## INVENTORY MANAGEMENT SYSTEM

#### By

#### AKSHAY AMAN GERA (CE-02)

**A project submitted**

**In**

**partial fulfillment of the requirements for the degree of**

**BACHELOR OF TECHNOLOGY**

**in Computer Engineering**

**Internal Guide External Guide**

*Prof.ANKIT.P.VAISHNAV*

*Assistant Professor*

*Dept. of Comp. Engg*



**Faculty of Technology Department of Computer Engineering**

**Dharmsinh Desai University**

#### March 2019

CERTIFICATE

This is to certify that the project work titled

Inventory Management System

is the bonafide work of

AKSHAY AMAN GERA

CE-02

16CEUON018

carried out in the partial fulfillment of the degree of Bachelor of Technology in Computer Engineering at Dharmsinh Desai University in the academic session

December 2018 to April 2019.

Prof Ankit.P.Vaishnav Dr. C. K. Bhensdadia

Asst. Prof Head,

Dept. of Computer Engg. Dept. of Computer Engg.



**Faculty of Technology Department of Computer Engineering**

**Dharmsinh Desai University**

#### March 2019

**ACKNOWLEDGEMENTS**

I would like to take this opportunity to thank Professor Ankit.P.Vaishnav for his invaluable guidance and support, because of which this project was possible . I would also like to thank the professors of this subject for providing this opportunity to work on a concept of utmost importance such as this and for imparting all the necessary knowledge required to make this project possible

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| Chapter |  | Page |
| 1. | Introduction | 1 |
| 2. | About the System | 3 |
| 3. | Analysis | 10 |
| 4. | Design | 12 |
| 5. | Implementation | 14 |
| 6. | Testing and Screenshots | 16 |
| 7. | Conclusion | 22 |
|  | Bibliography |  |

**List of Figures**

**Figure 1:** Use Case Diagram 11

**Figure 2:** Entity Relationship Diagram 11

**Figure 3:** Home Page 18

**Figure 4:** Home Page Search by name 18

**Figure 5:** Home Page Filter by Category 18

**Figure 6:** Home Page Update 19

**Figure 7:** Add Product Successfully 19

**Figure 8:** Add Product page record exists message 19

**Figure 9:** Add Product page required field message 20

**Figure 10:** Home page JAVA 20

**Figure 11:** Add Product Required Field JAVA 20

**Figure 12:** Update Product Required Field JAVA 21

**Figure 13:** Delete Product Required Field JAVA 21

**Chapter 1**

Introduction

**1.1 Introduction**: This introduction to the Report provides an overview of the entire Report by providing purpose, scope of project, references, meaning of special terms used in this document, the aim of this document is to give an in-depth insight into the Inventory Management System.

**1.2 Purpose**: The purpose of this document is to present a detailed description of the Inventory Management System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will work according to input. This document is intended for the developers of the system and for any future development purposes and it is also intended for any examining authority who will need every possible detail about this system to understand it fully. This document demonstrates the working of the system through workflow and modelling diagrams and also includes screenshots from the actual working system to further help the reader visualize how the system works.

**1.3 Overview**: The following sections involve details of how the system works functionally and what are its standard guidelines and operating expectations. Also the remaining sections of this document provide a general description, including characteristics of the users of this project, the product's hardware, and the functional and data requirements of the product.

**Chapter 2**

About The System

**2.1** Windows Communication Foundation (WCF) is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another. A service endpoint can be part of a continuously available service hosted by IIS, or it can be a service hosted in an application. An endpoint can be a client of a service that requests data from a service endpoint. The messages can be as simple as a single character or word sent as XML, or as complex as a stream of binary data.

WCF provides features such as service orientation, Interoperability, Security etc which makes it a very important framework for building services.

The SRS document(System requirement specification) is as shown next:

**2.2 Introduction**

#### Purpose

The purpose of this document is to present a detailed description of the requirements of Inventory Management System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate. This software requirements specification provides a complete description of all the function and specifications of system.

#### Document Conventions

All headings in this document are in bold. And the font used in this document is Times New Roman.

#### Intended Audience and Reading Suggestions

Developers working on this system are advised to read this Document.

#### Product Scope

The Inventory Management System is a CRUD based data management system designed using Windows Communication Foundation Services where the CRUD operations are itself services which can be used by various clients to perform operations on the database. The primary data structure of this system is product which consists of ID, Name, Price, Stock, and Category. This Database is on the server side. Clients can be on any platform which support WCF services, for the purposes of this project C# ASP.NET and JAVA’s Servlets were used as two different clients consuming the Service

#### References

* + - 1. IEEE Std 830-1998: IEEE Recommended Practice for Software Requirement Specifications
      2. Software Engineering : A practitioner's Approach by Roger S Pressman

### 

### **Overall description**

#### Product perspective

This System aims to provide CRUD based operations on a Relational DBMS, an Inventory in this case which manages the current products it has

.

#### Design and implementation constraints

Frontend is implemented using HTML and CSS . Backend is implemented in ASP.NET using C# and Servlets in JAVA for the two different clients. The WCF services are designed using C#

#### Operating Environment

The system works on all internet browsers flawlessly.

#### 

### **Functional Requirements**

### Get Products

### Input: User clicks on Refresh

### Output: The current inventory is displayed

### Process: Service to get all is called by the client

### 

### 2.4.2Get Products by Category

### Input: User selects category

### Output: Products are displayed based on Category

### Process: Service to get products by category is called by the client

### 2.4.3 Add Product

### Input: User inputs Product details

### Output: Success Message | Record already exists message

### Process: Add Product service is called and data is added to the Db

### Update Product

### Input: User inputs id and new details

### Output: Success Message

### Process: Update services are called and the data is then updated in the Db

### Delete Product

### Input: User inputs Product Id or clicks on delete button

### Output: Data view refreshed

### Process: Delete Product service is called

### **Nonfunctional requirements**

#### Performance Requirements

* Any page of the application should not take more than 6 seconds to load on a DSL broadband connection.
* The system may be throttled or slowed down on heavy loads to ensure service for everybody. By throttling is meant that certain functionality may be unavailable during heavy server load.
* The application should be able to support many concurrent users without any performance degradation and in future, installing additional hardware components.

#### Reliability/Availability Requirements

* The system has to be online 24 hours a day, 7 days a week. There is no place for an extended downtime.
* The Mean Time Between Failure (if any) should not be less than 3 months.
* In case of a failure that leads to a system outrage the Mean Time to Repair should not be more than 2 hours.

#### Usability

* The user interface of the system should be very user friendly**.**

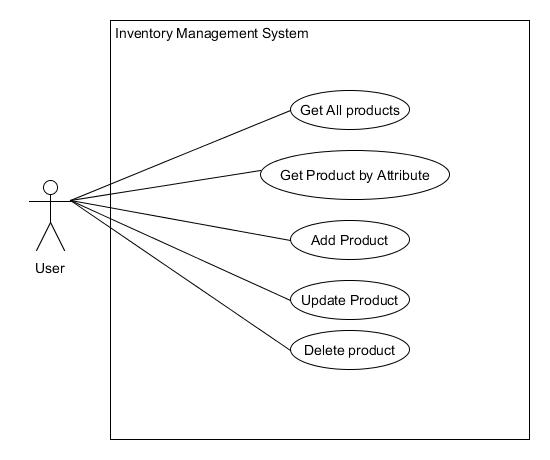
#### Design Constraints

* Html and css are to be used to design the front end
* ASP.NET using C# and JAVA should be used for the backend
* RDBMS is to be used

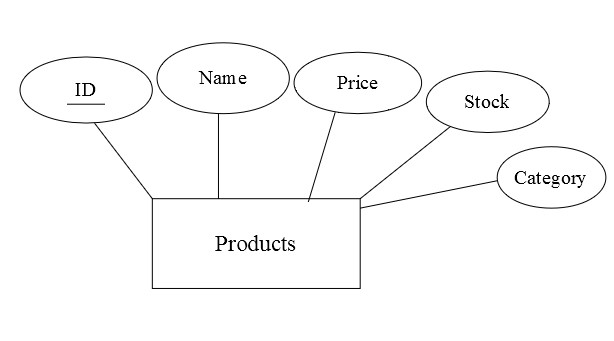
**Chapter 3**

Analysis

**Figure 1:** Use Case Diagram



**Figure 2:** Entity Relationship Diagram



**Chapter 4**

Design

### **4.1 Database**

The database in this system is the InventoryDB, The table used is Products table which has the following attributes

1. ID-Primary key for this table
2. Name-Name of the product
3. Price-The price of the particular product
4. Stock-No. of Units of the product in the inventory
5. Category-The category of the product

### **4.2 Front End Interface**

The Front End Interface is built on two platforms,

1. ASP.NET using C# - The UI is User friendly and consists of two pages. The first page Displays all products and allows for update and deletion, it also allows users to view products of a particular category
2. JAVA – The UI consists of four pages, The Primary Page is the Display Products page and the rest of the Pages are for Addition, Updation and Deletion of a product.

### **4.3 Validations**

Validations performed are for required fields during addition of a product or deletion and to check if the price and stock of product is not going below zero.

## Chapter 5 Implementation



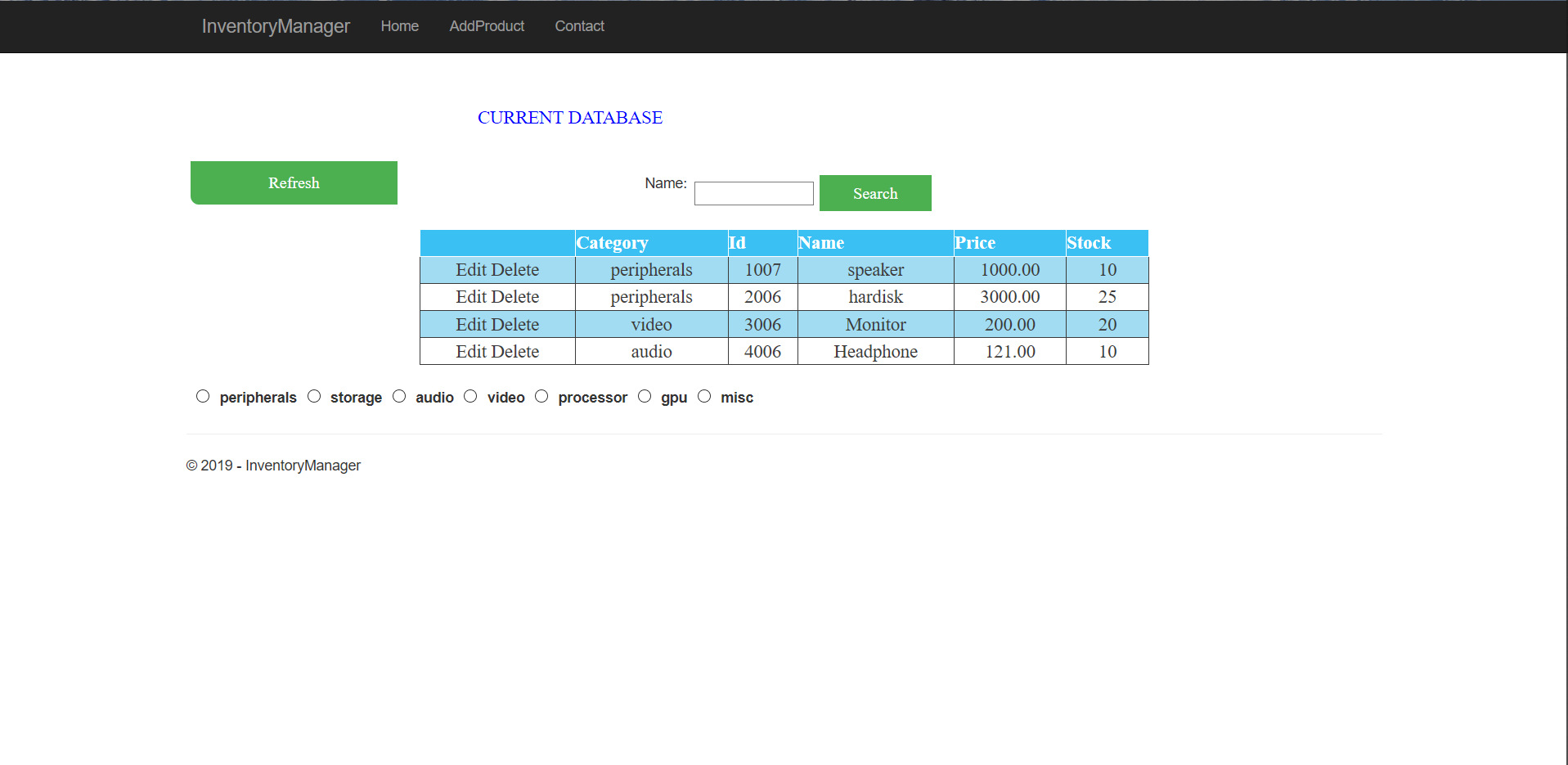
**5.1 Operations in WCF Service library**

1. GetProducts : Fetches and returns the list of all products currently in the inventory database.
2. GetProductWithName : Fetches and returns the product with the input name.
3. GetProductWithID : Fetches and returns the product with the input ID.
4. GetProductsOfCategory : Fetches and returns the list of all products in a particular Category.
5. AddProduct : Adds a new product to the database using user input.
6. UpdateName : Updates the name of the product
7. UpdatePrice : Updates the price of the product
8. Update Stock : Updates the Stock of the product
9. DeleteProduct : Deletes a product from the database using ID

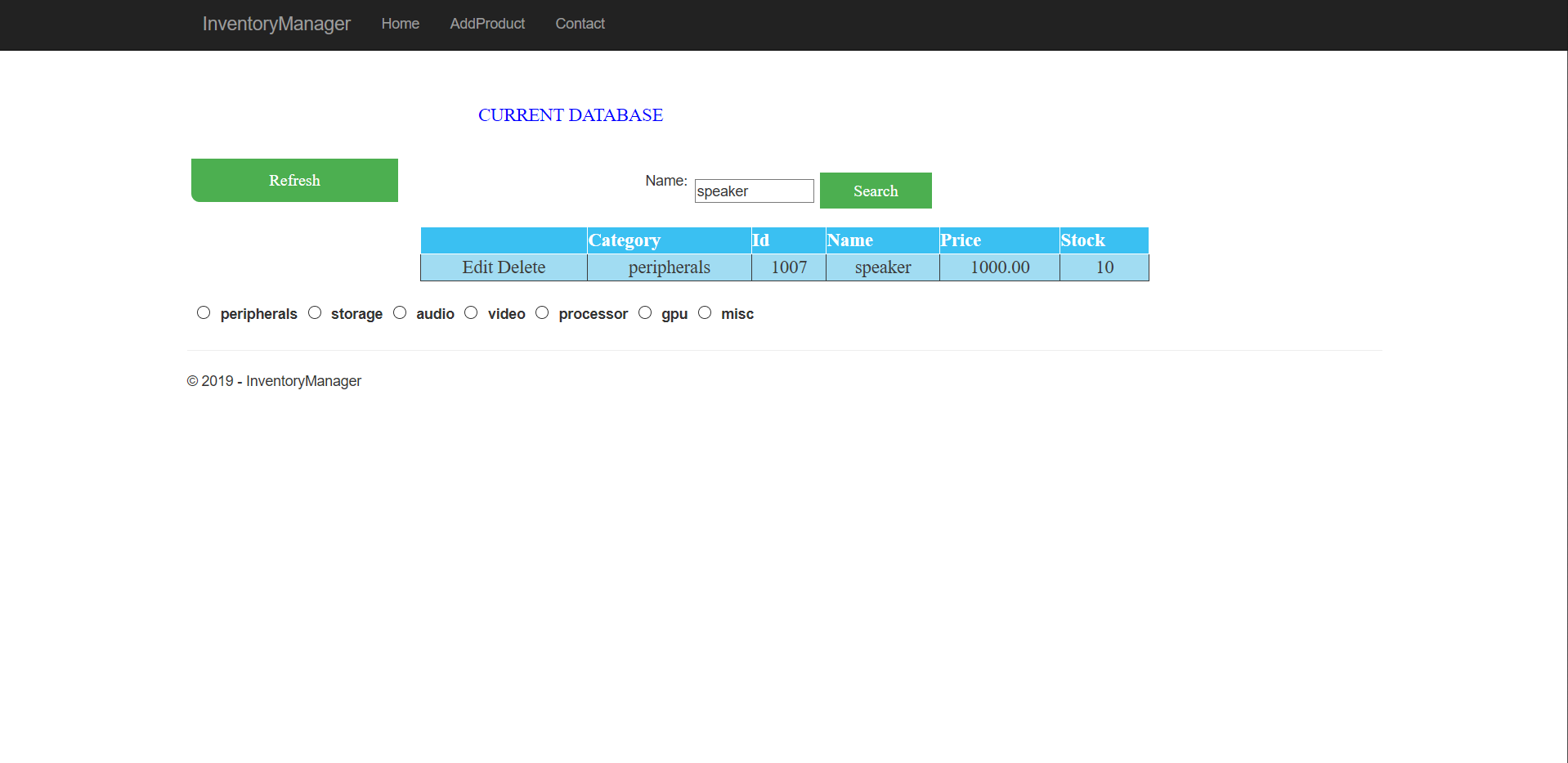
**Chapter 6**

**Testing and Screenshots**

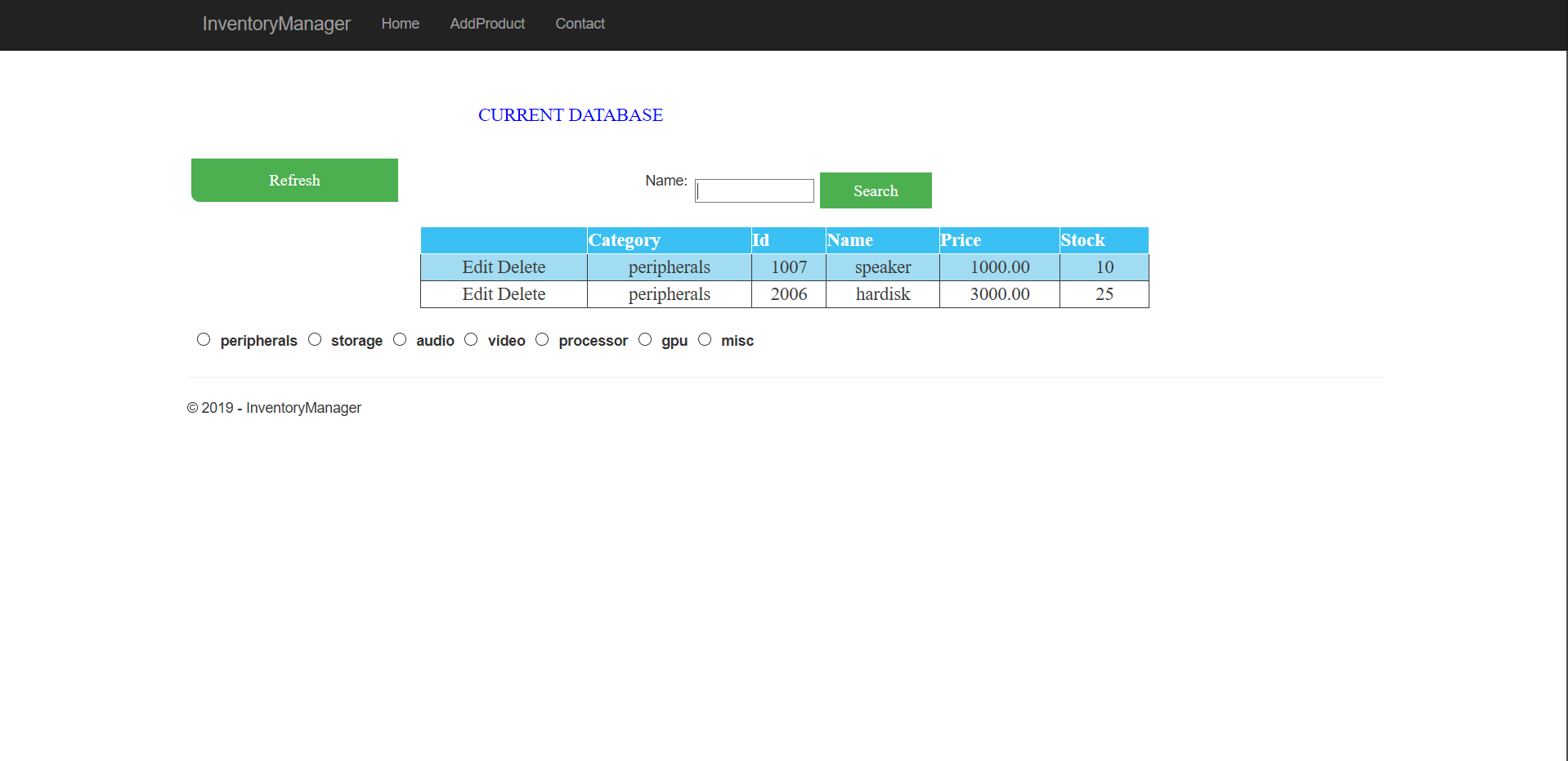
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test case ID | Test Scenario | Test Steps | Tests Data | Expected Result(s) | Pass or Fail |
| T01 | Input product name(C# client) | 1. Enter Correct product name | Name of product | Data if product exists else empty row | Pass |
| T02 | Select Category  (C# client) | 1. Click Radio Button | Category | Products of specific category | Pass |
| T03 | Edit and Update(C# client) | 1. Click Edit 2. Update Row | Updation Data | Row Updated | Pass |
| T04 | Add Product(C# client) | 1. Enter Product Details and click submit | Product Details | Success Message | Pass |
| T05 | Add Product of same name(C# client) | 1. Enter Product of same name | Product Details with same name | Record Exists Message | Pass |
| T06 | Submitting without input in add product (Both Clients) | 1. Click Submit | Empty fields | Required field message | Pass |
| T07 | Submitting without input in update product(Java client) | 1. Click Submit | Empty Fields | Required field message | Pass |
| T08 | Submitting without input in delete page | 1. Click submit | Empty Fields | Required field message | Pass |



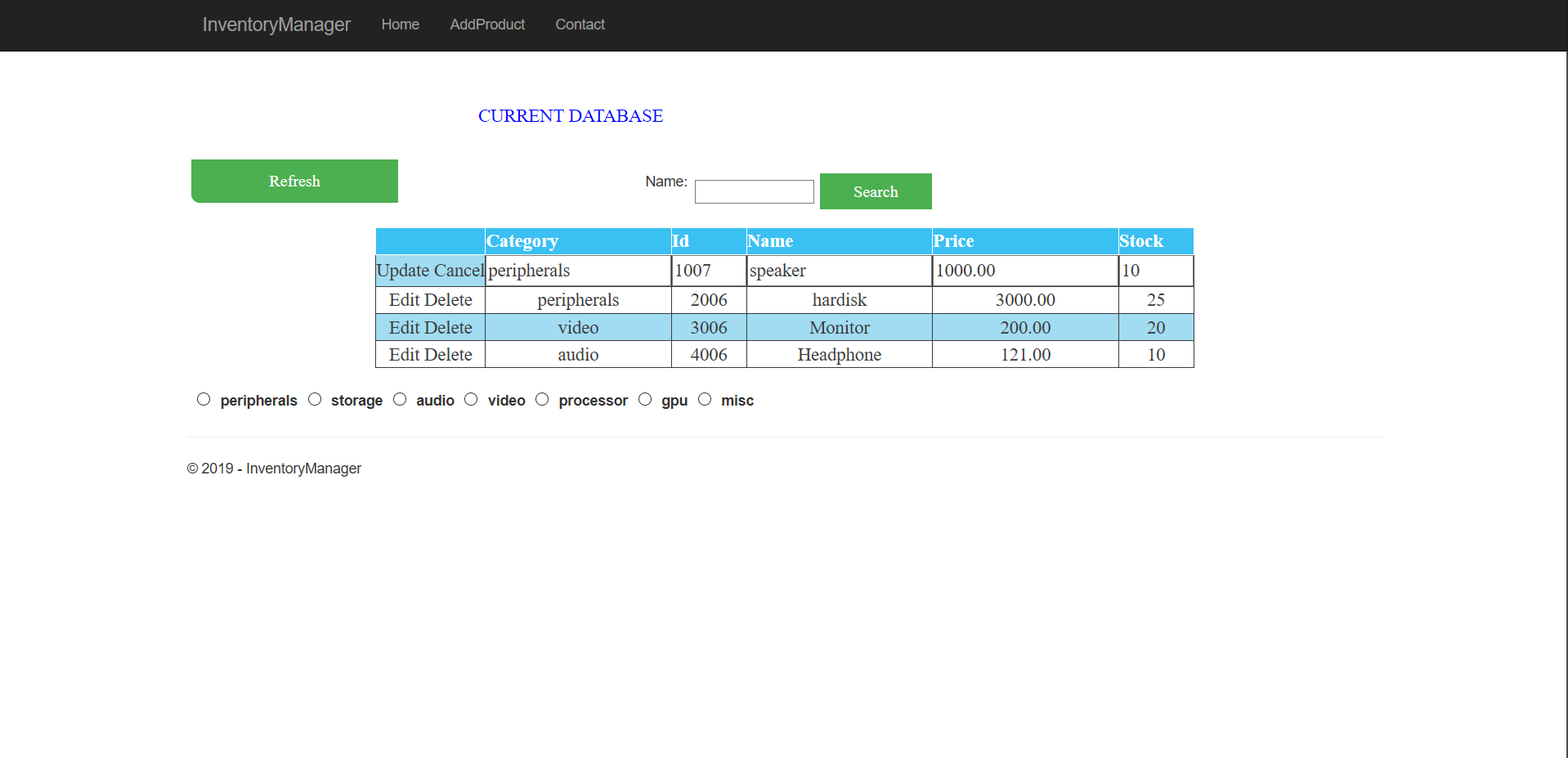
**Figure 3:** Home Page



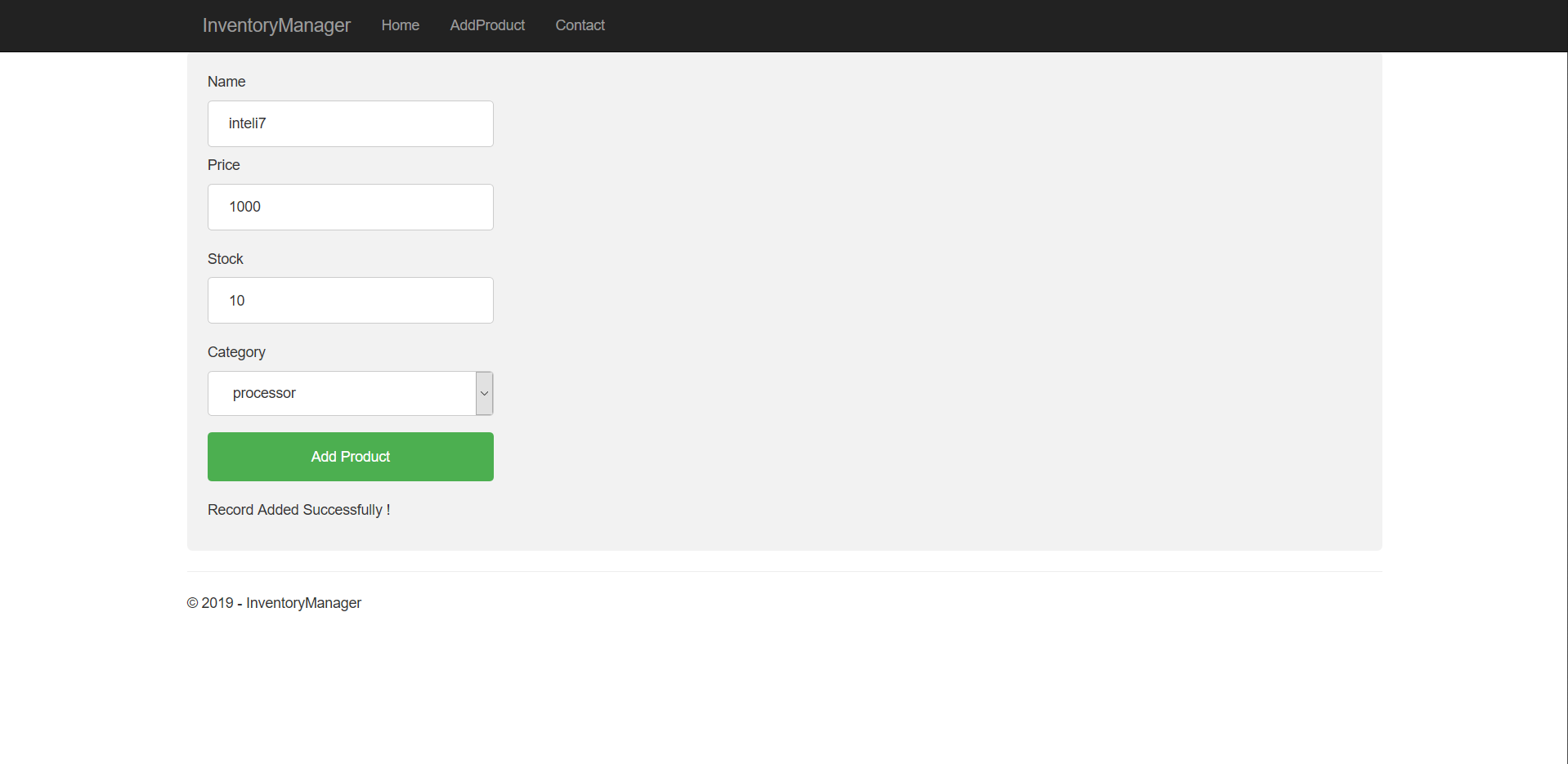
**Figure 4:** Home Page Search by name



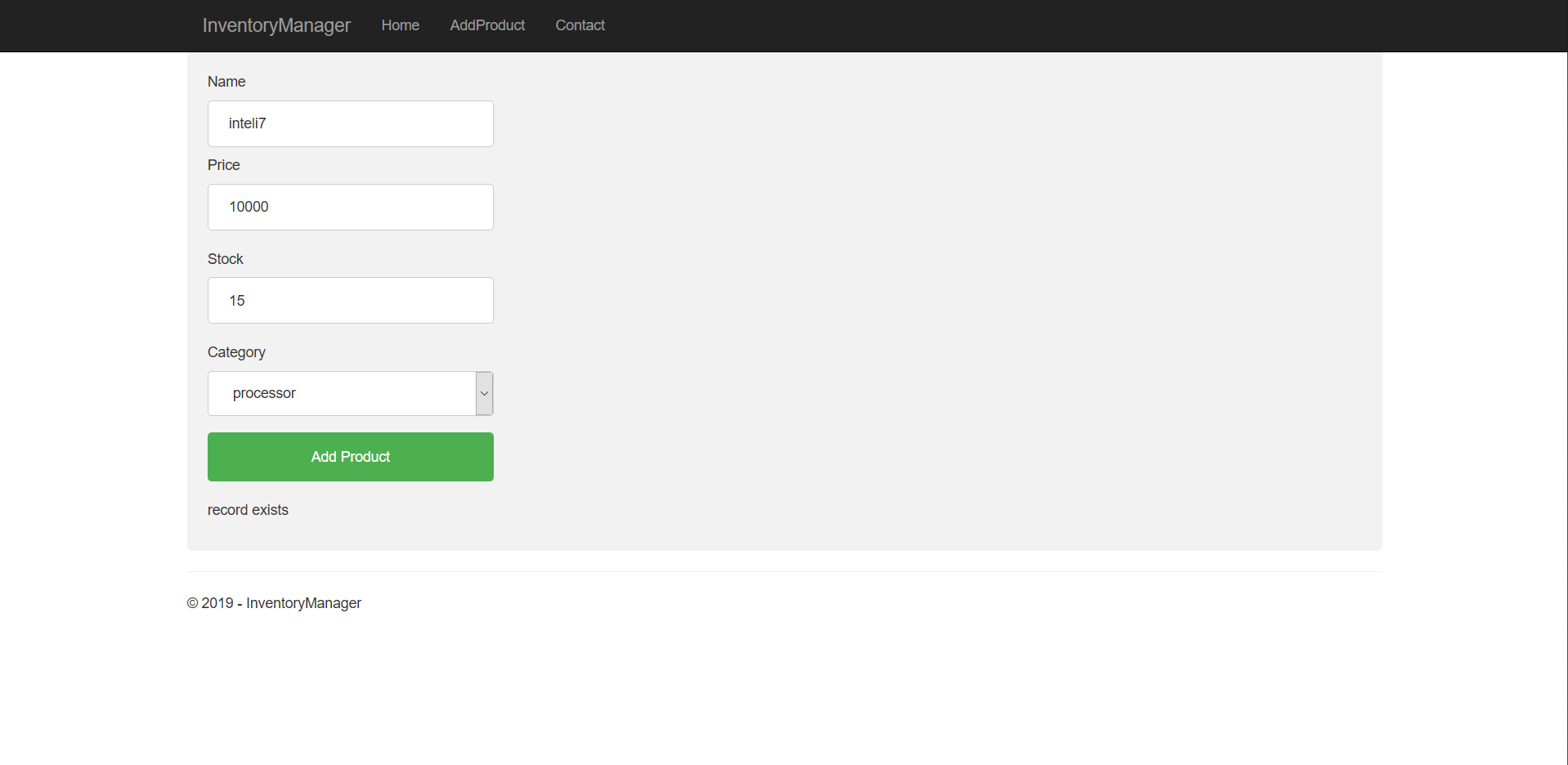
**Figure 5:** Home Page Filter by Category



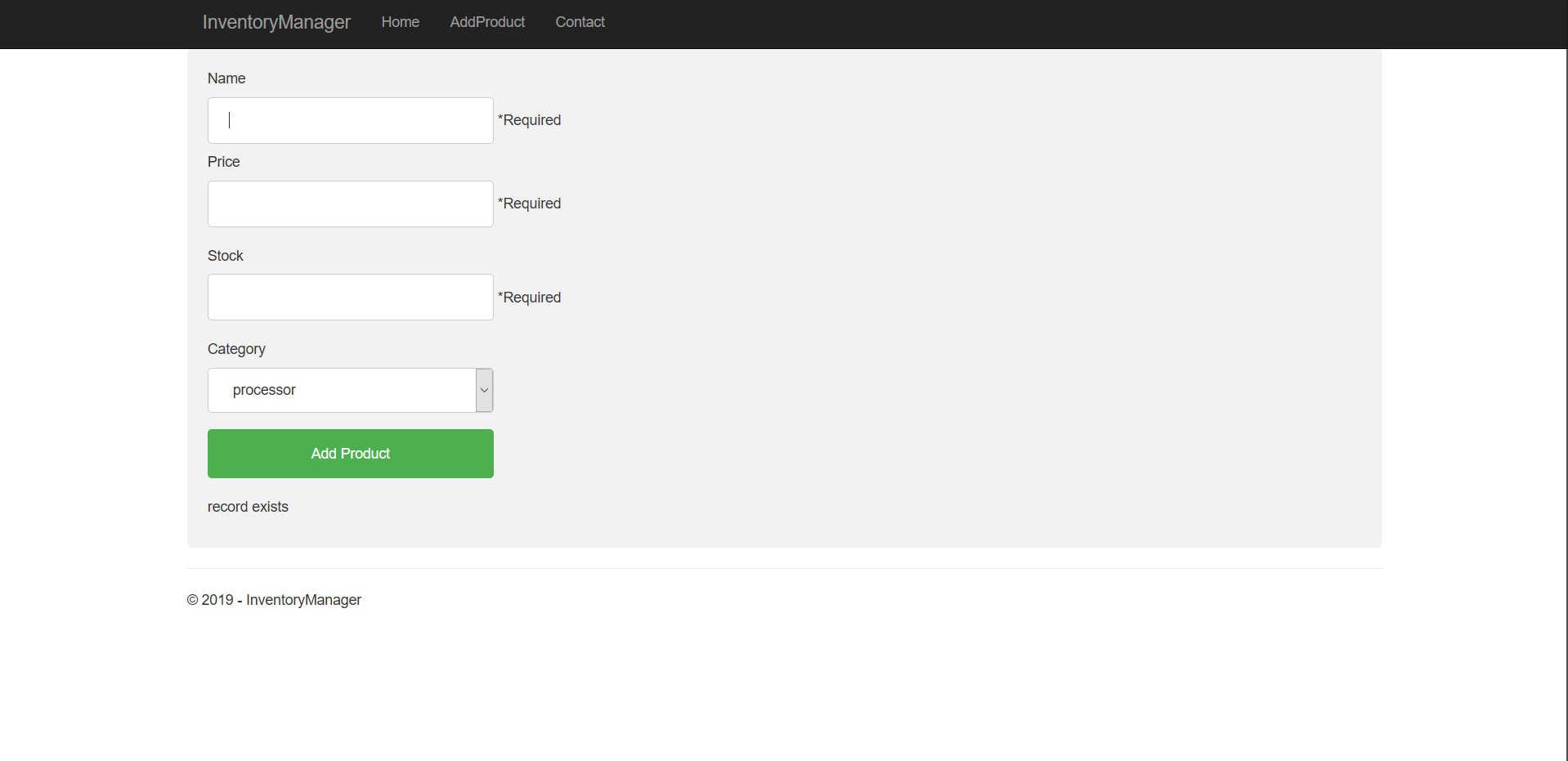
**Figure 6:** Home Page Update



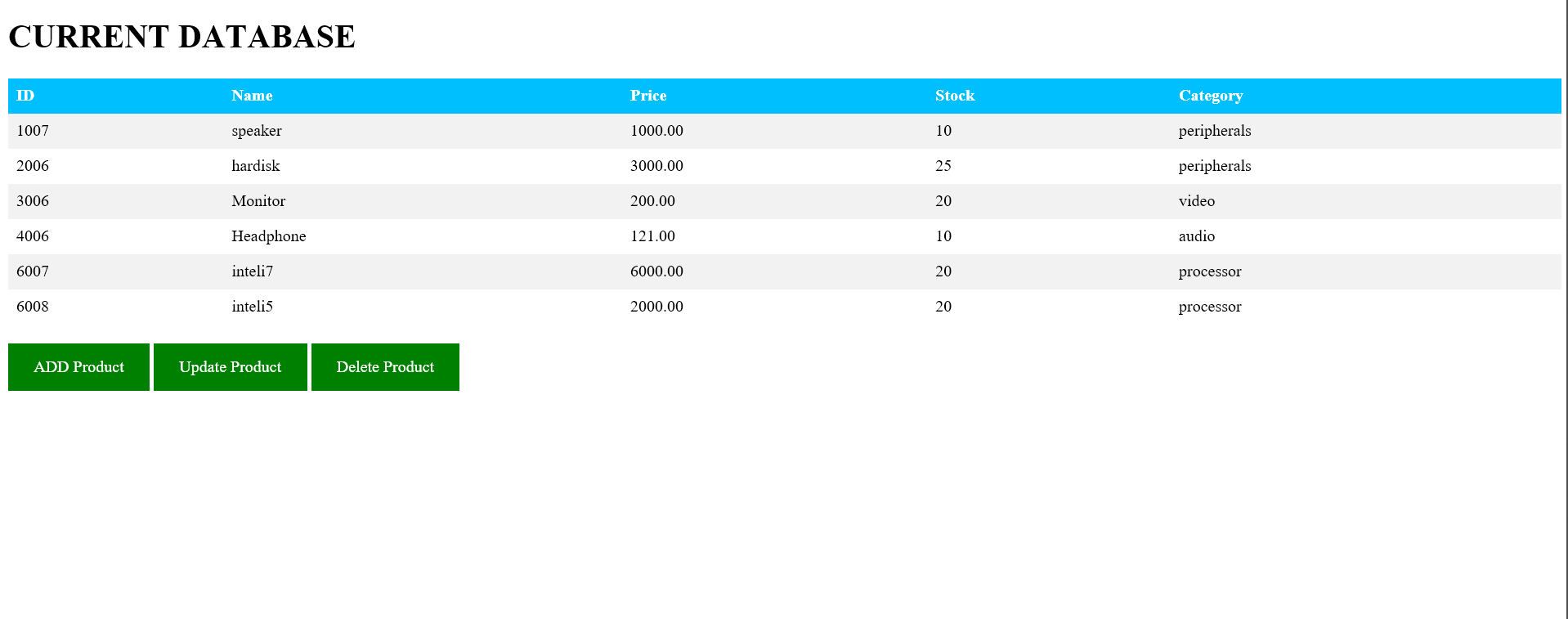
**Figure 7:** Add Product Successfully



**Figure 8:** Add Product page record exists message



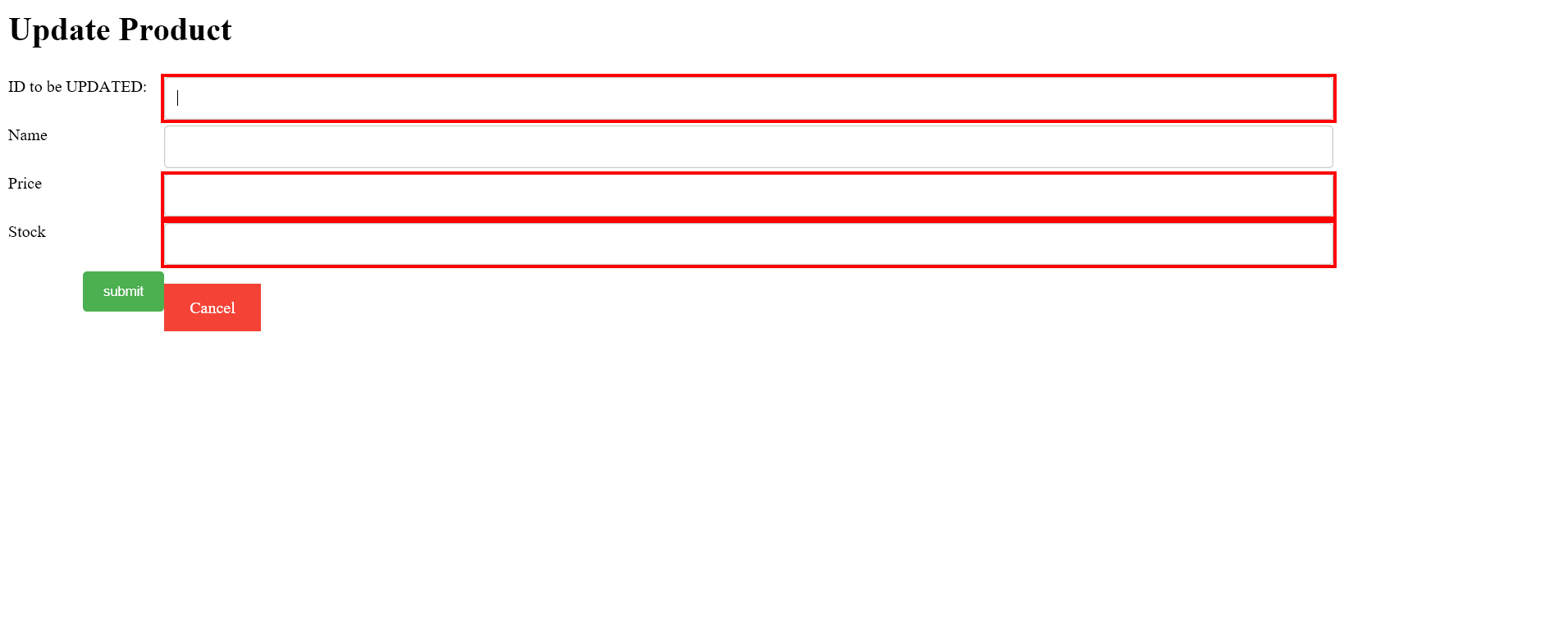
**Figure 9:** Add Product page required field message



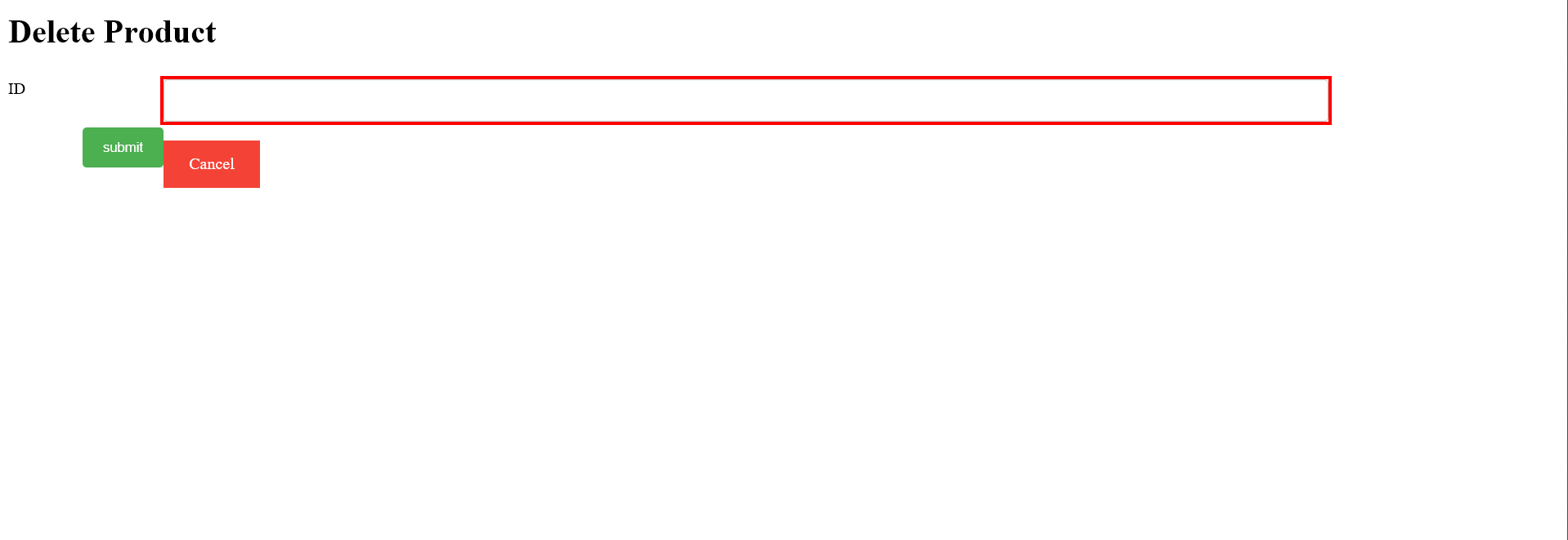
**Figure 10:** Home Page JAVA



**Figure 11:** Add Product Required Field JAVA



**Figure 12:** Update product required field JAVA



**Figure 13:** Delete Product required field JAVA

**Chapter 7**

**Conclusion**

Microsoft’s Windows Communication Foundation framework designed to build service libraries is a .NET based framework as it provides all the necessary tools to build a Service-oriented architecture application. In this project we see that the services created using WCF are being consumed by two clients operating seamlessly on different platforms proving its capabilities of interoperability and reusability which are standards every SOA application must follow. WCF uses the concept of contract to hide its implementation as seen in the inventory interface using a file called the WSDL which serves as Semantic and Syntactic data.

Thus we can conclude that WCF is a very important framework when building enterprise level services since it offers most of the standards required by a purely service-oriented architecture based application. Using this project’s CRUD based operations (Inventory DB), the use of WCF services and how they are consumed by a client is demonstrated and it is seen that WCF offers a lot of control on how the service should be designed while also providing means to achieve standards of Service Orientation. Visual Studio also serves as an important IDE since it provides auto-generated code to achieve some trivial functionality.

**Bibliography**

1. https://docs.microsoft.com/en-us/dotnet/framework/wcf/
2. <https://www.c-sharpcorner.com/>
3. www.youtube.com