**Technical Documentation**

**Jade Gorton - 7100288111**

**Unit Name**

**C#**

**Assessment 1 – Technical Document**

**Table of Contents**

Contents

[Introduction 3](#_Toc420484487)

[Script 3](#_Toc420484488)

[Requirements 5](#_Toc420484489)

[Site Map 6](#_Toc420484490)

[Version control 6](#_Toc420484491)

[Final Product Solution 6](#_Toc420484492)

[Design patterns 7](#_Toc420484493)

[Debugging Techniques used 8](#_Toc420484494)

[Logging for .Net framework 9](#_Toc420484495)

[Test Data and test summary results 10](#_Toc420484496)

[Test Plan 11](#_Toc420484497)

[Interface Design layout and the Final Product 11](#_Toc420484498)

[Sign off 11](#_Toc420484499)

[References 12](#_Toc420484500)

# Introduction

This report is created to document process of creating a webpage to display in. Process required creation of program, plans and diagrams. This ASP.net Program, created database management system handle access databases without requiring purchase licence.

# 

# Script

After sending few emails regarding requirements this program needs to meet and specifications for the program. I managed put together script of appointments we had and what I view as requirements.

“Mister Sparkly” is a company that manufactures and sells laundry detergent.

The company started its operations in 2009 and has been growing rapidly and now has several employees that work in various departments namely: Administration, Sales, Purchasing, Production, and Logistics.

Mister Sparkly purchases raw materials from a variety of suppliers by sending them with a purchase order. The raw materials go through a series of process to produce laundry detergent that gets bottled in to different popular brands.

The sales agents sell the detergents to the increasing number of retailers in their allocated area in various states and a retailer is assigned to one sales agent. The retailers can **only** place their orders with the sales agents and the agents forward the orders to the Administration department to be processed by creating a Sales Order.

With Mister Sparkly’s growth, the company would like to have an integrated system that would handle the company’s sales and purchasing, as well as managing its products, raw materials, suppliers, customers, and employees.

Mister Sparkly would like the system to do the following:

* Ability to manage the growing list of suppliers
* Ability to manage the list of materials
* Manage the purchasing of raw materials from suppliers
* Ability to manage the growing list of retailers
* Ability to manage the list of different brands of detergents
* Handle the sales of detergent to the retailers by the company’s sales agents
* Ability to manage the growing list of employees including the sales agents.

The security of the system is paramount and so they would like to assign different access rights to different users. Some users may have limited access to the features of the application. For example, users in Purchasing may enter new materials but cannot enter new products; some users in Sales are able to enter sales but can only read the product details; or some users may have administration rights where they can access every feature of the application. Each user will have their own user logins and one user will only be allocated with one login; there will be no cases where one user will have multiple access rights.

Mister Sparkly wants the Ordering / Sales system done in the first stage of the project and wants it to start immediately. The Purchasing system will be done in the second stage of the project and will start upon advice by Mister Sparkly.

# 

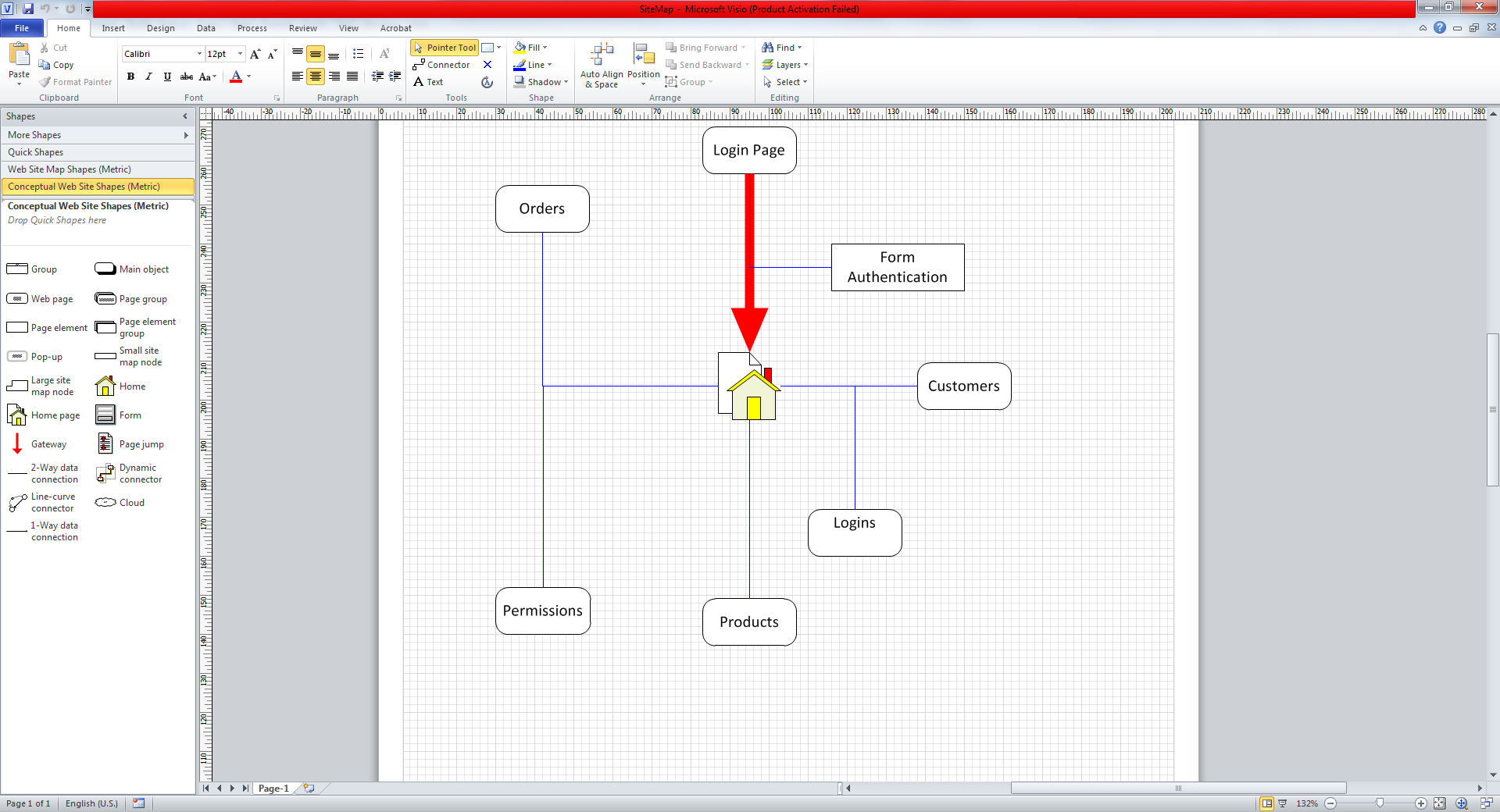
# 

# Requirements

Mr Sparkly required me prepare application handle their distribution of products in branches of their company and products they sell, including ingredients they purchase. This program will use Access engine, create database GUI for user enter and manipulate data. Each form be separated into different menus user can navigate. Users have login that allows them denying or allowing them read or write to database. Program stores everything into access database.

Requirement is create asp.net website handling database connection. Once page is created testing for application for meeting requirements be made. Once testing completed, ongoing maintenance happen make sure quality continues throughout its lifetime.

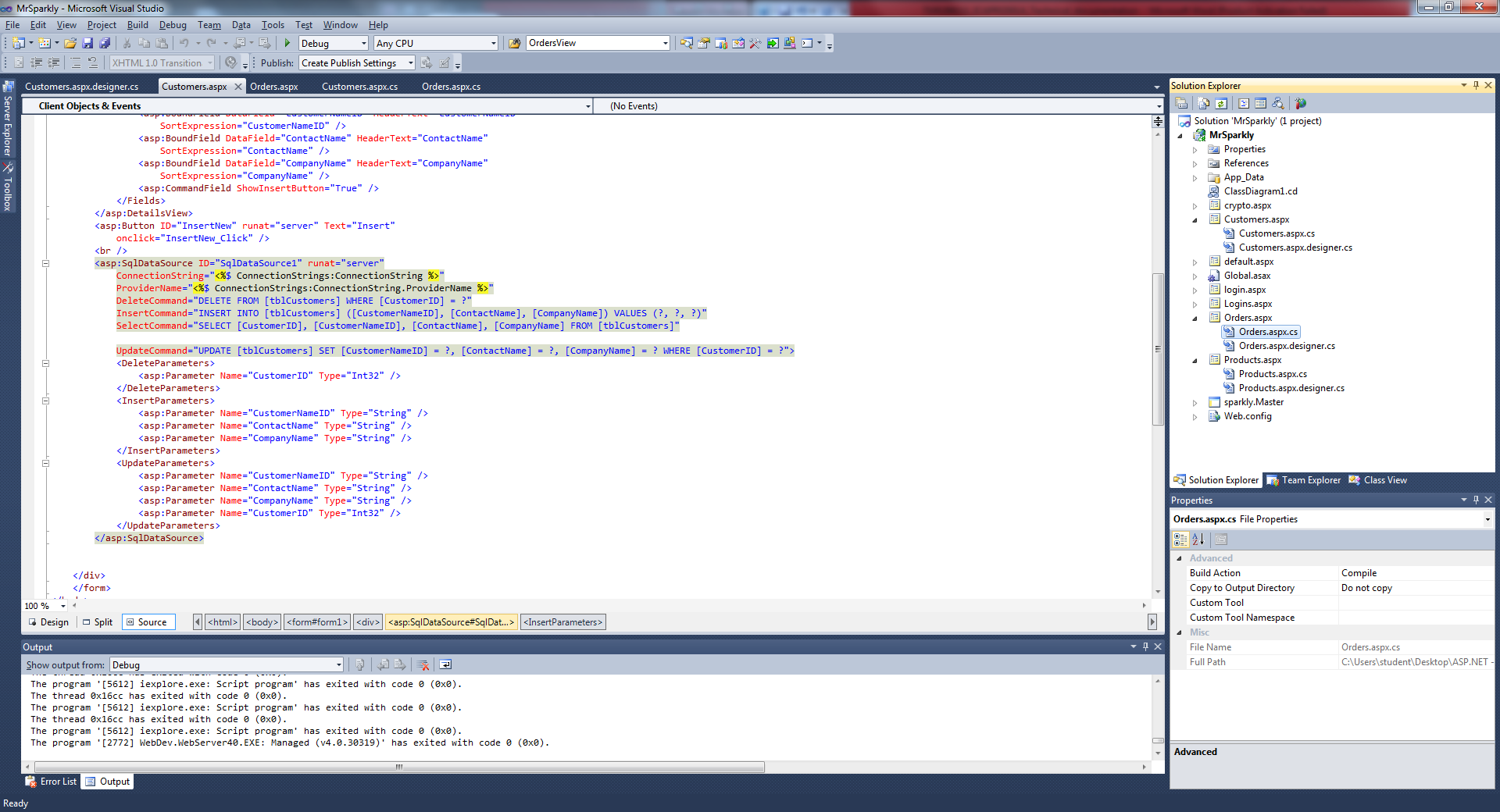
# Site Map



# Version control

Version Control is control changes regarding any additional features and creation of new and maintenance of existing features. Version control allows for documentation of changes also and others audit and review others editor’s changes made to program. Software for version control considered for this project is GitHub. GitHub allows for collaboration and control over multiple versions of program is maintained and edited.

# Final Product Solution



# Design patterns

Design patterns are general description of common reoccurring problem with a solution. These patterns occur when a general problem with general solution can be described with a design pattern. Design pattern uses formal approach for describing a design problem with appropriate solution and other factors that would require tweaking for specific solutions.

In OOP a pattern can also contain classes to describe objects, classes and attributes and dependences for general approach solve the problem. When repeated problem occurs in programmer arsenal where similar solution can be provided for multiple problems into general problem and provide a pattern for providing solution for this problem and you have design pattern.

Design patterns come in many different flavours including from strategy for algorithms for dealing with high level strategies describing how exploit application characteristics on computation platform. Most strategies regarding design patterns are structural design patterns and implementing Strategy patterns. These deal with structure of application being developed. Design pattern for how structure of classes and objects created.

Examples of design pattern can be called Singleton pattern. Purpose of singleton pattern is creating an object that can only have one instance. And how create singleton is class where constructors are private and only accessed through static public method usually called instance. By limiting creation of user created instances of Singleton you can control all uses of singleton pattern by calling instance and instantiating a new instance only if no pre-existing instance.   
  
To achieve a singleton pattern you need a static class that has private constructors. And instance method that returns instance of the class. It will check if instance is in memory and if not creates new instance using constructor however if one already created it simply return instance in memory.   
  
This self-imposed limitation on programmer so only one instance can exist at one time, an example for using singleton would be when dealing with menu screen and transitioning through menu screen within a game. You wouldn’t want creation of new menu screen whenever player quits or enters a new game & pauses. So you have single class that handles creation of menu and checks if been used before.

Another patterns includes interface allows for in C# .net framework is Iterator Pattern where allows you iterate through each element in collection requires a pattern for iterating. Interfaces allows anything with IEnumerable interface that enforces pattern enumeration requires. This one examples for C#

int[] values = new int[] {1, 2, 3, 4, 5};

foreach(int i in values)

{

Console.Write(i.ToString() + " ");

}

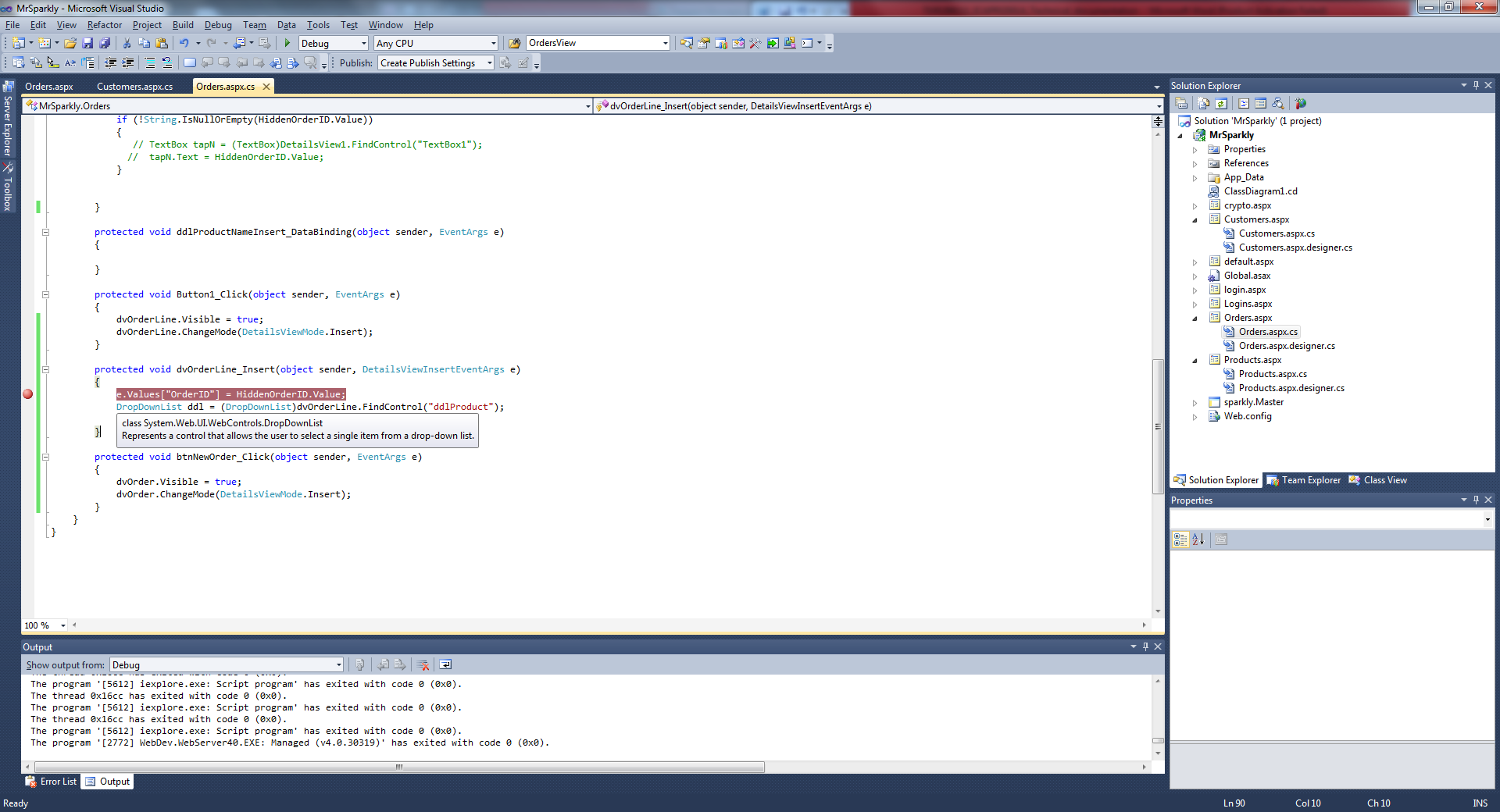
You can use IEnumerable interface create your own custom objects that can be iterated over using for each. Custom code must handle organizing and how it handles interaction over variables. However this pattern for enabling collection of objects be iterated over is invaluable when implemented correctly.

# 

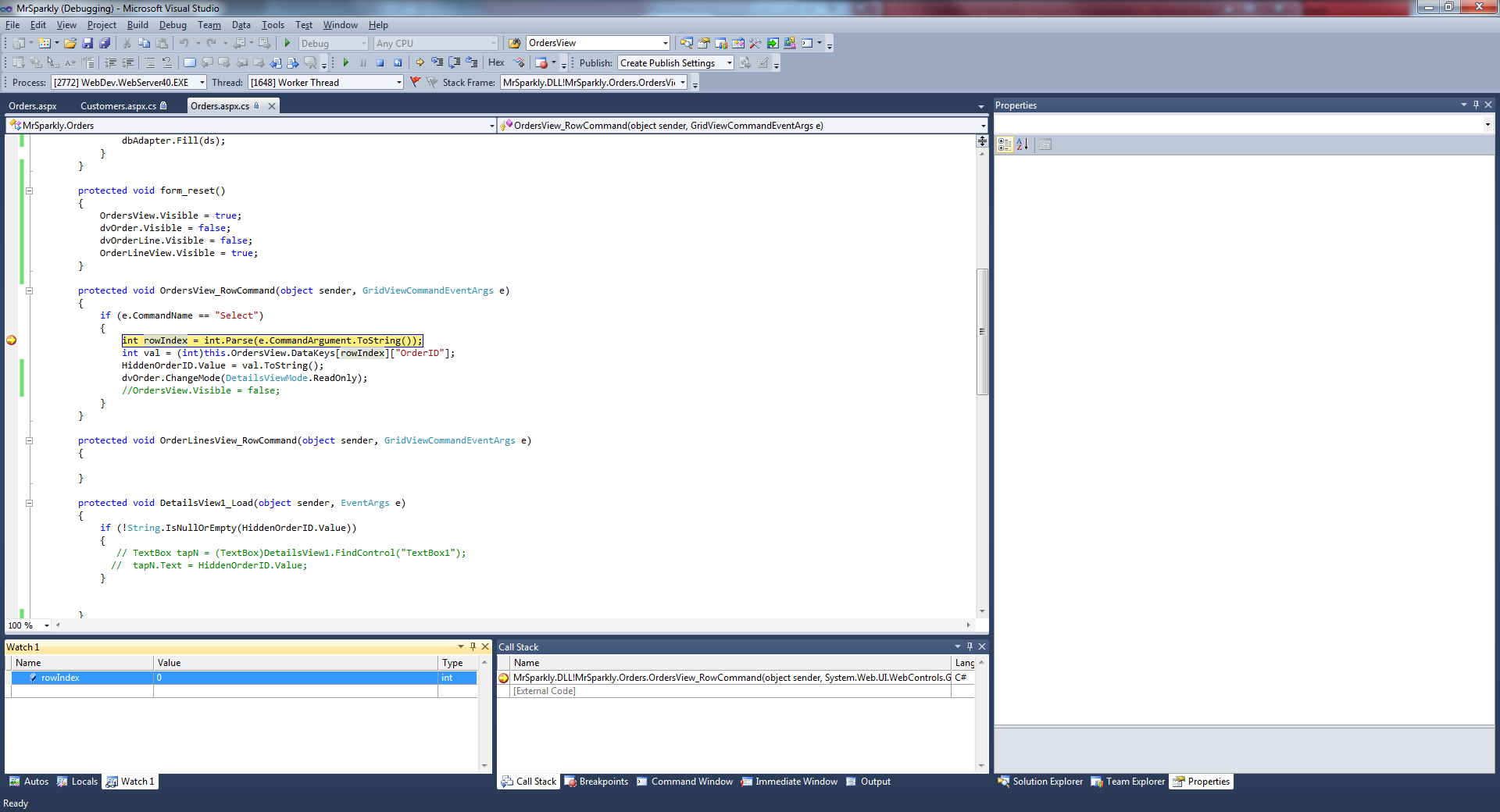
# Debugging Techniques used

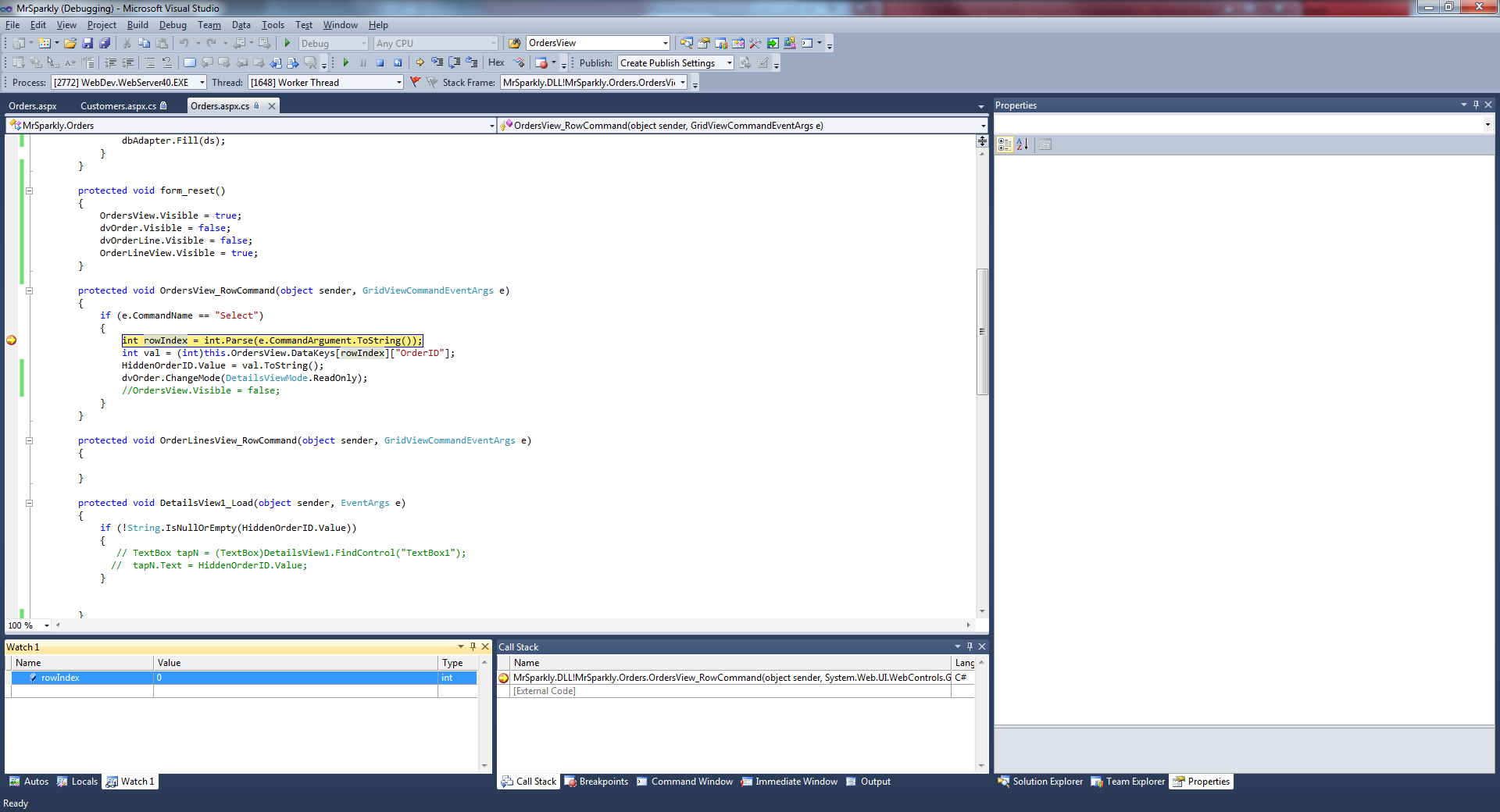
Debugging used for asp.net program was built in debugging for visual Studio 2010.  
List follows of features used for debugging.

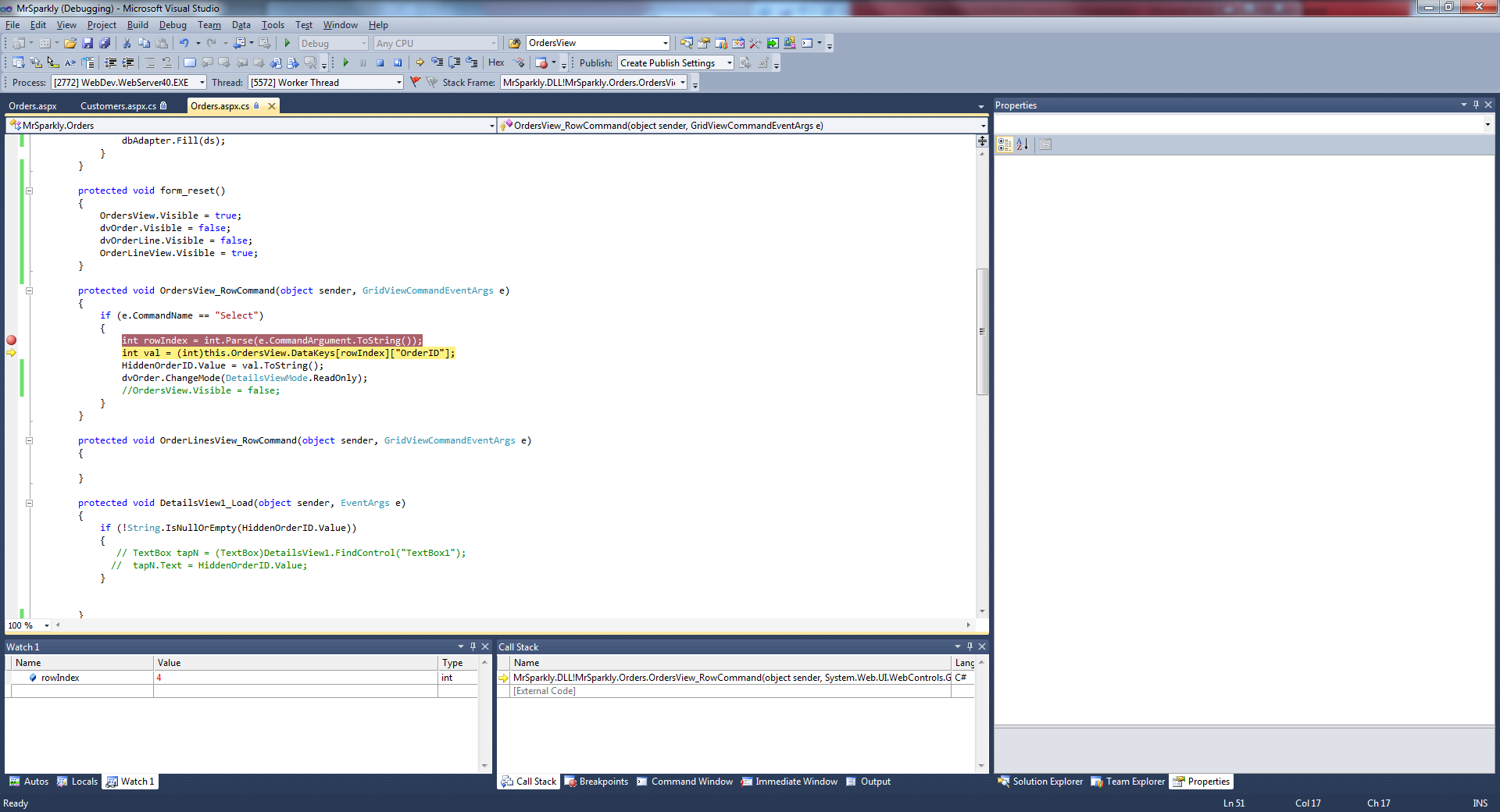
* Break points
* Watchers
* Conditional breakpoints
* Asserts / visual studio testing suite



Break points used determine if correct value for Hidden field was working during testing phase.







Watchers where used view values while executing program using step through in visual studio debugging. This allows values be seen while debugger steps through each line code.

Using Debug.Assert in code to communicate as part documentation while debugging application and maintaining it. Asserts communicate the assumptions made in program when assumptions for asserts are met it keeps silent not doing anything. Asserts are really just make sure everything is running on correct parameters and asserts used during development phase make sure that no changes within code create unexpected errors. When running on debug mode and assertion fails it will breakpoint to assertion.

Why would you use assertions? Well assert is debugging tool allow you make assumptions and then define it within true & false test condition. Also allows you communicate what went wrong and where also. An assertion would generally be used when you expect input be managed, for example user entering password and not allowed enter empty string. If for some reason string was empty you wouldn’t handle this code because external user input validation wouldn’t allow you continue till had valid input. Another example is using Asserts in automated testing; you also able provide assumptions for your automatic testing work within and cover additional areas your test scripts don’t cover or able handle. Since if any fatal error or failure of test will show up within results it allows for invaluable information.

Asserts allow you create your own error messages so best understand what lead to the error in first place. Allows you write up own messages so know exactly where error occurred. So any failures external methods or assumptions needed be made can be handled using Asserts. So implementing interfaces and other libraries you can document them with asserts better communicate also what your program functions assume data being passed to them are.

# Logging for .Net framework

Using System.Diagnostics built in .net 2.0 would provide flexibility and without need implementing external libraries. Additionally most of external library’s re-invent the wheel extending create features for particular solution to program allows for features for logging. Overloading System.Diagnostics framework where you changes suit problem. Using Trace you can add new Trace Listeners for console and files. This write to text file all application, using system.Diagnostics.TextWriterTraceListener can write to text file that contains related trace calls.

Using event logs can keep track of significant changes or events within application Events are logs are used for debugging on users systems. But Event logs can also be used track important changes within Program systems, and allow for important business and legal requirements be logged within application. Usually anything requiring legal attention for system events are handled by domain server, however sometimes changes made by program must also be tracked. Using Event logs allows you track & debug any major changes to system or errors. Events also allow you log who made changes if you incorporated user logins.

Using systems debug you have asserts, Write, Writeline, WriteIf, WriteLineIf all these methods help with debugging your code. Debug write line can help with visualising Debugging without needing rely heavily on breakpoints. You be able write useful information or print out any information or variables when running in debug mode. Using these features will include debug outputs that developer can see outputs in debugger console.

Example what is possible using System.Diagnostics included in zip

# Test Data and test summary results

Scope of this project is within requirements for software given in Mr Sparkly script. Requirements be fulfilled by testing or defects logged and corrected. Any additions to requirements will not be considered for this testing document

This summary will provide testing results to show successes and failure listing defects that where encounter in testing. A checklist be provided for testing requirements, and what is required go ahead with the test.

Requirements out of scope are deployment & maintenance for program. No tests be written for these consideration.

Testing results, results of testing surprising all basic tests regarding functionality on clients end all passed with few minor failures.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test cases planned | Test Cases Executed | TCs Pass | TCs Incomplete | TCs Failed |
| 22 | 18 | 10 | 4 | 4 |

Testing types preformed are Black box testing and white box testing relating some features.   
Automated testing also h ad consideration and regarding security been performed. Using Test method features within .net framework allowed for some testing regarding user login be automated.  
  
Testing for the web pages all done for internet explorer. Other testing platforms have been considered however not documented because they out of scope.

# 

# 

# Test Plan

Test Plan included in Zip File sent as separate word document

# Interface Design layout and the Final Product

Final Product is in Zip File sent you.

# Sign off

Client approves of Technical documentation, reviewed functionality and purpose of program and approves of program be created and hand off to client. Program meets client requirements and currently will not require any further testing and development because product meets client requirements and be handed off for client to use.

Signature: \_\_\_\_\_\_\_\_\_\_\_ date: \_\_\_/\_\_\_/\_\_\_\_\_

# References

Wikipedia (<http://en.wikipedia.org/wiki/Multiple_inheritance#The_diamond_problem>)

<http://www.cs.utah.edu/~germain/PPS/Topics/interfaces.html>

<http://stackoverflow.com/questions/2866987/what-is-the-definition-of-interface-in-object-oriented-programming>

<http://en.wikipedia.org/wiki/Multiple_inheritance#The_diamond_problem>