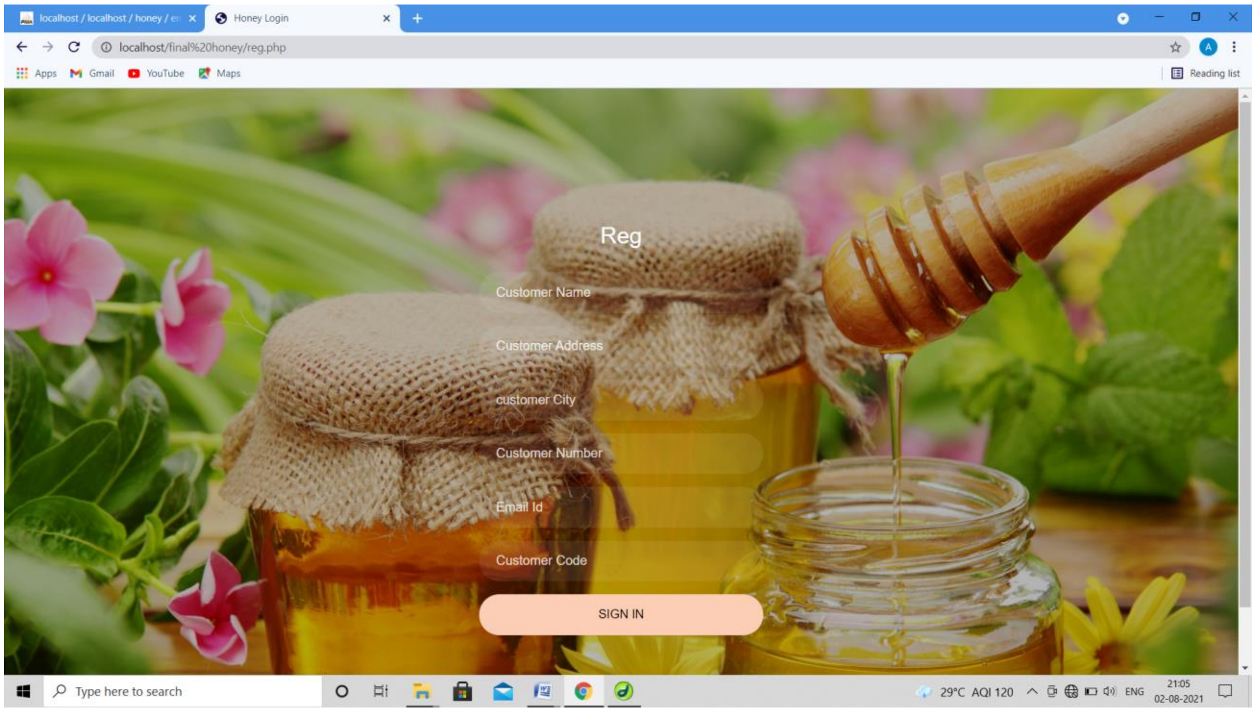
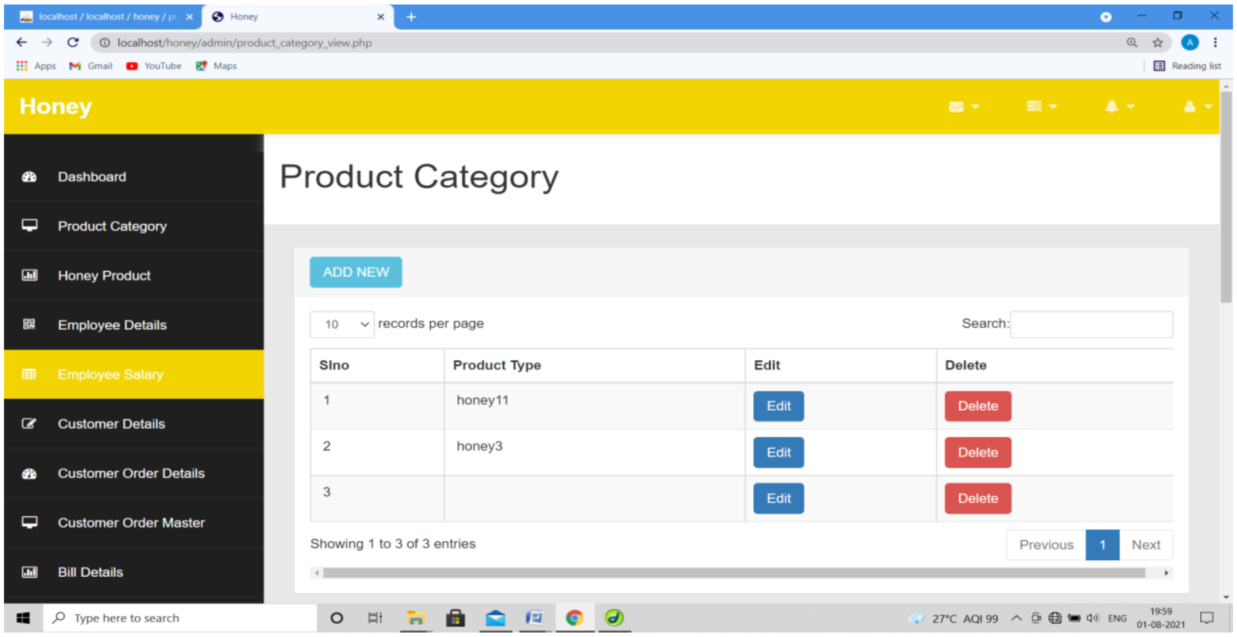


Fig 4. A VIEW OF PRODUCT CATEGORY PAGE

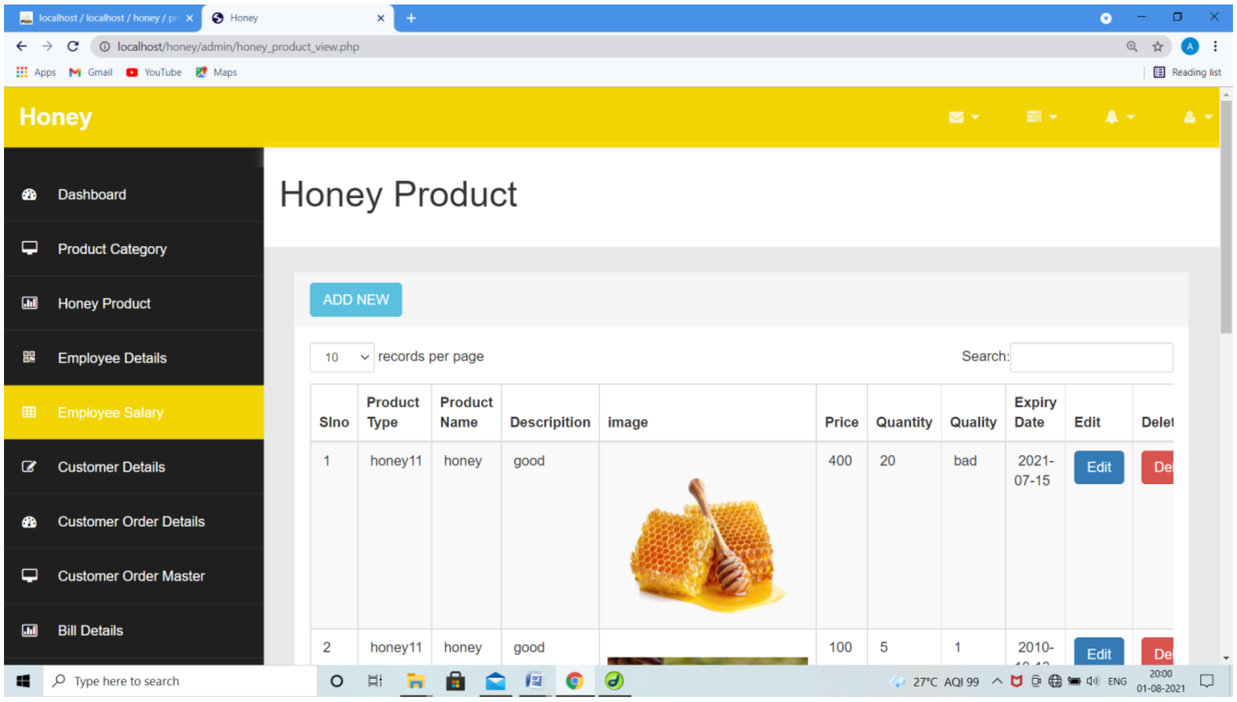


Product Category

Slno	Product Type	Edit	Delete
1	honey11	Edit	Delete
2	honey3	Edit	Delete
3		Edit	Delete

Showing 1 to 3 of 3 entries

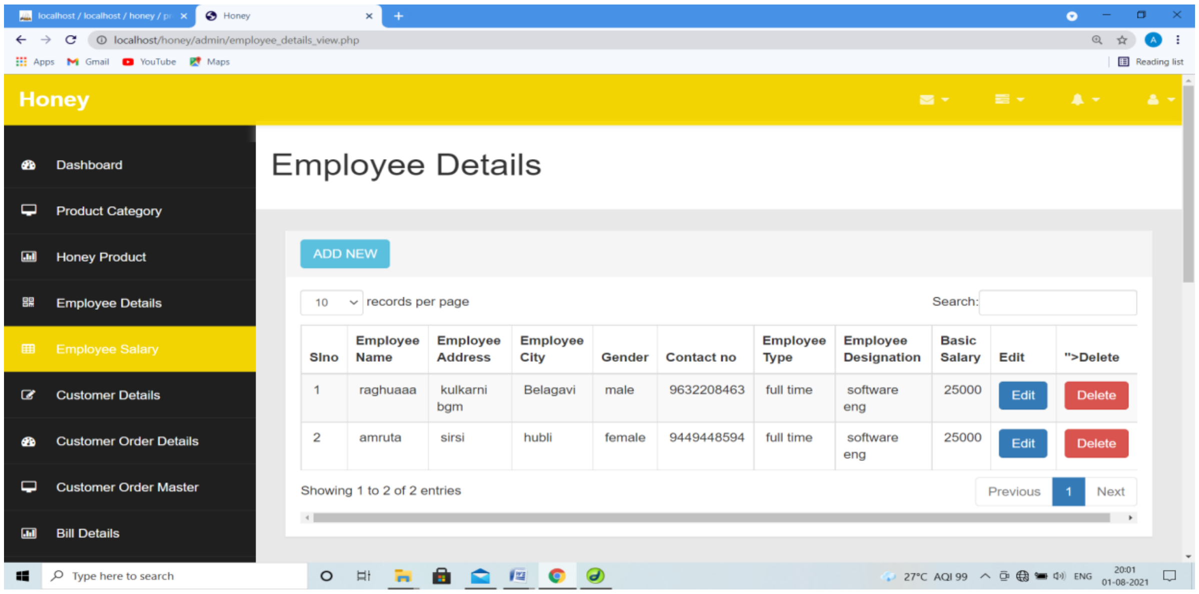
Fig 5. A VIEW OF HONEY PRODUCT PAGE



Honey Product

Slno	Product Type	Product Name	Descripition	image	Price	Quantity	Quality	Expiry Date	Edit	Delet
1	honey11	honey	good		400	20	bad	2021-07-15	Edit	De
2	honey11	honey	good		100	5	1	2010-	Edit	De

Fig 6. A VIEW OF EMPLOYEE DETAILS PAGE



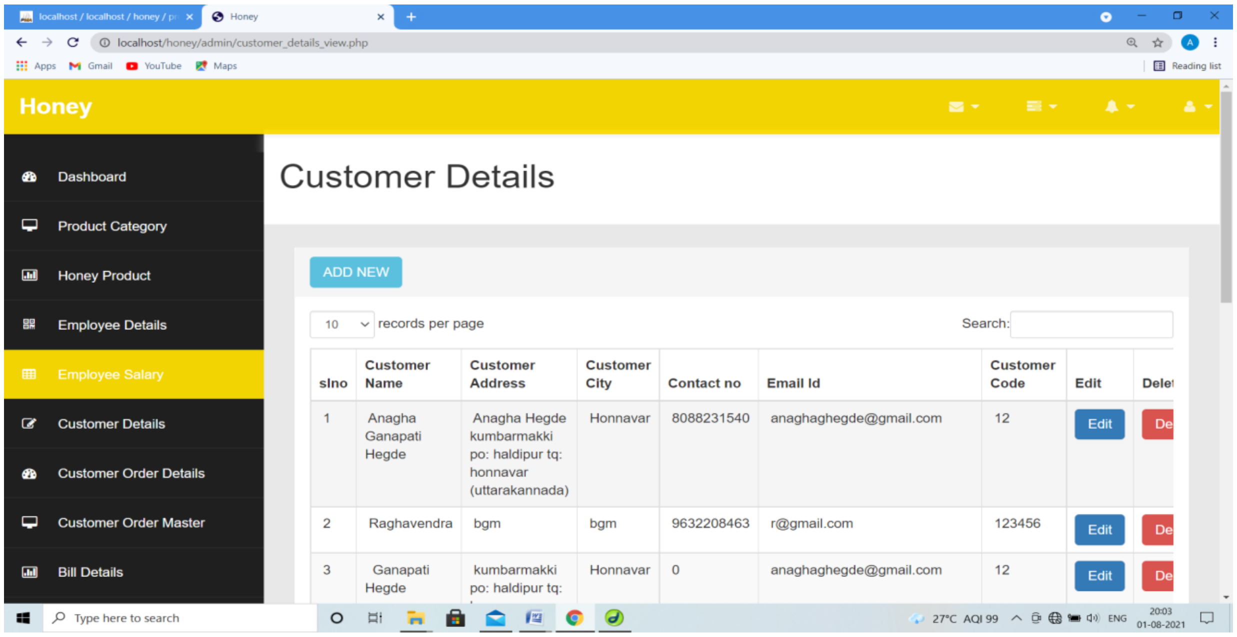
Employee Details

10 records per page Search:

Slno	Employee Name	Employee Address	Employee City	Gender	Contact no	Employee Type	Employee Designation	Basic Salary	Edit	">Delete
1	raghuaaa	kulkarni bgm	Belagavi	male	9632208463	full time	software eng	25000	Edit	Delete
2	amruta	sirsi	hubli	female	9449448594	full time	software eng	25000	Edit	Delete

Showing 1 to 2 of 2 entries Previous 1 Next

Fig 7. A VIEW OF CUSTOMER DETAILS PAGE

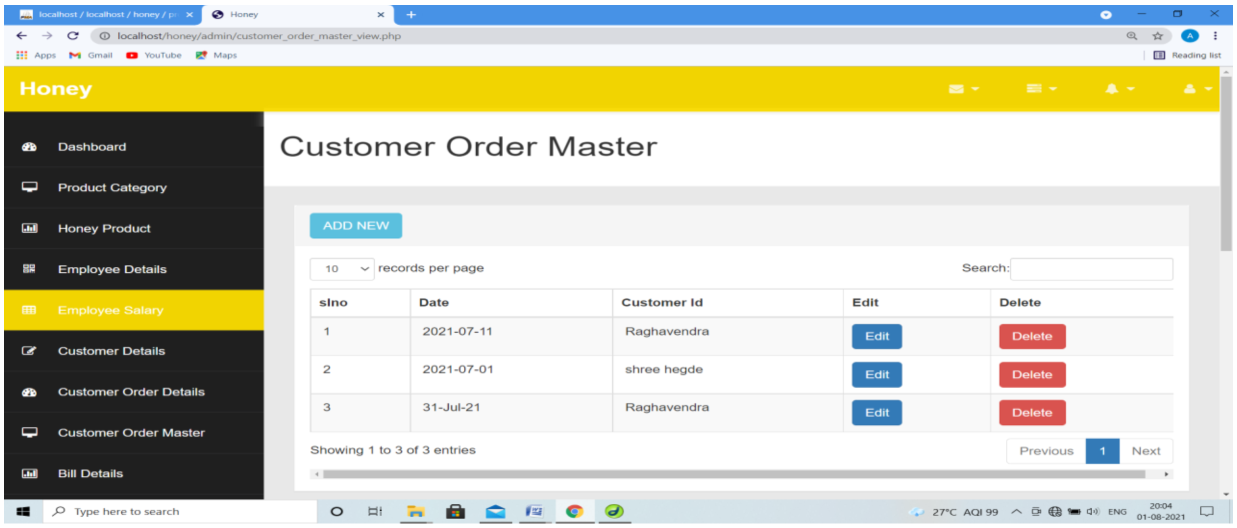


Customer Details

10 records per page Search:

slno	Customer Name	Customer Address	Customer City	Contact no	Email Id	Customer Code	Edit	Delet
1	Anagha Ganapati Hegde	Anagha Hegde kumbarmakki po: haldipur tq: honnavar (uttarakannada)	Honnavar	8088231540	anaghaghegde@gmail.com	12	Edit	De
2	Raghavendra	bgm	bgm	9632208463	r@gmail.com	123456	Edit	De
3	Ganapati Hegde	kumbarmakki po: haldipur tq:	Honnavar	0	anaghaghegde@gmail.com	12	Edit	De

Fig 8. A VIEW OF CUSTOMER ODRER MASTER PAGE



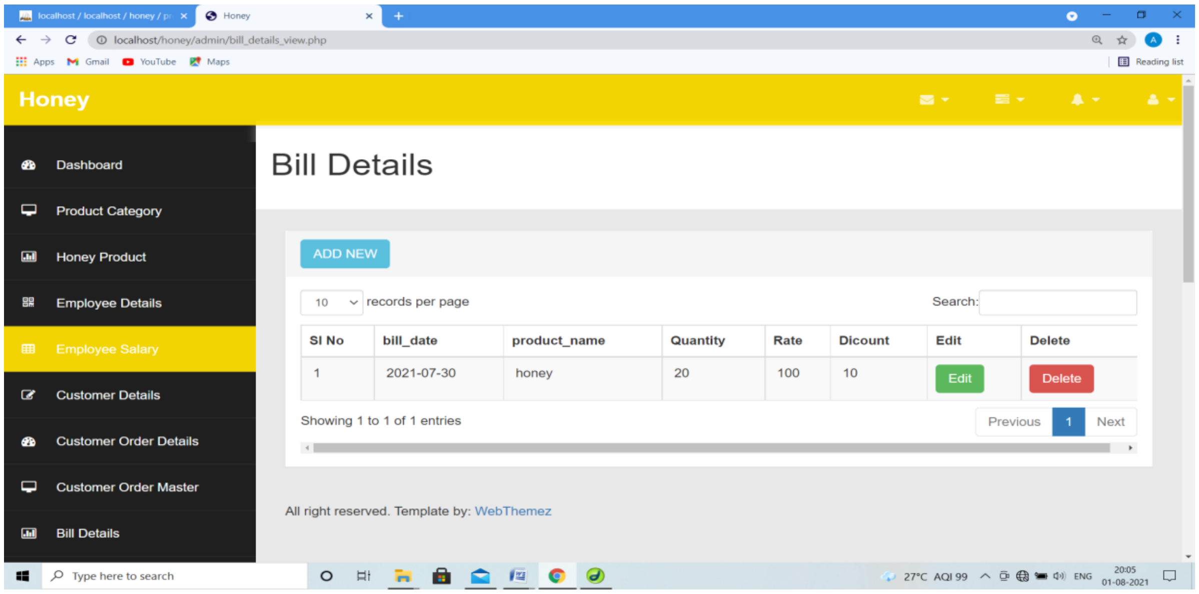
Customer Order Master

10 records per page Search:

slno	Date	Customer Id	Edit	Delete
1	2021-07-11	Raghavendra	Edit	Delete
2	2021-07-01	shree hegde	Edit	Delete
3	31-Jul-21	Raghavendra	Edit	Delete

Showing 1 to 3 of 3 entries Previous 1 Next

Fig 9. A VIEW OF BILL DETAILS PAGE



Bill Details

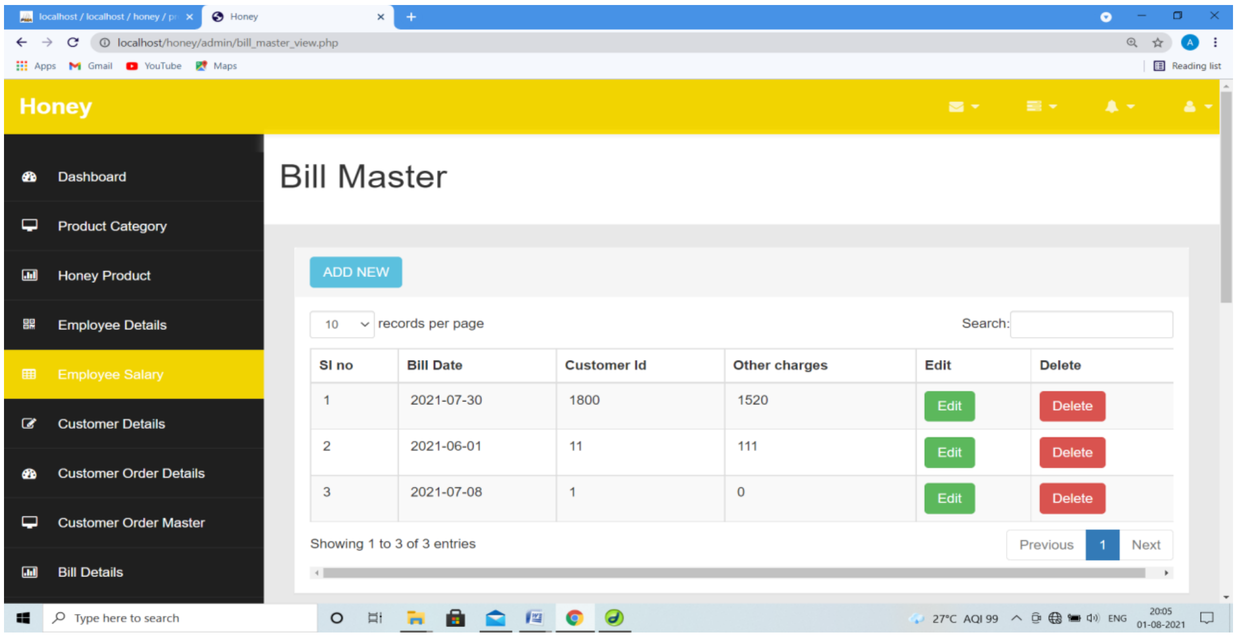
10 records per page Search:

Sl No	bill_date	product_name	Quantity	Rate	Dicount	Edit	Delete
1	2021-07-30	honey	20	100	10	Edit	Delete

Showing 1 to 1 of 1 entries Previous 1 Next

All right reserved. Template by: WebThemez

Fig 10. A VIEW OF BILL MASTER PAGE



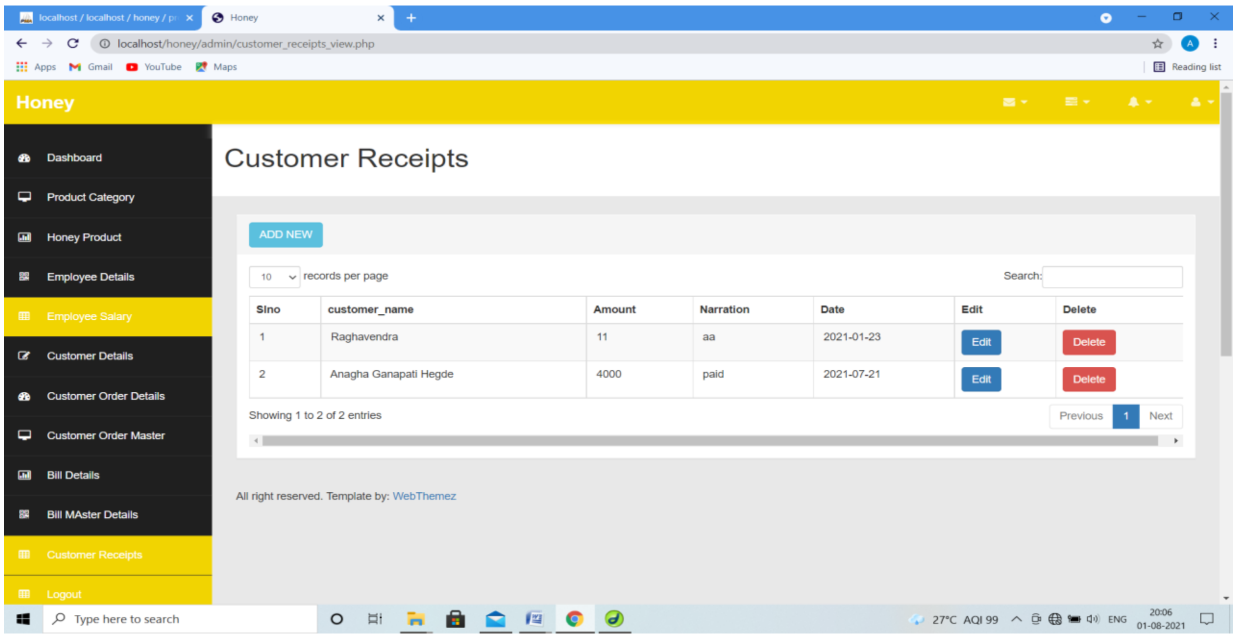
Bill Master

10 records per page

Sl no	Bill Date	Customer Id	Other charges	Edit	Delete
1	2021-07-30	1800	1520	Edit	Delete
2	2021-06-01	11	111	Edit	Delete
3	2021-07-08	1	0	Edit	Delete

Showing 1 to 3 of 3 entries

Fig 11. A VIEW OF CUSTOMER RECEIPTS PAGE



Customer Receipts

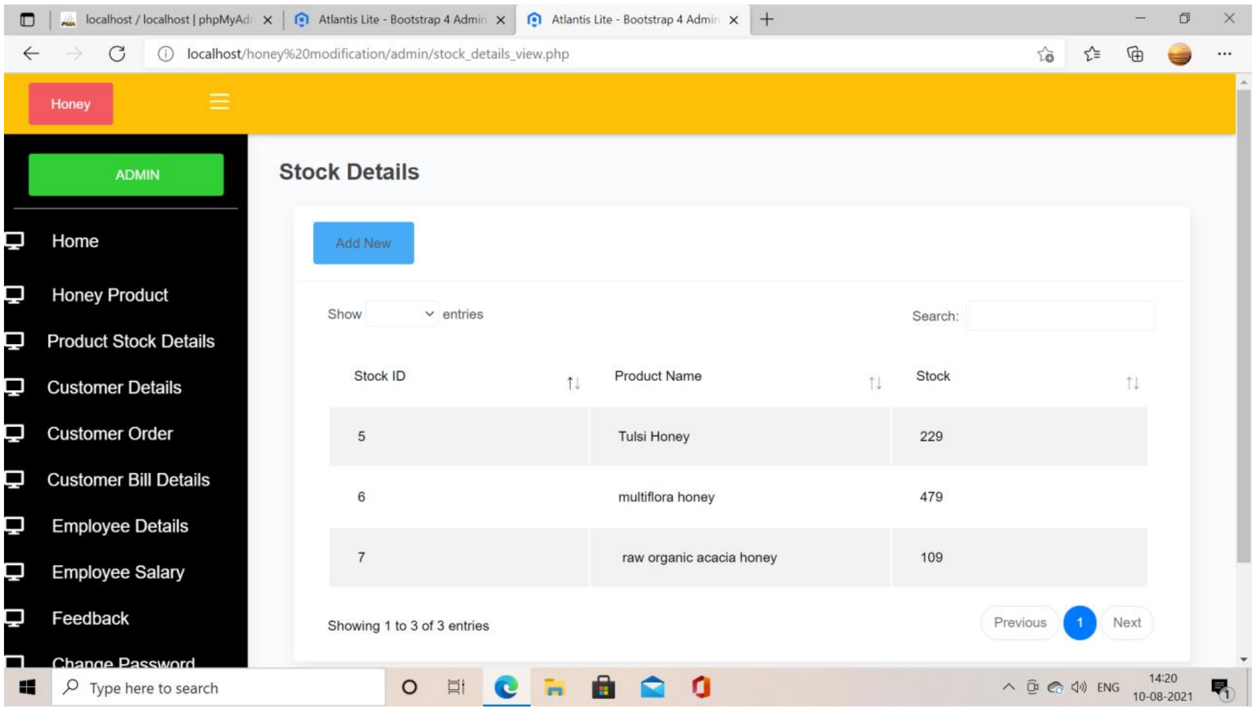
10 records per page

Slno	customer_name	Amount	Narration	Date	Edit	Delete
1	Raghavendra	11	aa	2021-01-23	Edit	Delete
2	Anagha Ganapati Hegde	4000	paid	2021-07-21	Edit	Delete

Showing 1 to 2 of 2 entries

All right reserved. Template by: WebThemez

Fig 12. A VIEW OF EMPLOYEE DETAILS PAGE



Stock Details

Stock ID	Product Name	Stock
5	Tulsi Honey	229
6	multiflora honey	479
7	raw organic acacia honey	109

Showing 1 to 3 of 3 entries

9.SOURCE CODE

HONEY PRODUCT TABLE

Form:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<?php include('metadata.php');?>
<?php include('header.php');?>
<?php include('sidebar.php');?>
</head>
<body>
<!--/. NAV TOP   -->
<!-- /. NAV SIDE   -->
<div id="page-wrapper" >
<div class="header">
<h1 class="page-header">
Honey Product
</h1>
</div>
```

```
<div id="page-inner">
<div class="row">
<div class="col-lg-12">
<div class="panel panel-default">
<div class="panel-heading">
</div>
<div class="panel-body">
<div class="row">
<div class="col-lg-6">
<!-- /.col-lg-6 (nested) -->
<?php include("cal.php");?>
<form name="formID" id="formID" method="post"   enctype="multipart/form-data"
action="honey_product_insert.php">
<table width="554" height="543" border="0" align="center">
<tr>
<td>Category ID </td>
<td><select name="category_id" id="category_id" class="validate[required] form-control" />
<option value="">Select Product Category Name</option>
<?php
include("../db connect/db_connect.php");
$slno=1;
$sql="select * from product_category";
$res=mysqli_query($conn,$sql);
while($row=mysqli_fetch_array($res))
{
?>
<option value="<?php echo $row['product_category_id'];?>"><?php echo $row['product_type'];?></option>
<?php
}
?>
</select></td>
</tr>
<tr>
<td>Product Name</td>
<td><input name="product_name" type="text" id="product_name"
class="validate[required,custom[onlyLetter]] form-control" /></td>
</tr>
<tr>
<td>Description</td>
<td><input name="description" type="text" id="description" class="validate[required] form-control" /></td>
</tr>
<tr>
<td>Image</td>
<td><input name="image" type="file" id="image" class="validate[required] form-control"></td>
</tr>
<tr>
<td>Price</td>
<td><input name="price" type="text" id="price" class="validate[required] form-control" /></td>
</tr>
<tr>
<td>Quantity</td>
<td><input name="quantity" type="text" id="quantity" class="validate[required] form-control" /></td>
```

```
</tr>
<tr>
<td>Quality</td>
<td><input name="quality" type="text" id="quality" class="validate[required] form-control" /></td>
</tr>
<tr>
<td>Expiry Date</td>
<td>
<?php
//$date_default = "";
$date_default =date('Y-m-d');
// $year=date('Y');
$syear=date('Y-m-d');
$myCalendar = new tc_calendar("Expiry_date", true, true);
$myCalendar->setIcon("calendar/images/iconCalendar.gif");
$myCalendar->setDate(date('d', strtotime($date_default))
, date('m', strtotime($date_default)), date('Y', strtotime($date_default)));
$myCalendar->dateAllow("$syear","2030-01-01");
$myCalendar->setPath("calendar/");
$myCalendar->setYearInterval(date('Y'),2030);
$myCalendar->setAlignment('left', 'bottom');
$myCalendar->writeScript();
?>
</td>
</tr>
<tr>
<td colspan="2"><div align="center">
<input type="submit" name="Submit" value="Submit" class="btn btn-info">
<input type="reset" name="Reset" value="Reset" class="btn btn-danger">
</div></td>
</tr>
</table>
<p>&nbsp;</p>
</form>
</div>
<!-- /.col-lg-6 (nested) -->
</div>
<!-- /.row (nested) -->
</div>
<!-- /.panel-body -->
</div>
<!-- /.panel -->
</div>
<!-- /.col-lg-12 -->
</div>
<?php include('footer.php');?>
<?php include("val.php"); ?>
</body>
</html>
```

Insert:

```
<?php

include("../db connect/db_connect.php");

$category_id=$_POST['category_id'];

$product_name=$_POST['product_name'];

$description=$_POST['description'];

//$image=$_POST['image'];

$price=$_POST['price'];

$quantity=$_POST['quantity'];

$quality=$_POST['quality'];

$expiry_date=$_POST['Expiry_date'];

$image=$_FILES['image']['name'];

$tmp_location=$_FILES['image']['tmp_name'];

$target="../img/".$image;

move_uploaded_file($tmp_location,$target);

$sql="insert into honey_product
values(null,'$category_id','$product_name','$description','$image','$price','$quantity','$quality','$expiry_date')";

mysqli_query($conn,$sql);?>

<script>

alert("honey product values Are Inserted..!!");

document.location="honey_product_form.php";

</script>
```

View:

```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
```

```
 <?php include('metadata.php');?>

 <?php include('header.php');?>

 <?php include('sidebar.php');?>

</head>

<body>

 <!--/. NAV TOP   -->

 <!-- /. NAV SIDE   -->

 <div id="page-wrapper" >

 <div class="header">

 <h1 class="page-header">
```

Honey Product

```
   </h1>

   <!-- <ol class="breadcrumb">

     <li><a href="#">Home</a></li>

     <li><a href="#">Tables</a></li>

     <li class="active">Data</li>

     </ol>   -->

     </div>

     <div id="page-inner">

     <div class="row">

     <div class="col-md-12">

     <!-- Advanced Tables -->

     <div class="panel panel-default">

     <div class="panel-heading">

     <a href="honey_product_form.php" class="btn btn-info">ADD NEW</a>

     </div>

     <div class="panel-body">
     <div class="table-responsive">
     <table class="table table-striped table-bordered table-hover" id="dataTables-example">
     <thead>
     <tr>
     <th>Slno</th>
     <th>Product Type </th>
     <th>Product Name </th>
     <th>Descripition</th>
     <th>image</th>
     <th>Price</th>
     <th>Quantity</th>
     <th>Quality</th>
     <th>Expiry Date </th>
     <th>Edit</th>
     <th>Delete</th>
     </tr>
     </thead>
   <tbody>
   <?php
 include("../db connect/db_connect.php");
 $slno=1;
 $sql="select * from   honey_product hd, product_category pc where hd.category_id=pc.product_category_id";
 $res=mysqli_query($conn,$sql);
 while($row=mysqli_fetch_array($res))
 {
 ?>
 <tr>
 <td>&nbsp;<?php echo $slno++; ?></td>
```

```
  <td>&nbsp;<?php echo $row['product_type'];?></td>

  <td>&nbsp;<?php echo $row['product_name'];?></td>

  <td>&nbsp;<?php echo $row['description'];?></td>

  <td>&nbsp;<img src="../img/<?php echo $row['image'];?>" width="200" height="150"></td>

  <td>&nbsp;<?php echo $row['price'];?></td>

  <td>&nbsp;<?php echo $row['quantity'];?></td>

  <td>&nbsp;<?php echo $row['quality'];?></td>

  <td>&nbsp;<?php echo $row['expiry_date'];?></td>

  <td><a href="honey_product_edit.php?honey_product_id=<?php echo $row['honey_product_id'];?>" class="btn
btn-primary">Edit</a></td>

   <td><a href="honey_product_delete.php?honey_product_id=<?php echo $row['honey_product_id'];?>"
class="btn btn-danger">Delete</a></td>

 </tr>

 <?php

}

?>

</tbody>

 </table>

 </div>

 </div>

 </div>

 <!--End Advanced Tables -->

 </div>

 </div>

 <!-- /. ROW   -->

 <?php include('footer.php');?>

  </body>

</html>
```

Delete:

```
<?php

include("../db connect/db_connect.php");

$honey_product_id=$_REQUEST['honey_product_id'];

$sql="delete from honey_product where honey_product_id='$honey_product_id'";

mysqli_query($conn,$sql);

?>

<script>

alert("Values are Deleted..!!");

document.location="honey_product_view.php";

</script>
```

Edit:

```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<?php include('metadata.php');?>

 <?php include('header.php');?>

 <?php include('sidebar.php');?>

</head>

<body>

 <!--/. NAV TOP   -->

 <!-- /. NAV SIDE   -->

 <div id="page-wrapper" >

  <div class="header">

  <h1 class="page-header">

   Honey Product

   </h1>

  </div>

 <div id="page-inner">
 <div class="row">

 <div class="col-lg-12">

 <div class="panel panel-default">

 <div class="panel-heading">

 </div>

 <div class="panel-body">

 <div class="row">

 <div class="col-lg-6">

 <!-- /.col-lg-6 (nested) -->

 <?php

 include("../db connect/db_connect.php");

 $honey_product_id=$_REQUEST['honey_product_id'];

 $sql="select * from honey_product where honey_product_id='$honey_product_id'";

 $res=mysqli_query($conn,$sql);

 $row=mysqli_fetch_array($res);

?>

<?php include("val.php"); ?>

<form name="formID" id="formID"   method="post" enctype="multipart/form-data"
action="honey_product_update.php">

<input type="hidden" name="honey_product_id" value="<?php echo $row['honey_product_id'];?>"
class="validate[required] form-control" />

 <table width="556" height="607" border="0" align="center">

 <tr>

   <td>product_type</td>

   <td><select name="category_id" id="category_id" class="validate[required] form-control" />

   <option value="">Select Product Category Name</option>

     <?php

     $sql2="select * from product_category";

     $res2=mysqli_query($conn,$sql2);
```

```
    while($row2=mysqli_fetch_array($res2))

    {

    ?>

    <option value="<?php echo $row2['product_category_id'];?>" <?php
if($row2['product_category_id']==$row['category_id']) { ?> selected<?php } ?>><?php echo
$row2['product_type'];?></option>

 <?php

 }

 ?>

 </select></td>

 </tr>

 <tr>

 <td>Product_Name</td>

 <td><input name="product_name" type="text" id="product_name" value="<?php echo $row['product_name'];
?>" class="validate[required,custom[onlyLetter]] form-control" /></td>

 </tr>

 <tr>

 <td>Description</td>

 <td><input name="description" type="text" id="description" value="<?php echo $row['description']; ?>"
class="validate[required] form-control" /></td>

</tr>

<tr>

 <td>Image</td>

 <td>&nbsp;<img src="../img/<?php echo $row['image'];?>" width="200" height="150">

 <input name="image" type="file" id="image" class="validate[required]"></td>

 </tr>

  <tr>

  <td>Price</td>

 <td><input name="price" type="text" id="price" value="<?php echo $row['price']; ?>" class="validate[required]
form-control" /></td>
```

```
 </tr>

 <tr>

 <td>Quantity</td>

<td><input name="quantity" type="text" id="quantity" value="<?php echo $row['quantity']; ?>"
class="validate[required] form-control" /></td>

 </tr>

 <tr>

 <td>Quality</td>

 <td><input name="quality" type="text" id="quality" value="<?php echo $row['quality']; ?>"
class="validate[required] form-control" /></td>

</tr>

<tr>

 <td>Expiry_Date</td>

<td><input name="Expiry_date" type="Date" id="Expiry_date" value="<?php echo $row['expiry_date']; ?>"
class="validate[required,custom[date]] form-control" /></td>

 </tr>

 <tr>

 <td colspan="2"><div align="center">

  <input type="submit" name="Submit" value="Submit" class="btn btn-success">

  <input type="reset" name="Reset" value="Reset" class="btn btn-danger">

  </div></td>

  </tr>

  </table>

  <p>&nbsp;</p>

  </form>

  </div>

  <!-- /.col-lg-6 (nested) -->

  </div>

  <!-- /.row (nested) -->
```

```
   </div>
   <!-- /.panel-body -->
   </div>
  <!-- /.panel -->
   </div>
   <!-- /.col-lg-12 -->
     </div>
  <?php include('footer.php');?>
  <?php include("val.php"); ?>
</body>
</html>
```

Update:

```
<?php
include("../db connect/db_connect.php");
$category_id=$_POST['category_id'];
$product_name=$_POST['product_name'];
$description=$_POST['description'];
//$image=$_POST['image'];
$price=$_POST['price'];
$quantity=$_POST['quantity'];
$quality=$_POST['quality'];
$expiry_date=$_POST['Expiry_date'];
$image=$_FILES['image']['name'];
$tmp_location=$_FILES['image']['tmp_name'];
$target="../img/".$image;
move_uploaded_file($tmp_location,$target);
$honey_product_id=$_POST['honey_product_id'];
```

```
$sql="update honey_product set
category_id='$category_id',product_name='$product_name',description='$description',image='$image',price='$price'
,quantity='$quantity',quality='$quality',expiry_date='$expiry_date' where honey_product_id='$honey_product_id'";

mysqli_query($conn,$sql);

?>

<script>

alert("honey product values are updated..!!");

document.location="honey_product_view.php";

</script>
```

CUSTOMER DETAILS TABLE

Form:

```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<?php include('metadata.php');?>

  <?php include('header.php');?>

  <?php include('sidebar.php');?>

</head>

<body>

<!--/. NAV TOP   -->

<!-- /. NAV SIDE   -->

<div id="page-wrapper" >

  <div class="header">

  <h1 class="page-header">

    Customer Details

</h1>

</div>
```

```
<div id="page-inner">
<div class="row">
<div class="col-lg-12">
<div class="panel panel-default">
<div class="panel-heading">
</div>
 <div class="panel-body">
 <div class="row">
 <div class="col-lg-6">
 <!-- /.col-lg-6 (nested) -->
 <form name="formID" id="formID" method="post" action="customer_details_insert.php">
 <table width="608" height="351" border="0" align="center">
 <tr>
 <td width="260">Customer Name</td>
 <td width="285"><input name="customer_name" type="text" id="customer_name"
class="validate[required,custom[onlyLetter]] form-control" /></td>
 </tr>
 <tr>
 <td>Customer Address </td>
 <td><div align="center">
 <input name="customer_address" type="text" id="customer_address" class="validate[required]
form-control"></textarea>
</div></td>
</tr>
<tr>
  <td>Customer City </td>
  <td><input name="customer_city" type="text" id="customer_city" class="validate[required] form-control" /></td>
   </tr>
   <tr>
   <td>Contact Number </td>

   <td><input name="contact_number" type="text" id="contact_number" class="validate[required,custom[mobile]]
form-control" /></td>

   </tr>

   <tr>

   td>Email ID</td>

  <td><input name="email_id" type="text" id="email_id" class="validate[required,custom[email]] form-control"
/></td>

  </tr>

  <tr>

  <td>Customer Code</td>

  <td><input name="customer_code" type="text" id="customer_code" class="validate[required] form-control"
/></td>

  </tr>

  <tr>

   <td colspan="2"><div align="center">

   <input type="submit" name="Submit" value="Submit" class="btn btn-info">

   <input type="reset" name="Reset" value="Reset" class="btn btn-danger">

   </div></td>

   </tr>

  </table>

</form>

</div>

 <!-- /.col-lg-6 (nested) -->

 </div>

   <!-- /.row (nested) -->

   </div>

   <!-- /.panel-body -->

    </div>
```

```
      <!-- /.panel -->

      </div>

      <!-- /.col-lg-12 -->

      </div>

     <?php include('footer.php');?>

     <?php include("val.php"); ?>

     </body>

      </html>
```

Insert:

```
   <?php

  include("../db connect/db_connect.php");

$customer_name=$_POST['customer_name'];

$customer_address=$_POST['customer_address'];

$customer_city=$_POST['customer_city'];

$customer_number=$_POST['customer_number'];

$email_id=$_POST['email_id'];

$customer_code=$_POST['customer_code'];

  $sql="insert into customer_details
values(null,'$customer_name','$customer_address','$customer_city','$customer_number','$email_id','$customer_cod
e')";mysqli_query($conn,$sql);

$sql1="insert into login values(null,'$customer_code','$customer_code','customer','who i
am','$customer_code','active')";

mysqli_query($conn,$sql1);

?>

<script>

alert("customer details inserted..!");

document.location="customer_details_form.php";

</script>
```

Delete:

```
<?php
include("../db connect/db_connect.php");
$customer_id=$_REQUEST['customer_id'];
$sql="delete from customer_details where customer_id='$customer_id'";
mysqli_query($conn,$sql);
?>
<script>
alert("values are Deleted..");
document.location="customer_details_view.php";
</script>
```

Edit:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <?php include('metadata.php');?>
  <?php include('header.php');?>
  <?php include('sidebar.php');?>
</head>
<body>
  <!--/. NAV TOP   -->
  <!-- /. NAV SIDE   -->
  <div id="page-wrapper"
  <div class="header">
  <h1 class="page-header">
    Customer Details
    </h1
```

```
   </div>

   <div id="page-inner">

     <div class="row">

     <div class="col-lg-12">

     <div class="panel panel-default">

     <div class="panel-heading">

     </div>

     <div class="panel-body">

       <div class="row">

         <div class="col-lg-6">

         <!-- /.col-lg-6 (nested) -->

       <?php

     include("../db connect/db_connect.php");

   $customer_id=$_REQUEST['customer_id'];

   $sql="select * from   customer_details where customer_id='$customer_id'";

   $res=mysqli_query($conn,$sql);

   $row=mysqli_fetch_array($res);

   ?>

   <?php include("val.php"); ?>

<form name="formID" id="formID" method="post" action="customer_details_update.php">

<input type="hidden" name="customer_id" value="<?php echo $row['customer_id']; ?>"
class="validate[required,custom[onlyLetter]] form-control" />

<table width="654" height="369" border="0" align="center">

 <tr>

 <td width="213">Customer Name</td>

<td width="265"><input name="customer_name" type="text" id="customer_name" value="<?php echo
$row['customer_name'];?>"class="validate[required,custom[onlyLetter]] form-control" /></td>

 </tr>

 <tr>
 <td>Customer Address </td>

 <td><div align="center">

 <input name="customer_address" type="text" id="customer_address" value="<?php echo
$row['customer_address'];?>" class="validate[required] form-control"></textarea></td>

 </tr>

 <tr>

 <td>Customer City </td>

 <td><input name="customer_city" type="text" id="customer_city" value="<?php echo $row['customer_city'];?>"
class="validate[required] form-control" /></td>

</tr>

<tr>

 <td>Contact Number </td>

 <td><input name="contact_number" type="text" id="contact_number" value="<?php echo $row['contact_no'];?>"
class="validate[required,custom[mobile]] form-control" /></td>

 </tr>

 <tr>

 <td>Email ID</td>

  <td><input name="email_id" type="text" id="email_id" value="<?php echo $row['email_id'];?>"
class="validate[required,custom[email]] form-control" /></td>

  </tr>

    <tr>

    <td height="42">Customer Code</td>

    <td><input name="customer_code" type="text" id="customer_code" value="<?php echo

     $ row['customer_code'];?>" class="validate[required] form-control" /></td>

    </tr>

    <tr>

     <td height="31" colspan="2"><div align="center">

     <input type="submit" name="Submit" value="Submit" class="btn btn-success">

      <input type="reset" name="Reset" value="Reset" class="btn btn-danger">

      </div></td>
```

```
        </tr>
      </table>
      </form>
      </div>
    <!-- /.col-lg-6 (nested) -->
  </div>
    <!-- /.row (nested) -->
    </div>
    <!-- /.panel-body -->
    </div>
    <!-- /.panel -->
    </div>
    <!-- /.col-lg-12 -->
    </div>
 <?php include('footer.php');?>
 <?php include("val.php"); ?>
</body>
</html>
```

View:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
 <?php include('metadata.php');?>
 <?php include('header.php');?>
 <?php include('sidebar.php');?>
</head>
<body>
```

```
 <!--/. NAV TOP   -->
 <!-- /. NAV SIDE   -->
 <div id="page-wrapper" >
<div class="header">
<h1 class="page-header">
 Customer Details
 </h1>
<!-- <ol class="breadcrumb">
<li><a href="#">Home</a></li>
<li><a href="#">Tables</a></li>
 <li class="active">Data</li>
</ol>   -->
</div>
<div id="page-inner">
<div class="row">
<div class="col-md-12">
 <!-- Advanced Tables -->
<div class="panel panel-default">
 <div class="panel-heading">
<a href="customer_details_form.php" class="btn btn-info">ADD NEW</a>
 </div>
<div class="panel-body">
<div class="table-responsive">
<table class="table table-striped table-bordered table-hover" id="dataTables-example">
 <thead>
 <tr>
 <th>slno</th>
 <th >Customer Name</th>
 <th >Customer Address </th>

 <th >Customer City </th>

 <th>Contact no </th>

 <th >Email Id </th>

 <th >Customer Code </th>

 <th >Edit</th>

 <th >Delete</th>

 </tr>

 </thead>

 <tbody>

  <?php

  include("../db connect/db_connect.php");

  $slno=1;

  $sql="select * from   customer_details   ";

  $res=mysqli_query($conn,$sql);

  while($row=mysqli_fetch_array($res))

   {

   ?>

  <tr>

  <td>&nbsp;<?php echo $slno++; ?></td>

  <td>&nbsp;<?php echo $row['customer_name']; ?></td>

  <td>&nbsp;<?php echo $row['customer_address']; ?></td>

  <td>&nbsp;<?php echo $row['customer_city']; ?></td>

  <td>&nbsp;<?php echo $row['contact_no']; ?></td>

  <td>&nbsp;<?php echo $row['email_id']; ?></td>

  <td>&nbsp;<?php echo $row['customer_code']; ?></td>

  <td><a href="customer_details_edit.php?customer_id=<?php echo $row['customer_id'];?>" class="btn
btn-primary">Edit</a></td>
<td><a href="customer_details_delete.php?customer_id=<?php echo $row['customer_id'];?>" class="btn
btn-danger">Delete</a></td>

 </tr>

 <?php

  }

  ?>

  </tbody>

  </table>

  </div>

  </div>

  </div>

   <!--End Advanced Tables -->

   </div>

    </div>

    <!-- /. ROW   -->

    <?php include('footer.php');?>

    </body>

    </html>
```

Update:

```
<?php

include("../db connect/db_connect.php");

$customer_name=$_POST['customer_name'];

$customer_address=$_POST['customer_address'];

$customer_city=$_POST['customer_city'];

$contact_no=$_POST['contact_number'];

$email_id=$_POST['email_id'];

$customer_code=$_POST['customer_code'];
```

```
$customer_id=$_POST['customer_id'];

$sql="update customer_details set
customer_name='$customer_name',customer_address='$customer_address',customer_city='$customer_city',contact_
no='$contact_no',email_id='$email_id',customer_code='$customer_code' where customer_id='$customer_id'";

mysqli_query($conn,$sql);

?>

<script>

alert("customer details updated..!");

document.location="customer_details_view.php";

</script>
```

10.TESTING

10. 1 SYSTEM TESTING

10.1.1 Introduction:

Testing is a process of executing a program with the indent of finding an error. Testing is a crucial element of software quality assurance and presents ultimate review of specification, design and coding. System Testing is an important phase. Testing represents an interesting anomaly for the software. Thus a series of testing are performed for the proposed system before the system is ready for user acceptance testing. The code is tested at various levels in software testing. Unit, system and user acceptance testings are often performed.

Testing Objectives

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a probability of finding an as yet undiscovered error.
- A successful test is one that uncovers an undiscovered error.

Testing Principles

- All tests should be traceable to end user requirements.
- Tests should be planned long before testing begins.
- Testing should begin on a small scale and progress towards testing in large.
- Exhaustive testing is not possible.
- To be most effective testing should be conducted by a independent third party.

The primary objective for test case design is to derive a set of tests that has the highest livelihood for uncovering defects in software. To accomplish this objective two different categories of test case design techniques are used. They are:

- White box testing.
- **White Box Testing:** White Black box testing.

White-Box Testing and Black-Box Testing

box testing focus on the program control structure. Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed.

Black Box Testing: Black box testing is designed to validate functional requirements without regard to the internal workings of a program. Black box testing mainly focuses on the information domain of the software, deriving test cases by partitioning input and output in a manner that provides through test coverage. Incorrect and missing functions, interface errors, errors in data structures, error in functional logic are the errors falling in this category.

Testing strategies

A strategy for software testing must accommodate low-level tests that are necessary to verify that all small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

There are two general strategies for testing software. They are as follows:

Code Testing: This examines the logic of the program. To follow this test, cases are developed such that every path of program is tested.

Specification Testing: Specification Testing examines the specification, starting what the program should do and how it should perform under various conditions. Then test cases are developed for each condition and combinations of conditions and to be submitted for processing.

10.1.2 Levels of Testing

The stages of Testing Process are:

Unit Testing: Individual components are tested to ensure that they operate correctly. Each component tested independently without other system components. Ex. Check for Username and Password with the table, after the next module is loaded.

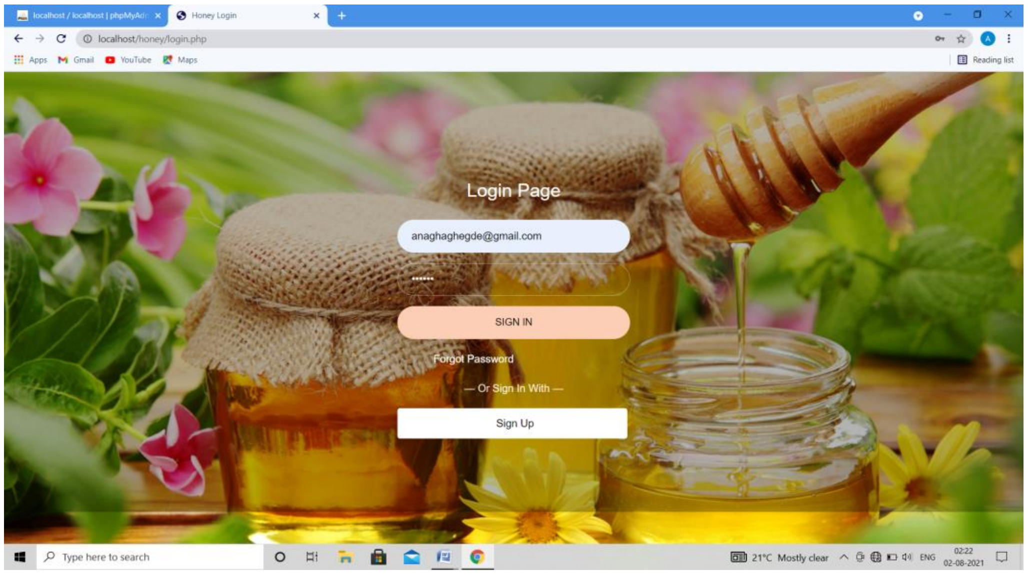
Integration Testing: Integration testing is a systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated with interfacing. This testing is done using the bottom-up approach to integrate the software components of the software system in to functioning whole.

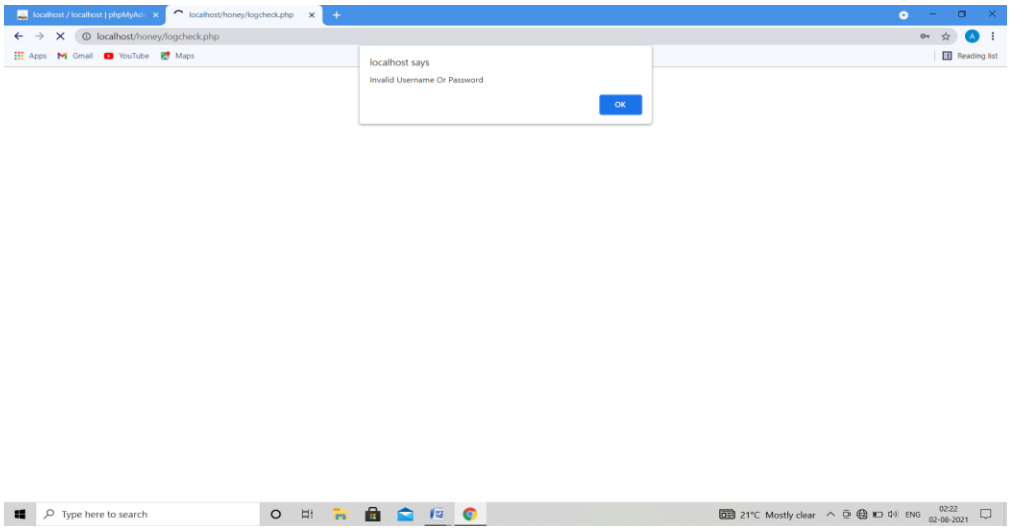
System Testing: System testing is actually a series of different tests whose primary purpose is fully to exercise the computer-based system. The system tests that where applied are recovery testing and performance testing. Finally a review or audit is conducted which is a final evaluation that occurs only after operating the system long enough for user to have gained a familiarity with it. System testing was done by the inspection team to verify that all the functionality identified is the software requirement specification has been implemented. Defects that crept in the system has been found defect free and is working well. System testing is concerned with interfaces, design logic, control flow recovery, procedures throughput, capacity and timing characteristics of the entire system. For blank field, alphabets, number and special character validation.

Acceptance Testing: User acceptance of the system is the key factor for the success of any system. This is done by user. The system is given to the user and they test it with live data. Acceptance testing involves the planning and execution of functional test. Performance tests, stress tests in order to demonstrate that the implemented system satisfies its requirements. Two sets of acceptance test can be run, those developed by the customer. The system has been tested for its performance at unit level by the individuals through performance testing that is designed to test the run time performance of the software. The performance of the fully integrated system is tested and was found good.

10.2 Validating The Tables

Login Form




localhost says

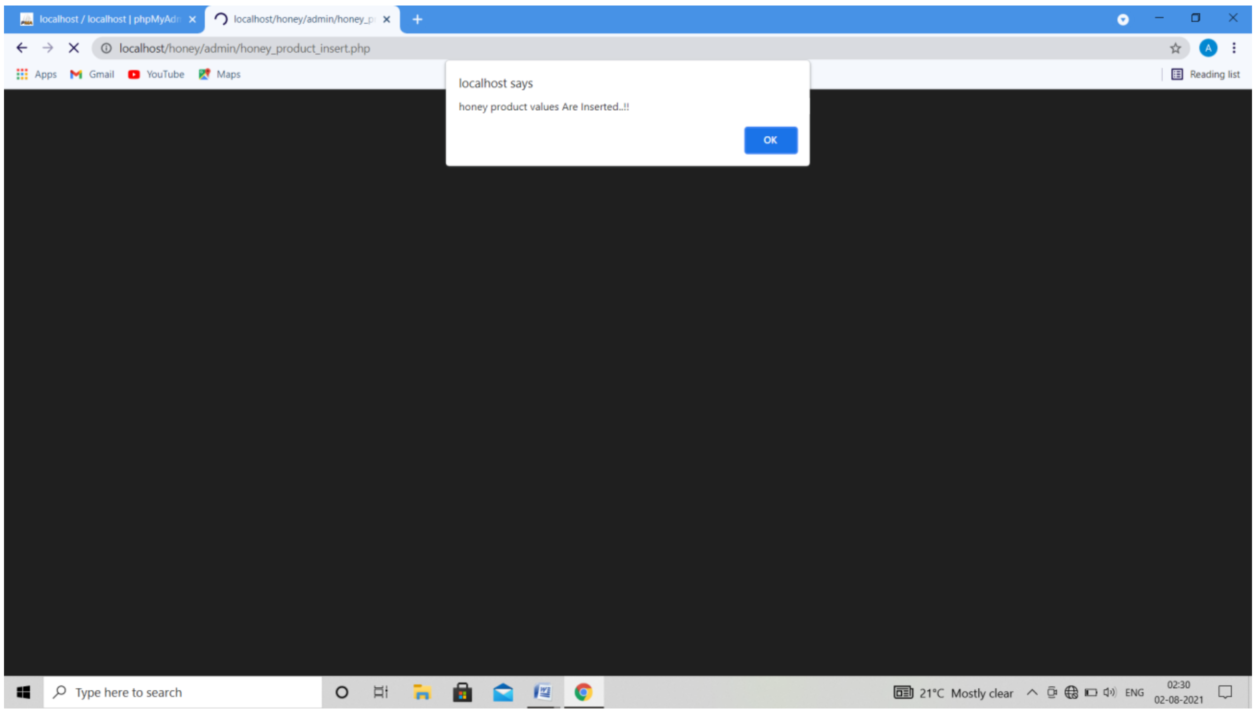
Invalid Username Or Password

Fig (10.2.1): login

The above screenshot is showing the error message that we should enter valid username and password.

HONEY PRODUCT Table

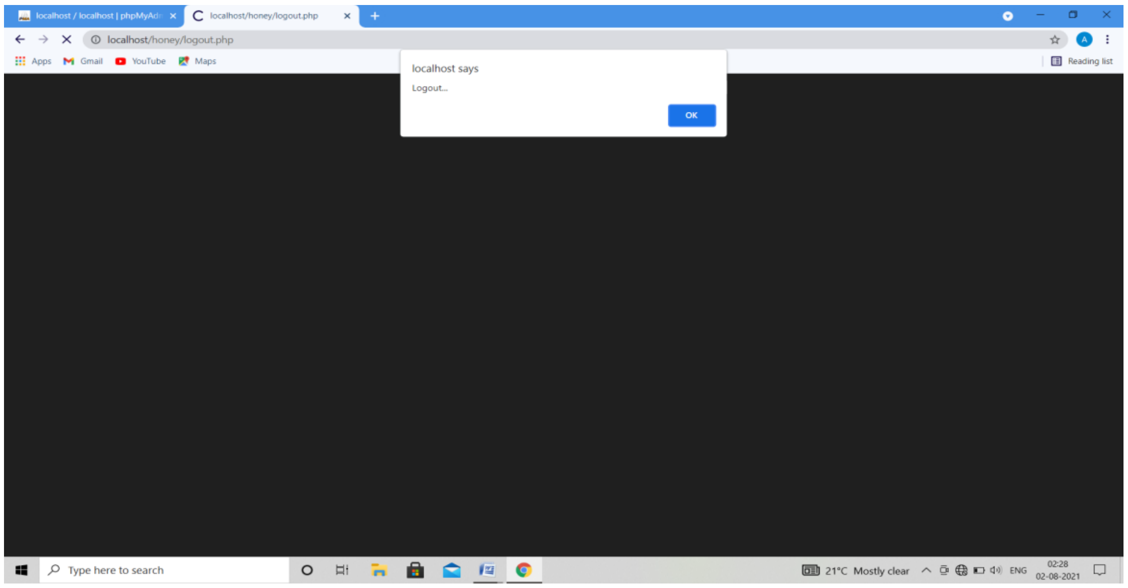
Adding a new value to the table


localhost says

honey product values Are Inserted..!!

Fig (10.2.3):Honey Product

Login Form

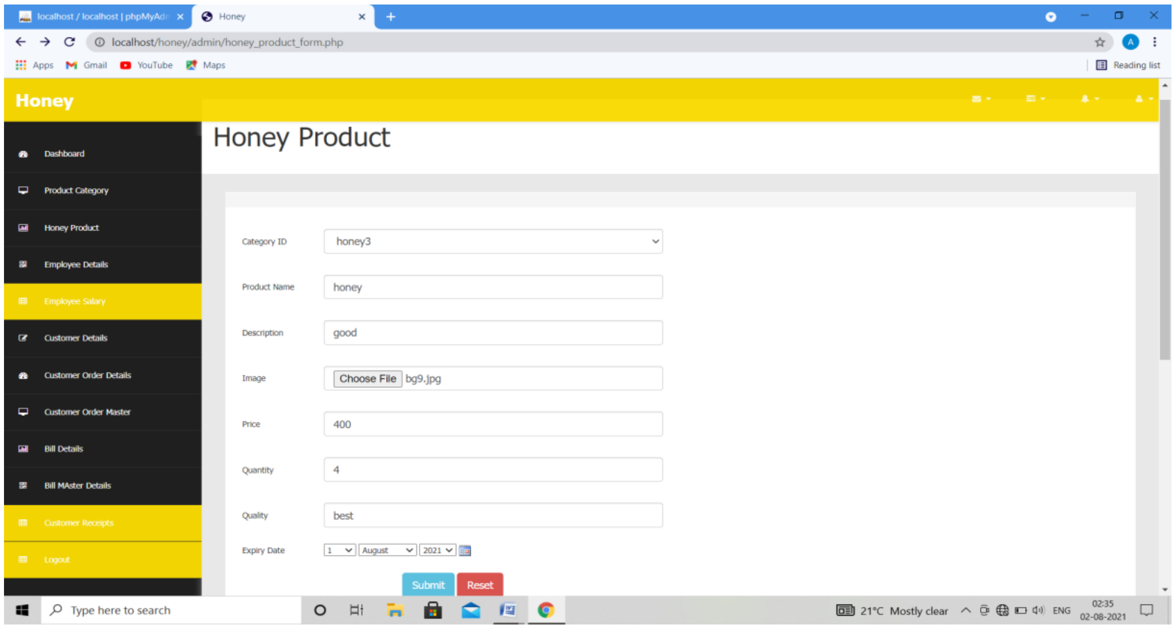

localhost says

Logout...

Fig (10.2.4): Logout

Honey Product

Product Image

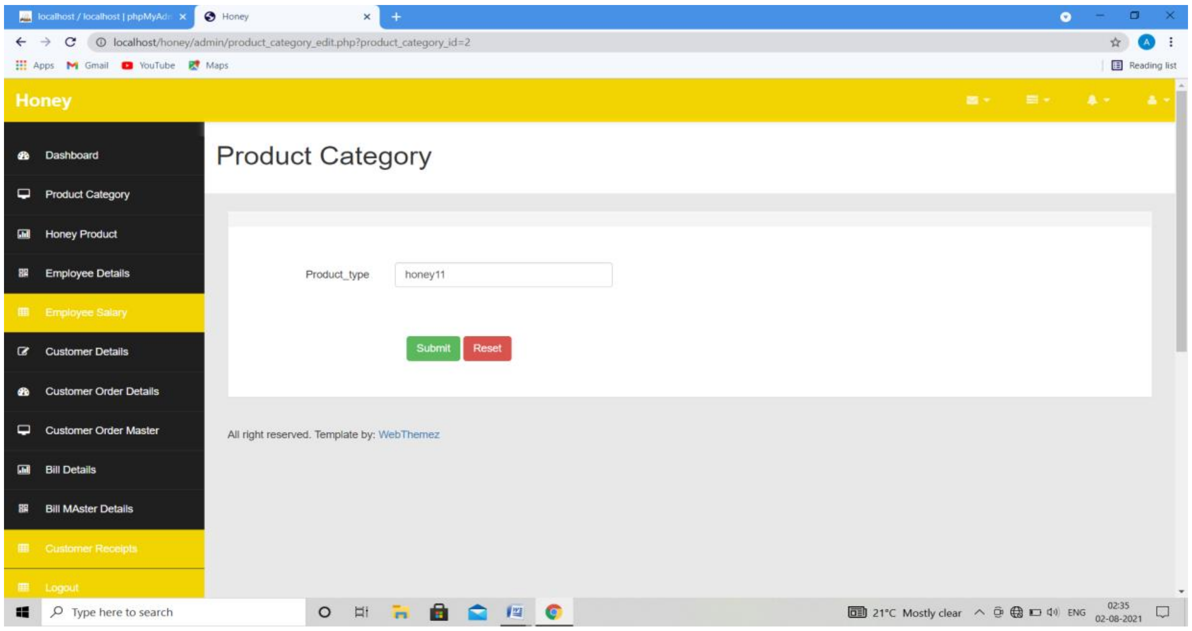


Honey Product

Category ID	honey3
Product Name	honey
Description	good
Image	Choose File bg9.jpg
Price	400
Quantity	4
Quality	best
Expiry Date	1 August 2021

Fig (10.2.5): Honey product Category

Product Category



localhost/honey/admin/product_category_edit.php?product_category_id=2

Product Category

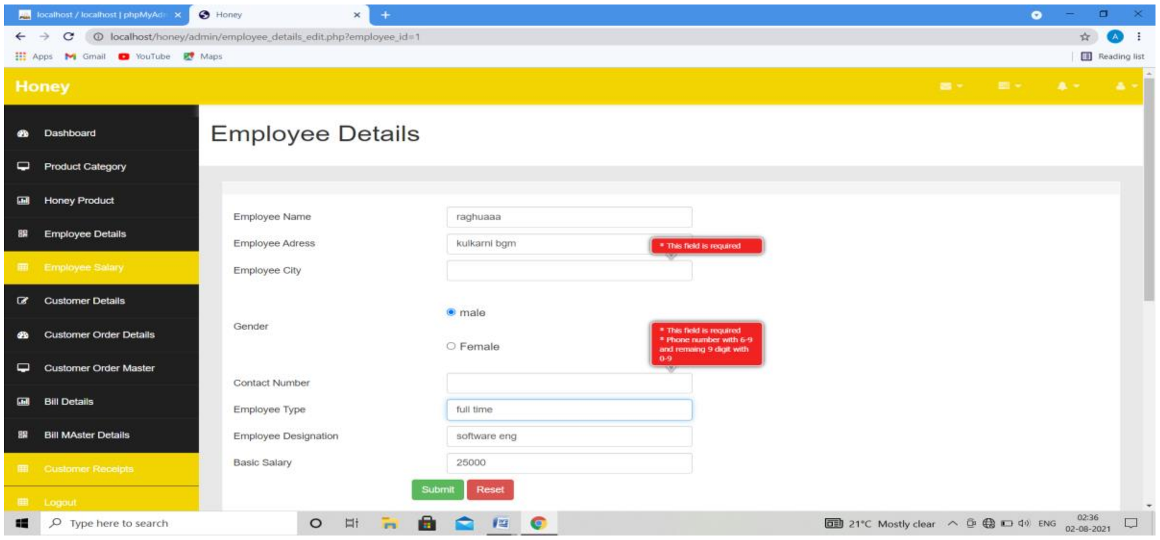
Product_type: honey11

All right reserved. Template by: WebThemez

Fig (10.2.6): Product Category

Employee Details

Employee City



localhost/honey/admin/employee_details_edit.php?employee_id=1

Employee Details

Employee Name: raghuaaa

Employee Adress: kulkarni bgm

* This field is required

Employee City:

Gender: [✓ male] Female

* This field is required
* Phone number with 6-9 and remaing 9 digit with 0-9

Contact Number:

Employee Type: full time

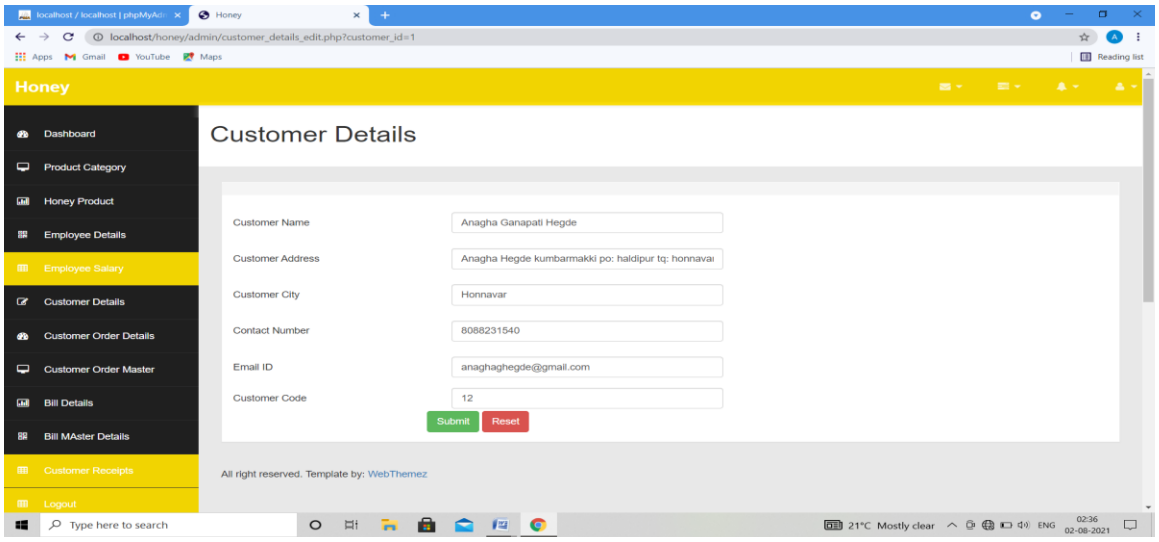
Employee Designation: software eng

Basic Salary: 25000

Fig (10.2.6): Employee Details

Customer Details

Email ID

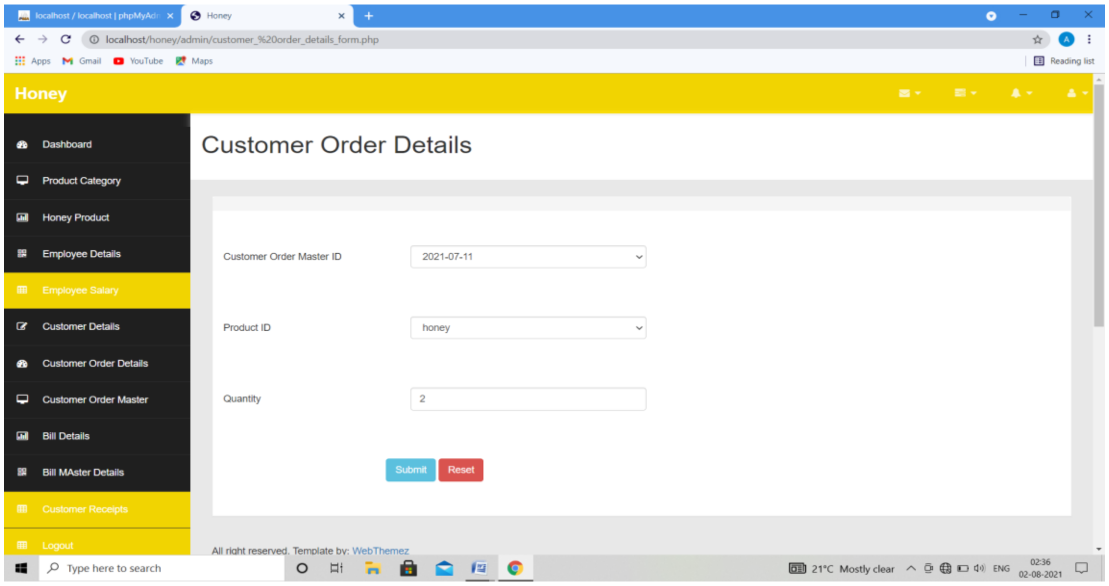


Customer Details

Customer Name	Anagha Ganapati Hegde
Customer Address	Anagha Hegde kumbarmakki po: haldipur tq: honnavar
Customer City	Honnavar
Contact Number	8088231540
Email ID	anaghaghegde@gmail.com
Customer Code	12

All right reserved. Template by: WebThemez

Fig (10.2.7): Customer Details



Customer Order Details

Customer Order Master ID	2021-07-11
Product ID	honey
Quantity	2

All right reserved. Template by: WebThemez

Fig (10.2.5): Customer Order Details

10.3 Functional Testing

Test No	Test Case	Expected Result	Actual Result	Result
1	Valid Username and Password	It should display respective page according to user type.	Respective Home is displayed	Fig (10.2.1)
2	Invalid Username and Password	It should give appropriate error message saying “Enter proper User-ID and Password”	Error message displayed	Fig (10.2.2)
3	Add/Update /Delete Member Details	Add/Delete/Update action is taken.	Added/Updated/Deleted Member message Displayed	Fig (10.2.3)
4	Logout	It should logout correctly and should not go to the home page	Logout message displayed and login page is shown	Fig (10.2.4)
5	Insert PDF/Image file	Choose file option should promt for selecting PDF/Image	Selected file is opened for selecting PDF/Image	Fig (10.2.5)

6	Blank field while inserting/ updating	It should give appropriate error message	Display appropriate error message	Fig (10.2.6)
7	Valid E-mail id with specific domain	It should insert the Email-id into the database while editing/inserting	E-mail id is stored while editing/inserting	Fig (10.2.7)
8	Valid mobile number	It must take correct number while inserting/updating	Mobile number is saved when inserting/updating	Fig (10.2.8)
9	Invalid mobile number	It should give appropriate message as “Mobile number entered is incorrect”	It will display the error message	Fig (10.2.9)
10	Registering distributor	Respective distributor information should be inserted	Respective distributor should receive ordered details with bill generated	Fig (10.2.10)

11.CONCLUSION

Software is said to have attained its objective only when it need all requirements of the user, further the user himself is the person to judge the success of the system. Every attempt has been made to ensure that the system is fully functional and works effectively and efficiently. The system has been tested with simple data to cover all possible options and checked for all outputs. Since the system is flexible and modular, further modification of this packge can be easily incorporated.

Importance of the system:

- Less manual work.
- Increased efficiency.
- Decreases the rate of errors.
- It reduces the time consumption.
- Quick (instant) result.