# A MAJOR PROJECT

On

**SUPERMARKET BILLING SYSYTEM USING QR CODE BLOCKCHAIN TECHNOLOGY**

*Submitted*

*In partial fulfilment for the requirement for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**By**

**TANGELLA NAGARAJU 19641A05D1**

**HUMERA ANWAR 19641A05G2**

**RENUKUNTLA SOMESH 19641A05C2**

**YERRABELLI HARISH RAO 19641A05C3**

***Under the Guidance of***

**Mrs. A. Swetha**

Associate Professor, Department of CSE.

### DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

### VAAGDEVI COLLEGE OF ENGINEERING

(UGC Autonomous, Accredited by NAAC with “A”)

Bollikunta, Khila Warangal (Mandal), Warangal Urban – 506005(T.S)

(2019-2023)

**VAAGDEVI COLLEGE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

(UGC Autonomous, Accredited by NBA, Accredited by NAAC with “A”)

Bollikunta, Khila Warangal (Mandal), Warangal Urban –506005(T.S)



**CERTIFICATE**

This is to certify that the Major Project entitled **“HAND GESTURE RECOGNITION USING CNN AND CONVERSION TO SPEECH”** is submitted by **TANGELLA NAGARAJU (19641A05D1), HUMERA ANWAR (19641A05G2), RENUKUNTLA SOMESH (19641A05C2), YERRABELLI HARISHRAO (19641A05C3)**in partial fulfilment of the requirements for the award of the Degree in Bachelor of Technology in Computer Science and Engineering during the academic year 2022-2023.

**Project Guide Head of the Department**

**Mrs. A. Swetha Dr. N. SATHYAVATHI**

**External Examiner**

# ACKNOWLEDGEMENT

We wish to take this opportunity to express my sincere gratitude and deep sense of respect to beloved **Dr. K. Prakash, Principal,** Vaagdevi College of Engineering for making me available all the required assistance and for his support and inspiration to carry out this industry Oriented Major Project in the institute.

We extend our heartfelt thanks to **Dr. N. Sathyavathi, Head of the Department of CSE**, Vaagdevi College of Engineering for providing us necessary infrastructure and thereby giving us freedom to carry out the Industry Oriented Major Project.

We express heartfelt thanks to the guide, **Mrs. A.Swetha, Associate** **Professor**, Department of CSE for his constant support and giving necessary guidance for completion of this Industry Oriented Major Project.

We are also thankful to **Mrs. K. Rekha Devi, Mr. Ch. Aravind Kumar** for their valuable suggestions, encouragement and motivations for completing this project successfully.

We are thankful to all other faculty members for their encouragement. We convey our heartfelt thanks to the lab staff for allowing me to use the required equipment whenever needed.

Finally, we would like to take this opportunity to thank our family for their support through the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work.

**TEAM MEMBERS**

**TANGELLA NAGARAJU 19641A05D1**

**HUMERA ANWAR 19641A05G2**

**RENUKUNTLA SOMESH 19641A05C2**

**YERRABELLI HARISHRAO 19641A05C3**

**DECLARATION**

We hereby declare that this project entitled **" HAND GESTURE RECOGNITION USING CNN AND CONVERSION TO SPEECH "** is submitted in partial fulfilment of requirements for the award of bachelor of technology at **VAAGDEVI COLLEGE OF ENGINEERING** affiliated to Jawaharlal Nehru Technology University. The report has not been submitted either in part or full for degree earlier to this University.

**TEAM MEMBERS**

**TANGELLA NAGARAJU 19641A05D1**

**HUMERA ANWAR 19641A05G2**

**RENUKUNTLA SOMESH 19641A05C2**

**YERRABELLI HARISHRAO 19641A05C3**

**ABSTRACT**

We generally go to supermarkets to purchase the essential needs such as groceries which are required for our day-to-day life, we see that the billing in supermarket is done by scanning the barcode which is present on the product, due to this the billing process consumes lot of time and even the customer has to wait longer time in the line at the counter when there are some issues in scanning the barcode.

So, in order to solve the issues, the billing system using website has introduced where it is implemented with some added functionality to traditional supermarket billing. This system is faster bill generation for customers. With the of website the system adds the product and gives the information of the product such as name and price of the product and calculates the bill quickly. By this system we can reduce the time and increase the accuracy.

**LIST OF CONTENTS**

**S.NO TITLE PAGE NO**

1 INTRODUCTION

1.1 EXISTING SYSTEM 7-10

1.2 PROPOSED SYSTEM

1.3 SYSTEM STUDY

2 LITERATURE SURVEY 11

3 SYSTEM REQUIRMENTS

3.1 HARDWARE REQUIRMENTS 12

3.2 SOFTWARE REQUIRMENTS

4 SYSTEM DESIGN 13

5 UML DIAGRAMS

5.1 USE CASE DIAGRAM

5.2 STATE CHART DIAGRAM

5.3 CLASS DIAGRAM 14-20

5.4SEQUENCE DIAGRAMS

5.5 COMPONENT DIAGRAM

6 **IMPLEMENTATION**

6.1 MODULES

6.2 SYSTEM ENVIRONMENT 20-47

6.3 SAMPLE CODE

7 **SYSTEM TESTING** 48-51

7.1 UNIT TESTING

7.2 INTEGRATION TESTING

7.3 FUNCTIONAL TEST

7.4 SYSTEM TEST

7.5 WHITE BOX TESTING

7.6 BLACK BOX TESTING

7.7 TEST CASES

8 **OUTPUTS** 52-55

9 CONCLUSION AND FUTURE SCOPE 56-57

10 REFERENCES 58

# 1. INTRODUCTION

In this modern era, the people have income time to spend and less time to spend. So they generally opt for supermarkets for groceries. Supermarket is the place where the consumers come to purchase their daily using products and pay for that. So there is need to calculate how many products are sold and to generate the bill for the customers. We can see that the barcodes are widely used in many grocery supermarkets for billing. These barcodes are placed on each and every product that is available in the supermarket.

After picking up the required items the customer goes to the check -out counters ,they scan the barcode on the products to calculate the bill for the customers. Every product has to be scanned separately consuming huge time and consumes lots of time of the customers. Particularly in huge stores whenever many customers comes for looking in exceedingly a day and thousands of products have to be compelled to be scanned. This makes the task difficult for the human workers and even the long queues of shoppers are seen at the stores. In several cases, the barcode is either broken or even there is also downside in reading barcode, because of lighting effects ,low resolution etc.

A barcode-based billing is also expensive as it requires laser light for barcoding of all the products. Supermarket billing system using webcam deals with the automation of supermarket. This will help user to work in a highly effective and in a friendly environment. The billing consumes considerably less time and energy of the customers. This system is implemented by considering the predefined images of the products that comes to the supermarket and are stored in the local system. other python libraries for the identification of images are used in this system.

The website will capture the images of the products when the customer comes for billing. After capturing the images of products, it will find the objects which are predefined then it compares with the stored images, the software part will calculate the bill. The supermarket billing system using webcam is built to help supermarkets calculate and dispalys bills and serve the customers in a faster and efficient manner. So the customer need not to wait for longer time in the queue for the billing process. Supermarket billing system using webcam is developed with the objective of making the system reliable, easier, fast and more informative.

## 1.1 EXISTING SYSTEM

In today’s fast paced society, it’s very hard to be competitive without using cutting-edge technology available in market. After years of business, the data has grown much. It is becoming a challenge for person to manage that data in an effective way. To be more productive in order processing, he needs a solution which can facilitate their current processes with use of technology and software. With increased amount of orders, it is becoming difficult for salesperson to manage orders in effective and efficient manner. It is very hard to go through all paper work and backtracking orders. If there is any complain or review of any order, it takes large amount of effort and time to backtrack and fix the problem. This results in loss of resources, increased time, and low output.

**Drawbacks of Manual System (Current System)**

**Time consuming:** Getting the required information from the available data takes a lot of time. Changing, editing and updating the information contained in several files are a slow and time consuming process.

**Poor communication:** A manual system requires employees and managers to write down each time an item is removed from the inventory. If one employee forgets to mention that the last coffee product has been removed from the inventory the admin or manager expects the item to still available for a customers during sale.

**Need of Effort:** In manual system, an Item’s record is maintained in separate files so it takes much effort to collect data from several Stores for and if we want to change or delete the data of any transaction then it has to be changed or deleted from all the files and places it stored.

**Needs Large Space:** In manual work done data item has to be stored at several places, similarly student’s record is maintained in separate registers. It requires more storage space.

## 1.2 PROPOSED SYSTEMS

This software project is a traditional supermarket billing system with some added functionality. This system is built for fast data processing and bill generation for supermarket customers. The billing system consists of an sql database and effective front end designed in Asp.net. The billing database is a vast collection of product name, price and other product specific data. A product when billed is searched from the database and its price is added to the bill based upon the product quantity. The system also contains discounts on various products so that the product is offered at discounted price while billing. The supermarket billing system is built to help supermarkets calculate and display bills and serve the customer in a faster and efficient manner. This software project consists of an effective and easy Gui to help the employee in easy bill calculation and providing an efficient customer service.

**1.3 SYSTEM STUDY**

**FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**2. LITERATURE SURVEY**

Object detection and tracking are the two main tasks in multi camera surveillance. A new object detection algorithm using mean shift (MS) is introduced. The detected objects are then tracked by a new object are then tracked by a new object tracking algorithm using a Bayesian Kalman filter with simplified Gaussian mixture (BKF-SGM). A new BKF-SGM is improved with MS algorithm a more robust tracking performance is obtained.

The experimental results shows that the proposed object detection algorithm yields improved results over conventional object detection methods and the proposed tracking algorithm can successfully handle complex tasks with good performance. Image processing techniques minimizes the manual tasks of recognizing. In this, an image processing system for automatic segmentation and prediction is proposed on the basis of color and shape features are being performed.

OpenCV python software is used to perform the required image processing operation. Object detection and recognition is the primary and foundation for intelligent service to understand the surrounding environment and make decisions. Here, aiming at the accuracy and real-time performance of object detection and recognition of service in complex scenes, an end to end object detection and recognition algorithm based on deep learning is proposed.

The deep convolution neural network is adopted to enhance the feature representation capability of the model by enhancing the convolution module function. When the local features and global features are fully fused ,the natural multi scale detection recognition is realized on multiple receptive fields. The results shows that the algorithm has both good accuracy and real-time performance.

**3. SYSTEM REQUIREMENT**

**3.1 HARDWARE REQUIREMENTS**

* System : Pentium IV 2.4 GHz.
* Hard Disk : 40 GB.
* Floppy Drive : 1.44 Mb.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 512 Mb.

**3.2 SOFTWARE REQUIREMENTS**

* Operating System **:** Windows
* Coding Language : C#
* Frontend : asp.net,
* Middleware Technology : ADO.NET
* Backend Technology : MS SQL

# 4.SYSTEM DESIGN

**4.1 SYSTEM ARCHITECTURE**

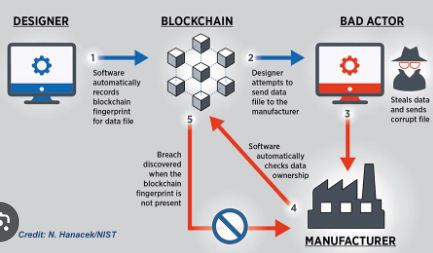
****

Fig 4.1 SYSTEM ARCHITECTURE

**5. UNIFIED MODELLING LANGUAGE (UML) DIAGRAMS**

**DIAGRAMS**

A Unified Modelling language diagram is a diagram based on the UML (Unified Modelling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

UML is a modern approach to modelling and documenting software. In fact, it's one of the most popular business process modelling techniques. It is based on diagrammatic representations of software components. As the old proverb says: "a picture is worth a thousand words". By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

Mainly UML has used as a general-purpose modelling language on the field of software engineering. However, it has row found its way in the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts hey provide both a more standardized way of modelling workflows as well as a wider range of features to improves readability.

**5.1 USE CASE DIAGRAM**

In the Unified Modelling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, use a set of specialized symbols and connectors. An effective use case diagram can help the team discuss and represent.

* Scenarios in which system or application interacts with people, organizations, or external systems.
* Goals that your system or application helps those entities (known as actors) achieve the scope of the system.

UML use case diagrams are ideal for:

* Representing the goals of system-user interactions.
* Defining and organizing functional requirements in a system.
* Specifying the context and requirements of a system.
* Modelling the basic flow of events in a use case



Fig 5.1 Use Case Diagram

**5.2 STATE CHART DIAGRAM**

State chart diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control

# 

Fig 5.2 State Chart Diagram

**5.3 CLASS DIAGRAM**

Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modelling its classes, attributes, operations, and relationships between objects. It is a static diagram that represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The class shape itself consists of a rectangle with three rows. The top row contains the name of the class, the middle row contains the attributes of the class, and the bottom section express the methods or operations that the class may use. Classes and subclasses are grouped together to show the static relationship between each object.

The purpose of the class diagram can be summarized as:

* Analysis and design of the static view of an application.
* Describe responsibilities of a system.
* Base for component and deployment diagrams.
* Forward and reverse engineering.

****

Fig 5.3 Class Diagram

**5.4 SEQUENCE DIAGRAMS**

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

****

Fig 5.4 Sequence Diagram

**5.5 COMPONENT DIAGRAM**

A component diagram is a type of UML diagram that shows the structure of a software system in terms of its components. Components are physical units of software that can be compiled or linked into an executable system. They can be libraries, executables, files, or other physical entities.

Component diagrams are used to show the relationships between components, such as dependency, aggregation, and composition. Dependency shows that one component uses another component. Aggregation shows that one component contains another component. Composition shows that one component owns another component.

Component diagrams can also be used to show the interfaces between components. An interface is a contract that defines the behavior of a component. It specifies the operations that can be performed on the component and the data that can be passed to and from the component.

Component diagrams are a useful tool for visualizing and documenting the structure of a software system. They can be used to communicate the system architecture to developers, testers, and other stakeholders.

****

Fig 5.5 Component Diagram

**6. IMPLEMENTATION**

The objective of the coding or programming phase is to translate the design of the system produced during the design phase into code in a given programming language, which can be executed by a computer and that performs the computation specified by the design. The coding phase affects both testing and maintenance. The goal of coding is not to reduce the implementation cost, but the goal should be to reduce the cost of later phases.

**6.1 MODULES**

The system after careful analysis has been identified to be presented with the following modules and roles.

The modules involved are:

1 Administrator

2. Users

**ADMINISTRATOR**

The administrator is the super user of this application. Only admin have access into this admin page. Admin may be the owner of the shop. The administrator has all the information about all the users and about all products.

This module is divided into different sub-modules.

1.Add Items

2. Add Products

3. View Users

4. View Orders

**ADD PRODUCTS**

The shopping cart project contains different kind of products. The products can be classified into different categories by name. Admin can add new products into the existing system with all its details including an image.

**DELETE PRODUCTS**

Administrator can delete the products based on the stock of that particular product.

**USER:**

A new user will have to register in the system by providing essential details in order to view the products in the system. The admin must accept a new user by unblocking him.

**LOGIN:** A user must login with his user name and password to the system after registration.

**VIEW PRODUCTS:** User can view the list of products based on their names after successful login. A detailed description of a particular product with product name, products details, product image, price can be viewed by users.

**SEARCH PRODUCT:** Users can search for a particular product in the list by name.

**ADD TO CART:** The user can add the desired product into his cart by clicking add to cart option on the product. He can view his cart by clicking on the cart button. All products added by cart can be viewed in the cart. User can remove an item from the cart by clicking remove.

**6.2 SOFTWARE ENVIRONMENT**

# MICROSOFT .NET FRAMEWORK

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet.

The .NET Framework is designed to fulfill the following objectives:

* To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.
* To provide a code-execution environment that minimizes software deployment and versioning conflicts.
* To provide a code-execution environment that guarantees safe execution of code, including code created by an unknown or semi-trusted third party.
* To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
* To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.
* To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime and the .NET Framework class library. The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that ensure security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code. The class library, the other main component of the .NET Framework, is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services.

The .NET Framework can be hosted by unmanaged components that load the common language runtime into their processes and initiate the execution of managed code, thereby creating a software environment that can exploit both managed and unmanaged features. The .NET Framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts.

For example, ASP.NET hosts the runtime to provide a scalable, server-side environment for managed code. ASP.NET works directly with the runtime to enable Web Forms applications and XML Web services, both of which are discussed later in this topic.

Internet Explorer is an example of an unmanaged application that hosts the runtime (in the form of a MIME type extension). Using Internet Explorer to host the runtime enables you to embed managed components or Windows Forms controls in HTML documents. Hosting the runtime in this way makes managed mobile code (similar to Microsoft® ActiveX® controls) possible, but with significant improvements that only managed code can offer, such as semi-trusted execution and secure isolated file storage.

The following illustration shows the relationship of the common language runtime and the class library to your applications and to the overall system. The illustration also shows how managed code operates within a larger architecture.

## FEATURES OF THE COMMON LANGUAGE RUNTIME

The common language runtime manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features are intrinsic to the managed code that runs on the common language runtime.

With regards to security, managed components are awarded varying degrees of trust, depending on a number of factors that include their origin (such as the Internet, enterprise network, or local computer). This means that a managed component might or might not be able to perform file-access operations, registry-access operations, or other sensitive functions, even if it is being used in the same active application.

The runtime enforces code access security. For example, users can trust that an executable embedded in a Web page can play an animation on screen or sing a song, but cannot access their personal data, file system, or network. The security features of the runtime thus enable legitimate Internet-deployed software to be exceptionally featuring rich. The runtime also enforces code robustness by implementing a strict type- and code-verification infrastructure called the common type system (CTS). The CTS ensures that all managed code is self-describing. The various Microsoft and third-party language compilers generate managed code that conforms to the CTS. This means that managed code can consume other managed types and instances, while strictly enforcing type fidelity and type safety.

In addition, the managed environment of the runtime eliminates many common software issues. For example, the runtime automatically handles object layout and manages references to objects, releasing them when they are no longer being used. This automatic memory management resolves the two most common application errors, memory leaks and invalid memory references.

The runtime also accelerates developer productivity. For example, programmers can write applications in their development language of choice, yet take full advantage of the runtime, the class library, and components written in other languages by other developers. Any compiler vendor who chooses to target the runtime can do so. Language compilers that target the .NET Framework make the features of the .NET Framework available to existing code written in that language, greatly easing the migration process for existing applications.

While the runtime is designed for the software of the future, it also supports software of today and yesterday. Interoperability between managed and unmanaged code enables developers to continue to use necessary COM components and DLLs.

The runtime is designed to enhance performance. Although the common language runtime provides many standard runtime services, managed code is never interpreted. A feature called just-in-time (JIT) compiling enables all managed code to run in the native machine language of the system on which it is executing. Meanwhile, the memory manager removes the possibilities of fragmented memory and increases memory locality-of-reference to further increase performance.

Finally, the runtime can be hosted by high-performance, server-side applications, such as Microsoft® SQL Server™ and Internet Information Services (IIS). This infrastructure enables you to use managed code to write your business logic, while still enjoying the superior performance of the industry's best enterprise servers that support runtime hosting.

## .NET FRAMEWORK CLASS LIBRARY

The .NET Framework class library is a collection of reusable types that tightly integrate with the common language runtime. The class library is object oriented, providing types from which your own managed code can derive functionality. This not only makes the .NET Framework types easy to use, but also reduces the time associated with learning new features of the .NET Framework. In addition, third-party components can integrate seamlessly with classes in the .NET Framework.

For example, the .NET Framework collection classes implement a set of interfaces that you can use to develop your own collection classes. Your collection classes will blend seamlessly with the classes in the .NET Framework.

As you would expect from an object-oriented class library, the .NET Framework types enable you to accomplish a range of common programming tasks, including tasks such as string management, data collection, database connectivity, and file access. In addition to these common tasks, the class library includes types that support a variety of specialized development scenarios. For example, you can use the .NET Framework to develop the following types of applications and services:

* Console applications.
* Scripted or hosted applications.
* Windows GUI applications (Windows Forms).
* ASP.NET applications.
* XML Web services.
* Windows services.

For example, the Windows Forms classes are a comprehensive set of reusable types that vastly simplify Windows GUI development. If you write an ASP.NET Web Form application, you can use the Web Forms classes.

## CLIENT APPLICATION DEVELOPMENT

Client applications are the closest to a traditional style of application in Windows-based programming. These are the types of applications that display windows or forms on the desktop, enabling a user to perform a task. Client applications include applications such as word processors and spreadsheets, as well as custom business applications such as data-entry tools, reporting tools, and so on. Client applications usually employ windows, menus, buttons, and other GUI elements, and they likely access local resources such as the file system and peripherals such as printers.

Another kind of client application is the traditional ActiveX control (now replaced by the managed Windows Forms control) deployed over the Internet as a Web page. This application is much like other client applications: it is executed natively, has access to local resources, and includes graphical elements.

In the past, developers created such applications using C/C++ in conjunction with the Microsoft Foundation Classes (MFC) or with a rapid application development (RAD) environment such as Microsoft® Visual Basic®. The .NET Framework incorporates aspects of these existing products into a single, consistent development environment that drastically simplifies the development of client applications. The Windows Forms classes contained in the .NET Framework are designed to be used for GUI development. You can easily create command windows, buttons, menus, toolbars, and other screen elements with the flexibility necessary to accommodate shifting business needs.

For example, the .NET Framework provides simple properties to adjust visual attributes associated with forms. In some cases the underlying operating system does not support changing these attributes directly, and in these cases the .NET Framework automatically recreates the forms. This is one of many ways in which the .NET Framework integrates the developer interface, making coding simpler and more consistent.

Unlike ActiveX controls, Windows Forms controls have semi-trusted access to a user's computer. This means that binary or natively executing code can access some of the resources on the user's system (such as GUI elements and limited file access) without being able to access or compromise other resources. Because of code access security, many applications that once needed to be installed on a user's system can now be safely deployed through the Web. Your applications can implement the features of a local application while being deployed like a Web page.

## SERVER APPLICATION DEVELOPMENT

Server-side applications in the managed world are implemented through runtime hosts. Unmanaged applications host the common language runtime, which allows your custom managed code to control the behavior of the server. This model provides you with all the features of the common language runtime and class library while gaining the performance and scalability of the host server.

The following illustration shows a basic network schema with managed code running in different server environments. Servers such as IIS and SQL Server can perform standard operations while your application logic executes through the managed code.

**SERVER-SIDE MANAGED CODE**

ASP.NET is the hosting environment that enables developers to use the .NET Framework to target Web-based applications. However, ASP.NET is more than just a runtime host; it is a complete architecture for developing Web sites and Internet-distributed objects using managed code. Both Web Forms and XML Web services use IIS and ASP.NET as the publishing mechanism for applications, and both have a collection of supporting classes in the .NET Framework.

XML Web services, an important evolution in Web-based technology, are distributed, server-side application components similar to common Web sites. However, unlike Web-based applications, XML Web services components have no UI and are not targeted for browsers such as Internet Explorer and Netscape Navigator. Instead, XML Web services consist of reusable software components designed to be consumed by other applications, such as traditional client applications, Web-based applications, or even other XML Web services. As a result, XML Web services technology is rapidly moving application development and deployment into the highly distributed environment of the Internet.

If you have used earlier versions of ASP technology, you will immediately notice the improvements that ASP.NET and Web Forms offers. For example, you can develop Web Forms pages in any language that supports the .NET Framework. In addition, your code no longer needs to share the same file with your HTTP text (although it can continue to do so if you prefer). Web Forms pages execute in native machine language because, like any other managed application, they take full advantage of the runtime. In contrast, unmanaged ASP pages are always scripted and interpreted. ASP.NET pages are faster, more functional, and easier to develop than unmanaged ASP pages because they interact with the runtime like any managed application.

The .NET Framework also provides a collection of classes and tools to aid in development and consumption of XML Web services applications. XML Web services are built on standards such as SOAP (a remote procedure-call protocol), XML (an extensible data format), and WSDL ( the Web Services Description Language). The .NET Framework is built on these standards to promote interoperability with non-Microsoft solutions.

For example, the Web Services Description Language tool included with the .NET Framework SDK can query an XML Web service published on the Web, parse its WSDL description, and produce C# or Visual Basic source code that your application can use to become a client of the XML Web service. The source code can create classes derived from classes in the class library that handle all the underlying communication using SOAP and XML parsing. Although you can use the class library to consume XML Web services directly, the Web Services Description Language tool and the other tools contained in the SDK facilitate your development efforts with the .NET Framework.

If you develop and publish your own XML Web service, the .NET Framework provides a set of classes that conform to all the underlying communication standards, such as SOAP, WSDL, and XML. Using those classes enables you to focus on the logic of your service, without concerning yourself with the communications infrastructure required by distributed software development.

Finally, like Web Forms pages in the managed environment, your XML Web service will run with the speed of native machine language using the scalable communication of IIS.

**ACTIVE SERVER PAGES.NET**

ASP.NET is a programming framework built on the common language runtime that can be used on a server to build powerful Web applications. ASP.NET offers several important advantages over previous Web development models:

* **Enhanced Performance:** ASP.NET is compiled common language runtime code running on the server. Unlike its interpreted predecessors, ASP.NET can take advantage of early binding, just-in-time compilation, native optimization, and caching services right out of the box. This amounts to dramatically better performance before you ever write a line of code.
* **World-Class Tool Support:** The ASP.NET framework is complemented by a rich toolbox and designer in the Visual Studio integrated development environment. WYSIWYG editing, drag-and-drop server controls, and automatic deployment are just a few of the features this powerful tool provides.
* **Power and Flexibility:** Because ASP.NET is based on the common language runtime, the power and flexibility of that entire platform is available to Web application developers. The .NET Framework class library, Messaging, and Data Access solutions are all seamlessly accessible from the Web. ASP.NET is also language-independent, so you can choose the language that best applies to your application or partition your application across many languages. Further, common language runtime interoperability guarantees that your existing investment in COM-based development is preserved when migrating to ASP.NET.
* **Simplicity:** ASP.NET makes it easy to perform common tasks, from simple form submission and client authentication to deployment and site configuration. For example, the ASP.NET page framework allows you to build user interfaces that cleanly separate application logic from presentation code and to handle events in a simple, Visual Basic - like forms processing model. Additionally, the common language runtime simplifies development, with managed code services such as automatic reference counting and garbage collection.
* **Manageability:** ASP.NET employs a text-based, hierarchical configuration system, which simplifies applying settings to your server environment and Web applications. Because configuration information is stored as plain text, new settings may be applied without the aid of local administration tools. This "zero local administration" philosophy extends to deploying ASP.NET Framework applications as well. An ASP.NET Framework application is deployed to a server simply by copying the necessary files to the server. No server restart is required, even to deploy or replace running compiled code.
* **Scalability and Availability:** ASP.NET has been designed with scalability in mind, with features specifically tailored to improve performance in clustered and multiprocessor environments. Further, processes are closely monitored and managed by the ASP.NET runtime, so that if one misbehaves (leaks, deadlocks), a new process can be created in its place, which helps keep your application constantly available to handle requests.
* **Customizability and Extensibility:** ASP.NET delivers a well-factored architecture that allows developers to "plug-in" their code at the appropriate level. In fact, it is possible to extend or replace any subcomponent of the ASP.NET runtime with your own custom-written component. Implementing custom authentication or state services has never been easier.
* **Security:** With built in Windows authentication and per-application configuration, you can be assured that your applications are secure.
* **Language Support:** The Microsoft .NET Platform currently offers built-in support for three languages: C#, Visual Basic, and JScript.

**ASP.NET WEB FORMS**

The ASP.NET Web Forms page framework is a scalable common language runtime programming model that can be used on the server to dynamically generate Web pages.

Intended as a logical evolution of ASP (ASP.NET provides syntax compatibility with existing pages), the ASP.NET Web Forms framework has been specifically designed to address a number of key deficiencies in the previous model. In particular, it provides:

* The ability to create and use reusable UI controls that can encapsulate common functionality and thus reduce the amount of code that a page developer has to write.
* The ability for developers to cleanly structure their page logic in an orderly fashion (not "spaghetti code").
* The ability for development tools to provide strong WYSIWYG design support for pages (existing ASP code is opaque to tools).

ASP.NET Web Forms pages are text files with an .aspx file name extension. They can be deployed throughout an IIS virtual root directory tree. When a browser client requests .aspx resources, the ASP.NET runtime parses and compiles the target file into a .NET Framework class. This class can then be used to dynamically process incoming requests. (Note that the .aspx file is compiled only the first time it is accessed; the compiled type instance is then reused across multiple requests).

An ASP.NET page can be created simply by taking an existing HTML file and changing its file name extension to .aspx (no modification of code is required). For example, the following sample demonstrates a simple HTML page that collects a user's name and category preference and then performs a form post back to the originating page when a button is clicked:

ASP.NET provides syntax compatibility with existing ASP pages. This includes support for <% %> code render blocks that can be intermixed with HTML content within an .aspx file. These code blocks execute in a top-down manner at page render time.

Code-Behind Web Forms: ASP.NET supports two methods of authoring dynamic pages. The first is the method shown in the preceding samples, where the page code is physically declared within the originating .aspx file. An alternative approach--known as the code-behind method--enables the page code to be more cleanly separated from the HTML content into an entirely separate file.

Introduction to ASP.NET Server Controls:

In addition to (or instead of) using <% %> code blocks to program dynamic content, ASP.NET page developers can use ASP.NET server controls to program Web pages. Server controls are declared within an .aspx file using custom tags or intrinsic HTML tags that contain a runat="server" attributes value. Intrinsic HTML tags are handled by one of the controls in the System.Web.UI.HtmlControls namespace. Any tag that doesn't explicitly map to one of the controls is assigned the type of System.Web.UI.HtmlControls.HtmlGenericControl.

Server controls automatically maintain any client-entered values between round trips to the server. This control state is not stored on the server (it is instead stored within an <input type="hidden"> form field that is round-tripped between requests). Note also that no client-side script is required.

In addition to supporting standard HTML input controls, ASP.NET enables developers to utilize richer custom controls on their pages. For example, the following sample demonstrates how the <asp:adrotator> control can be used to dynamically display rotating ads on a page.

* + ASP.NET Web Forms provide an easy and powerful way to build dynamic Web UI.
  + ASP.NET Web Forms pages can target any browser client (there are no script library or cookie requirements).
  + ASP.NET Web Forms pages provide syntax compatibility with existing ASP pages.
  + ASP.NET server controls provide an easy way to encapsulate common functionality.
  + ASP.NET ships with 45 built-in server controls. Developers can also use controls built by third parties.
  + ASP.NET server controls can automatically project both up level and down level HTML.
  + ASP.NET templates provide an easy way to customize the look and feel of list server controls.
  + ASP.NET validation controls provide an easy way to do declarative client or server data validation.

**Crystal Reports:** Crystal Reports for Visual Basic .NET is the standard reporting tool for Visual Basic.NET; it brings the ability to create interactive, presentation-quality content — which has been the strength of Crystal Reports for years — to the .NET platform. With Crystal Reports for Visual Basic.NET, you can host reports on Web and Windows platforms and publish Crystal reports as Report Web Services on a Web server.

To present data to users, you could write code to loop through record sets and print them inside your Windows or Web application. However, any work beyond basic formatting can be complicated: consolidations, multiple level totals, charting, and conditional formatting are difficult to program.

With Crystal Reports for Visual Studio .NET, you can quickly create complex and professional-looking reports. Instead of coding, you use the Crystal Report Designer interface to create and format the report you need. The powerful Report Engine processes the formatting, grouping, and charting criteria you specify.

**Report Experts:** Using the Crystal Report Experts, you can quickly create reports based on your development needs:

Choose from report layout options ranging from standard reports to form letters, or build your own report from scratch.

* + Display charts that users can drill down on to view detailed report data.
  + Calculate summaries, subtotals, and percentages on grouped data.
  + Show TopN or BottomN results of data.
  + Conditionally format text and rotate text objects.

**BACK-END TECHNOLOGY**

About Microsoft SQL Server 2000: Microsoft SQL Server is a Structured Query Language (SQL) based, client/server relational database. Each of these terms describes a fundamental part of the architecture of SQL Server.

**Database:** A database is similar to a data file in that it is a storage place for data. Like a data file, a database does not present information directly to a user; the user runs an application that accesses data from the database and presents it to the user in an understandable format.

A database typically has two components: the files holding the physical database and the database management system (DBMS) software that applications use to access data. The DBMS is responsible for enforcing the database structure, including:

* + Maintaining the relationships between data in the database.
  + Ensuring that data is stored correctly and that the rules defining data relationships are not violated.
  + Recovering all data to a point of known consistency in case of system failures.

**Relational Database:** There are different ways to organize data in a database but relational databases are one of the most effective. Relational database systems are an application of mathematical set theory to the problem of effectively organizing data. In a relational database, data is collected into tables (called relations in relational theory).

When organizing data into tables, you can usually find many different ways to define tables. Relational database theory defines a process, normalization, which ensures that the set of tables you define will organize your data effectively.

**CLIENT/SERVER**

In a client/server system, the server is a relatively large computer in a central location that manages a resource used by many people. When individuals need to use the resource, they connect over the network from their computers, or clients, to the server.

Examples of servers are: In a client/server database architecture, the database files and DBMS software reside on a server. A communications component is provided so applications can run on separate clients and communicate to the database server over a network. The SQL Server communication component also allows communication between an application running on the server and SQL Server.

Server applications are usually capable of working with several clients at the same time. SQL Server can work with thousands of client applications simultaneously. The server has features to prevent the logical problems that occur if a user tries to read or modify data currently being used by others.

While SQL Server is designed to work as a server in a client/server network, it is also capable of working as a stand-alone database directly on the client. The scalability and ease-of-use features of SQL Server allow it to work efficiently on a client without consuming too many resources.

**STRUCTURED QUERY LANGUAGE (SQL)**

To work with data in a database, you must use a set of commands and statements (language) defined by the DBMS software. There are several different languages that can be used with relational databases; the most common is SQL. Both the American National Standards Institute (ANSI) and the International Standards Organization (ISO) have defined standards for SQL. Most modern DBMS products support the Entry Level of SQL-92, the latest SQL standard (published in 1992).

### SQL Server Features

Microsoft SQL Server supports a set of features that result in the following benefits:

### Ease of installation, deployment, and use: SQL Server includes a set of administrative and development tools that improve your ability to install, deploy, manage, and use SQL Server across several sites.

**Scalability:** The same database engine can be used across platforms ranging from laptop computers running Microsoft Windows® 95/98 to large, multiprocessor servers running Microsoft Windows NT®, Enterprise Edition.

### Data warehousing: SQL Server includes tools for extracting and analyzing summary data for online analytical processing (OLAP). SQL Server also includes tools for visually designing databases and analyzing data using English-based questions.

### System integration with other server software: SQL Server integrates with e-mail, the Internet, and Windows.

### Databases: A database in Microsoft SQL Server consists of a collection of tables that contain data, and other objects, such as views, indexes, stored procedures, and triggers, defined to support activities performed with the data. The data stored in a database is usually related to a particular subject or process, such as inventory information for a manufacturing warehouse.

SQL Server can support many databases, and each database can store either interrelated data or data unrelated to that in the other databases. For example, a server can have one database that stores personnel data and another that stores product-related data. Alternatively, one database can store current customer order data, and another; related database can store historical customer orders that are used for yearly reporting. Before you create a database, it is important to understand the parts of a database and how to design these parts to ensure that the database performs well after it is implemented.

**Normalization theory:** Relations are to be normalized to avoid anomalies. In insert, update and delete operations. Normalization theory is built around the concept of normal forms. A relation is said to be in a particular form if it satisfies a certain specified set if constraints. To decide a suitable logical structure for given database design the concept of normalization, which are briefly described below.

* 1st Normal Form (1 N.F): A relation is said to be in 1 NF is and only if all unaligned domains contain values only. That is the fields of an n-set should have no group items and no repeating groups.
* 2nd Normal Form (2 N.F): A relation is said to be in 2 NF is and only if it is in 1 NF and every non key attribute is fully dependent on primary key. This normal take care of functional dependencies on non-key attributes.
* 3rd Normal Form (3 N.F): A relation is said to be in 3 NF is and only if it is in 2 NF and every non key attribute is non transitively dependent on the primary key. This normal form avoids the transitive dependencies on the primary key.
* Boyce code Normal Form (BCNF): This is a stronger definition than that of NF. A relation is said to be in BCNF if and only if every determinant is a Candidate key.
* 4th Normal Form (4 NF): A relation is said to be in 4 NF if and only if whenever there exists a multi valued dependency in a relation say A->->B then all of the relation are also functionally dependent on A (i.e. A->X for all attributes x of the relation.).
* 5th Normal Form (5 NF) OR Projection Join Normal Form (PJNF): A relation R is in 5 NF. if and only if every join dependency in R is implied by the candidate key on R. A relation can’t be non-loss split into two tables but can be split into three tables. This is called Join Dependency.

**MIDDLEWARE TECHNOLOGY**

**ActiveX Data Objects.Net Overview:** ADO.NET is an evolution of the ADO data access model that directly addresses user requirements for developing scalable applications. It was designed specifically for the web with scalability, statelessness, and XML in mind.

ADO.NET uses some ADO objects, such as the Connection and Command objects, and also introduces new objects. Key new ADO.NET objects include the Dataset, Data Reader, and Data Adapter.

The important distinction between this evolved stage of ADO.NET and previous data architectures is that there exists an object -- the Dataset -- that is separate and distinct from any data stores. Because of that, the Dataset functions as a standalone entity. You can think of the Dataset as an always disconnected record set that knows nothing about the source or destination of the data it contains. Inside a Dataset, much like in a database, there are tables, columns, relationships, constraints, views, and so forth.

A Data Adapter is the object that connects to the database to fill the Dataset. Then, it connects back to the database to update the data there, based on operations performed while the dataset held the data. In the past, data processing has been primarily connection-based. Now, in an effort to make multi-tiered apps more efficient, data processing is turning to a message-based approach that revolves around chunks of information. At the center of this approach is the Data Adapter, which provides a bridge to retrieve and save data between a Dataset and its source data store. It accomplishes this by means of requests to the appropriate SQL commands made against the data store.

The XML-based Dataset object provides a consistent programming model that works with all models of data storage: flat, relational, and hierarchical. It does this by having no 'knowledge' of the source of its data, and by representing the data that it holds as collections and data types. No matter what the source of the data within the Dataset is, it is manipulated through the same set of standard APIs exposed through the Dataset and its subordinate objects.

While the Dataset has no knowledge of the source of its data, the managed provider has detailed and specific information. The role of the managed provider is to connect, fill, and persist the Dataset to and from data stores. The OLE DB and SQL Server .NET Data Providers (System.Data.OleDb and System.Data.SqlClient) that are part of the .Net Framework provide four basic objects: the Command, Connection, Data Reader and Data Adapter. In the remaining sections of this document, we'll walk through each part of the Dataset and the OLE DB/SQL Server .NET Data Providers explaining what they are, and how to program against them. The following sections will introduce you to some objects that have evolved, and some that are new. These objects are:

* Connections. For connection to and managing transactions against a database.
* Commands. For issuing SQL commands against a database.
* Data Readers. For reading a forward-only stream of data records from a SQL Server data source.
* Datasets. For storing, removing and programming against flat data, XML data and relational data.
* Data Adapters. For pushing data into a Dataset, and reconciling data against a database.

When dealing with connections to a database, there are two different options: SQL Server .NET Data Provider (System.Data.SqlClient) and OLE DB .NET Data Provider (System.Data.OleDb). In these samples we will use the SQL Server .NET Data Provider. These are written to talk directly to Microsoft SQL Server. The OLE DB .NET Data Provider is used to talk to any OLE DB provider (as it uses OLE DB underneath).

Connections: Connections are used to 'talk to' databases, and are represented by provider-specific classes such as SQLConnection. Commands travel over connections and result sets are returned in the form of streams which can be read by a Data Reader object, or pushed into a Dataset object.

Commands: Commands contain the information that is submitted to a database, and are represented by provider-specific classes such as SQLCommand. A command can be a stored procedure call, an UPDATE statement, or a statement that returns results. You can also use input and output parameters, and return values as part of your command syntax. The example below shows how to issue an INSERT statement against the North wind database.

**Data Readers:** The Data Reader object is somewhat synonymous with a read-only/forward-only cursor over data. The Data Reader API supports flat as well as hierarchical data. A Data Reader object is returned after executing a command against a database. The format of the returned Data Reader object is different from a record set. For example, you might use the Data Reader to show the results of a search list in a web page.

**Datasets:** The Dataset object is similar to the ADO Record set object, but more powerful, and with one other important distinction: the Dataset is always disconnected. The Dataset object represents a cache of data, with database-like structures such as tables, columns, relationships, and constraints. However, though a Dataset can and does behave much like a database, it is important to remember that Dataset objects do not interact directly with databases, or other source data. This allows the developer to work with a programming model that is always consistent, regardless of where the source data resides. Data coming from a database, an XML file, from code, or user input can all be placed into Dataset objects. Then, as changes are made to the Dataset they can be tracked and verified before updating the source data. The Get Changes method of the Dataset object actually creates a second Dataset that contains only the changes to the data. This Dataset is then used by a Data Adapter (or other objects) to update the original data source. The Dataset has many XML characteristics, including the ability to produce and consume XML data and XML schemas. XML schemas can be used to describe schemas interchanged via Web Services. In fact, a Dataset with a schema can actually be compiled for type safety and statement completion.

**Data Adapters (OLEDB/SQL):** The Data Adapter object works as a bridge between the Dataset and the source data. Using the provider-specific SqlDataAdapter (along with its associated SqlCommand and SqlConnection) can increase overall performance when working with a Microsoft SQL Server databases. For other OLE DB-supported databases, you would use the OleDbDataAdapter object and its associated OleDbCommand and OleDbConnection objects. The Data Adapter object uses commands to update the data source after changes have been made to the Dataset. Using the Fill method of the Data Adapter calls the SELECT command; using the Update method calls the INSERT, UPDATE or DELETE command for each changed row. You can explicitly set these commands in order to control the statements used at runtime to resolve changes, including the use of stored procedures. For ad-hoc scenarios, a Command Builder object can generate these at run-time based upon a select statement. However, this run-time generation requires an extra round-trip to the server in order to gather required metadata, so explicitly providing the INSERT, UPDATE, and DELETE commands at design time will result in better run-time performance.

* + ADO.NET is the next evolution of ADO for the .Net Framework.
  + ADO.NET was created with n-Tier, statelessness and XML in the forefront. Two new objects, the Dataset and Data Adapter, are provided for these scenarios. ADO.NET can be used to get data from a stream, or to store data in a cache for updates.
  + There is a lot more information about ADO.NET in the documentation.
  + Remember, you can execute a command directly against the database in order to do inserts, updates, and deletes. You don't need to first put data into a Dataset in order to insert, update, or delete it.

Also, you can use a Dataset to bind to the data, move through the data, and navigate data relationships

# SAMPLE CODE

**6.3.1 CODE FOR REGISTRATION**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.Configuration;

using System.Data;

public partial class UserRegistration : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

SqlConnection con = new SqlConnection(ConfigurationManager.ConnectionStrings["con"].ConnectionString);

protected void Button1\_Click(object sender, EventArgs e)

{

if (con.State == ConnectionState.Open)

{

con.Close();

}

con.Open();

SqlCommand cmd = new SqlCommand();

cmd.CommandType = CommandType.StoredProcedure;

cmd.CommandText = "sp\_UserRegistration";

cmd.Parameters.AddWithValue("@username", txtUname.Text);

cmd.Parameters.AddWithValue("@password", txtPswd.Text);

cmd.Parameters.AddWithValue("@MobileNo", txtMobile.Text);

cmd.Parameters.AddWithValue("@email", txtEmailID.Text);

cmd.Parameters.AddWithValue("@fName", txtFname.Text);

cmd.Parameters.AddWithValue("@LName", txtLanme.Text);

cmd.Parameters.AddWithValue("@Address", txtAddress.Text);

cmd.Parameters.AddWithValue("@State", txtState.Text);

cmd.Parameters.AddWithValue("@country", txtCountry.Text);

cmd.Parameters.AddWithValue("@ZipCode", txtZipCode.Text);

cmd.Connection = con;

int res = cmd.ExecuteNonQuery();

if (res> 0)

{

ClientScript.RegisterStartupScript(Page.GetType(), "validation", "<script language='javascript'>alert('Registration Successfully')</script>");

}

ClientScript.RegisterStartupScript(Page.GetType(), "validation", "<script language='javascript'>alert('Registration not Successfully')</script>");

}

}

* + 1. **CODE FOR HOMEPAGE**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class Admin\_Home : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["aname"] == null || string.IsNullOrEmpty(Session["aname"].ToString()))

{

Response.Redirect("~/Login.aspx/?r=Admin/Home.aspx");

}

}

}

* + 1. **CODE FOR ADD FOOD ITEMS**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.IO;

using System.Data;

using System.Configuration;

public partial class Admin\_AddFoodItems : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

//if (Session["aname"] == null || string.IsNullOrEmpty(Session["aname"].ToString()))

//{

// Response.Redirect("~/Login.aspx/?r=Admin/AddFoodItems.aspx");

//}

if (!IsPostBack)

{

BindGridView();

}

}

SqlConnection con = new SqlConnection(ConfigurationManager.ConnectionStrings["con"].ConnectionString);

protected void BindGridView()

{

if (con.State == ConnectionState.Open)

{

con.Close();

}

DataTable dt = new DataTable();

SqlDataAdapter da = new SqlDataAdapter("Select ItemId,ItemName,Rate,ItemPath,ManufacturingDate,ExpiryDate,Quantity from AddItem", con);

con.Open();

da.Fill(dt);

con.Close();

if (dt.Rows.Count > 0)

{

GridView1.DataSource = dt;

GridView1.DataBind();

}

}

protected void btnSubmit\_Click(object sender, EventArgs e)

{

if (FileUpload1.PostedFile != null)

{

string FileName = Path.GetFileName(FileUpload1.PostedFile.FileName);

//Save files to disk

FileUpload1.SaveAs(Server.MapPath("../Images/" + FileName));

con.Open();

//Add Entry to DataBase

//String strConnString = System.Configuration.ConfigurationManager.ConnectionStrings["con"].ConnectionString;

//SqlConnection con = new SqlConnection(strConnString);

//string strQuery = "spAddItems";

//SqlCommand cmd = new SqlCommand(strQuery);

SqlCommand cmd = new SqlCommand();

cmd.CommandType = CommandType.Text;

cmd.CommandText = "INSERT INTO AddItem(ItemName,Rate,ItemPath,CId,ManufacturingDate,ExpiryDate,Quantity) VALUES (@ItemName,@Rate,@ItemPath,@CId,@ManufacturingDate,@ExpiryDate,@Quantity)";

cmd.Parameters.AddWithValue("@ItemName",txtItemName.Text);

cmd.Parameters.AddWithValue("@CId", DropDownList1.SelectedIndex);

cmd.Parameters.AddWithValue("@Rate", txtRate.Text);

cmd.Parameters.AddWithValue("@ManufacturingDate", TextBox1.Text);

cmd.Parameters.AddWithValue("@ExpiryDate", TextBox2.Text);

cmd.Parameters.AddWithValue("@Quantity", TextBox3.Text);

cmd.Parameters.AddWithValue("@ItemPath", "../Images/" + FileName);

cmd.Connection = con;

try

{

cmd.ExecuteNonQuery();

ClientScript.RegisterStartupScript(Page.GetType(), "validation", "<script language='javascript'>alert('Added successfully')</script>");

BindGridView();

txtItemName.Text = "";

txtRate.Text = "";

}

catch (Exception ex)

{

Response.Write(ex.Message);

}

finally

{

con.Close();

con.Dispose();

}

}

}

protected void GridView1\_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)

{

GridView1.EditIndex = -1;

BindGridView();

}

protected void GridView1\_RowEditing(object sender, GridViewEditEventArgs e)

{

GridView1.EditIndex = e.NewEditIndex;

BindGridView();

}

protected void GridView1\_RowUpdating(object sender, GridViewUpdateEventArgs e)

{

string ItemId = GridView1.DataKeys[e.RowIndex].Value.ToString();

// find values for update

//TextBox = (TextBox)GridView1.Rows[e.RowIndex].FindControl("txtItemId");

TextBox ItemName = (TextBox)GridView1.Rows[e.RowIndex].FindControl("txtItemName");

TextBox ManufacturingDate = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox1");

TextBox ExpiryDate = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox2");

TextBox Quantity = (TextBox)GridView1.Rows[e.RowIndex].FindControl("TextBox3");

TextBox Rate = (TextBox)GridView1.Rows[e.RowIndex].FindControl("txtRate");

FileUpload FileUpload1 = (FileUpload)GridView1.Rows[e.RowIndex].FindControl("FileUpload1");

string path = "../Images/";

if (FileUpload1.HasFile)

{

path += FileUpload1.FileName;

//save image in folder

FileUpload1.SaveAs(MapPath(path));

}

else

{

// use previous user image if new image is not changed

Image img = (Image)GridView1.Rows[e.RowIndex].FindControl("img\_user");

path = img.ImageUrl;

}

SqlCommand cmd = new SqlCommand("update AddItem set ItemName='" + ItemName.Text + "',Quantity='" + Quantity.Text+ "',ExpiryDate='" + ExpiryDate.Text + "',ManufacturingDate='" + ManufacturingDate.Text + "',Rate='" + Rate.Text + "',ItemPath='" + path + "' where ItemId=" + ItemId + "", con);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

GridView1.EditIndex = -1;

BindGridView();

}

protected void GridView1\_RowDeleting(object sender, GridViewDeleteEventArgs e)

{

string ItemId = GridView1.DataKeys[e.RowIndex].Value.ToString();

con.Open();

SqlCommand cmd = new SqlCommand("delete from AddItem where ItemId=" + ItemId, con);

int result = cmd.ExecuteNonQuery();

con.Close();

if (result == 1)

{

BindGridView();

}

}

protected void GridView1\_PageIndexChanging(object sender, GridViewPageEventArgs e)

{

GridView1.PageIndex = e.NewPageIndex;

BindGridView();

}

}

# 7. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

### TYPES OF TESTS

**7.1 UNIT TESTING**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**7.2 INTEGRATION TESTING**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**7.3 FUNCTIONAL TEST**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

**Valid Input:** identified classes of valid input must be accepted.

**Invalid Input:** identified classes of invalid input must be rejected.

**Functions:** identified functions must be exercised.

**Output:** identified classes of application outputs must be exercised.

**Systems/Procedures**: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**7.4 SYSTEM TEST**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**7.5 WHITE BOX TESTING**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**7.6 BLACK BOX TESTING**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**7.7 TEST CASES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case**  **ID** | **Test**  **Objective** | **Excepted**  **Result** | **Obtained**  **Result** | **Pass/Fail** |
| TC001 | Verify admin login with username and password | It should display admin page | Loads admin page | Pass |
| TC002 | Verify user login with username and password | It should display user page | Loads user page | Pass |
| TC003 | User should select the items and add items to cart | It should display total items and add to cart option | Shows total items and items added to cart | Pass |
| TC004 | Show total amount of items in cart | It should display the details and total amount of all items | Shows the details and total amount of all items | Pass |
| TC005 | Remove item from cart | It should remove desired items from cart and display updated cart | Removes desired items and shows the updated cart details | Pass |
| TC006 | Proceed to checkout | It should display total details, amount of cart, shipping details and payment options | Shows the cart details, shipping details and payment options | Pass |

Fig 7.1 Showing various test cases

**8. OUTPUTS**

**OUTPUT 1:**

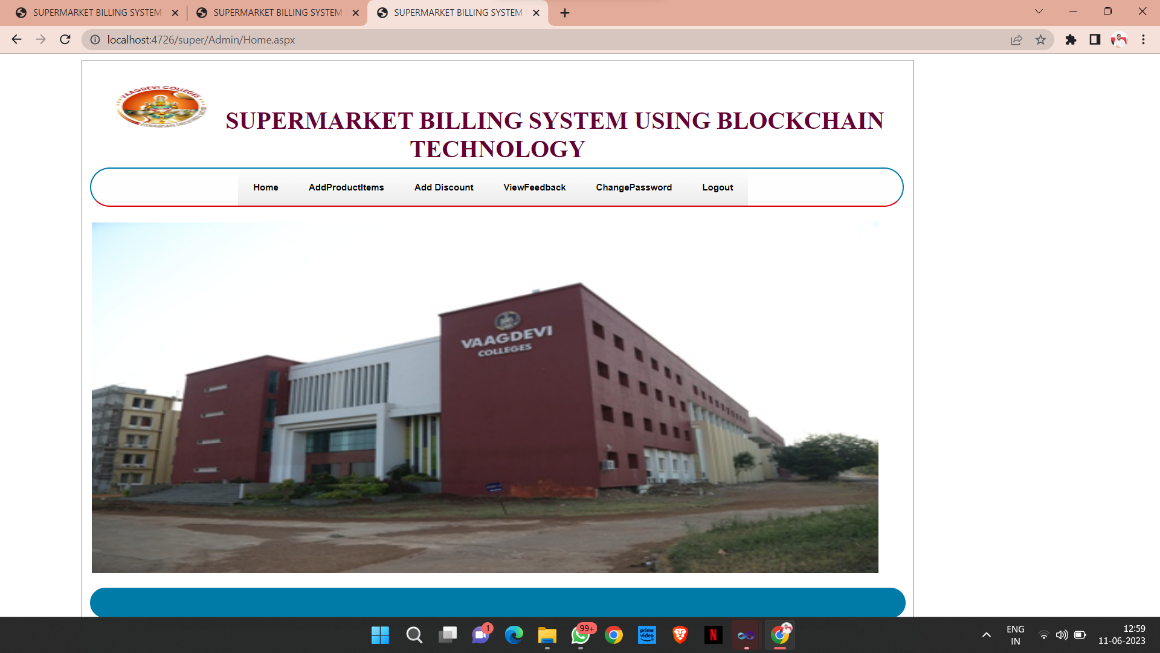
****

Fig 8.1 Home page

### OUTPUT 2:

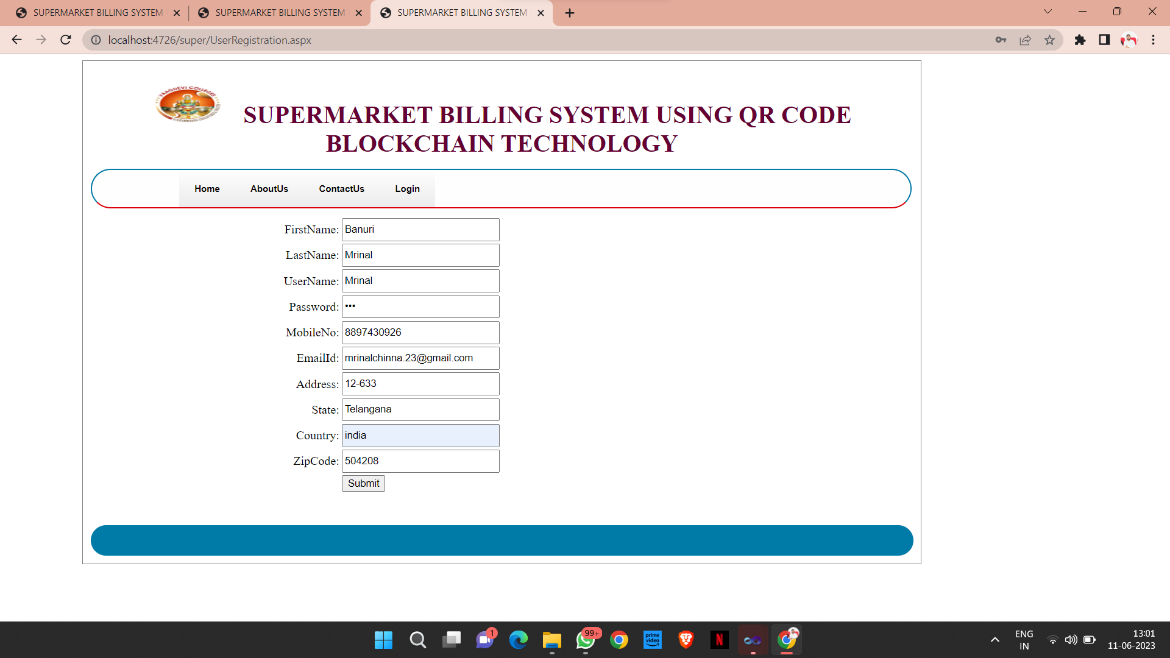


Fig 8.2 User Registration Page

**OUTPUT 3:**

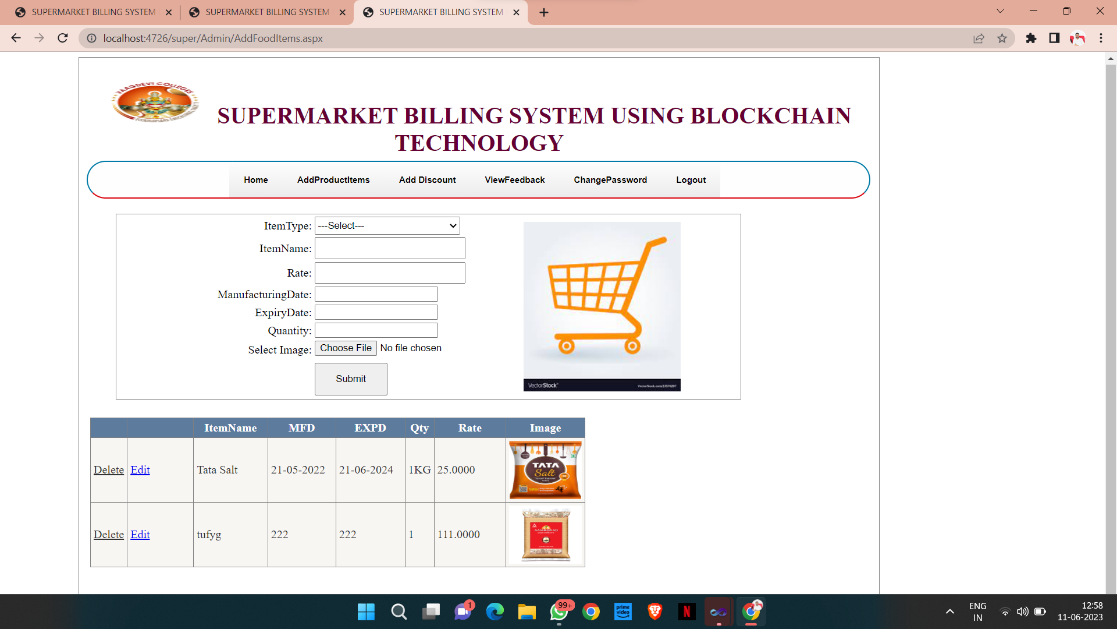


Fig 8.3 Add Products Page

**OUTPUT 4:**

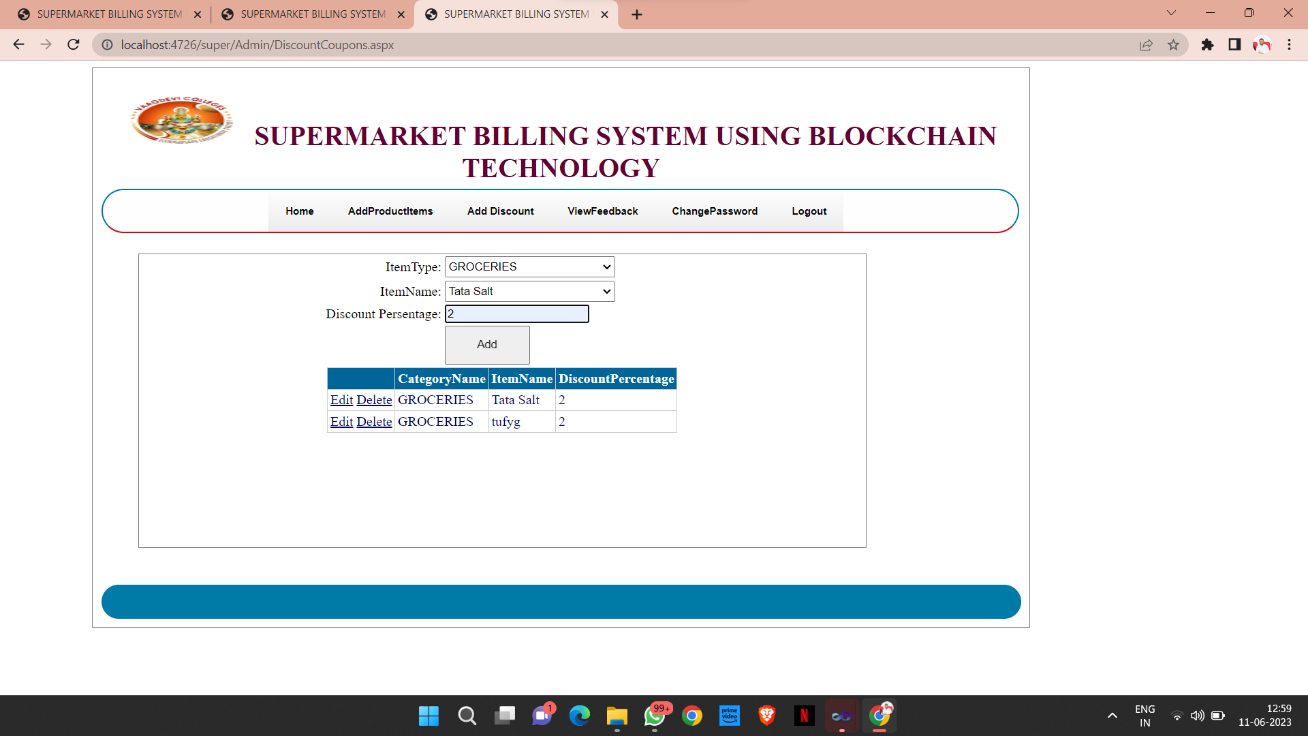
****

Fig 8.4 Add Discount Page

### OUTPUT 5:

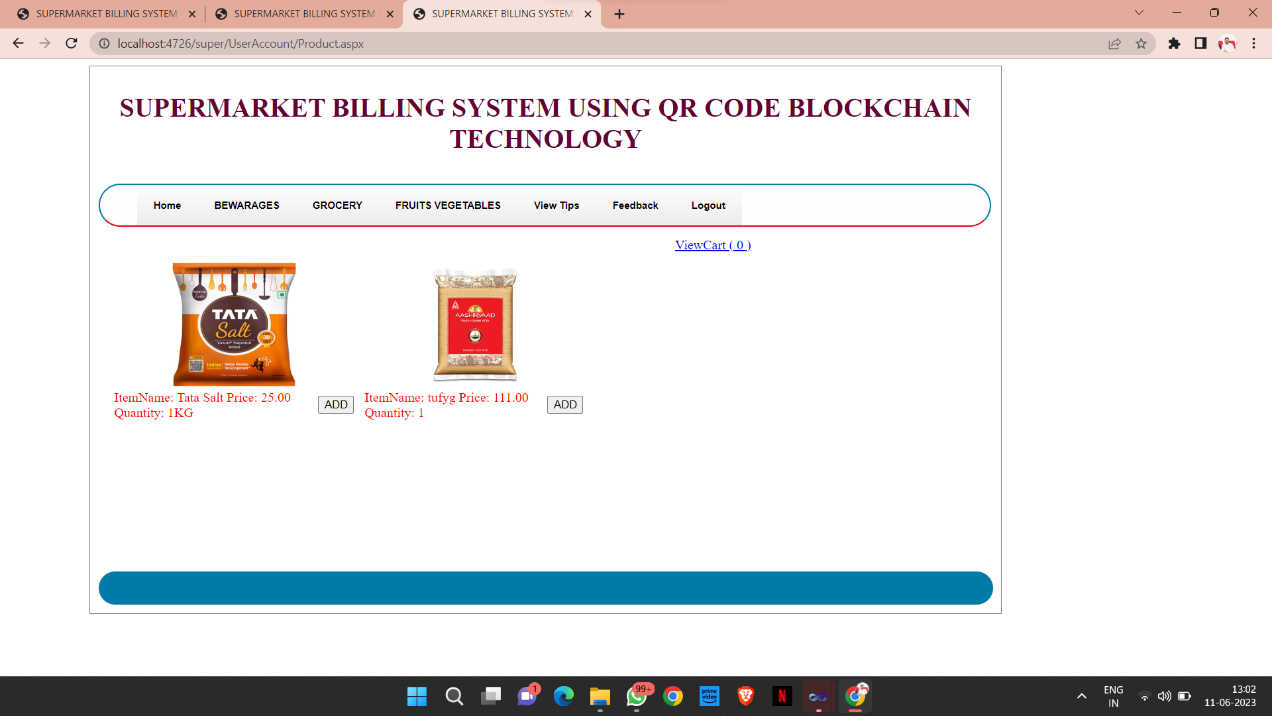


Fig 8.5 Products Display Page

**OUTPUT 6:**

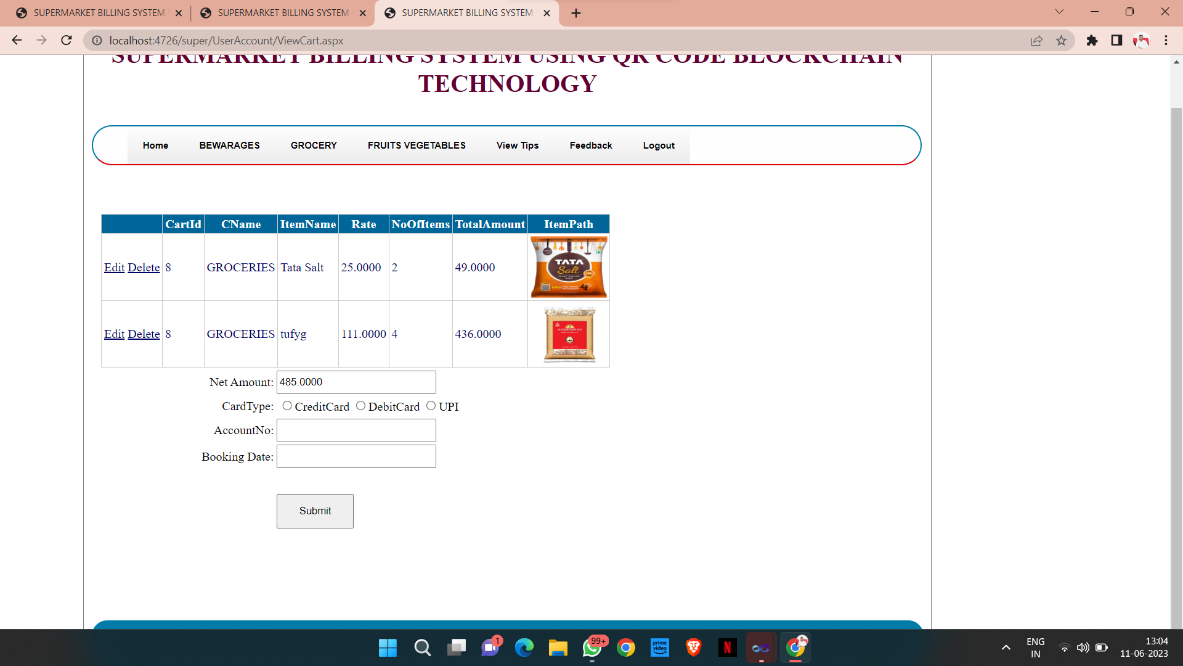
****

Fig 8.6 View Cart Page

**OUTPUT 7:**

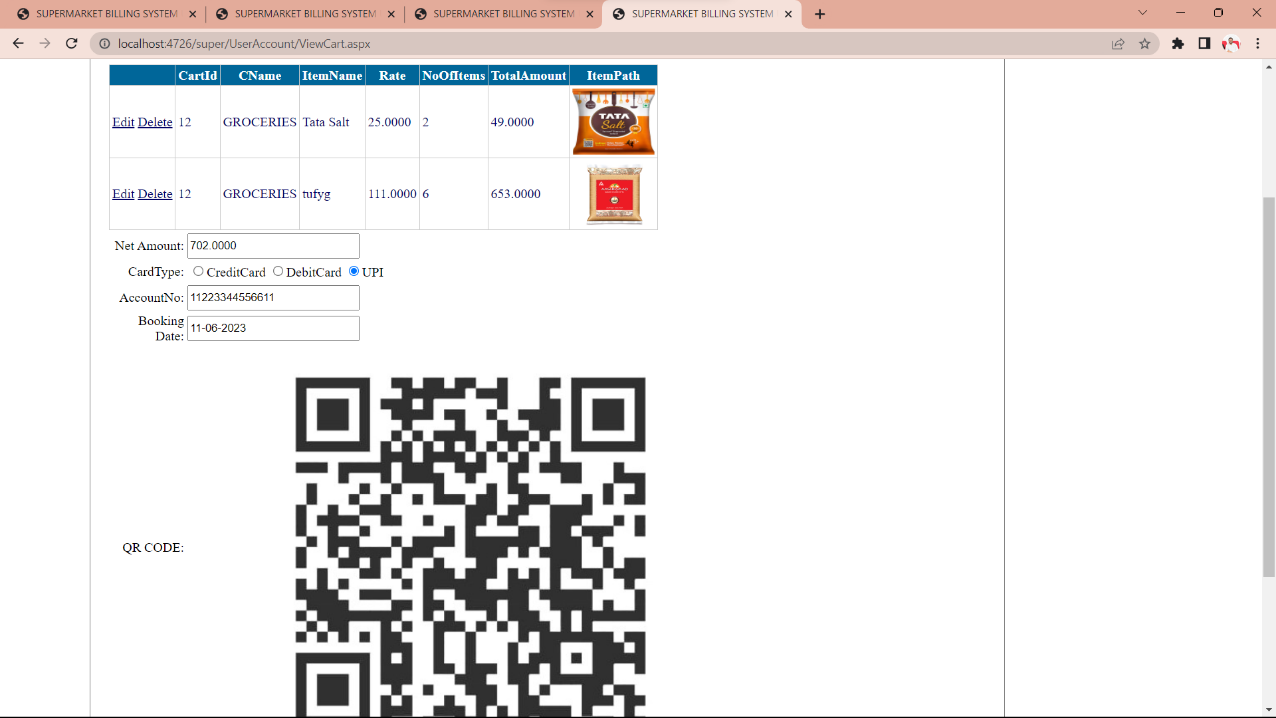
****

Fig 8.7 Payment Options Page

**OUTPUT 8:**

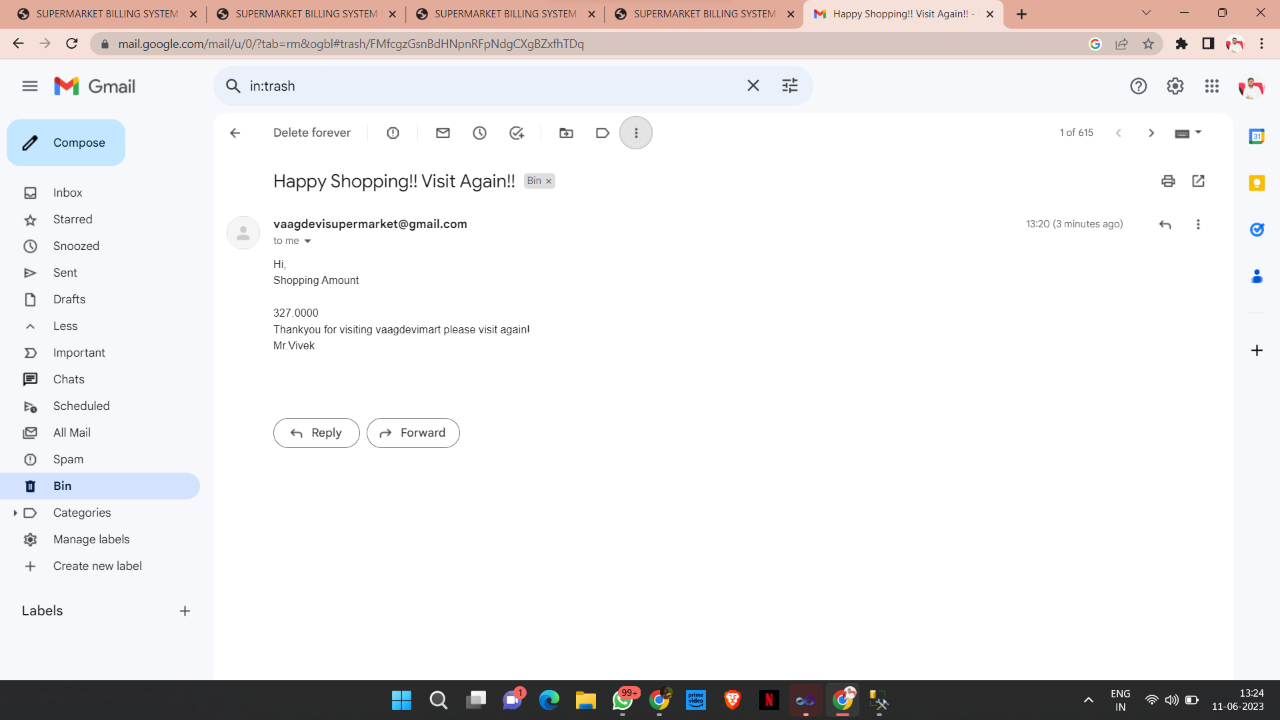
****

Fig 8.8 Payment Successful

### 9. CONCLUSIONS AND FUTURE SCOPE

### 

### CONCLUSION

### 

The supermarket billing system using webcam aim is to make a system that is useful to the retail shop by reducing time consumption in bill counter. No need for specialized hardware for installing the system the classroom. It can be assembled using a camera and computer. This system is advantageous over barcode billing system. This method is accurate enough and reliable.

In this project with the tremendous use of TensorFlow, the object detection is done. And every method has its own advantage and disadvantage. But in this we tried to overcome the existing system problems. When compared to the other system the proposed system has more advantages and secure as we make change only in the billing process. Using this system both costumers and management will get a better shopping experience.

The experiment results can be shown that the system can achieve high precision counting and high recognition accuracy. A. Conflict Of Interest The authors declare that they have no conflict of interest to report regarding the present study

**FUTURESCOPE**

The future scope of supermarket billing system using QR code blockchain technology is very promising. QR codes are a quick and easy way for customers to scan and pay for their groceries, while blockchain technology can help to ensure that transactions are secure and transparent.

Here are some of the benefits of using QR code blockchain technology for supermarket billing:

* + **Increased efficiency:** QR code scanning can help to speed up the checkout process, which can lead to shorter wait times for customers.
  + **Improved security:** Blockchain technology can help to prevent fraud and errors, as all transactions are recorded on a secure and transparent ledger.
  + **Enhanced customer experience:** Customers can pay for their groceries quickly and easily using their smartphones, which can lead to a more positive shopping experience.

As the adoption of QR code blockchain technology continues to grow, it is likely that we will see even more benefits for supermarkets and their customers. For example, supermarkets may be able to offer loyalty rewards or discounts to customers who pay using QR code blockchain technology. Additionally, blockchain technology could be used to track the provenance of food products, which could help to increase customer confidence in the food supply chain.

Overall, the future scope of supermarket billing system using QR code blockchain technology is very promising. This technology has the potential to improve efficiency, security, and the customer experience in supermarkets.

## 

**10.REFERENCES**

1. K. He, X. Zhang, S. Ren, and J. Sun, ‘‘Deep residual learning for image recognition,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2016, pp. 770–778.
2. Y.-D. Zhang, Y. Zhang, X.-X. Hou, H. Chen, and S.-H. Wang, ‘‘Sevenlayer deep neural network based on sparse autoencoder for voxelwise detection of cerebral microbleed,’’ Multimedia Tools Appl., vol. 77, no. 9, pp. 10521–10538, May 2018.
3. S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko, ‘‘Translating videos to natural language using deep recurrent neural networks,’’ 2014, arXiv:1412.4729. [Online]. Available: <https://arxiv.org/abs/1412.4729>
4. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, ‘‘Mastering the game of Go with deep neural networks and tree search,’’ Nature, vol. 529, no. 7587, pp. 484–489, 2016.
5. C. Bishop, Pattern Recognition and Machine Learning. New York, NY, USA: Springer, 2006.
6. Y. LeCun, Y. Bengio, and G. Hinton, ‘‘Deep learning,’’ Nature, vol. 521, no. 7553, pp. 436–444, 2015.
7. G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, ‘‘Towards a better understanding of context and context-awareness,’’ in Proc. Int. Symp. Handheld Ubiquitous Comput. Berlin, Germany: Springer, 1999, pp. 304–307.
8. C. Dwork, ‘‘Differential privacy,’’ in Encyclopedia of Cryptography and Security, H. C. A. van Tilborg and S. Jajodia, Eds. Boston, MA, USA: Springer, 2011.
9. L. Bottou, ‘‘Large-scale machine learning with stochastic gradient descent,’’ in Proc. COMPSTAT, 2010, pp. 177–186.
10. A. Kankanhalli, Y. Charalabidis, and S. Mellouli, ‘‘IoT and AI for smart government: A research agenda,’’ Government Inf. Quart., vol. 36, no. 2, pp. 304–309, 2019.