|  |  |
| --- | --- |
|  | **Cognizant Academy**  **Generic Billing System**  **ASP .Net MVC,**  **Entity Framework,**  **SQL Server**  **Integrated Capability Test**  **Version 1.0** |
|  |
|  |

Table of Contents

[1.0 Introduction 3](#_Toc38027157)

[1.0 Purpose of this document 3](#_Toc38027158)

[2.0 Definitions & Acronyms 3](#_Toc38027159)

[3.0 Project Overview 3](#_Toc38027160)

[4.0 Scope 4](#_Toc38027161)

[5.0 Hardware and Software Requirment 4](#_Toc38027162)

[2.0 System diagram 4](#_Toc38027163)

[3.0 Component Specification – Model 5](#_Toc38027164)

[4.0 Design of Home Pages of Generic Billing System 7](#_Toc38027168)

[1.0 Requirement flow 7](#_Toc38027169)

[2.0 Create Restaurant 7](#_Toc38027170)

[3.0 Technical guidelines 7](#_Toc38027171)

[5.0 Design of Prouduct Page of Generic Billing System 8](#_Toc38027168)

[1.0 Requirement flow 8](#_Toc38027169)

[2.0 Create Restaurant 8](#_Toc38027170)

[3.0 Technical guidelines 7](#_Toc38027171)

[6.0 Design of Billing Page of Generic Billing System 10](#_Toc38027168)

[1.0 Requirement flow 10](#_Toc38027169)

[2.0 Create Restaurant 11](#_Toc38027170)

[3.0 Technical guidelines 11](#_Toc38027171)

[7.0 Standards and Guidelines 13](#_Toc38027173)

[1.0 Controller & View 13](#_Toc38027174)

[8.0 Design constraints 13](#_Toc38027175)

[9.0 Code submission Instructions 1](#_Toc38027176)3

[10.0 Evaluation Areas 13](#_Toc38027177)

# **Introduction**

## Purpose of this document

The purpose of this document is to define the server side implementation of the Generic Billing System web application.

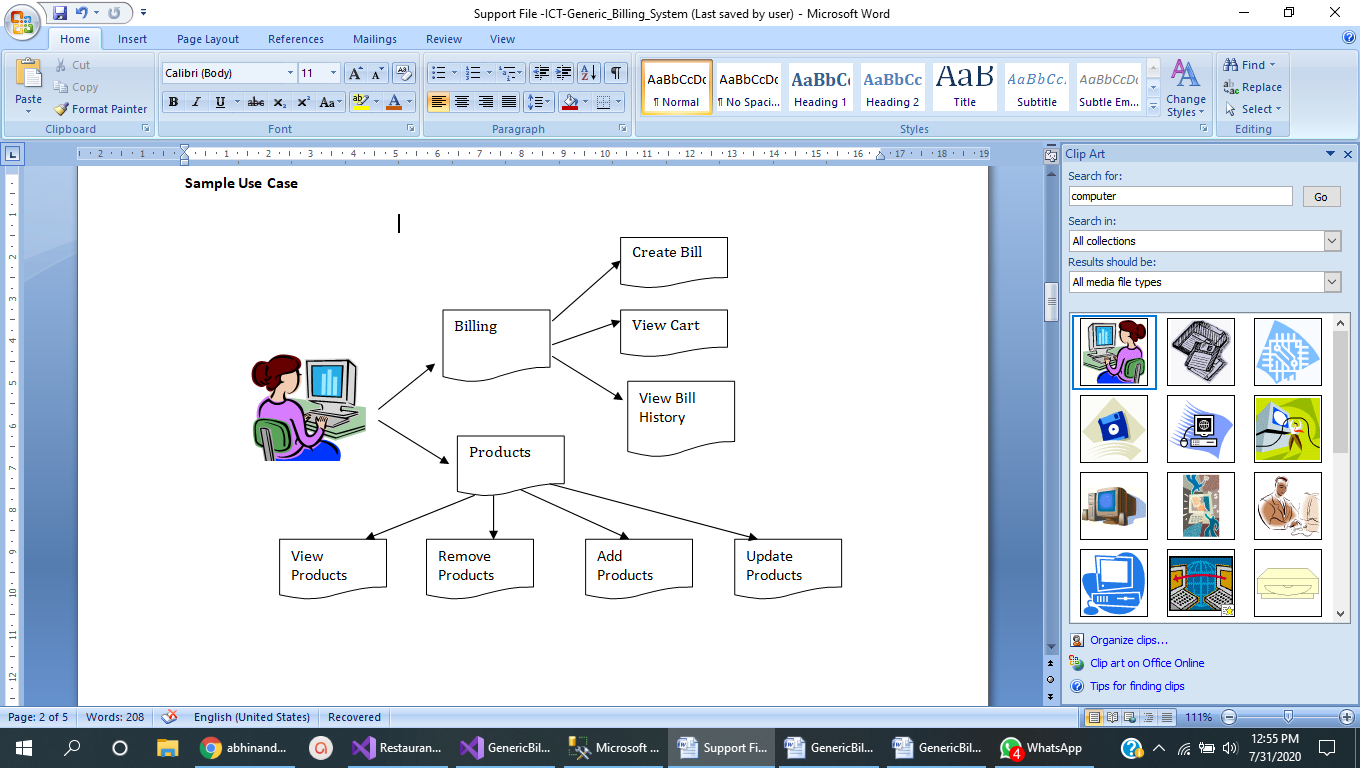
## Definitions& Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| ASP.NET MVC | ASP.Net MVC is a Web development framework built on top of ASP.Net with certain changes in the internal workings of web page rendering. |

## Project Overview

This Project contains development of web application to automate Generic Sales and billing application. Any retail shop can use this application in generic way and they can maintain Products information and can perform billing operations. The Retail outlet has to feed all the products information first before making bills.

Use Case Diagram

****

## Scope

1. Creation of ASP.Net MVC web application for Generic Billing application

## Hardware and Software Requirment

1. Hardware Requirement:
   1. Developer PC with 8GB RAM
2. Software Requirement
   1. IE or Chrome
   2. .Net Framework 4.5
   3. Visual Studio Professional Edition 2015
   4. SQL Server enterprise edition 2014

# **System diagram**

System Admin

Add / Edit / Remove / View all Products

Create Bill

View Cart

Show Bill Histories

Generic Billing System Database

## **3. 0 Component Specification– Models**

1. Create Web Application of MVC Template in Visual Studio. Add Entity Framework in to your application using NUGet Package manager.

2. Create following Models and enumerated types in models folder

|  |  |
| --- | --- |
| public enum UnitOfMeasure  {    Grams,  Kgs,  Milli\_Litre,  Litres,  Pcs,  Boxes,  Metre  } | public enum ProductStatus  {  Live,  Removed  } |

**Table: Products**

|  |  |  |
| --- | --- | --- |
| **Property** | **Data type** | **Data Annotations / Additional Details** |
| ProdCode | Integer | [Key] |
| Name | Char(30) | [Required] [Display(Name = "Product Name")] |
| UnitOfMeasure | Custom Enum type | [Display(Name = "Unit of Measure")] |
| Price | Float | [Range(1,5000,ErrorMessage = "Price Should be between 1 and 5000")] [Display(Name = "Price")] |
| Product Status | Custom Enum Type | [Display(Name = "Product Status")] |

**Table: Billing**

|  |  |  |
| --- | --- | --- |
| **Property** | **Data type** | **Data Annotations / Additional Details** |
| Bill Number | Integer | [Key] |
| Date | Datetime | No Annotations (System Generated field) |
| Items | List<BilledItem> | Virtual type Navigation Property for EF |

**Table: BilledItem**

|  |  |  |
| --- | --- | --- |
| **Property** | **Data type** | **Data Annotations / Additional Details** |
| Bill Number | Integer | [Key] [Column(Order = 1)] |
| ProdCode | Integer | [Key] [Column(Order = 2)] |
| Quantity | Integer | [Required] |
| SoldPrice | Float | No Annotations (Read Only field) |
| Product | Product | Virtural type Navigation field for EF of Product Type |
| Billing | Billing | Virtural type Navigation field for EF of Billing Type |

3. Create BillingDBContext model in Models folder like below

using System.Data.Entity;

public class BillingDBContext:DbContext

{

public DbSet<Product> Products { get; set; }

public DbSet<Billing> Billings { get; set; }

public DbSet<BilledItem> BilledItems { get; set; }

public BillingDBContext() : base("BillingDBConStr"){}

}

4. Create ConnectionString block in web.config file as below

<connectionStrings>

<add name="BillingDBConStr" connectionString="Data Source=(local);Initial Catalog=ICT\_GBDB;Integrated Security=True" providerName="System.Data.SqlClient"/>

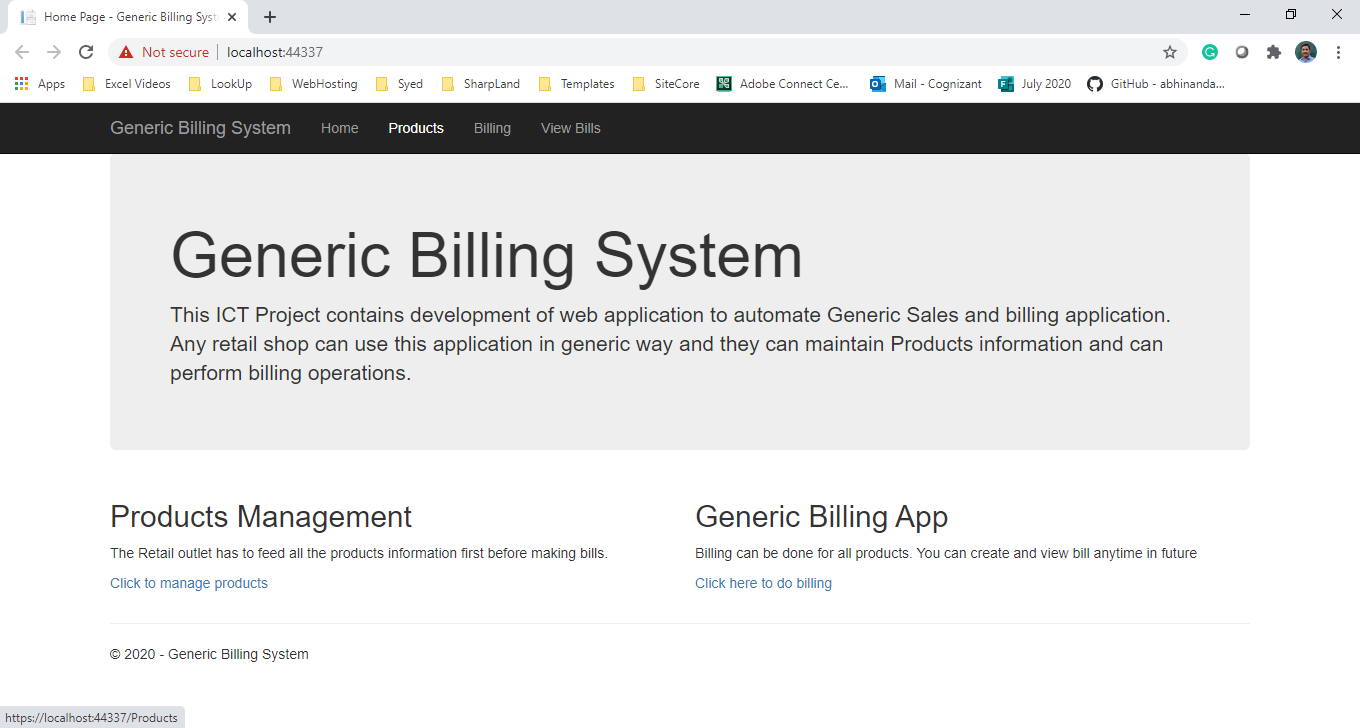
</connectionStrings>

Note: Data Souce can be assigned as per your SQL SERVER INSTANCE name.

# **4.0 Design of HomePage of Generic Billing System**

## Requirement flow

When Application user launches **Generic Billing application**, The following Home page has to be displayed with all redirections links*.*



**Technical Guidelines**

Create Home controller with facilities to transfer to other controllers

**Home Controller**

**Products Controller**

**Billing Controller**

With following action methods

public ActionResult Index()

{

return View();

}

public void Products()

{

//code toredirect

}

public void Billing()

{

//code to redirect

}

# **5.0 Design of Products Page of Generic Billing System**

**Home Controller**

Add Products

(Create)

List Products

(Read)

Remove Products

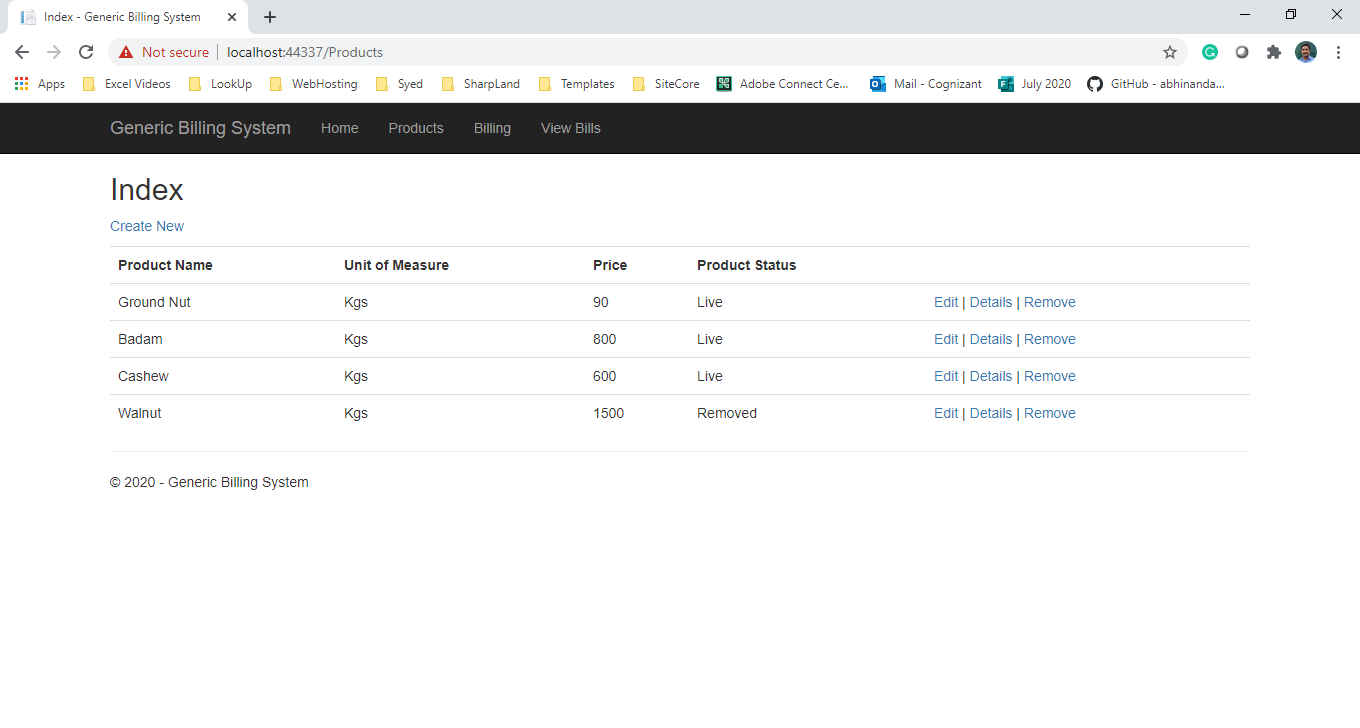
(Delete)

Remove Products

(Update)

## Requirement flow

**Products Page with Create / Edit / Remove and View Details Options**



## Technical Guidelines

1. In Products Controller, declare **public static BillingDBContext db;** as global variable to all action methods will be used in all action methods to access Database context.

2. Define following action methods

public ActionResult Index()

{

//Add Code to display products from products property of DBContext

}

[HttpGet]

public ActionResult Create()

{

//Add Code to render UI to create new product (add view create.cshtml)

}

[HttpPost]

[ActionName("Create")]

public ActionResult Create\_Post(Product Prod)

{

//Add Code to Store Product record into database

}

[HttpGet]

public ActionResult Details(int id)

{

//Add code to display individual product details (add view details.cshtml)

}

[HttpGet]

public ActionResult Edit(int id)

{

//Add code to render UI to edit individual product details (add view edit.cshtml)

}

[HttpPost]

[ActionName("Edit")]

public ActionResult Edit\_Post(Product prod)

{

//Add Code to Store Edited Product record into database

}

[HttpGet]

public ActionResult Delete(int id)

{

//Add Code to render delete UI for delete confirmation (add view delete.cshtml)

}

[HttpPost]

[ActionName("Delete")]

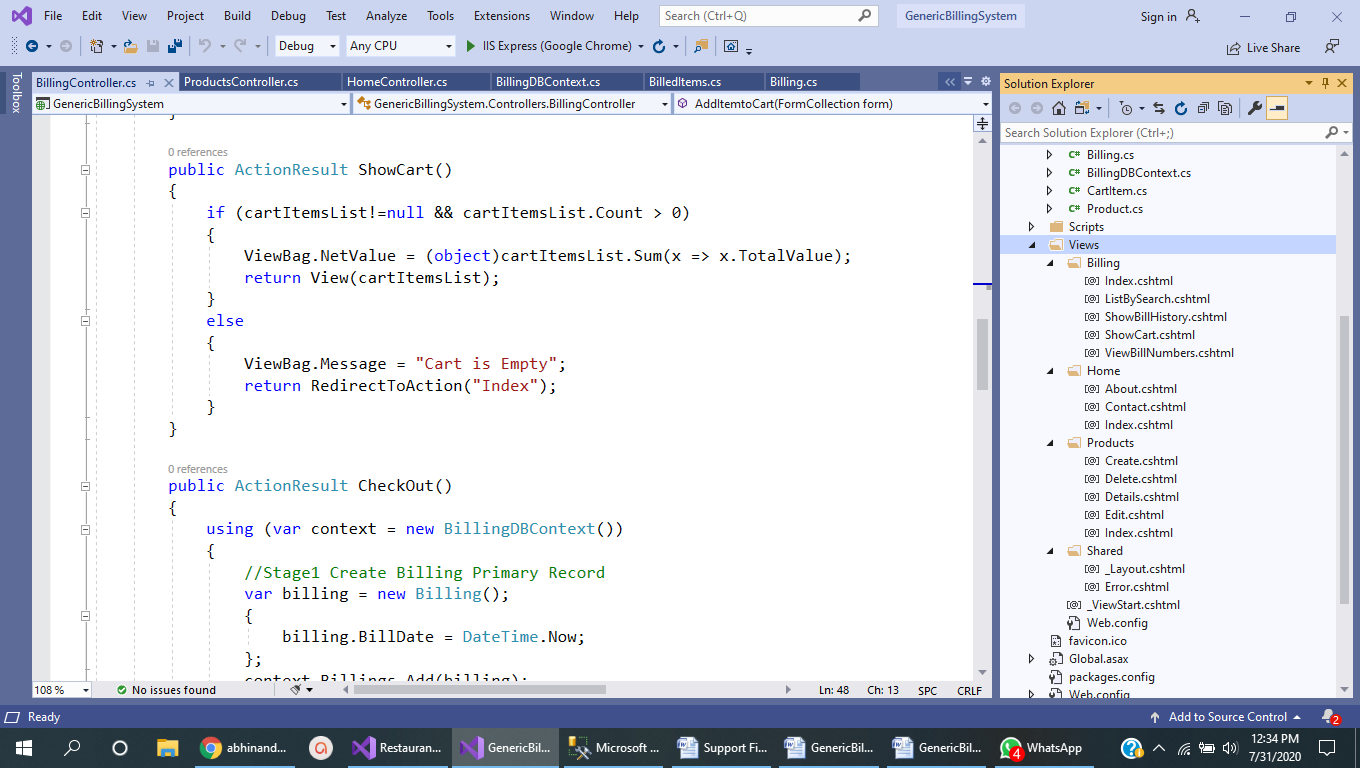
public ActionResult Delete\_Post(int id)

{

//Add Code to delete product

}

**Views Needed for this controller actions**



# **6.0 Design of Billing Page of Generic Billing System**

## Requirement flow

Billing controls should have following facilities as below,

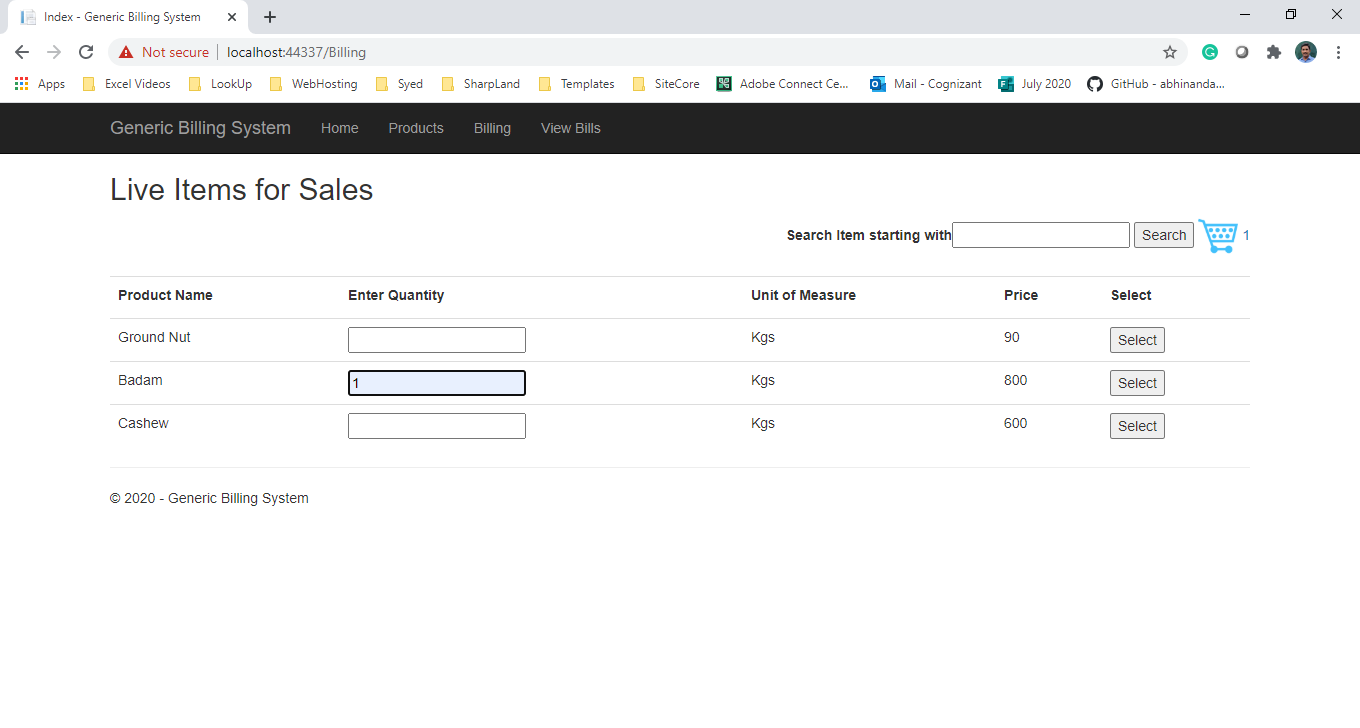
Billing Page (Quanitty has to be entered and User has to click Select to add it to cart)

**Billing Controller**

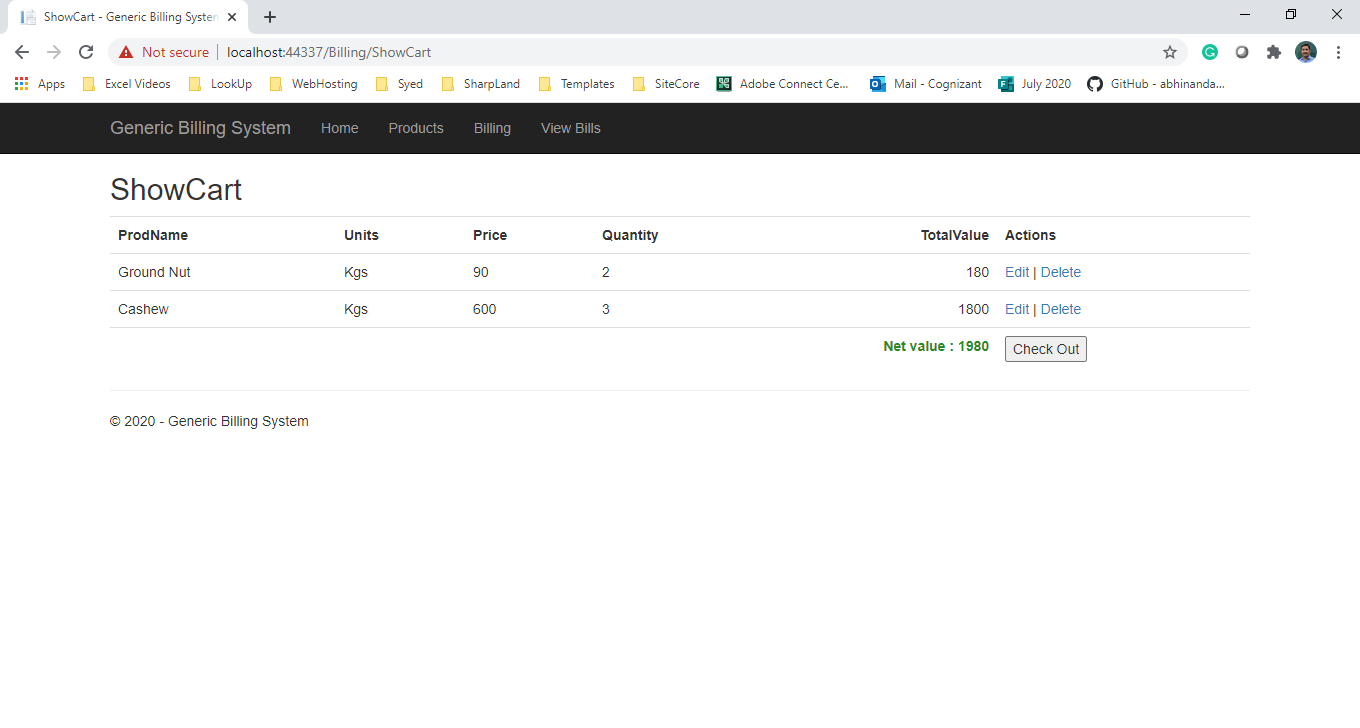
Create Bill (Create)

View Bill History

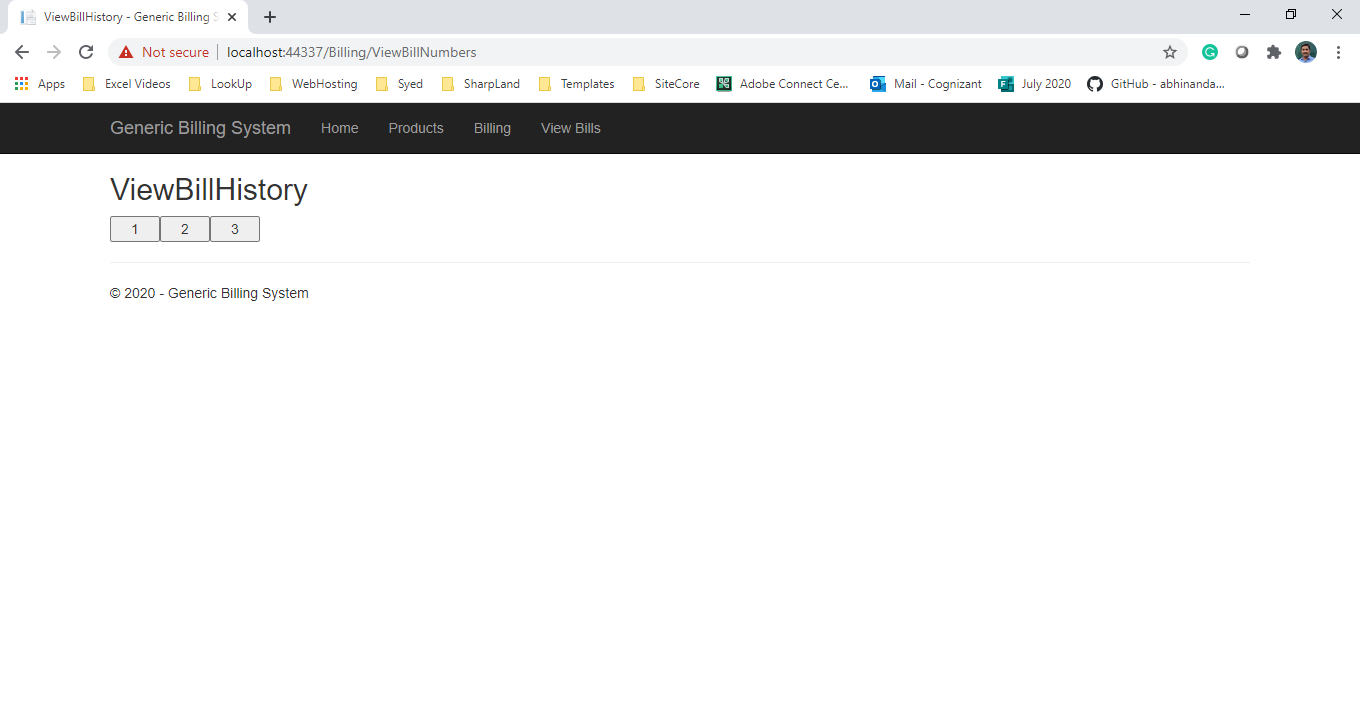
View Bill (Read)



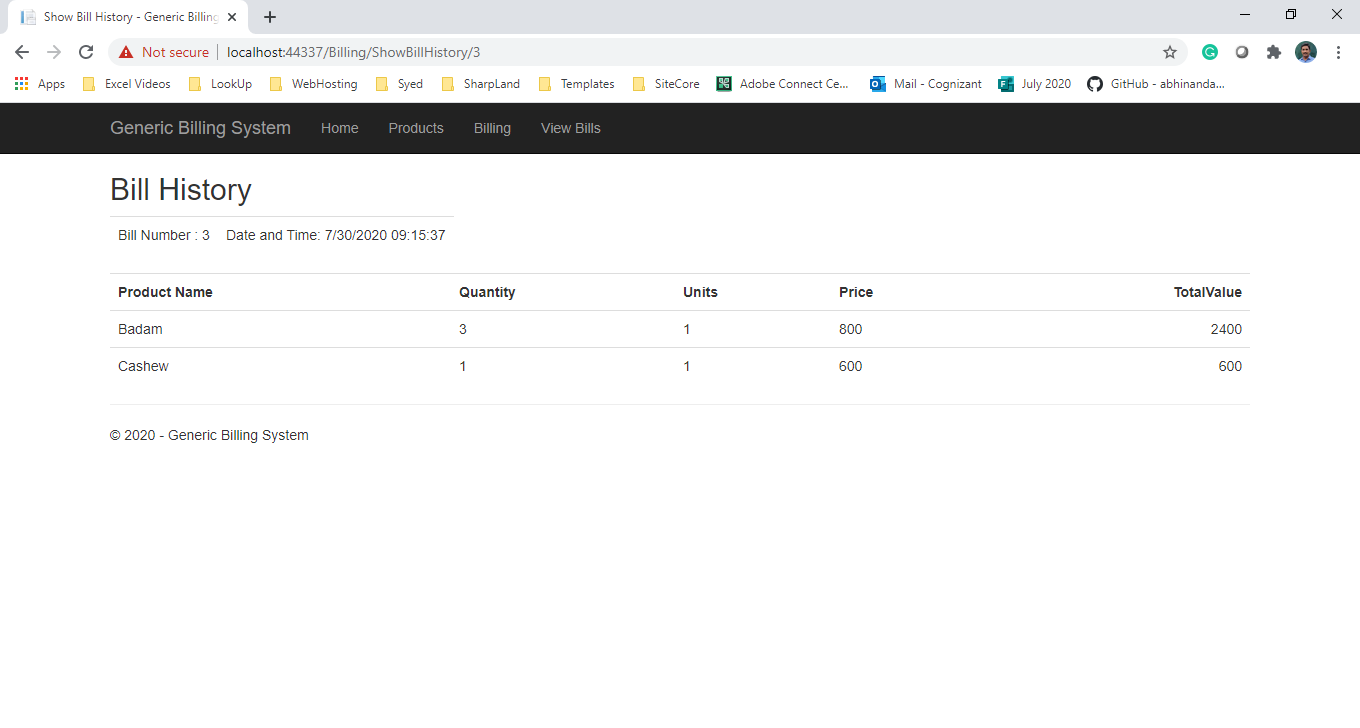
View Cart Page with Check out Option



Bill Numbers Listing



Bill History Viewing



## Technical Guidelines

1. In Billing Controller, declare **public static BillingDBContext db;** will be used in all action methods to access Database. Define following action methods.

2.  **public static List<CartItem> cartItemsList=new List<CartItem>();** will be used to as local storage to store billed items of individual sales.

3. Declare following action methods in billingController

public ActionResult Index()

{

//Add Code

}

[HttpPost]

public ActionResult ListBySearch(string txtSearch)

{

//Add Code

}

[HttpPost]

public ActionResult AddItemtoCart(FormCollection form)

{

// Add Code to populate CartItem and add it to cartItemList

}

public ActionResult ShowCart()

{

//Add code to show cart if it is not empty

}

public ActionResult CheckOut()

{

//Stage1 Create Billing Primary Record, Fetch auto generated bill Number

//Stage2 (Create Billing Records)

}

public ActionResult ViewBillNumbers()

{

//Add code to fetch and show bill numbers

}

public ViewResult ShowBillHistory(int id)

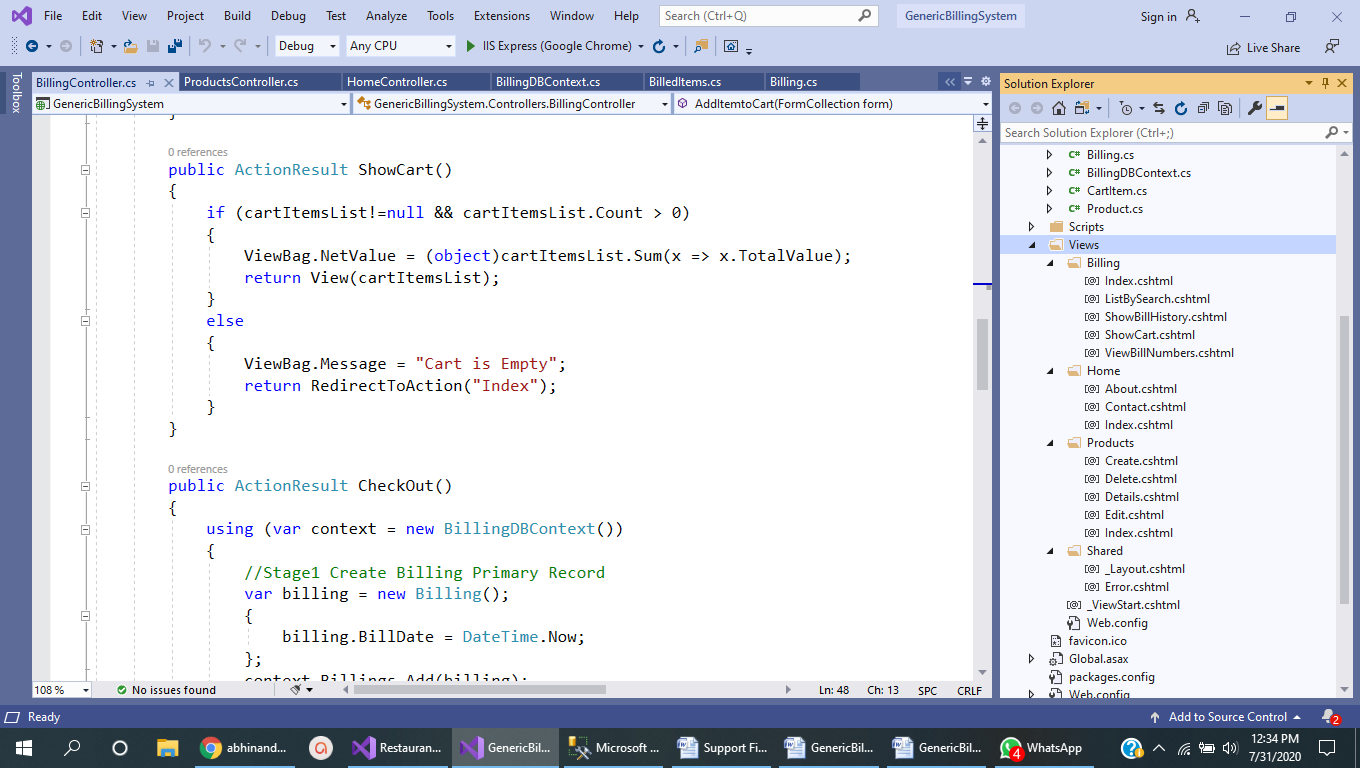
{

//Stage1 – Show Bill Primary Details

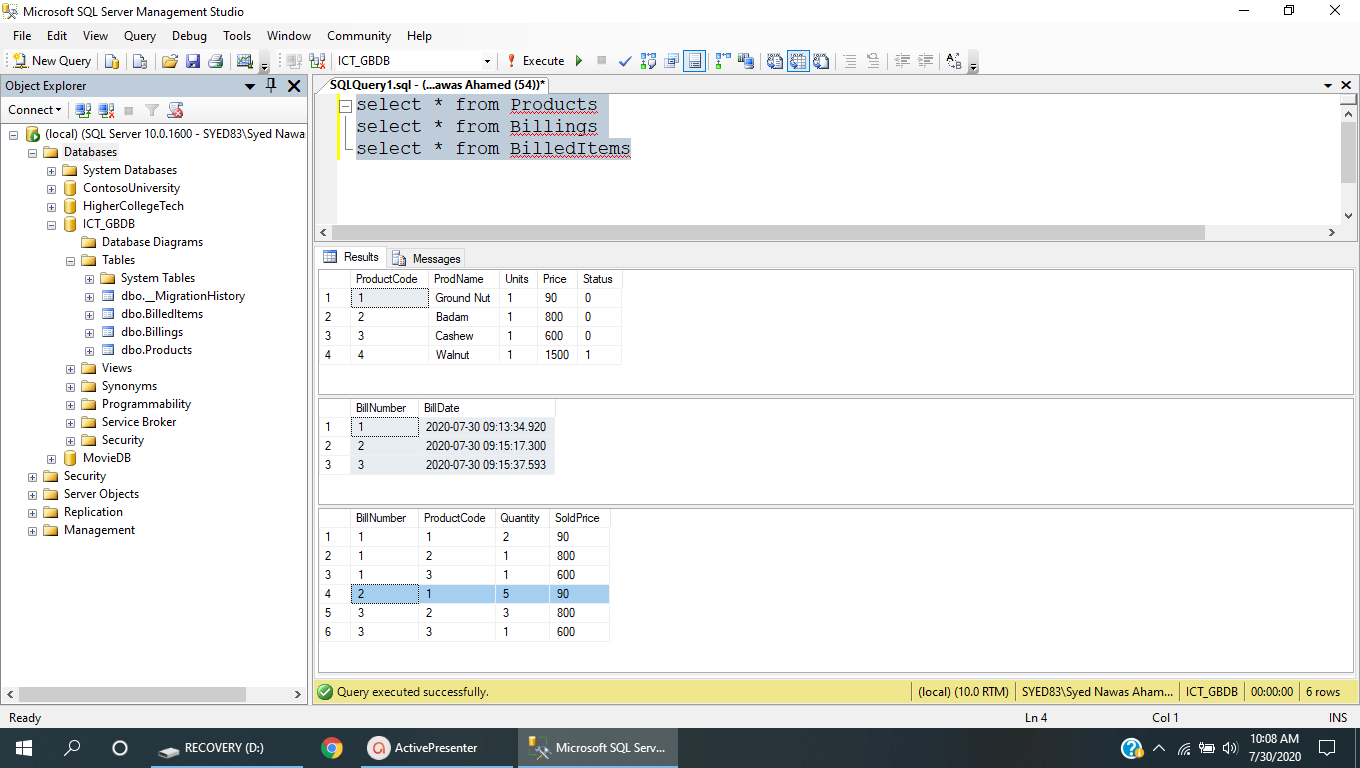
//Stage2 – Show Billed Products details

}

Views Needed for this overall application



Example Database state after successful billing operations



# 7.0 Standards and Guidelines

## Controller& View

* Action methods should have a meaningful name
* Remove unused Action methods
* There should not be any hard coded connection string values in code. It has to be referenced from Web.config file
* Database connection string should be set in the ConnectionStrings section of Web.config and NOT in the AppSettings
* Meaningful names should be given to the controls created in View

8.0 Design constraints

Required packages are already supplied with ‘Using’ statement. So do not try to add packages using Nugetpackages.

9.0 Code submission Instructions

* Do not change the code skeleton given, as your code will be auto-evaluated.
* Your last submitted solution will be considered for detailedevaluation.
* Make sure to submit the solution before the specified time limit. You will not be allowed to submit the solution once the mention time for the assessment is over.

10. Evaluation Areas

|  |  |
| --- | --- |
| 1 | Launched application lands in the Home page |
| 2 | Home Page UI contains the product and billing details |
| 3 | Link ‘Producs’ navigates to Products page |
| 5 | Link ‘Billing’ navigates to Billing page |
| 4 | All Pages related with Products CRUD operations have to be implemented |
| 5 | Implement and Test Create bill, View Bill and Bill History options |
| 6 | Implementation of BillingDBContext and all controller action methods of Home,Product and Billing controllers. |
| 7. | Ensure UI accepting valid data. Display appropriate error messages for invalid inputs |
| 8. | Navigations should be fine without null redirections. |