**1. PREAMBLE**

**1.1 Introduction**

MILKEES is a web-based milk subscription application software for Epic monk. The main aim of " MILKEES " is to eliminate the conventional form of milk delivery which is via phone call or word of mouth subscription. It maintains the details of customer, payments, subscription details, sales report, milk required for the next day etc.

**1.2 About the Organisation**

Epic monk is a Marketing Process Outsourcing firm based in Cochin, Kerala with not so nerdy, but creative and practically skilled people. Their services include in 4 verticals, viz; Online Marketing, Offline Marketing, Mobile App and Research & Analysis.

**1.3 Objective of the Project**



The objective of the project is to develop a web application for a dairy farm which makes it easy for the customer to subscribe for milk online. The product came into origin when there was a need for a more organized system for milk delivery. There was no subscription model as of now.

**1.4 Scope of the Project**

This system gives the current system a new model which is organized and user friendly. The dairy farm administrator can view the amount of packets being sold. They can predict beforehand how many packets would be required to be sold the next day. It keeps the dairy farm administrator more aware about the whole process and they can analyze their productivity and sales.

There are two users, admin who manages and monitors the entire system and customer who will be able to register online.



**2. SOFTWARE REQUIREMENT SPECIFICATION**

**2.1 System Study**

**2.1.1 Existing System**

Currently the customers have no formal registration process to enrol for monthly or weekly subscription of milk. The customers are delivered milk mainly by word of mouth schedule or by calling up the dairy farm owner to send the required packets of milk. Also if one is a regular customer and they are not available to receive the milk for the next day,



then this can result in wastage of milk.

**2.1.2 Proposed System**

By using the MILKEES website, a customer can register to the website and log onto it. The customer can decide what mode of delivery is required by the user ie, it can either be monthly or weekly. This kind of system makes monitoring easier. Also there are added facilities for customer to edit the delivery day provided the updation must be done before hand night itself. The cost of that milk will be adjusted with the next cycle of

subscription. This system also results in reduced wastage of milk. The owner can monitor the entire process as well as estimate the amount of packets required for each day based on the customer schedule or orders.

**Advantages**

 Administrator can monitor the entire system.

 Customers can subscribe online.

 Ensures the customer satisfaction.

**2.2 Functional Specifications**

The system has two modules:

**Module 1: Customer Management**

This module deals with the registration of customers which includes subscribing for monthly or weekly type of subscription. Monthly means the milk will be delivered every day for 30 days and weekly means milk would be delivered for the next 7 days. Once the customer subscribes and makes payment offline/online they can log onto the site and view their plan. Also they can edit their details and in case they do not require milk for a range of days, the updation should be made beforehand itself and that amount will be adjusted with the next cycle of subscription.



**Module 2: Administration**

The module deals with the privileges an administrator gets. The administrator can view the members and their subscription. Once they see that the customer has registered and if payment has been done, the administrator (dairy farm owner) would approve the customer (if payment done offline) and offer service selected by the customer. Also the customer can view the amount of milk required for the next day and the sales report for each month. The administrator basically monitors and verifies the whole process.

**2.3 User Characteristics**

**Admin**

The admin logs into the system. The admin has certain privileges such verifying and approving the registered customers who have made payment. Admin can view all the customer details and sales report. Admin has full access to the system and can monitor the entire system.

**Customer**

A customer can register online and subscribe for either monthly or weekly kind of subscription. Once the customer makes the payment via offline/online to the dairy farm, the admin approves the customer and then the customer can view their plan and if they want to edit their details they can do so. Also if they do not require milk for the next day, they should request for cancellation beforehand itself.

**2.4 System Specification**

**2.4.1 Software Interfaces**

 Development Configuration

 Machine– Windows 7 or above /Linux distros, MySqL

 Implementation Configuration

 Client machine- Windows 7 or above /Linux distros



 Server Machine- Windows 7 or above/Linux distros, WAMP or XAMP and MySqL

**2.4.2 Hardware Requirements**

 Development Configuration

 Machine (*Minimum Requirement*)

 Processor –Intel i3, AMD and above version.

 RAM – 512MB and above.

 Hardware Device – A Monitor and Keyboard with Mouse.

 Hard disk – Min 1 GB.

 Implementation Configuration

 Client Machine (*Minimum Requirement*)

 Hardware Device – A computer with a web browser.

 RAM – 512 MB and above (Recommended 1GB).

 Hard Disk – Min 60 MB

**2.4.3 About the Software/Tools.**

**JAVA**

Java is a set [of computer software](https://en.wikipedia.org/wiki/Computer_software) and specifications developed by[S](https://en.wikipedia.org/wiki/Sun_Microsystems)un

[Microsystems,](https://en.wikipedia.org/wiki/Sun_Microsystems) which was later acquired by th[e Oracle Corporation](https://en.wikipedia.org/wiki/Oracle_Corporation), that provides a system for developing[application software](https://en.wikipedia.org/wiki/Application_software) and deploying it in a[c](https://en.wikipedia.org/wiki/Cross-platform)ross- [platfo](https://en.wikipedia.org/wiki/Cross-platform)rm computing environment. Java is used in a wide variety of[comput](https://en.wikipedia.org/wiki/Computing_platform)ing [platfo](https://en.wikipedia.org/wiki/Computing_platform)rms from[embedded devices](https://en.wikipedia.org/wiki/Embedded_device) a[nd mobile phones](https://en.wikipedia.org/wiki/Mobile_phone) [to enterpr](https://en.wikipedia.org/wiki/Enterprise_server)ise [servers](https://en.wikipedia.org/wiki/Enterprise_server) and[supercompute](https://en.wikipedia.org/wiki/Supercomputer)[rs. Java applets](https://en.wikipedia.org/wiki/Java_applet), which are less common than standalone Java applications, run in secure,[sandboxed](https://en.wikipedia.org/wiki/Sandbox_(computer_security)) environments to provide many features of native applications and can be embedded in[HTML](https://en.wikipedia.org/wiki/HTML) pages.



JavaScript often abbreviated as JS, is [a high-leve](https://en.wikipedia.org/wiki/High-level_programming_language)[l, dynami](https://en.wikipedia.org/wiki/Dynamic_programming_language)[c, weakl](https://en.wikipedia.org/wiki/Weak_typing)y [type](https://en.wikipedia.org/wiki/Weak_typing)[d, prototype-based,](https://en.wikipedia.org/wiki/Prototype-based_programming) [multi-paradigm,](https://en.wikipedia.org/wiki/Multi-paradigm_programming_language) a[nd interprete](https://en.wikipedia.org/wiki/Interpreted_language)[d programming language](https://en.wikipedia.org/wiki/Programming_language). Along[side HTM](https://en.wikipedia.org/wiki/HTML)L a[nd CSS](https://en.wikipedia.org/wiki/CSS), JavaScript is one of the three core technologies of[World Wide Webcontent production](https://en.wikipedia.org/wiki/World_Wide_Web). It is used to make webpages interactive and provide online programs, including video games. The majority of[websites](https://en.wikipedia.org/wiki/Website) employ it, and all modern[web browse](https://en.wikipedia.org/wiki/Web_browser)rs support it without the need for[plug-ins](https://en.wikipedia.org/wiki/Browser_extension) by means of a built-[in JavaScript engi](https://en.wikipedia.org/wiki/JavaScript_engine)ne.

**JSP**

JSP allows Java code and certain predefined actions to be interleaved with static web markup content, such as HTML, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, contain Java bytecode rather than machine code. Like any other Java program, they must be executed within a Java virtual machine (JVM) that interacts with the server's host operating system to provide an abstract, platform-neutral environment.

**Mysql**

MySQL is the world’s most popular open source database, enabling the

cost effective delivery of reliable, high-performance, and scalable web-based, cloud and embedded database applications, including all five of the top five websites. A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.



**3. SOFTWARE MODELING**

**3.1 Uml Diagrams**

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. It was initially started to capture the behavior of complex software and non-software system and now it has become an OMG standard.

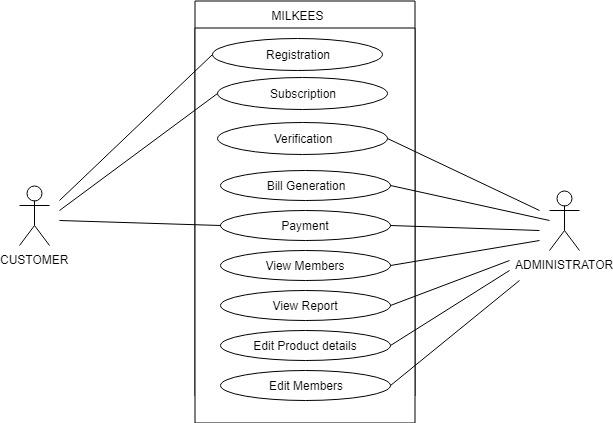
**3.1.1 Use Case Diagram**

The purpose of use case diagram is to capture the dynamic aspect of a system. But this definition is too generic to describe the purpose. Because other four diagrams (activity, sequence, collaboration and State chart) are also having the same purpose. So we will look into some specific purpose which will distinguish it from other four diagrams. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So when a system is analyzed to gather its functionalities use cases are prepared and actors are identified



.

Figure 3.1 Use Case Diagram



The below use case diagram shows the available operations of the system which are available for admin and customer.

**Admin**

1. Approve customers.

2. Verify payment.

3. View details.

4. Generate bill

5. Edit product details

**Customer**

1. Register online.

2. Make payment.

4. Request for cancellation.

**3.1.2 Class Diagram**

Class diagrams are arguably the most used UML diagram type. It is the main building block of any object oriented solution. It shows the classes in a system, attributes and operations of each class and the relationship between each class. In most modeling tools, a class has three parts, name at the top, attributes in the middle and operations or methods at the bottom. In large systems with many related classes, classes are grouped together to create class diagrams. Different relationships between classes are shown by different types of arrows.

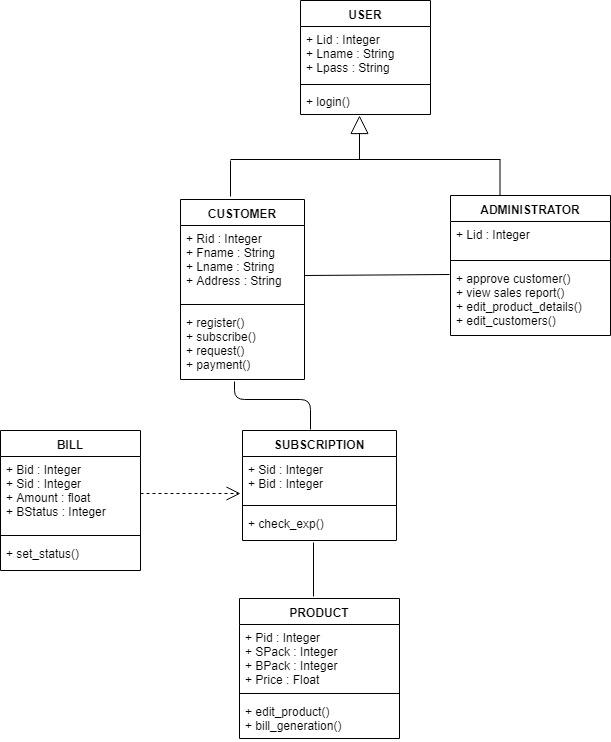


Figure 3.2 Class Diagram

Class diagram shows the information of the classes; the above diagram depicts the operations and the attributes of the classes and the types of attributes which are used in the class diagram. In the Above figure there are 6 classes.

**3.1.3 Sequence Diagram**

Sequence diagram is the most common kind of [interaction diagram](http://www.uml-diagrams.org/uml-25-diagrams.html#interaction-diagram), which focuses on [the message](http://www.uml-diagrams.org/interaction-message.html) interchange between a numbers of lifelines. Sequence diagram describes an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding occurrence specifications on the lifelines. The following nodes and edges are typically drawn in an Uml sequence diagram: lifeline, execution, specification, message, fragment, interaction, state, invariant, continuation, destruction occurrence.

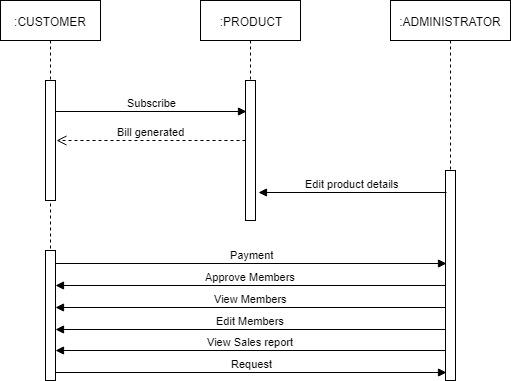


Figure 3.3 Sequence Diagram

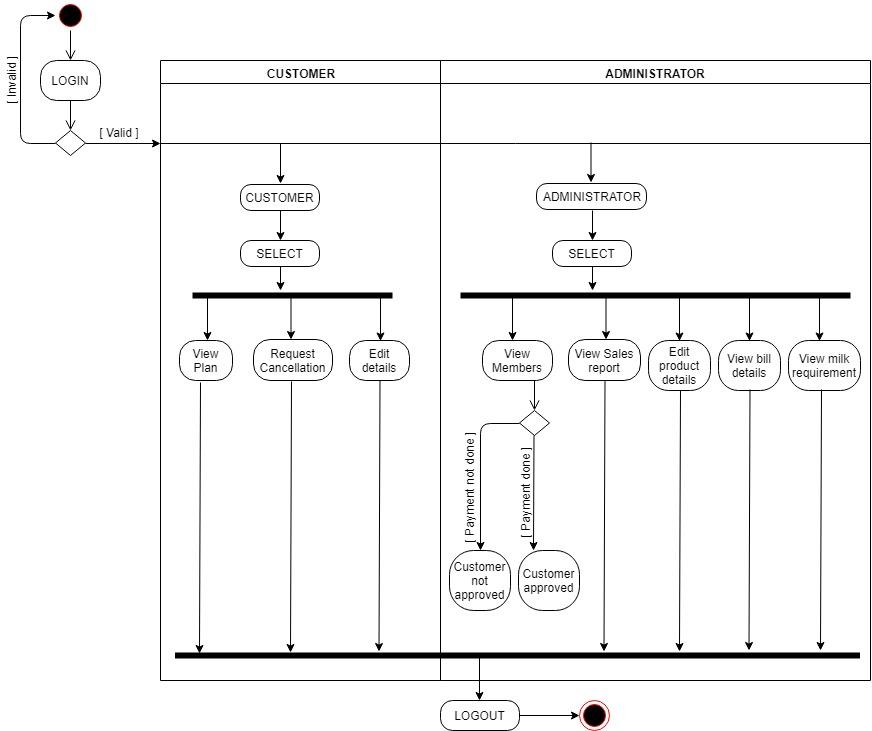
The Above diagrams show the life line of the each object. The first diagram contains information about how the admin controls the system. The second diagram is about how the Customer interacts with system.

**3.1.4 Activity Diagram**

The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.



Figure 3.4 Activity Diagram



The above diagram is about the flow of the system. The starting point then the flow within and once completed the stop.

**3.2 ER Diagram**

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a Data base. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases. At first glance an entity relationship diagram looks very much like a flowchart. It is the specialized symbols,

and the meanings of those symbols that make it unique.

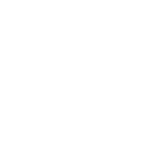
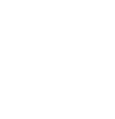
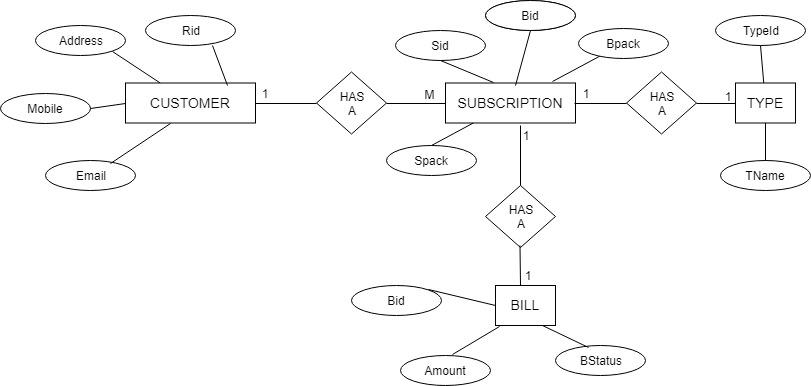
HAS A

Price

Pid

PRODUCT

Figure 3.5 ER Diagram



The above diagram contains the entities such as customer, issue history, car and staff from these entities it able to find the tables needed for the system. The ER Diagram also attributes for different entities.

**3.3 Database Design**

**3.3.1. Register**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Constraints** | **Type** | **Description** |
| Rid | PRIMARY KEY | INT | User id |
| Name | NOT NULL | VARCHAR(50) | First name |
| Address | NOT NULL | VARCHAR(50) | Address location |
| Mobile | NOT NULL | DOUBLE | Mobile number |
| Email | NOT NULL | VARCHAR(20) | Email id |
| User\_type | NOT NULL | VARCHAR(20) | Type of user-admin/user |
| Pass | NOT NULL | VARCHAR(20) | Password of the user |
| RStatus | NOT NULL | INT | Status of the account |

Table 3.2 Register Table



**3.3.2. Subscribe**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Constraints** | **Type** | **Description** |
| Sid | PRIMARY KEY | INT | Id of subscription |
| Rid | FOREIGN KEY | INT | Id of registration |
| Tid | FOREIGN KEY | INT | Id of type of subscription |
| Act\_date | NOT NULL | DATE | Date of activation |
| Exp\_date | NOT NULL | DATE | Date of expiry |
| Pid | NOT NULL | INT | Product Id |
| Total\_pack | NOT NULL | INT | Number of small pack |
| Bal\_pack | NOT NULL | INT | Number of big pack |

Table 3.3 Subscription Table

**3.3.3. Bill**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Constraints** | **Type** | **Description** |
| Bid | PRIMARY KEY | INT | Id of bill |
| Sid | FOREIGN KEY | INT | Id of subscription |
| Amount | NOT NULL | FLOAT | Total amount |
| BStatus | NOT NULL | INT | Status of bill |

**3.3.4. Type**

Table 3.4 Bill Table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Constraints | Type | Description |
| Tid | PRIMARY KEY | INT | Id of type of subscription |
| TName | NOT NULL | VARCHAR(20) | Name of type of subscription |

Table 3.5 Type Table



**3.3.5. Product**

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Constraints | Type | Description |
| Pid | PRIMARY KEY | INT | Id of product |
| Pname | NOT NULL | VARCHAR(20) | Name of pack |
| Price | NOT NULL | FLOAT | Price of pack |

Table 3.6 Product Table

**4.0 TESTING**

**4.1 Introduction**

Software Testing is the process of executing a program or system with the intent of finding errors. Testing involves any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. The scope of software testing includes examination of code as well as execution of that code in various environments and conditions as well as examining the quality aspects of code: does it do what it is supposed to do and do what it needs to do. Testing helps not only to uncover errors introduced during coding, but also locates errors committed during the previous phases. Thus the aim of testing is to uncover requirements, design or coding errors in the program.



**Unit Testing**

A unit is the smallest testable part of an application. Unit testing is a method of testing that verifies the individual units of source code are working properly. Rather than initially testing a program as a whole, testing is first focused on the smaller building blocks of the program. Unit testing eases the task of debugging and provide parallelism to program testing process by giving the opportunity to test multiple modules simultaneously. In this system the validity of fields in which data entered in each form and web form are checked. If the entered data are valid, then only further processing will take place.

**Integration Testing**

Data can be lost across an interface; one module can have an adverse effect on the other sub functions, when combined may not produce the desired functions. Integrated testing is the systematic testing to uncover the errors within the interface. This testing is done with simple data and the developed system has run successfully with this simple data. The need for integrated testing is to find the overall system performance. While developing the system, each module is developed individually and integrated with present system. Modules are integrated by adding the module as a reference in other modules.

**4.2 Test Cases**



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Data** | **Cases** | **Expected result** | **Actual result** | **Remark** |
| 1. | Password | User enter  \* Invalid password  \*Valid Password | Error Message:  Invalid password  ---- | Error Message:  Invalid password  ---- | As  expected |
| 2. | Admin  login | \*Enter incorrect  email and password  \*Correct email password | Error Message:  Email password mismatch  ---- | Error Message:  Email password mismatch  ---- | As  expected |
| 3. | User login | \*Enter incorrect  email and password  \*Correct email password | Error Message:  Email password mismatch  ---- | Error Message:  Email password mismatch  ---- | As  expected. |
| 4. | Phone  number | \*Enter invalid phone  number for registration  \*Valid phone number | Error Message:  Incorrect format  ---- | Error Message:  Incorrect format  ---- | As  expected. |
| 5. | Email | \*Enter invalid email  number for registration  \*Valid email number | Error Message:  Incorrect format  ---- | Error Message:  Incorrect format  ---- | As  expected. |

**5.0 IMPLEMENTATION DETAILS**

**5.1 Introduction**

Implementation is the stage in the project where the theoretical design is turned into a working system. Implementation is the final and important phase. The most critical stage for achieving a successful new system and for giving the users confidence that the new system will work and be effective. The system can be implemented only after thorough testing is done and only if it is found to working according to the specification. This method also offers the high security since it is implemented after identifying and handling all types of transactions while using the new system. Implementation phase include the training that should be provided for the chosen staff.



**5.2 Installation procedure**

The software can be installed in the following simple steps. In implementing machine

 Install mysql/Heidi sql/phpMyadmin

 Install or Enable IIS Manager

**5.3 Implementation plan**

In a direct cutover conversion, the old system is discarded and the new system takes over all at once, it is essentially turning the old system off and turning the new system on. This approach can be the least expensive of the different methods and can occur in the quickest time. A direct cutover conversion may be the only option if the old and new systems cannot co-exist in any form. The greatest risk is the impact that errors and failures would have on the organization. The timing of this type of conversion is a key element of its success. The riskiest strategy for new systems installation, the direct cutover conversion can be low cost and the benefits of the new system can be realized without delay.

**CONCLUSION**

Milkees is an efficient way of subscribing for milk. The ease of subscribing is one of the important factors of this system. This system provides functionalities for the administrator to view the entire system and be in control of the system. The administrator has privileges to view the members, edit the product, vie turnover report, milk required for the next day etc. The whole process is automated to a large extent



**7.0 APPENDIX**

**7.1 APPENDIX A**

**Sample source code/Pseudo code – Register.html**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>register</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter- bootstrap/4.0.0/css/bootstrap.min.css">



<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font- awesome/4.7.0/css/font-awesome.min.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Aladin">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Allerta+Stencil">

<link rel="stylesheet" href="assets2/css/styles.min.css">

</head>

<body>

<div>

<div class="container-fluid" style="background- image:url(&quot;assets2/img/b.jpg&quot;);">

<h1 class="text-center" style="padding-top:50px;font-family:'Allerta Stencil', sans-serif;font-size:58px;">REGISTRATION</h1>

<hr>

<form action="register\_act.jsp" method="post" >

<div class="form-row justify-content-center align-items-center" style="padding-bottom:100px;font-family:Aladin, cursive;">

<div class="col-12 col-md-6" id="message" style="background- color:rgba(156,152,152,0.59);">

<div class="form-group"><label for="from-comments" style="font- size:27px;font-family:Aladin, cursive;">Name\*</label>

<div class="input-group">

<div class="input-group-prepend"><span class="input-group- text"><i class="fa fa-user-o"></i></span></div><input class="form-control" type="text" name="name" required="" placeholder="Full Name" id="from- name"></div>

</div>

<div class="form-group"><label for="from-comments" style="font- size:27px;font-family:Aladin, cursive;">Email\*</label>

<div class="input-group">

<div class="input-group-prepend"><span class="input-group- text"><i class="fa fa-envelope-o"></i></span></div><input class="form-control" type="text" name="email" required="" placeholder="Email Address" id="from- email"></div>

</div>

<div class="form-row justify-content-start">

<div class="col-12 col-sm-6 col-md-12 col-lg-6">

<div class="form-group"><label for="from-comments" style="font- size:27px;font-family:Aladin, cursive;">Phone\*</label>

<div class="input-group">

<div class="input-group-prepend"><span class="input-group- text"><i class="fa fa-phone"></i></span></div><input class="form-control" type="text" name="phone" required="" placeholder="Phone Number" id="from- phone"></div>



</div>

</div>

</div>

<div class="form-group"><label for="from-comments" style="font- size:27px;font-family:Aladin, cursive;">Address\*</label><textarea class="form- control" rows="5" name="address" required="" placeholder="Enter address" id="from- comments"></textarea></div>

<div

class="form-group"><label for="from-comments" style="font- size:27px;font-family:Aladin, cursive;">Password\*</label>

<div class="input-group">

<div class="input-group-prepend"><span class="input-group- text"><i class="fa fa-user-o"></i></span></div><input class="form-control" type="text" name="pass" required="" placeholder="Password" id="from-pass"></div>

</div>

<div class="form-group"><label for="from-comments" style="font- size:27px;font-family:Aladin, cursive;">Confirm Password\*</label>

<div class="input-group">

<div class="input-group-prepend"><span class="input-group-text"><i class="fa fa-user-o"></i></span></div><input class="form-control" type="text" name="cpass" required="" placeholder="Confirm Password" id="from-pass2"></div>

</div>

<div class="form-group">

<div class="form-row">

<div class="col"><button class="btn btn-primary btn-block" type="submit" style="background-color:rgb(235,238,20);font- size:24px;color:rgb(18,13,13);">Submit <i class="fa fa-chevron-circle- right"></i></button></div>

</div>

</div>

</div>

</div>

</form>

</div>

</div>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/twitter- bootstrap/4.0.0/js/bootstrap.bundle.min.js"></script>

<script src="assets2/js/script.min.js"></script>

</body>

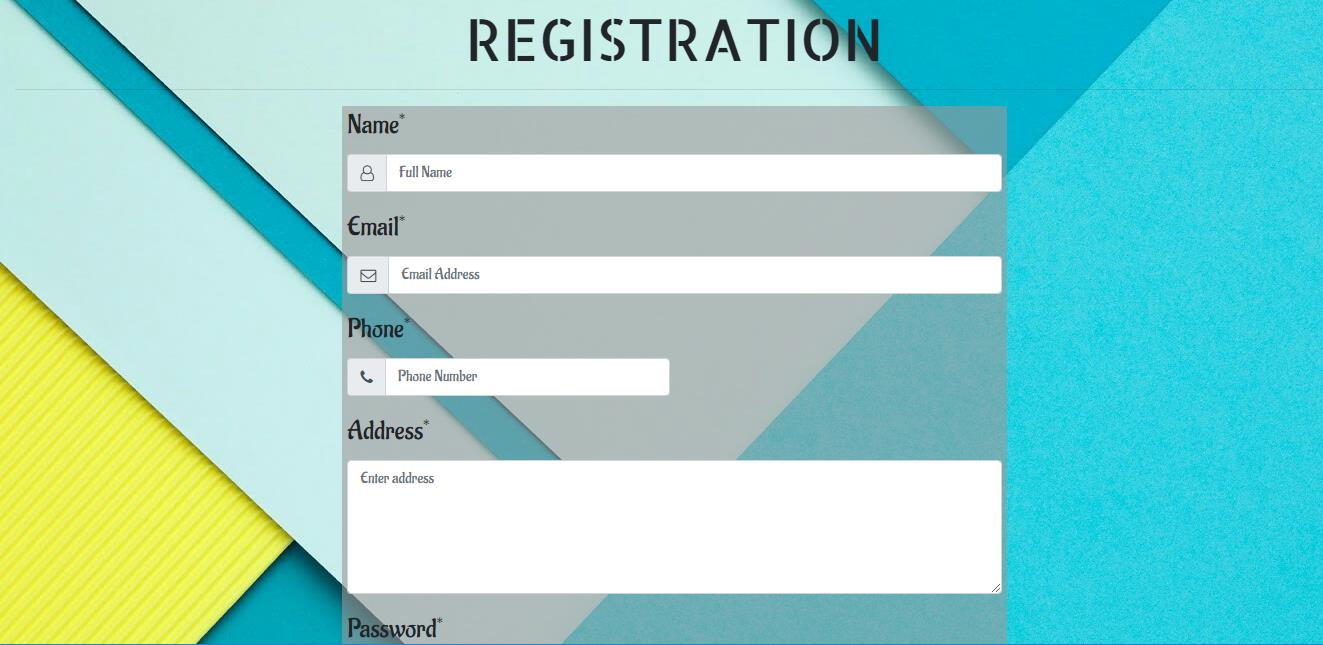
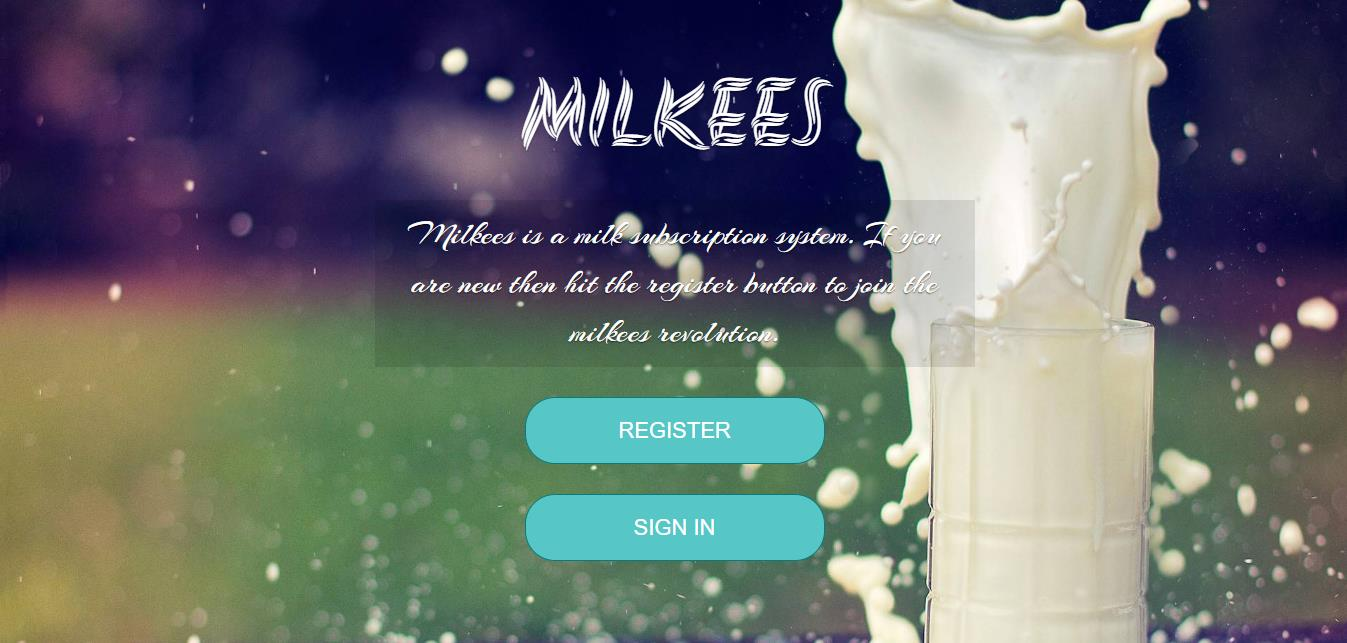
</html>



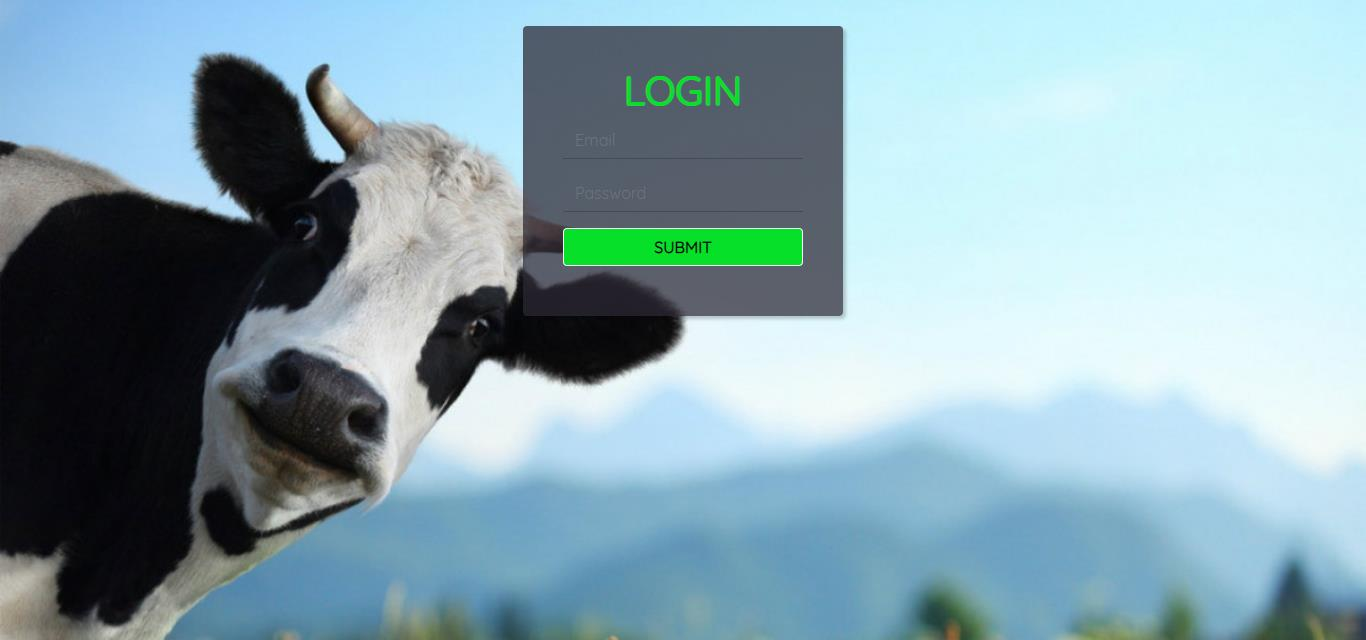
**7.2 APPENDIX B**

**7.2.1 Screenshots**

Main Page



Registration



Login Page

Admin Page



**7.3 APPENDIX C**

**7.3.1 BIBLIOGRAPHY**

**Books Referred**

 Elias M. Award.(2005)*System Analysis and Design.* Galgotia. Second Edition.

 Roger S. Pressman.(2001)*Software Engineering.* McGraw Hill. Fifth Edition.

 Craig Larman.(2004) *Applying UML and Patterns: An Introduction to Object oriented Analysis and Design And the Unified Process* Pearson Education. Third Edition.

**Internet Site**



 *“Retrieved from Software Developing Life Cycles”(March 2012).*

[http://www.sdlc](http://www.sdlc.ws).ws.

 *“Object Modelling Technique”(June 1999)*[.http://www.omt.org/technology/](http://www.omt.org/technology/uml)uml.

 *“Software Testing Dictionary”(2014).*

[http://www.tutorialspoint.com/sofware\_testing/impleme](http://www.tutorialspoint.com/sofware_testing/implementation)ntation