**KARNATAK UNIVERSITY**

**DHARWAD**

**Janata Shikshana Samiti’s**

**Shri Manjunatheshwra Institute of UG & PG Studies Vidyagiri, Dharwad-580004.**

****

**A PROJECT REPORT ON**

**“CUSTOMISED APPLICATION FOR GLASS BANGLES MANAGEMENT”**

**BACHELOR OF SCIENCE(CS)**

**OF**

**KARNATAK UNIVERSITY, DHARWAD**

**PROJECT GUIDED BY:**

**Miss. Padmavathi Shyadambi**

**Submitted By:**

**Pallavi Mindolkar Pooja Yaligar**

**BSC(cs) VI SEMESTER BSC(cs)VI SEMESTER**

**REG NO:18M10135 REGNO: 18M10139**

**DEPARTMENT OF COMPUTER SCIENCE**

**2020-2021**

**JANATA SHIKSHAN SAMITI’S**

**SHRI MANJUNATHESHWARA INSTITUTE OF UG & PG STUDIES, VIDYAGIRI, DHARWAD-580004**

****

CERTIFICATE

**This is to certify that Miss. PALLAVI M AND Miss. POOJA Y has satisfactorily completed project work entitled “ CUSTOMISED APPLICATION FOR GLASS BANGLES MANAGEMENT” for the partial fulfillment of BSC(cs) prescribed by Karanatak University, Dharwad during the academic year**

**2020-2021.**

**Miss. Padma Shyadambi Sri. Vivek M Laxmeshwar Dr. Ajith Prasad**

Project Guide [HOD] Computer Department Principal

Examiners:

1. 2)

**ACKNOWLEDGEMENT**

**The successful presentation of this project is an acknowledgement of the immense support extended by JSS Shri Manjunatheshwara Institute of UG & PG Studies, Vidyagiri, Dharwad Which has provided opportunities to fulfil the most cherished desired to reach our goal. We would like to express our heartfelt thanks to our President Shri. Vishwaprasanna teertha of Pejavar math of Udupi, Chairman Padmabhushan Pujya Dr. D. Veerendra Heggade and Dr. N. Vajrakumar, Secretory of JSS.**

**We would like to express our sincere gratitude to our beloved Principal Dr. Ajith Prasad who gave us inspiration and moral support. We offer our heartily gratitude to our H.O.D Shri. Vivek M. Laxmeshwar of Computer Science Department who gave us knowledge and an opportunity to do the project on “Customised Application for GLASS BANGLES INDUSTRY”, which also helped us in doing a lot of research and where expose to many new things.**

**We would also take this opportunity to offer our sincere gratitude to our** **Project Guide Miss. Padma Shyadambi for her excellent support throughout the development of this project and for providing the necessary information on our request at all times.**

**We also thank all Teaching and Non-Teaching staff of Computer Science Department for helping us in all aspects. We thank our parents for providing all the facilities to complete and make this project successful.**

**PALLAVI MINDOLKAR (18M10135)**

**POOJA YALIGAR (18M10139)**

**DECLARATION**

We, **Pallavi Mindolkar** and **Pooja Yaligar,** students of sixth semester BCA, Department of Computer Science, JSS SMI UG AND PG STUDIES VIDYAGIRI, DHARWAD, Karnatak University declare that the project entitled **“CUSTOMISED APPLICATION FOR GLASS BANGLE INDUSTRY”** has been submitted in partial fulfillment of the course requirement for the award of degree in BachelorofScience(cs)**,** KarnatakUniversity**,** Dharwad during the academic year 2020-2021. We have not submitted the matter embodied to any other university or institution for the award of any other degree.

**Date: Pallavi Mindolkar**

**Place: Dharwad Pooja Yaligar**

**CONTENTS**

**1. PROJECT SYNOPSIS………………………………………...1**

**2. FRAMEWORK………………………………………………….**

**2.1. HTML**

**2.2. DreamWeaver**

**2.3. PHP**

**2.4. WAMP**

**2.5. My Sql**

**3. PROJECT SUBJECT…………………………………………..**

**4. SOFTWARE REQUIREMENTS SPECIFICATION…………**

**4.1. Introduction**

**4.1.1. Purpose**

**4.1.2. Document Conventions**

**4.1.3. Scope**

**4.1.4. References**

**4.2. General Description**

**4.2.1. Product Perspective**

**4.2.2. Product Functionalities**

**4.2.3. User Characteristics**

**4.3. Specific Requirements**

**4.3.1. Functional Requirements**

**4.3.2. Other Non-Functioal Requirements**

**4.3.3. External Interface Requirements**

**5.DESIGN OF THE SYSTEM……………………………………**

**5.1. Introduction**

**5.2. Data Flow Diagram**

**5.3. ER- Diagram**

**5.4. Use-Case Diagram**

**6. IMPLEMENTATION…………………………………………..**

**6.1. Introduction**

**6.2. Database Evolution**

**7. DATABASE TABLES…………………………………………**

**7.1. Tables Used In Database**

**8. SCREEN SHOTS……………………………………………**

**9. SOURCE CODE………………………………………………**

**10.TESTING……………………………………………………….**

**10.1. Software Testing**

**10.1.1. Introduction**

**10.1.2. Levels Of Testing**

**10.2. Validating The Tables**

**10.3. Functional Testing**

**11. CONCLUTION………………………………………………..**

**12. BIBLIOGRAPHY……………………………………………..**

**12.1. Books**

**12.2. Websites**

1. **PROJECT SYNOPSIS**

**1.1 INTRODUCTION**

This project is designed to help traditional bangle manufacturing unit of industry to maintain record of order details, raw material details, finished stock details and invoice management. Nearly 80 employees are working in this industry and they have their own

vehicles to transport the products to different places. They manufacture different types of

glass bangles. These will be packed and distributed.

**1.2 INPUTS OF THE PROJECT**

* Customer details
* Production details
* Stock details
* Sales details
* Raw material details

**1.3 OUTPUT OF THE PROJECT**

* Payment report
* Production report
* Stock report
* Sales report
* Raw material report

**1.4** **PROCESS LOGIC**

Customer details

Production details

Stock details

Sales details

Raw material details

Payment report

Production report

Stock report

Sales report

Raw material report

**1.6 TOOLS/PLATFORM**

Browser (Google Chrome)

**1.7 FRONT END**

* HTML
* CSS

**1.8 BACK END**

* PHP
* MySql

**1.9 DURATION OF THE PROJECT**

2 Months

**1.10 MEMBERS OF THE PROJECT**

* **Pallavi Mindolkar 18M10135**
* **Pooja Yaligar 18M10139**

**1.11 LIMITATIONS**

* No multi- language support.
* No support for instant online reply.
* No advanced payment such as internet banking, credit card, debit card payment system.
* Applicable only for one branch.

**1.12 SCOPE OF THE APPLICATION**

* Online payment acceptance.
* Development of Android and IOS application

**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
* Designing 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 bangles. It helps to make bills and maintain the stock of all type of bangles 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.Vendor module**

* Vendor details are maintained.
* He confirms the orders placed by the admin on raw materials.

**3.Customer module**

* Customer details are maintained.
* He orders the products whichever are necessary.

**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,
* Online payment acceptance for wide range of services.

**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.
* Provides data security and huge maintenance of data.

**4.2.2 Product Functionalities**

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 authentication.

**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 password. Give him 3 chances.

**d) Order Placing**

* 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 vendors, distributors 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.

Internet

Mysql

Admin

Web Browser which manages user interaction with website

Wamp Server

**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:**

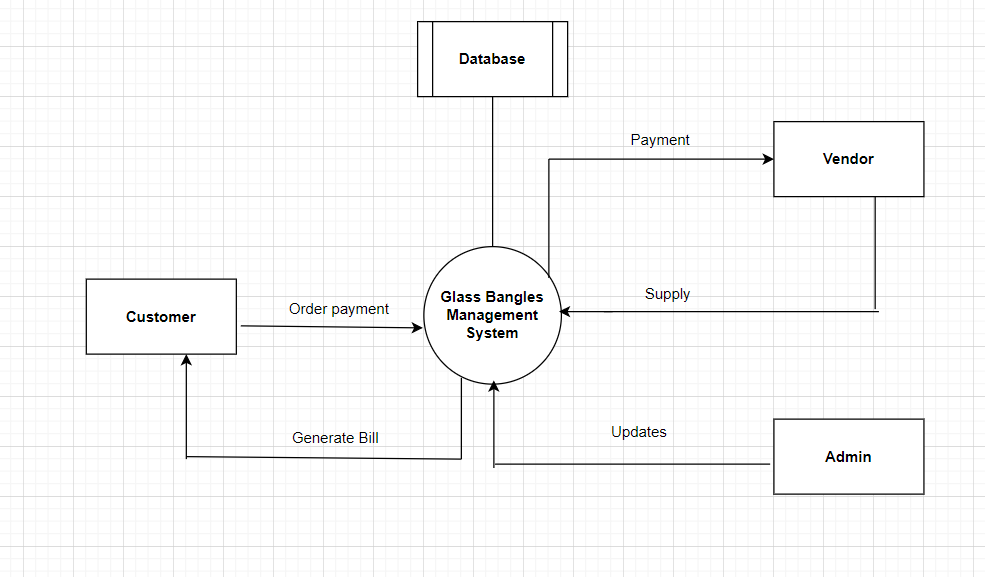
****

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

**Objectives:**

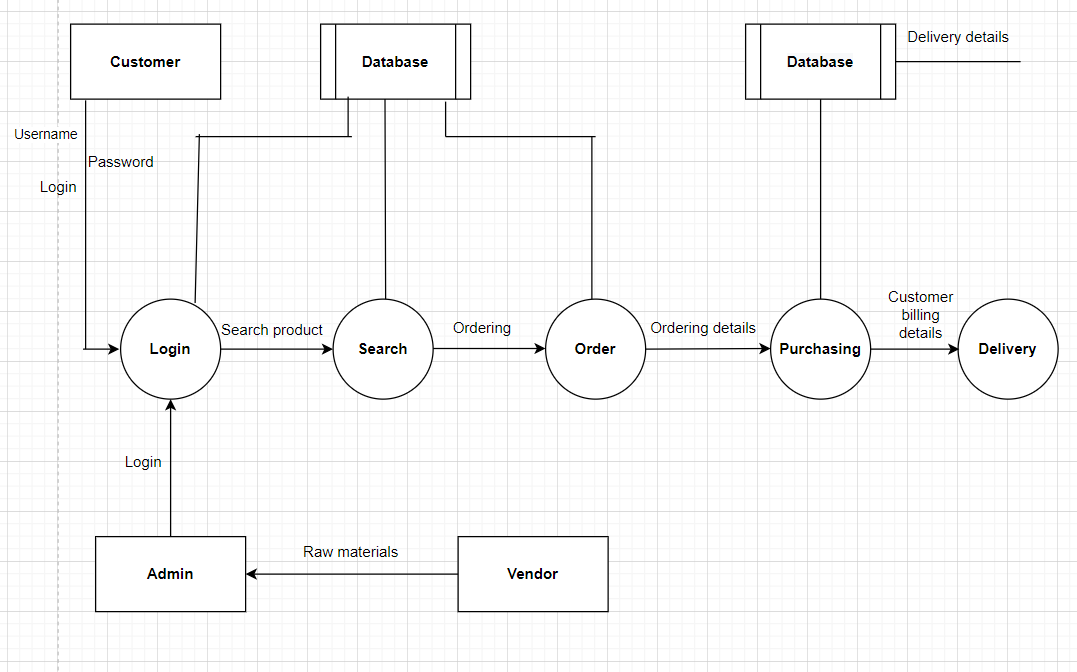
* To graphically document boundaries of a system.
* To provide hierarchy breakdown of the system.
* To show movement of information between a system and its environment.
* To document information flows within the system.
* To aid communication between users and developers.

**Context level diagram (zero level dfd)**

**

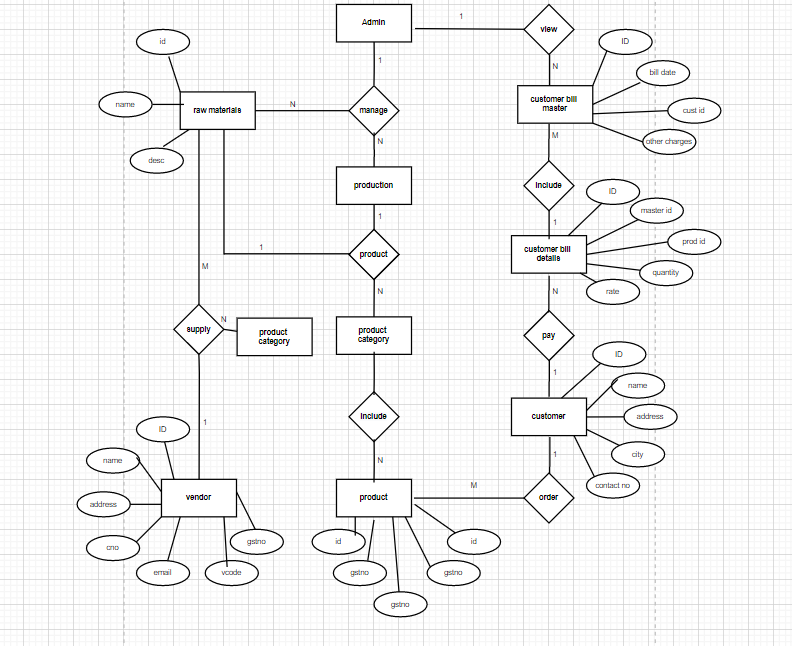
*Fig level 0 DFD: Bangle manufacturing industry*

**Context level diagram (one level DFD):**

****

*Fig level 1 DFD: Bangle manufacturing industry*

**5.3 ER 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 compil ed 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>

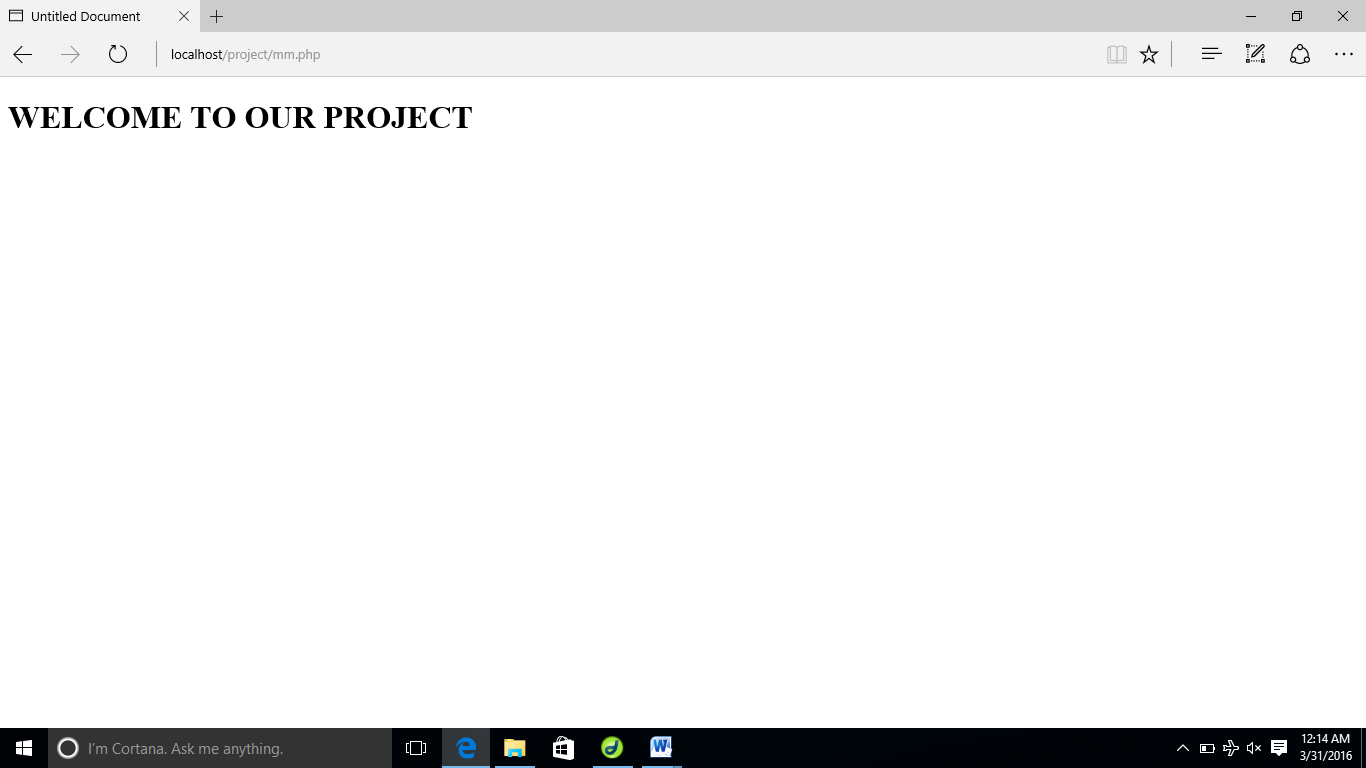
<body>

<h1>WELCOME TO OUR PROJECT</h1>

</body>

</html>

**Output:**



**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

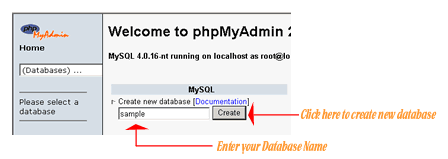
<?php

$con=mysql\_connect('localhost','root','');

mysql\_select\_db ('project', $con); ?>

**Steps to create a database in PHPMyAdmin**

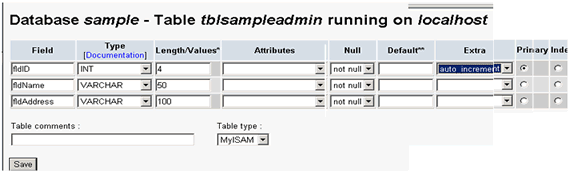
1) The following figure shows your PHPMyAdmin interface, just enter your database name and click the 'Create' button to create your database.



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



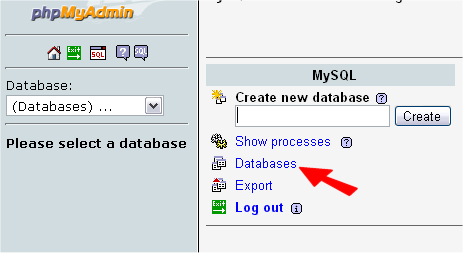
3)The next step is to create the fields, just enter values for each field name, type, length of the field, null option and mention whether it is a primary key or not. Then click the 'Save' button to complete your table creation.



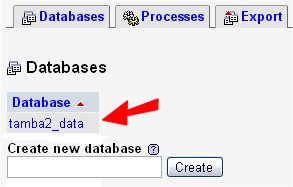
4) The following figure is displayed upon successful creation of your table

**Steps to Drop Table in PHPMyAdmin:**

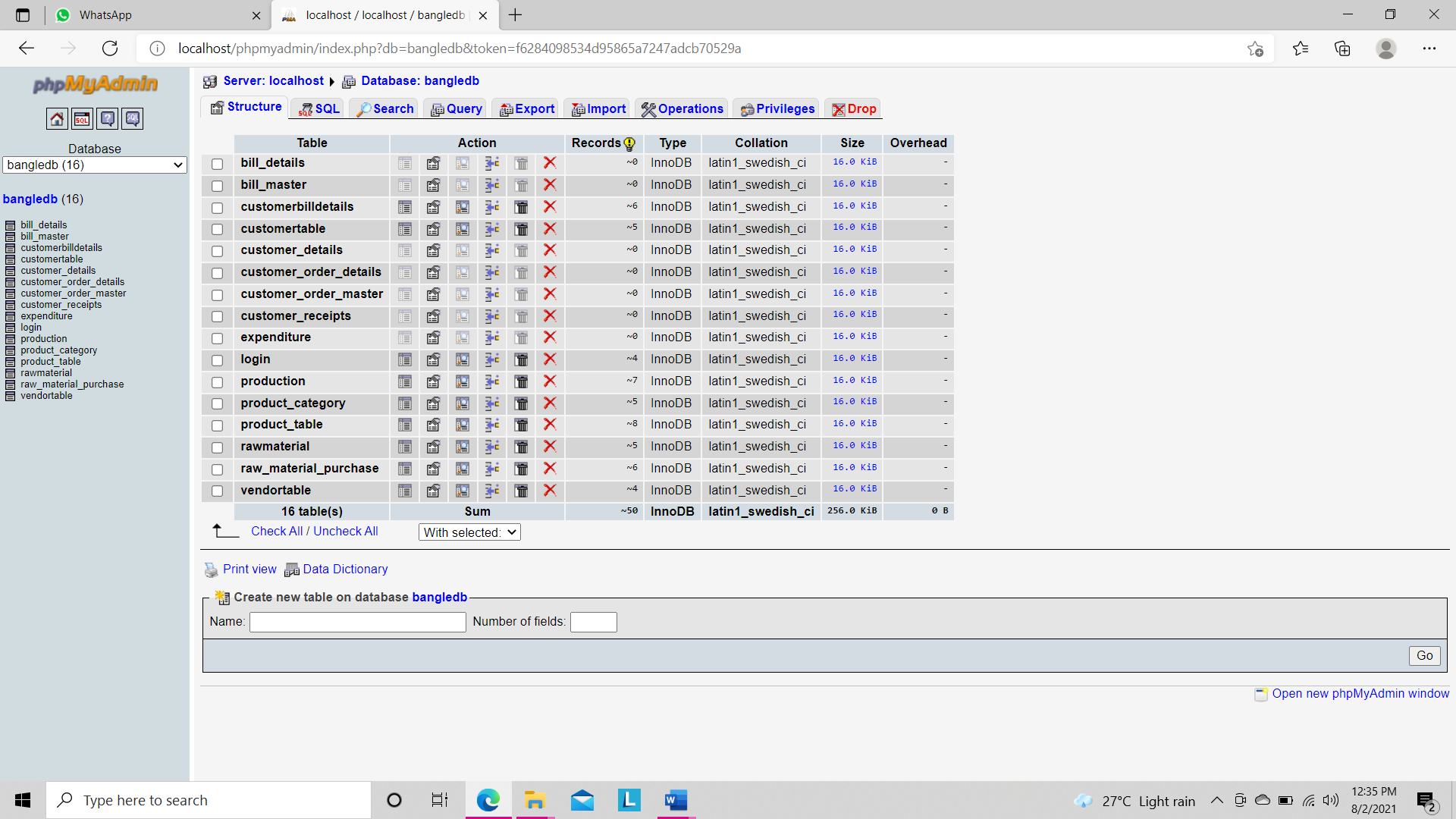
Login to phpmyadmin. Click 'databases'



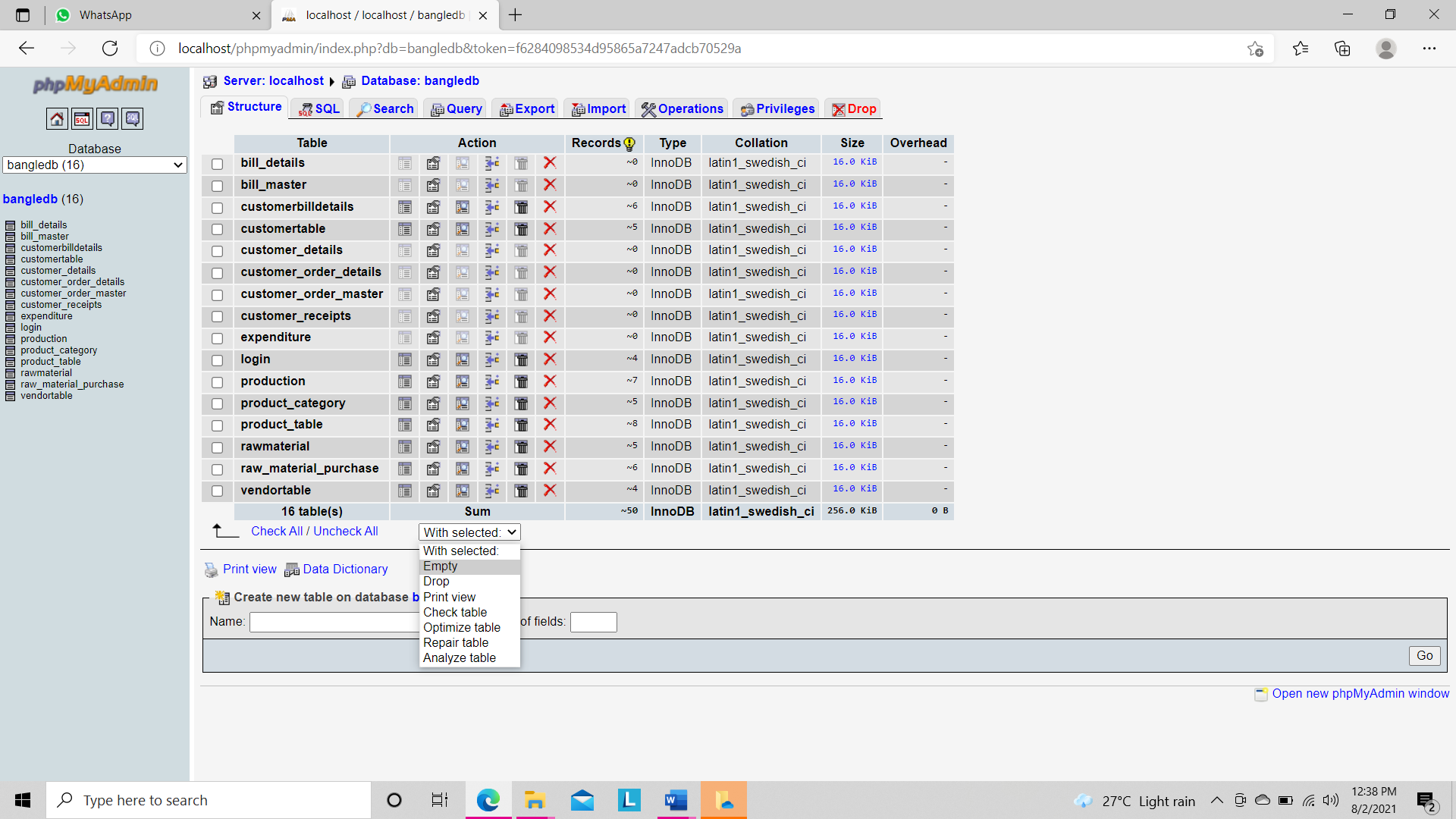
List of your databases will appear. Click the one that is your WordPress database.



Note the size of the 'wp\_bad\_behaviour\_log' table - this is one to be emptied in this example.



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.



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 !

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 table bill\_details

## Primary Key: ****bill\_details\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***bill\_details\_id*** | int(11) | Yes | NULL |
| bill\_master\_id | int(11) | Yes | NULL |
| product\_id | int(11) | Yes | NULL |
| quantity | int(11) | Yes | NULL |
| Rate | Double | Yes | NULL |
| discount | Double | Yes | NULL |

## Table structure for table bill\_master

## Primary Key: ****bill\_master\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***bill\_master\_id*** | int(11) | Yes | NULL |
| bill\_date | date | Yes | NULL |
| customer\_id | int(11) | Yes | NULL |
| other\_charges | double | Yes | NULL |

## Table structure for table customer\_details

## Primary Key: ****cust\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***customer\_id*** | int(11) | Yes | NULL |
| customer\_name | varchar(50) | Yes | NULL |
| customer\_address | varchar(200) | Yes | NULL |
| customer\_city | varchar(50) | Yes | NULL |
| contact\_no | bigint(12) | Yes | NULL |
| email\_id | varchar(100) | Yes | NULL |
| customer\_code | varchar(50) | Yes | NULL |

|  |
| --- |
|  |

|  |
| --- |
|  |

## Table structure for table customer\_order\_details

## Primary Key: ****customer\_order\_details\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***customer\_order\_details\_id*** | int(11) | Yes | NULL |
| customer\_order\_master\_id | int(11) | Yes | NULL |
| product\_id | int(11) | Yes | NULL |
| Quantity | int(11) | Yes | NULL |
| cust\_order\_status | varchar(50) | Yes | NULL |

|  |
| --- |
|  |

## Table structure for table customer\_order\_master

## Primary Key: ****customer\_order\_master\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***customer\_order\_master\_id*** | int(11) | Yes | NULL |
| Date | date | Yes | NULL |
| customer\_id | int(11) | Yes | NULL |

|  |
| --- |
|  |

## Table structure for table customer\_receipts

## Primary Key: ****customer\_receipts\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***customer\_receipts\_id*** | int(11) | Yes | NULL |
| customer\_id | int(11) | Yes | NULL |
| amount | double | Yes | NULL |
| narration | varchar(500) | Yes | NULL |
| Date | date | Yes | NULL |

|  |
| --- |
|  |

## Table structure for table expenditure

## Primary Key: ****expenditure\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***expenditure\_id*** | int(11) | Yes | NULL |
| expenditure\_name | varchar(200) | Yes | NULL |
| description | varchar(500) | Yes | NULL |
| total\_amount | double | Yes | NULL |
| given\_to | varchar(100) | Yes | NULL |
| voucher\_no | varchar(50) | Yes | NULL |
| given\_date | Date | Yes | NULL |

## Table structure for table login

## Primary Key: ****loginid****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***Loginid*** | int(10) | Yes | NULL |
| username | varchar(200) | Yes | NULL |
| password | varchar(200) | Yes | NULL |
| Type | varchar(100) | Yes | NULL |
| hint\_q | varchar(100) | Yes | NULL |
| hint\_a | varchar(100) | Yes | NULL |
| Status | varchar(100) | Yes | NULL |

## Table structure for table production

## Primary Key: ****production\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***production\_id*** | int(10) | Yes | NULL |
| prod\_id | int(20) | Yes | NULL |
| quantity | int(20) | Yes | NULL |
| product\_details | varchar(200) | Yes | NULL |
| Date | Date | Yes | NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Table structure for table product\_category

## Primary Key: ****prodcatid****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***prodcatid*** | int(10) | Yes | NULL |
| prodcatname | varchar(200) | Yes | NULL |
| Desc | varchar(200) | Yes | NULL |

## Table structure for table product\_table

## Primary Key: ****prod\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***prod\_id*** | int(10) | Yes | NULL |
| prod\_name | varchar(20) | Yes | NULL |
| prodcatid | int(10) | Yes | NULL |
| Price | int(10) | Yes | NULL |

## Table structure for table rawmaterial

## Primary Key: ****rawid****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***Rawid*** | int(10) | Yes | NULL |
| rawname | varchar(200) | Yes | NULL |
| rawdescription | varchar(200) | Yes | NULL |

## Table structure for table raw\_material\_purchase

## Primary Key: ****rawid****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***Rawmaterialpurchaseid*** | int(10) | Yes | NULL |
| Rawid | int(10) | Yes | NULL |
| Vendorid | int(10) | Yes | NULL |
| Totquantity | int(10) | Yes | NULL |
| Totalamount | int(10) | Yes | NULL |
| Rate | int(10) | Yes | NULL |
| Purchasedate | date | Yes | NULL |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## Table structure for table raw\_material\_purchase\_details

## Primary Key: ****rawid****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***raw\_material\_purchase\_details\_id*** | int(11) | Yes | NULL |
| raw\_material\_purchase\_id | int(11) | Yes | NULL |
| raw\_material\_id | int(11) | Yes | NULL |
| Quantity | int(11) | Yes | NULL |
| Rate | double | Yes | NULL |
| Discount | double | Yes | NULL |

## Table structure for table stock\_details

## Primary Key: ****stock\_id****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***stock\_id*** | int(11) | Yes | NULL |
| product\_id | int(11) | Yes | NULL |
| Stock | int(11) | Yes | NULL |

## Table structure for table vendortable

## Primary Key: ****vendorid****

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***vendorid*** | int(10) | Yes | NULL |
| vendorname | varchar(20) | Yes | NULL |
| vendorcontno | varchar(10) | Yes | NULL |
| vendoremail | varchar(20) | Yes | NULL |

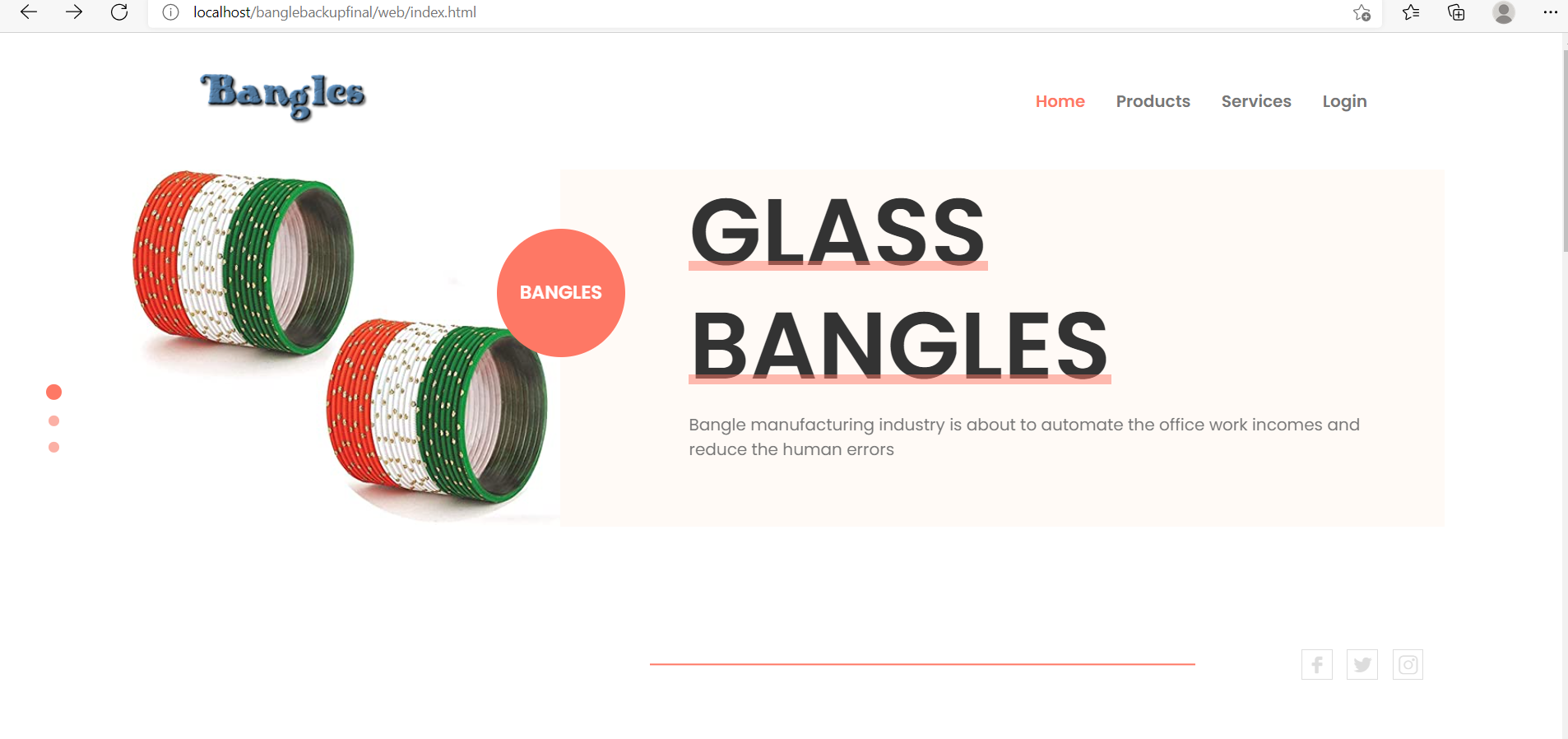
## Table structure for table vendor\_payment

## Primary Key: ****vendor\_payment\_id****

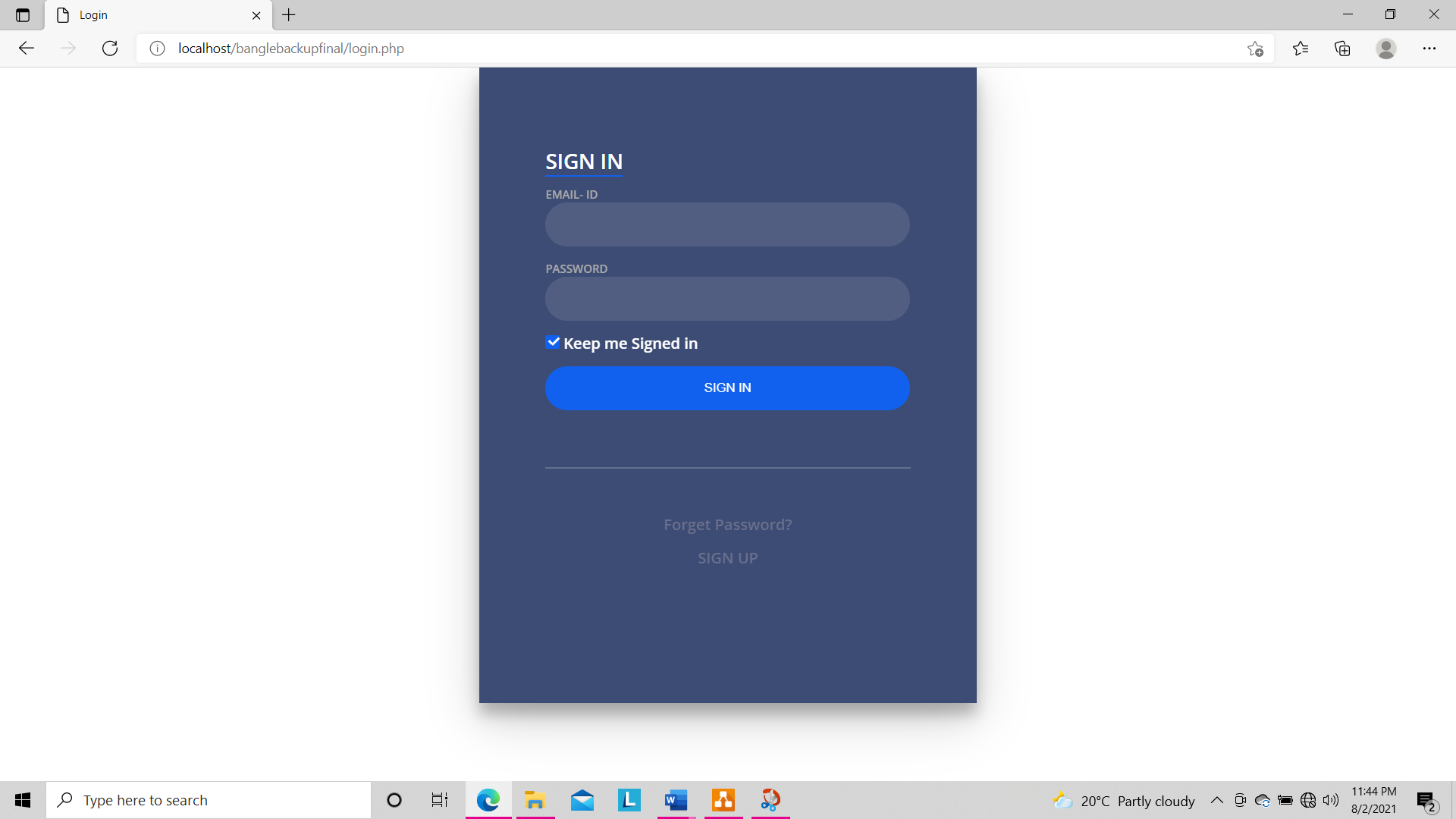
|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| ***vendor\_payment\_id*** | int(11) | Yes | NULL |
| vendor\_id | int(11) | Yes | NULL |
| amount | double | Yes | NULL |
| narration | varchar(500) | Yes | NULL |
| Date | date | Yes | NULL |

**8.SCREEN SHOTS**

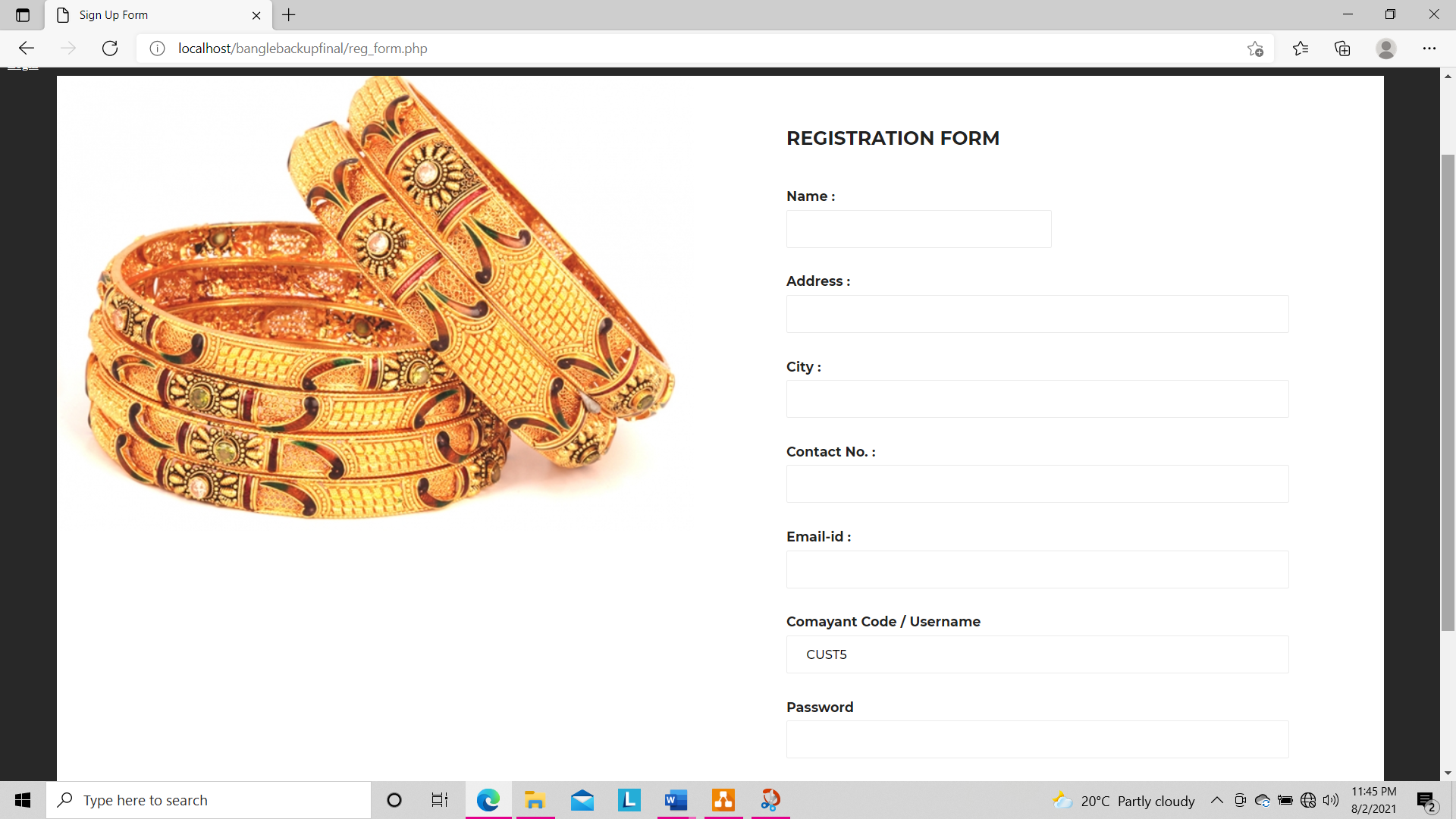
**Fig 1. A VIEW OF HOMEPAGE**



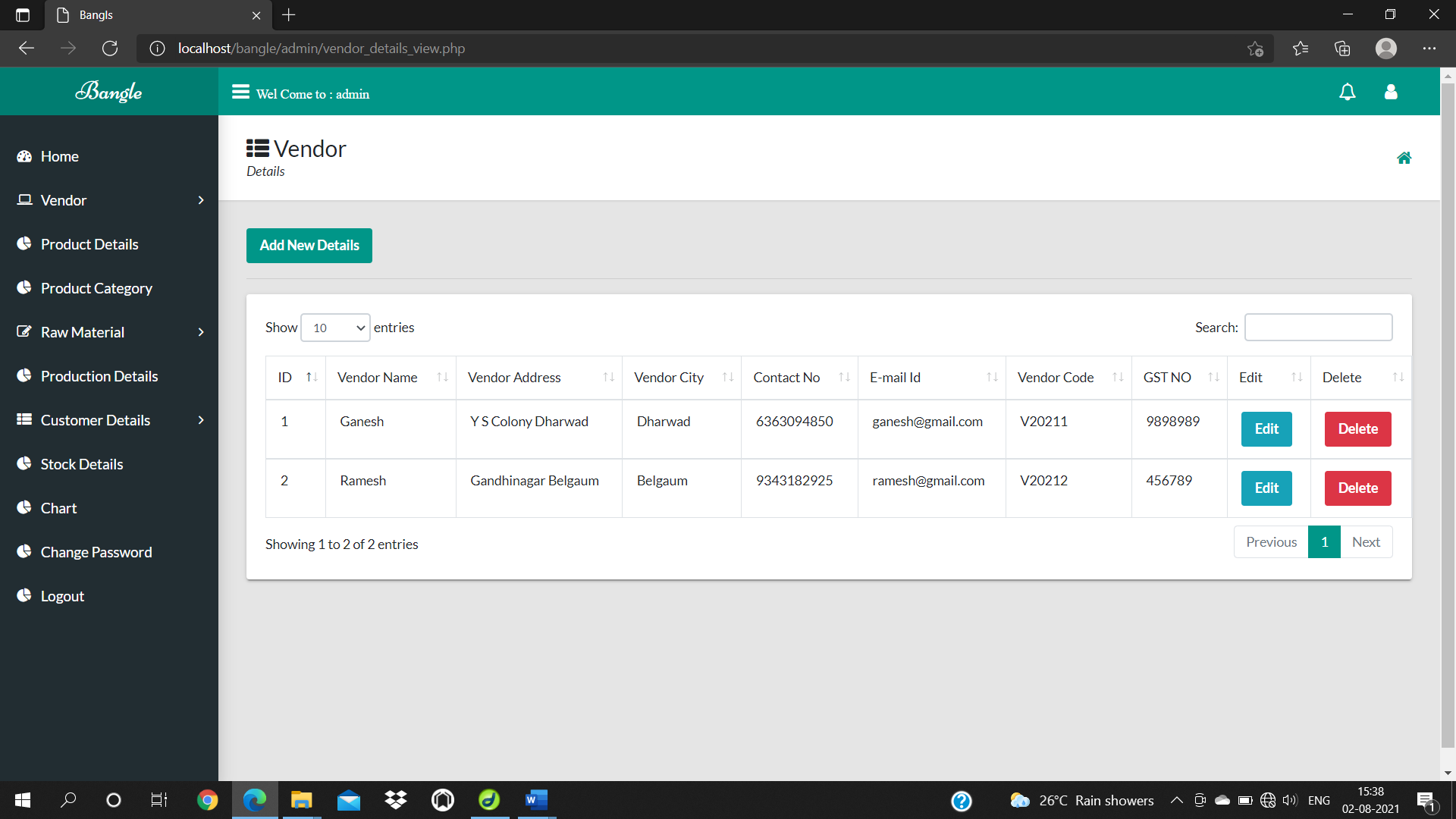
**Fig 2. A VIEW OF LOGIN PAGE**



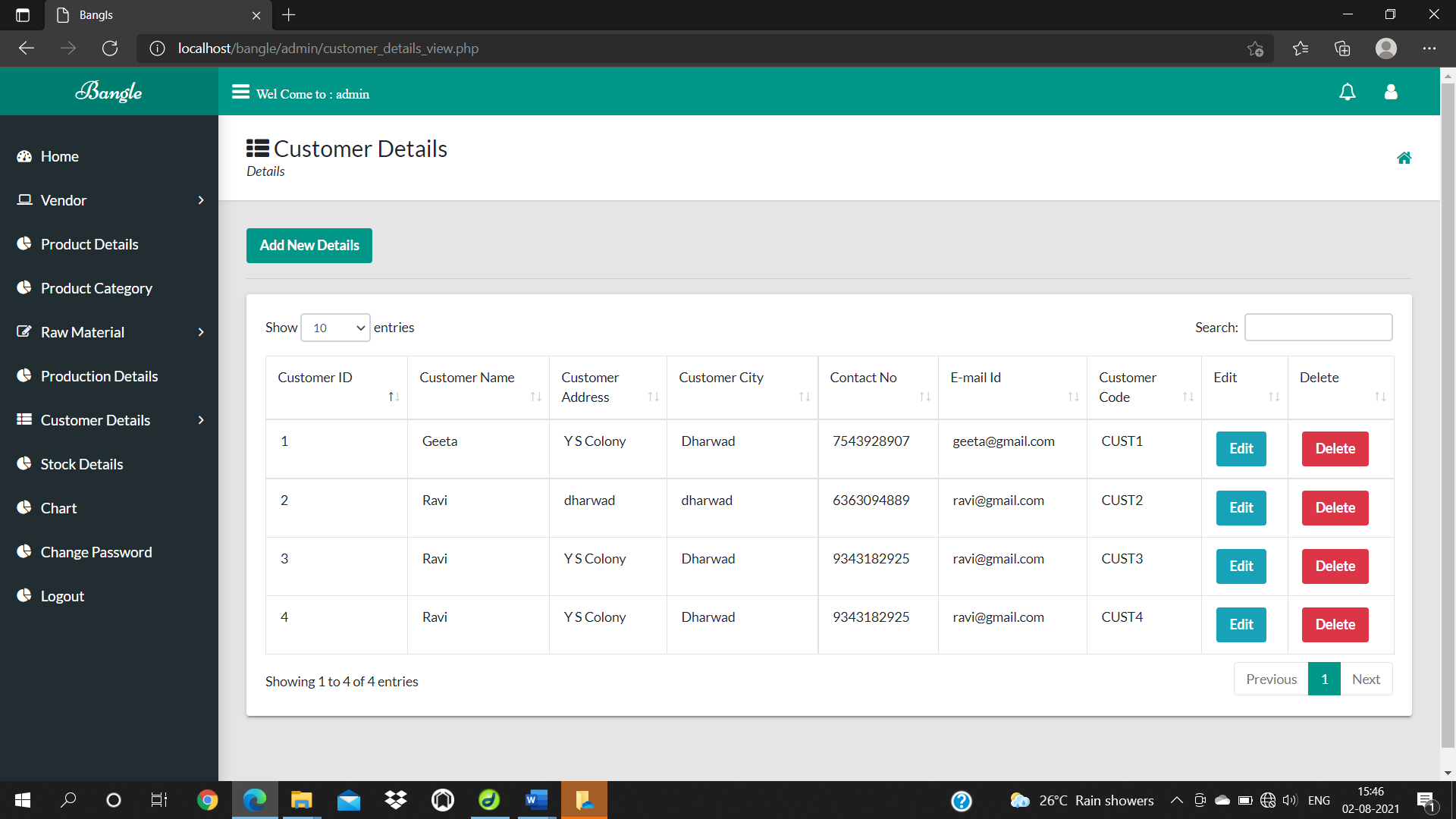
**Fig 3. A VIEW OF REGISTRATION PAGE**



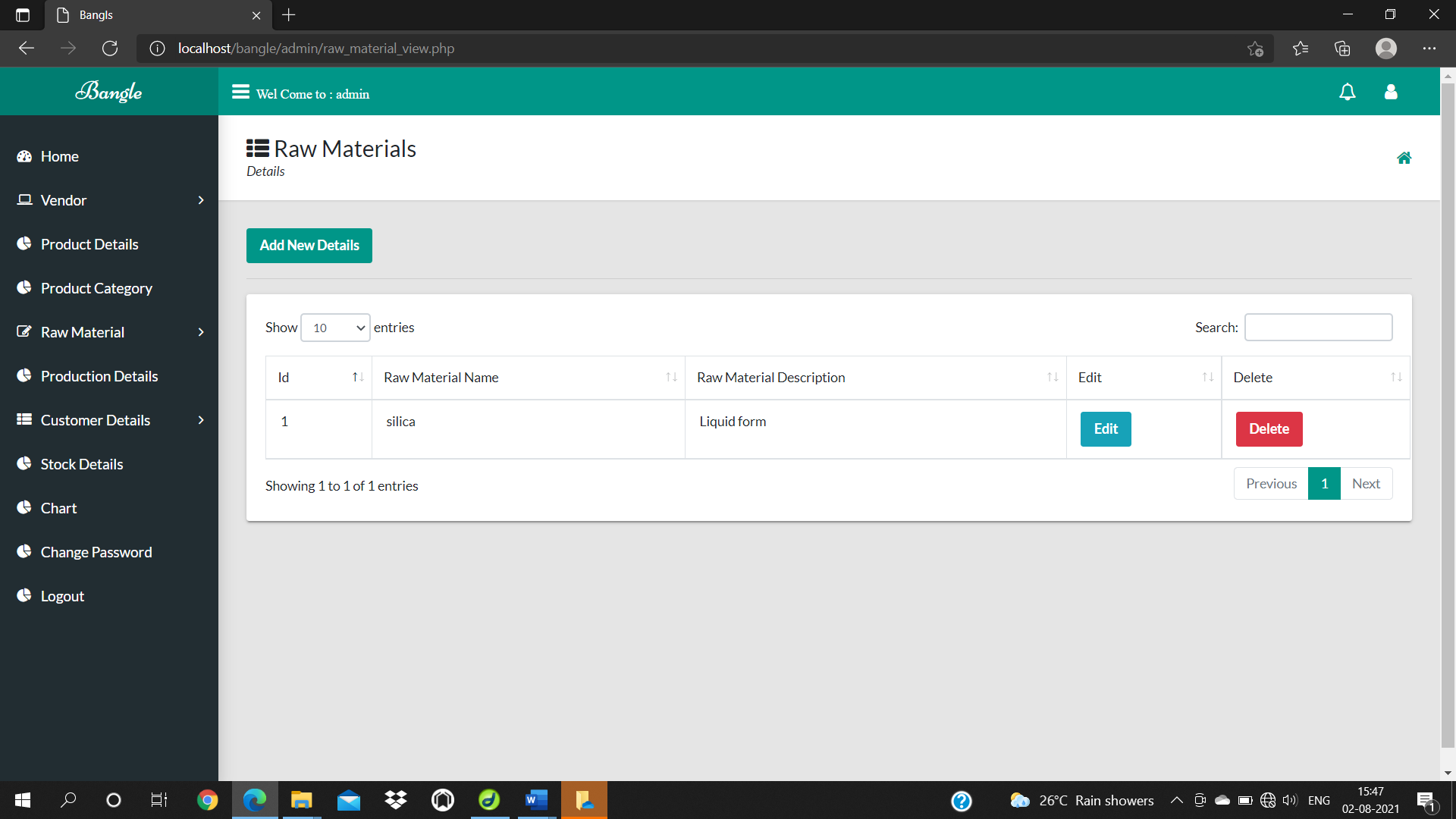
**Fig 4. A VIEW OF VENDOR TABLE**



**Fig 5.A VIEW OF CUSTOMER DETAILS TABLE**



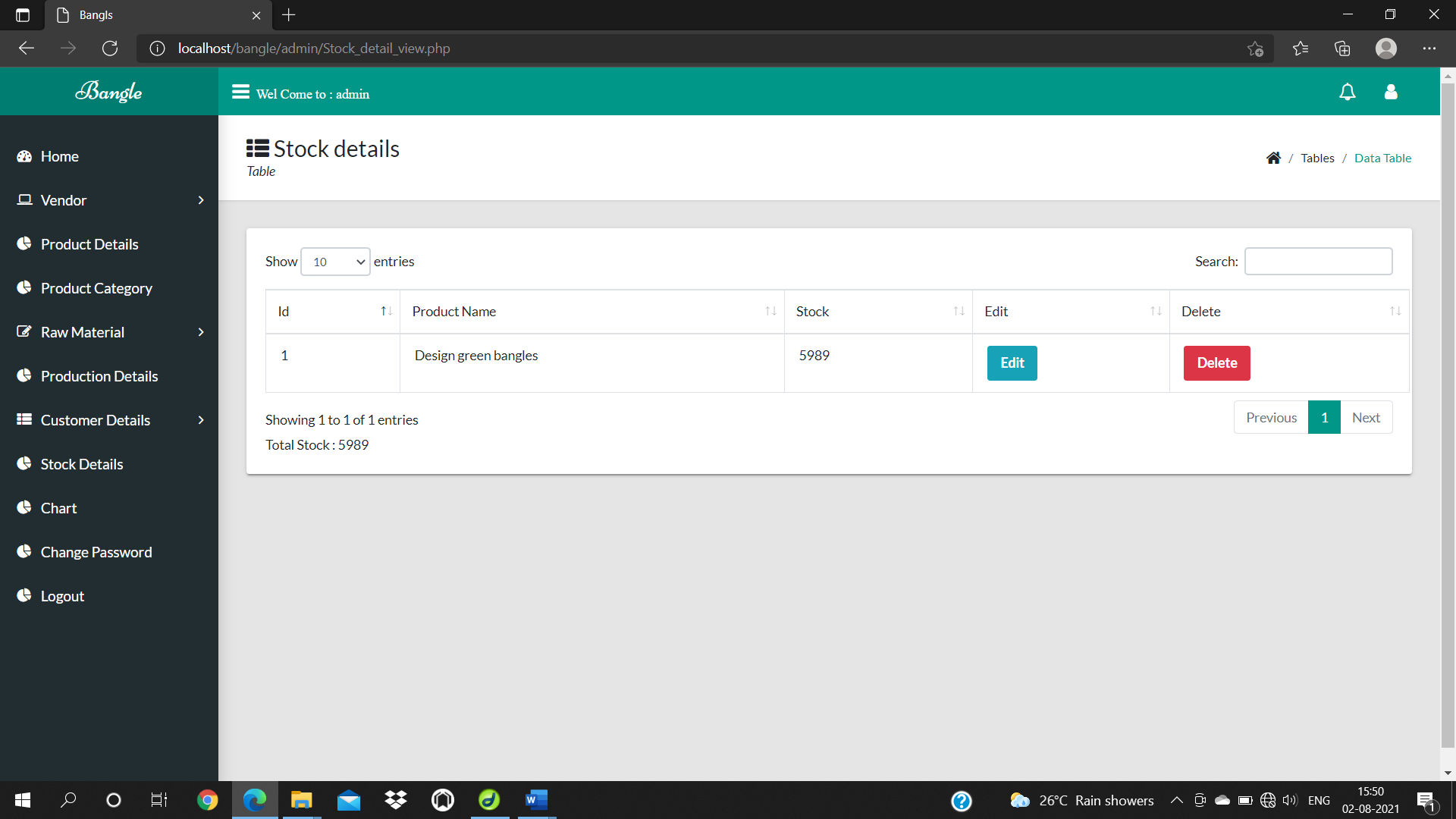
**Fig 6.A VIEW OF RAW MATERIAL TABLE**



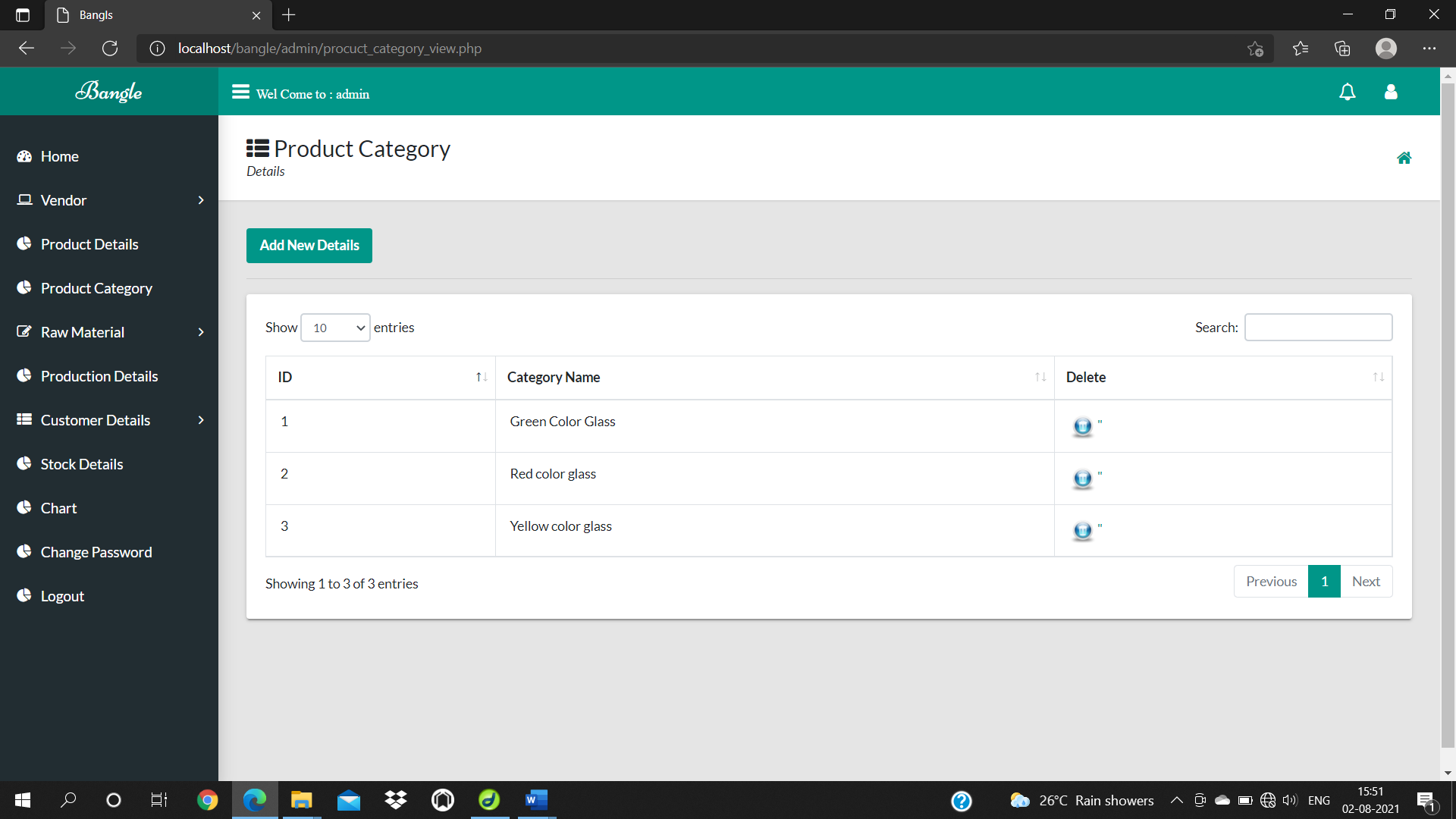
**Fig 7.A VIEW OF PRODUCTION TABLE**



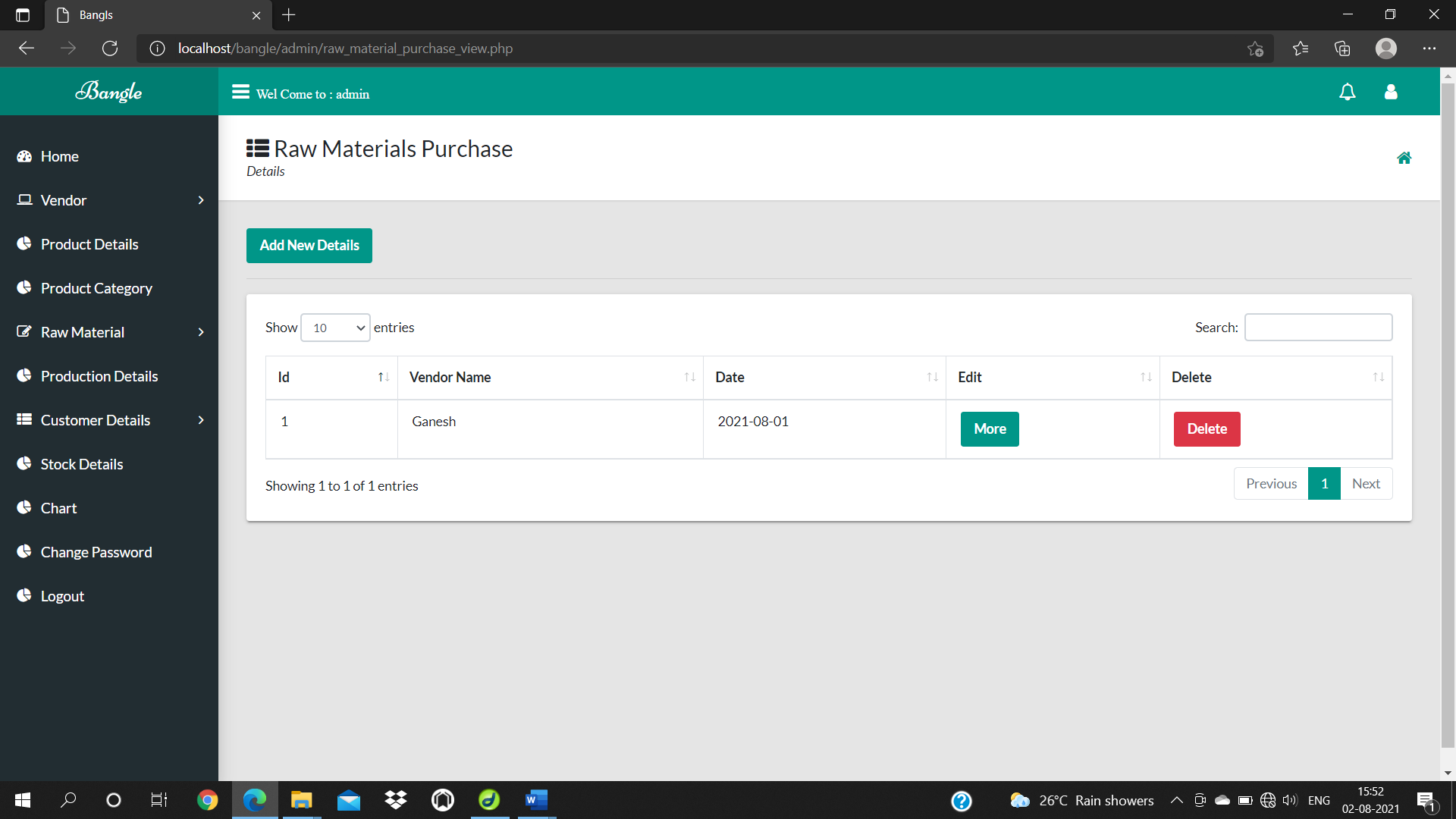
**Fig 8.A VIEW OF STOCK DETAILS TABLE**



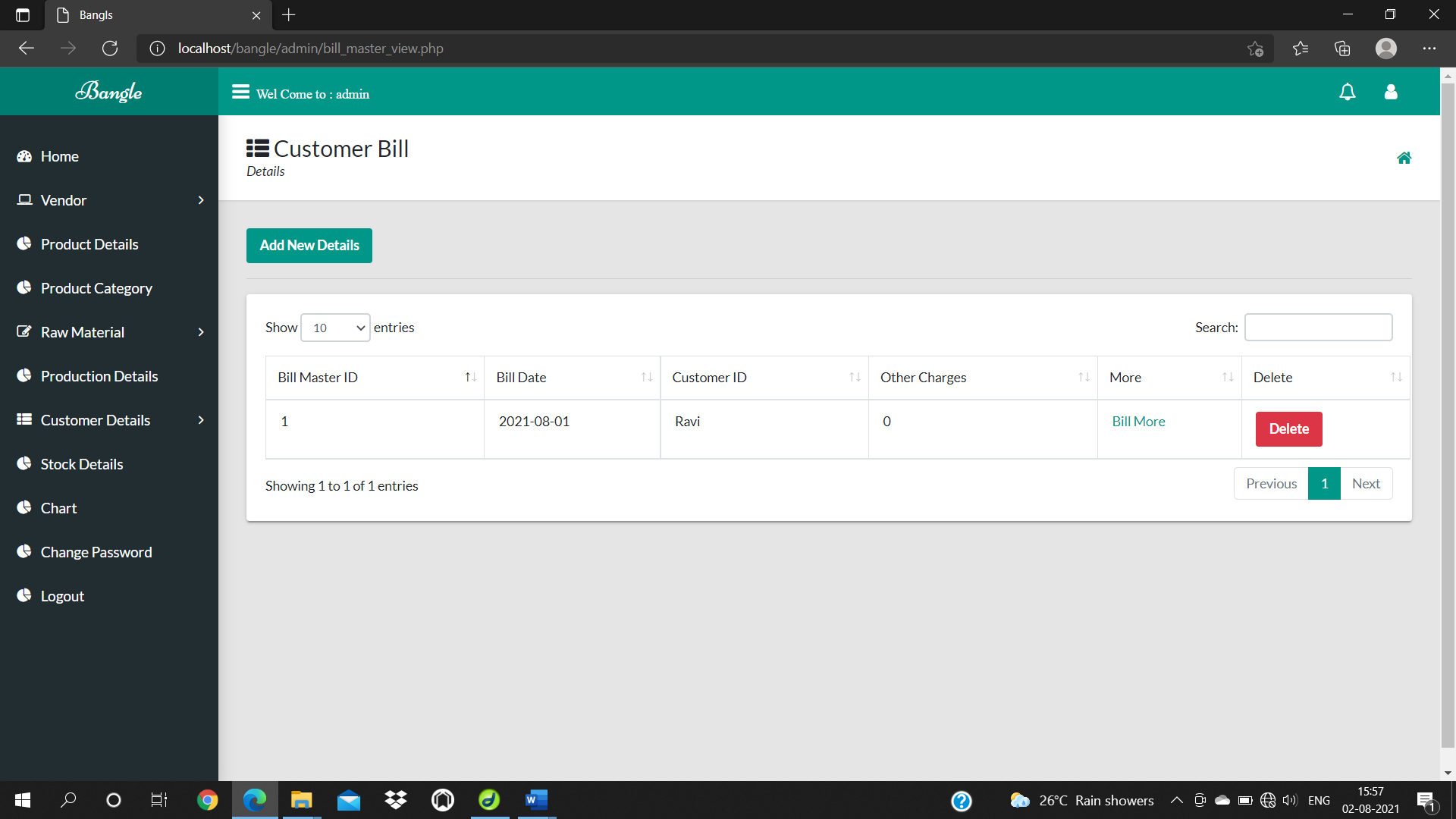
**Fig 9.A VIEW OF PRODUCT CATEGORY TABLE**



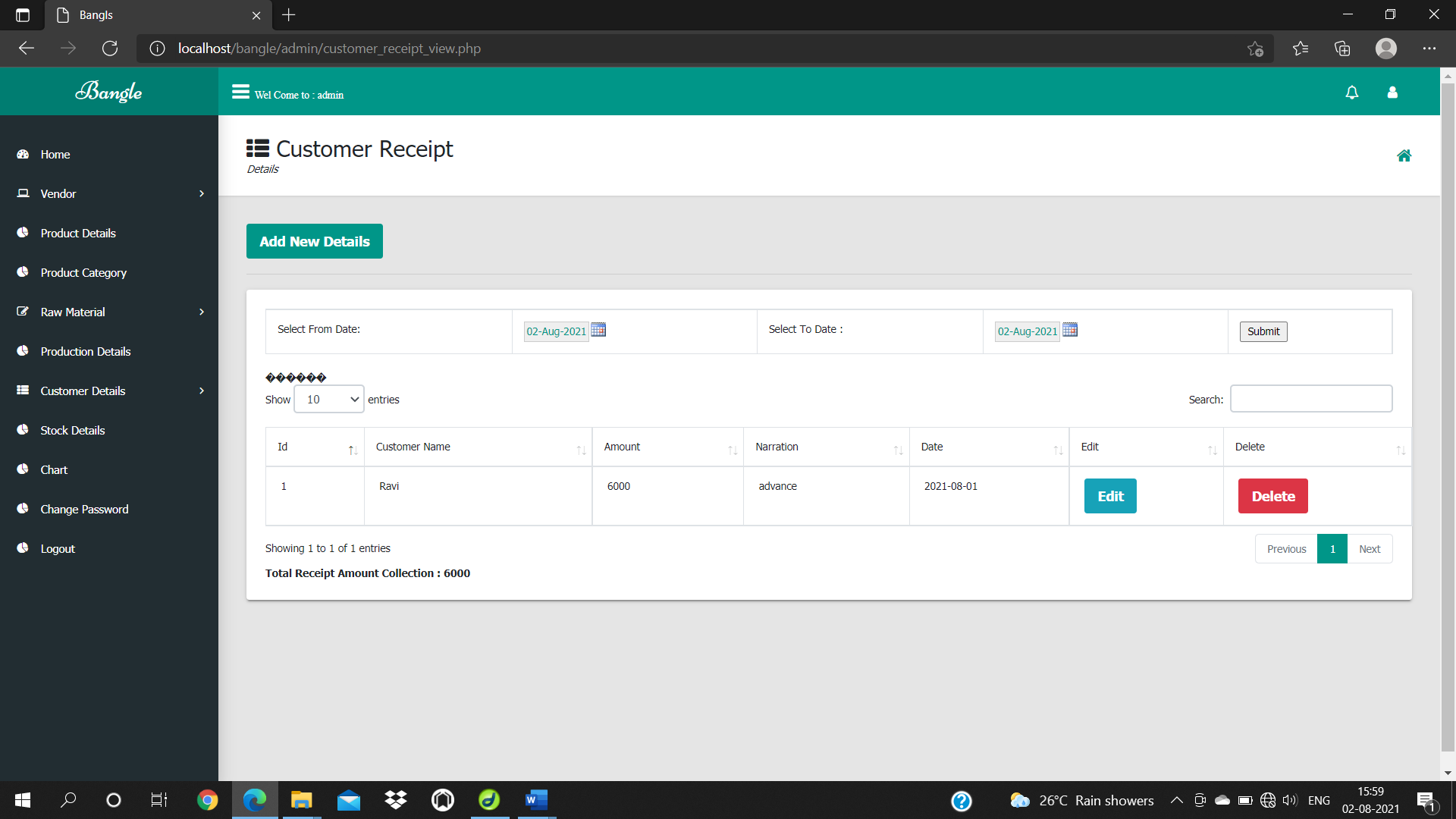
**Fig 10.A VIEW OF RAW MATERIAL PURCHASE TABLE**



**Fig 11.A VIEW OF CUSTOMER BILL TABLE**



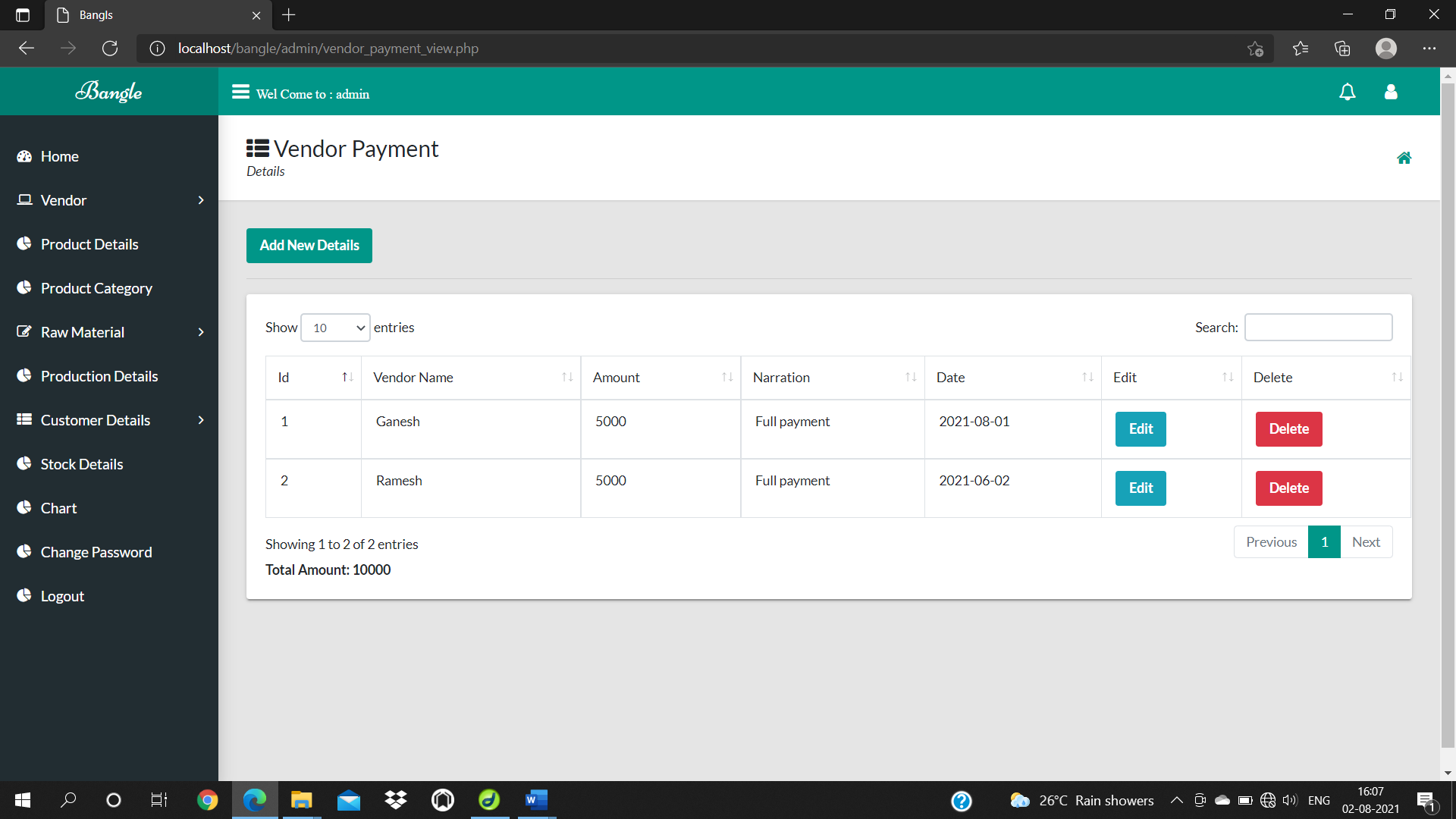
**Fig 12.A VIEW OF CUSTOMER RECEIPT TABLE**



**Fig 13.A VIEW OF CUSTOMER ORDER DETAILS TABLE**



**Fig 14.A VIEW OF VENDOR PAYMENT TABLE**



**9.SOURCE CODE**

**VENDOR 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">

Vendor Details <small></small>

</h1>

<ol class="breadcrumb">

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

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

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

</ol>

</div>

<div id="page-inner">

<div class="row">

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

<div class="panel panel-default">

<div class="panel-heading">

Vendor Details

</div>

<div class="panel-body">

<div class="row">

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

<?php include('val.php');?><form name="formID" ID="formID" method="post" action="vendor\_details\_insert.php">

<table width="423" height="198" border="0" align="center">

<tr>

<td>Vendor Name </td>

<td><input name="vendor\_name" type="text" id="vendor\_name" class="form-control validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>Vendor Address</td>

<td><textarea name="vendor\_address" id="vendor\_address" class="form-control validate[required]"></textarea></td>

</tr>

<tr>

<td>Vendor City </td>

<td><input name="vendor\_city" type="text" id="vendor\_city" class="form-control validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>Contact No</td>

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

</tr>

<tr>

<td>E-mail -Id</td>

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

</tr>

<tr>

<td>Vendor Code </td>

<td><input name="vendor\_code" type="text" id="vendor\_code" class="form-control validate[required,custom[onlyNumber]]"></td>

</tr>

<tr>

<td>GST -No </td>

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

</tr>

<tr>

<td colspan="2"><input name="Submit" type="submit" value="Submit" class="btn btn-success btn">

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

</tr>

</table>

</form>

</div>

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

</div>

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

</div>

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

</div>

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

</div>

<!-- /.panel --><?php include('footer.php');?>

</div>

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

</div>

</body>

</html>

**Insert**

<?php

$vendor\_name=$\_POST['vendor\_name'];

$vendor\_address=$\_POST['vendor\_address'];

$vendor\_city=$\_POST['vendor\_city'];

$contact\_no=$\_POST['contact\_no'];

$email\_id=$\_POST['email\_id'];

$vendor\_code=$\_POST['vendor\_code'];

$gst\_no=$\_POST['gst\_no'];

include('dbconnect.php');

$sql="insert into vendor\_details values(null,'$vendor\_name','$vendor\_address','$vendor\_city','$contact\_no','$email\_id','$vendor\_code','$gst\_no')";

mysql\_query($sql);

?>

<script language="javascript1.2">

alert('inserted');

document.location="Vendor\_detail\_form.php";

</script>

**Delete**

<?php

include('dbconnect.php');

$v\_id=$\_REQUEST['v\_id'];

$sql="delete from vendor\_details where vendor\_id='$v\_id' ";

mysql\_query($sql);

?>

<script>

alert("values is deleted..");

document.location="vendor\_details\_view.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">

Vendor Details <small></small>

</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">

Vendor Details Tables

</div>

<div class="panel-body">

<div class="table-responsive">

<a href="Vendor\_detail\_form.php" class="btn btn-primary">Add New Details</a>

<table class="table table-striped table-bordered table-hover" id="dataTables-example">

<thead>

<tr>

<td>ID</td>

<td>Vendor Name</td>

<td>Vendor Address </td>

<td>Vendor City </td>

<td>Contact No </td>

<td>E-mail Id </td>

<td>Vendor Code </td>

<td>GST NO </td>

<td>Edit</td>

<td>Delete</td>

</tr>

</thead>

<tbody>

<?php

include('dbconnect.php');

$sql=" select \* from vendor\_details ";

$res=mysql\_query($sql);

while($row=mysql\_fetch\_array($res))

{

?>

<tr>

<td>&nbsp;<?php echo $row['vendor\_id'];?></td>

<td>&nbsp;<?php echo $row['vendor\_name'];?></td>

<td>&nbsp;<?php echo $row['vendor\_address'];?></td>

<td>&nbsp;<?php echo $row['vendor\_city'];?></td>

<td>&nbsp;<?php echo $row['contact\_no'];?></td>

<td>&nbsp;<?php echo $row['email\_id'];?></td>

<td>&nbsp;<?php echo $row['vendor\_code'];?></td>

<td>&nbsp;<?php echo $row['gst\_no'];?></td>

<td>&nbsp;<a href="vendor\_details\_edit.php?v\_id=<?php echo $row['vendor\_id'];?>"><img src="../image/edit.jpg" width="30" height="30"></a></td>

<td>&nbsp;<a href="vendor\_details\_delete.php?v\_id=<?php echo $row['vendor\_id'];?>"onClick="return confirm('Are you sure want to delete')"><img src="../image/delete.png" width="30" height="30"></a></td>

</tr>

<?php

}

?>

</tbody>

</table>

</div>

</div>

</div>

<!--End Advanced Tables -->

</div>

</div>

<!-- /. ROW -->

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

</body>

</html>

**Update**

<?php

include('dbconnect.php');

$v\_id=$\_POST['v\_id'];

$vendor\_name=$\_POST['vendor\_name'];

$vendor\_address=$\_POST['vendor\_address'];

$vendor\_city=$\_POST['vendor\_city'];

$contact\_no=$\_POST['contact\_no'];

$email\_id=$\_POST['email\_id'];

$vendor\_code=$\_POST['vendor\_code'];

$gst\_no=$\_POST['gst\_no'];

$sql="update vendor\_details set vendor\_name='$vendor\_name',vendor\_address='$vendor\_address',vendor\_city='$vendor\_city',contact\_no='$contact\_no',email\_id='$email\_id',vendor\_code='$vendor\_code',gst\_no='$gst\_no' where vendor\_id='$v\_id' ";

mysql\_query($sql);

?>

<script>

alert("values is updted...");

document.location="vendor\_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">

Vendor Details <small></small>

</h1>

<ol class="breadcrumb">

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

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

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

</ol>

</div>

<div id="page-inner">

<div class="row">

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

<div class="panel panel-default">

<div class="panel-heading">

Vendor Details

</div>

<div class="panel-body">

<div class="row">

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

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

include('dbconnect.php');

$v\_id=$\_REQUEST['v\_id'];

$sql="select \* from vendor\_details where vendor\_id='$v\_id' ";

$res=mysql\_query($sql);

$row=mysql\_fetch\_array($res);

?>

<form name="formID" ID="formID" method="post" action="vendor\_details\_update.php">

<input type="hidden" name="v\_id" value="<?php echo $row['vendor\_id'];?>">

<table width="423" height="198" border="0" align="center">

<tr>

<td>Vendor Name </td>

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

</tr>

<tr>

<td>Vendor Address</td>

<td><textarea name="vendor\_address" id="vendor\_address" class="form-control validate[required]"><?php echo $row['vendor\_address'];?></textarea></td>

</tr>

<tr>

<td>Vendor City </td>

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

</tr>

<tr>

<td>Contact No</td>

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

</tr>

<tr>

<td>E-mail -Id</td>

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

</tr>

<tr>

<td>Vendor Code </td>

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

</tr>

<tr>

<td>GST -No </td>

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

</tr>

<tr>

<td colspan="2"><input name="Submit" type="submit" value="Submit" class="btn btn-success btn">

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

</tr>

</table>

</form>

</div>

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

</div>

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

</div>

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

</div>

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

</div>

<!-- /.panel --><?php include('footer.php');?>

</div>

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

</div>

</body>

</html>

**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 <small></small>

</h1>

<ol class="breadcrumb">

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

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

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

</ol>

</div>

<div id="page-inner">

<div class="row">

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

<div class="panel panel-default">

<div class="panel-heading">

Customer Details

</div>

<div class="panel-body">

<div class="row">

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

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

<form name="formID" ID="formID" method="post" action="customer\_detail\_insert.php">

<table width="423" height="198" border="0" align="center">

<tr>

<td>Customer Name </td>

<td><input name="customer\_name" type="text" id="customer\_name" class="form-control validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>Customer Address </td>

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

</tr>

<tr>

<td>Customer City</td>

<td><input name="customer\_city" type="text" id="customer\_city" class="form-control validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>Contact No </td>

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

</tr>

<tr>

<td>Email Id </td>

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

</tr>

<tr>

<td>Customer Code </td>

<td><input name="customer\_code" type="text" id="customer\_code"class="form-control validate[required,custom[onlyNumber]]"></td>

</tr>

<tr>

<td colspan="2"><input type="submit" name="Submit" value="Submit" class="btn btn-success btn">

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

</tr>

</table>

</form>

</div>

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

</div>

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

</div>

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

</div>

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

</div>

<!-- /.panel --><?php include('footer.php');?>

</div>

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

</div>

</body>

</html>

**Insert**

<?php

$customer\_id=$\_POST['customer\_id'];

$customer\_name=$\_POST['customer\_name'];

$customer\_address=$\_POST['customer\_address'];

$customer\_city=$\_POST['customer\_city'];

$contact\_no=$\_POST['contact\_no'];

$email\_id=$\_POST['email\_id'];

$customer\_code=$\_POST['customer\_code'];

include('dbconnect.php');

$sql="insert into customer\_details values(null,'$customer\_name','$customer\_address','$customer\_city','$contact\_no','$email\_id','$customer\_code')";

mysql\_query($sql);

?>

<script language="javascript1.2">

alert(' inserted');

document.location="Customer\_detail\_form.php";

</script>

**Delete**

<?php

include('dbconnect.php');

$c\_id=$\_REQUEST['c\_id'];

$sql="delete from customer\_details where customer\_id='$c\_id' ";

mysql\_query($sql);

?>

<script>

alert("values is deleted..");

document.location="customer\_details\_view.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">

Customer Details <small></small>

</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">

Customer Details Tables

</div>

<div class="panel-body">

<div class="table-responsive">

<a href="Customer\_detail\_form.php" class="btn btn-primary">Add New Details</a>

<table class="table table-striped table-bordered table-hover" id="dataTables-example">

<thead>

<tr>

<td width="86">Customer ID </td>

<td width="86">Customer Name </td>

<td width="63">Customer Address </td>

<td width="94">Customer City </td>

<td width="60">Contact No</td>

<td width="58">E-mail Id </td>

<td width="21">Customer Code </td>

<td width="24">Edit</td>

<td width="21">Delete</td>

</tr>

</thead>

<tbody>

<?php

include('dbconnect.php');

$sql=" select \* from customer\_details ";

$res=mysql\_query($sql);

while($row=mysql\_fetch\_array($res))

{

?>

<tr>

<td>&nbsp;<?php echo $row['customer\_id'];?></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>&nbsp;<a href="customer\_details\_edit.php?c\_id=<?php echo $row['customer\_id'];?>"><img src="../image/edit.jpg" width="30" height="30"></a></td>

<td>&nbsp;<a href="customer\_details\_delete.php?c\_id=<?php echo $row['customer\_id'];?> "onClick="return confirm('Are you sure want to delete')"><img src="../image/delete.png" width="30" height="30"></a></td>

</tr>

<?php

}

?>

</tbody>

</table>

</div>

</div>

</div>

<!--End Advanced Tables -->

</div>

</div>

<!-- /. ROW -->

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

</body>

</html>

**Update**

<?php

include('dbconnect.php');

$c\_id=$\_POST['c\_id'];

$customer\_id=$\_POST['customer\_id'];

$customer\_name=$\_POST['customer\_name'];

$customer\_address=$\_POST['customer\_address'];

$customer\_city=$\_POST['customer\_city'];

$contact\_no=$\_POST['contact\_no'];

$email\_id=$\_POST['email\_id'];

$customer\_code=$\_POST['customer\_code'];

$sql="update customer\_details set customer\_name='$customer\_name',customer\_address='$customer\_address',customer\_city='$customer\_city',contact\_no='$contact\_no',email\_id='$email\_id',customer\_code='$customer\_code' where customer\_id='$c\_id' ";

mysql\_query($sql);

?>

<script>

alert("values is updeted...");

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 <small></small>

</h1>

<ol class="breadcrumb">

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

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

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

</ol>

</div>

<div id="page-inner">

<div class="row">

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

<div class="panel panel-default">

<div class="panel-heading">

Customer Details

</div>

<div class="panel-body">

<div class="row">

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

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

<?php

include('dbconnect.php');

$c\_id=$\_REQUEST['c\_id'];

$sql="select \* from customer\_details where customer\_id='$c\_id' ";

$res=mysql\_query($sql);

$row=mysql\_fetch\_array($res);

?>

<form name="formID" ID="formID" method="post" action="customer\_details\_update.php">

<input type="hidden" name="c\_id" value="<?php echo $row['customer\_id'];?>">

<table width="423" height="198" border="0" align="center">

<tr>

<td>Customer Name </td>

<td><input name="customer\_name" type="text" id="customer\_name" value="<?php echo $row['customer\_name'];?>"></td>

</tr>

<tr>

<td>Customer Address </td>

<td><input name="customer\_address" type="text" id="customer\_address" value="<?php echo $row['customer\_address'];?>"></td>

</tr>

<tr>

<td>Customer City</td>

<td><input name="customer\_city" type="text" id="customer\_city" value="<?php echo $row['customer\_city'];?>"></td>

</tr>

<tr>

<td>Contact No </td>

<td><input name="contact\_no" type="text" id="contact\_no" value="<?php echo $row['contact\_no'];?>"></td>

</tr>

<tr>

<td>Email Id </td>

<td><input name="email\_id" type="text" id="email\_id" value="<?php echo $row['email\_id'];?>"></td>

</tr>

<tr>

<td>Customer Code </td>

<td><input name="customer\_code" type="text" id="customer\_code" value="<?php echo $row['customer\_code'];?>"></td>

</tr>

<tr>

<td colspan="2"><input type="submit" name="Submit" value="Submit" class="btn btn-success btn">

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

</tr>

</table>

</form>

</div>

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

</div>

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

</div>

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

</div>

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

</div>

<!-- /.panel --><?php include('footer.php');?>

</div>

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

</div>

</body>

</html>

**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.
* Black box testing.

# White-Box Testing and Black-Box Testing

**White Box Testing:** White 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.

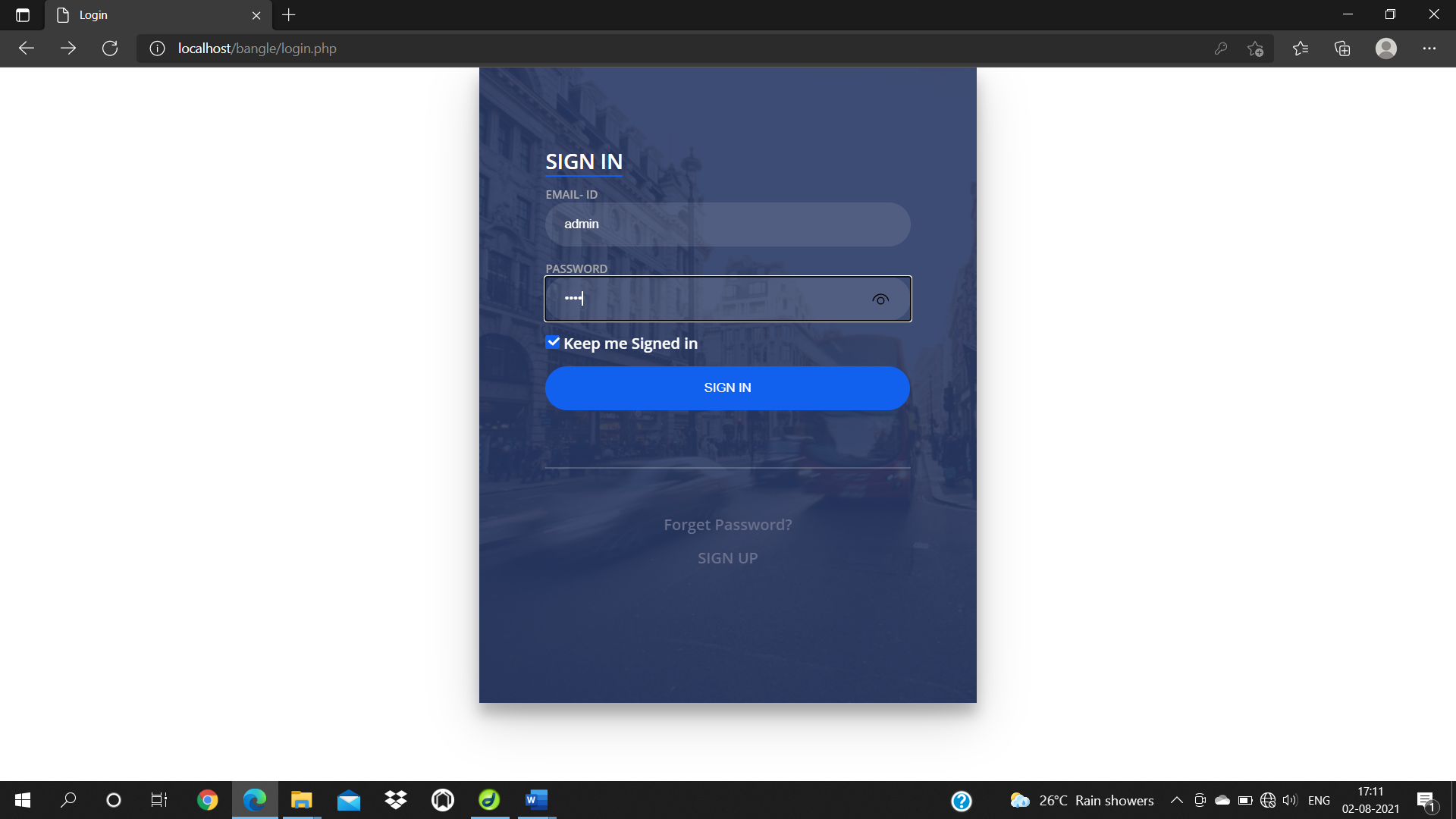
**Integration Testing:** Integration testing is a systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated with interfacing. This testing is done using the bottom-up approach to integrate the software components of the software system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

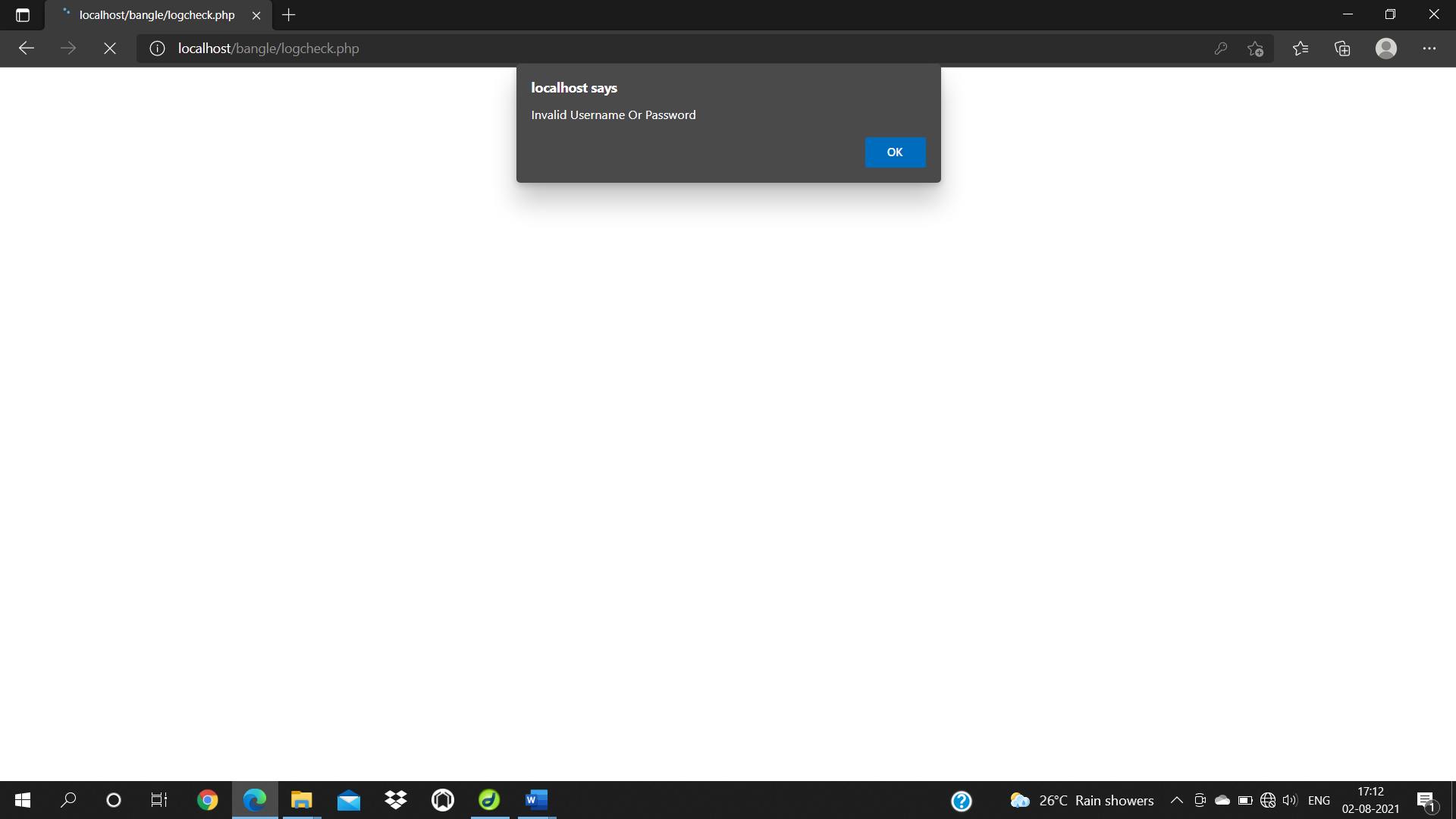
**Acceptance Testing:** User acceptance of the system is the key factor for the success of any system. This is done by user. The system is given to the user and they test it with live data. Acceptance testing involves the planning and execution of functional test. Performance tests, stress tests in order 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**



*Fig (10.2.1): login*



*Fig (10.2.2): invalid username or password*

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

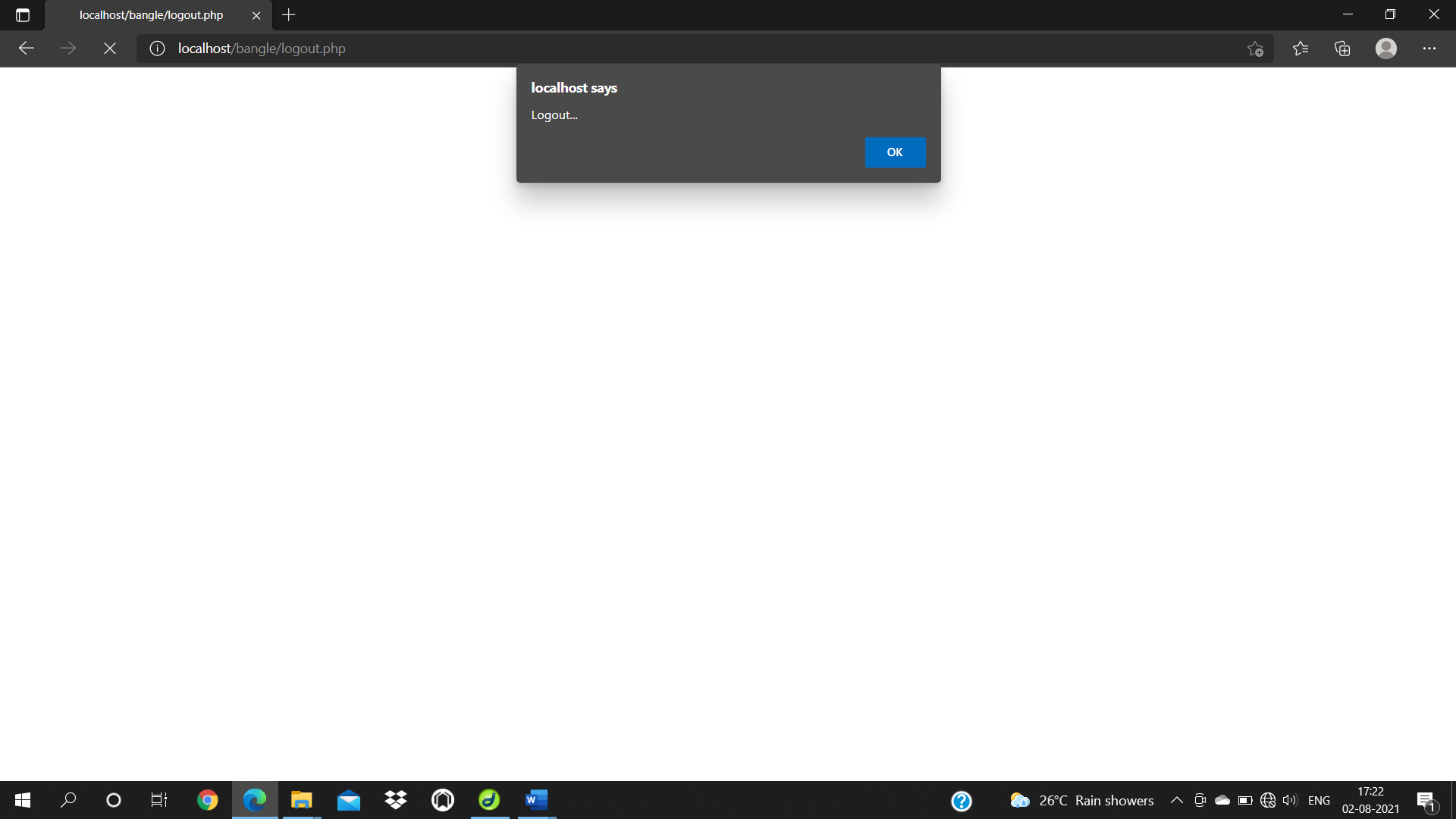
**VENDOR Table**

Adding a new value to the table



*Fig (10.2.3): Vendor Details*

**Logout Form**



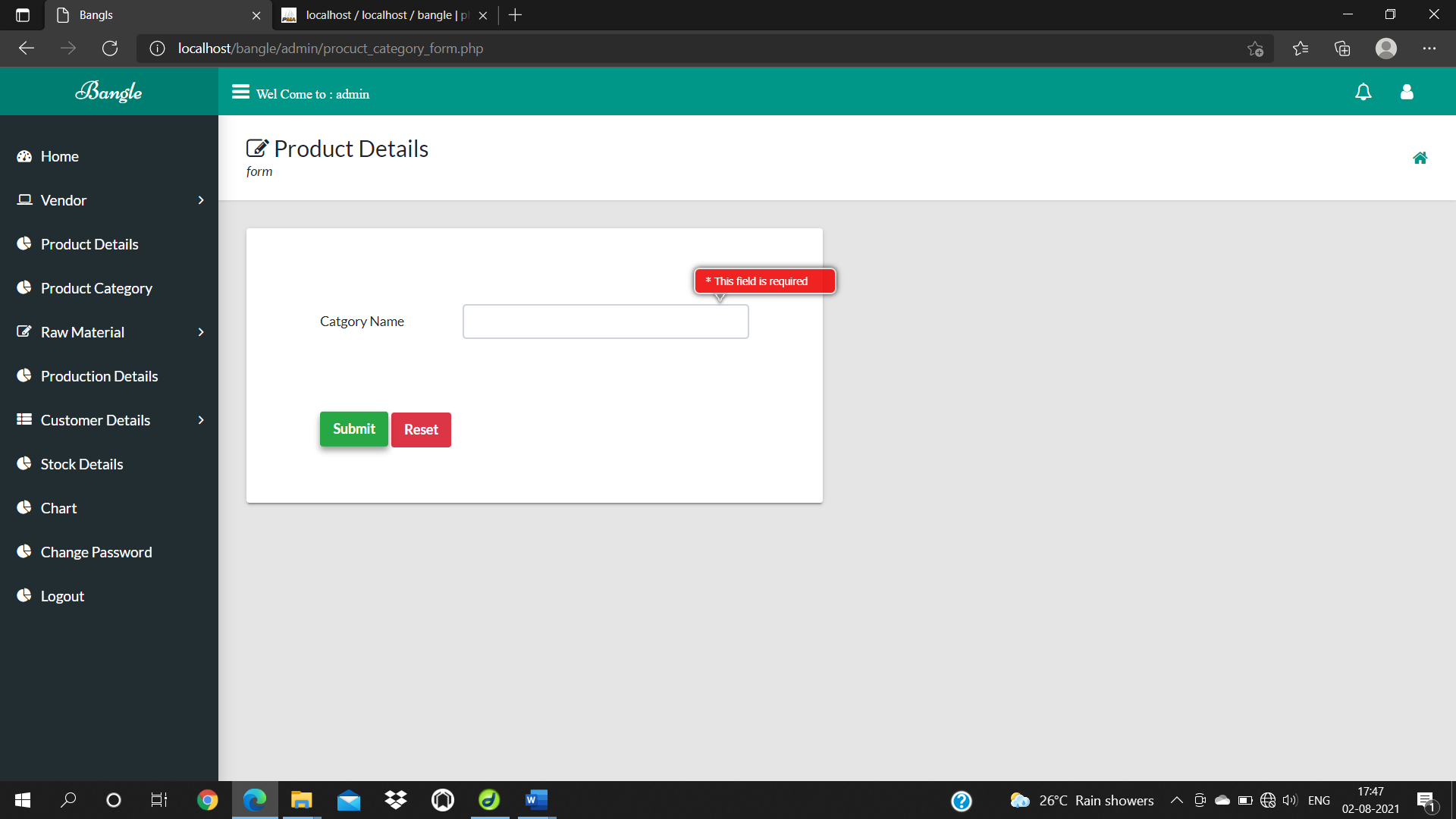
*Fig (10.2.4): Logout*

**Product Table**

****

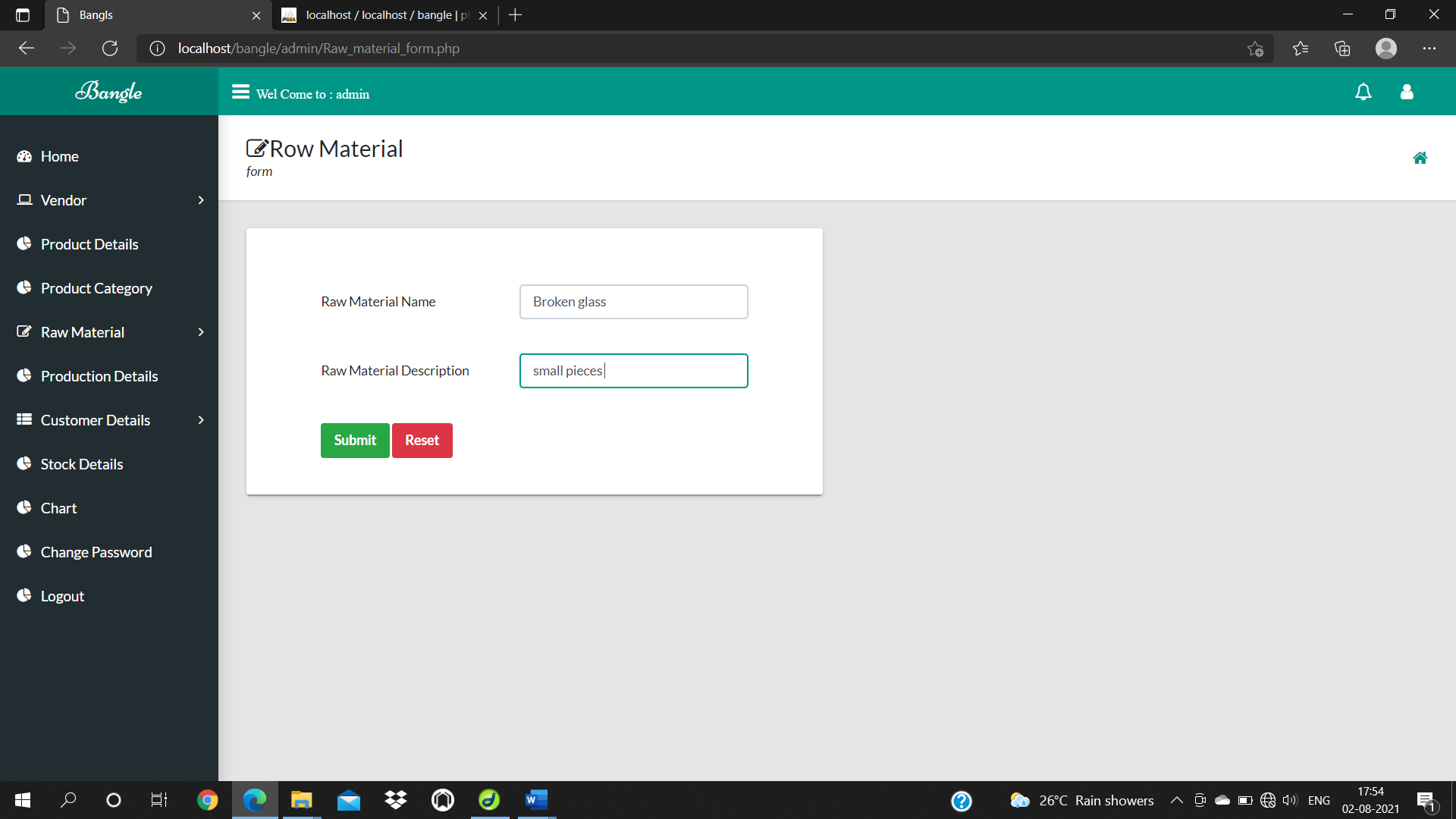
*Fig (10.2.5): Vendor Details*

**Product Category Details Table**



*Fig (10.2.6): Product Category Details*

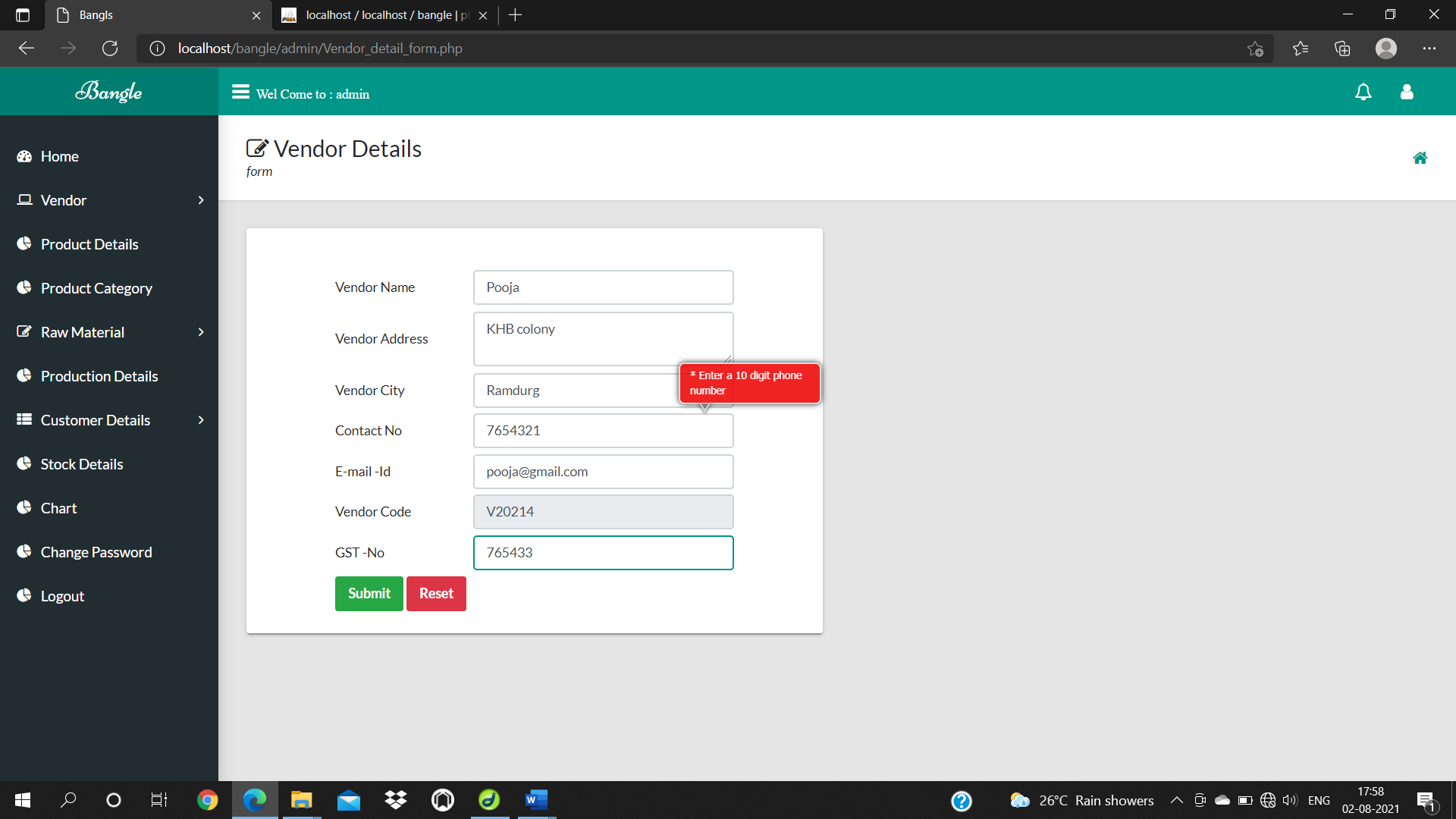
**Raw Material Details Table**



*Fig (10.2.6): Raw material Details*

**Vendor Table**

Contact number



*Fig (10.2.7): Vendor 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 (10.2.6) |
| 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) |

**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.

**12.BIBLIOGRAPHY**

**12.1 Books**

1. “Software Engineering”, by Ian Somerville, Sixth Edition, Pearson Education Ltd 2007.
2. “Web Programming”, by ‘Chris Bates’ Wiley Dreamtech India, 2nd Edition.
3. IEEE SRS Format.
4. Database Management Systems, by Navathe.

**12.2 Websites**

1. <http://www.hotscripts.com/category/php/> for Php
2. <http://en.wikipedia.org/wiki/PHP> for Php.

3. <http://www.mysql.com/click.php?e=35050> for MySql.

**References**

1.<https://www.w3schools.com/>

2.<https://getootstrap.com/>

3.<https://www.youtube.com/>

4.<https://stackoverflow.com/>

5.<https://docs.djangoproject.com/en/2.1/>