**TBPB Shop**

Team: Popescu Radu Andrei, Balosu Mihai,

Bacioiu Alexandru, Toma Alexandru

Lab Section: Software Engineering

Date: 27.03.2020

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **1.** | **I**[**NTRODUCTION**](#page4) | **2** |
| 1.[1](#page4) | [Purpose](#page4) | 2 |
| 1.[2](#page4) | [Scope](#page4) | 2 |
| 1.3 | Overview | 2 |
| 1.4 | Reference Material | 2 |
| 1.[5](#page4) | [Definitions and Acronyms](#page4) | 2 |
| **2.** | **S**[**YSTEM**](#page4) **O**[**VERVIEW**](#page4) | **2** |
| **3.** | **S**[**YSTEM**](#page4) **A**[**RCHITECTURE**](#page4) | **2** |
| [3.1](#page4) | [Architectural Design](#page4) | 2 |
| [3.2](#page5) | [Decomposition Description](#page5) | 3 |
| [3.3](#page5) | [Design Rationale](#page5) | 3 |
| **4.** | **D**[**ATA**](#page5) **D**[**ESIGN**](#page5) | **3** |
| [4.1](#page5) | [Data Description](#page5) | 3 |
| [4.2](#page5) | [Data Dictionary](#page5) | 3 |
| **5.** | **C**[**OMPONENT**](#page5) **D**[**ESIGN**](#page5) | **3** |
| **6.** | **H**[**UMAN**](#page6) **I**[**NTERFACE**](#page6) **D**[**ESIGN**](#page6) | **4** |
| 6.1 | Overview of User Interface | 4 |
| [6.2](#page6) | [Screen Images](#page6) | [4](#page6) |
| [6.3](#page6) | [Screen Objects and Actions](#page6) | [4](#page6) |
| **7.** | [**REQUIREMENTSMATRIX**](#page6) | **2** |

1. **INTRODUCTION**

**1.1 Purpose**

The purpose of this software design document is to describe the implementation, architecture and system design of the ecommerce TBPBShop website presented in the IEE Software Requirements Specifications document. This website is designed to buy/sell product of any kind, there are two types of users, admin users who can add new products to the eshop and client users that can only buy the products that are present in the eshop.

Audiences generally vary, however, the targeted audience consists of people who want to buy any type of products and people who want to sell their products.

**1.2 Scope**

This document describes the implementation details of the TBPBShop website. The scope of the website is to bring the classic shop to everyone without the necessity of going to a specific place to buy some products. Within this application, we want to make a friendly environment for the customers, where they can easily access any of the application features and make their “stay” as comfortable as we can.

Objectives relevant to this website include helping the customer with a friendly interface so that they can easily buy products, and also making sure the administrator has full control of the database in order to make sure that the website functions without problems and that the customers are satisfied, and also administrator users being the only users that can add new products to the shop.

Finally, the goal of website is to encourage more people to choose online shopping because it is simple, fast and more time saving. Maintaining a simple and friendly site is deemed of highest importance in order to hold customer attention.

**1.3 Overview**.

This document contains several representations of software components, interfaces and data necessary for implementation phase. It will show how the software system will be structured to satisfy the requirements. The IEEE Recommended Practice for Software Design Descriptions have been reduced in order to simplify this assignment while still retaining the main components and providing a general idea of a project definition report.

**1.4 Reference Material**

https://www.lucidchart.com/

https://www.toptal.com/freelance/why-design-documents-matter

https://en.wikipedia.org/wiki/Software\_design\_description

**1.5 Definitions and Acronyms**

|  |  |  |
| --- | --- | --- |
| Nr. Crt. | Acronym | Description |
| 1 | HTML | Hyper Text Markup Language |
| 2 | HTTP | Hyper Text Transfer Protocol |
| 3 | IIS | Internet Information Service |
| 4 | SSMS | SQL Server Management Studio |
| 5 | CSS | Cascading Style Sheets |
| 6 | GUI | Graphical User Interface |

|  |  |  |
| --- | --- | --- |
| Nr. Crt. | Name | Description |
| 1 | Authentication | The procedure (essentially approval) used by the approval authority in verifying that specification content is acceptable. Authentication does not imply acceptance or responsibility for the specified item to perform successfully. |
| 2 | Client | (1) A computer process that requests a service from another computer and accepts the server's responses; (2) the individual computers in a network computing system |
| 3 | Database | A collection of related data stored in one or more computerized files in a manner that can be accessed by users or computer programs via a database management system. |
| 4 | Database management system | An integrated set of computer programs that provide the capabilities needed to establish, modify, make available, and maintain the integrity of a database. |
| 5 | Functional requirement | A statement of a piece of required functionality or a behavior that a system will exhibit under specific conditions. These include inputs, outputs, calculations, external interfaces, communications, and special management information needs. Functional requirements are also called behavioral requirements because they address what the system does. |
| 6 | JavaScript | A programming language designed by Sun Microsystems, in conjunction with Netscape, that can be integrated into standard HTML pages. |
| 7 | SQL | (pronounced "es-que-el") stands for Structured Query Language. SQL is used to communicate with a database. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. However, the standard SQL commands such as "Select", "Insert", "Update", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database. This tutorial will provide you with the instruction on the basics of each of these commands as well as allow you to put them to practice using the SQL Interpreter. |

**2. SYSTEM OVERVIEW**

In this application implementation are used the following: C# ASP.NET, JavaScript, CSS, SQL and HTML. SQL represents the database where all the information about the products, users, roles are kept. HTML, CSS and JavaScript are used for the frontend design, to implement the user interface. In C# ASP>NET is written the code for the server side, where is implemented every functionality of the web application.

There will be more roles, both normal customers (clients) and administrators. Each client has the opportunity to register and log in. The log in can be made either using the existing shop account, which will be created using the register page, or the Facebook account.

There is also a section dedicated to administrators, who can manage the products, including all the attributes that we will set. There are also several product categories.

When logged in, each customer will be entitled to a shopping cart, where they can add any of the existing products in the store. In the shopping cart, he can perform operations such as increasing or decreasing the quantity of a specific product, or to definitely delete that product from the list. The price and quantities will be updated, with the addition or deletion of a product from the shopping list.

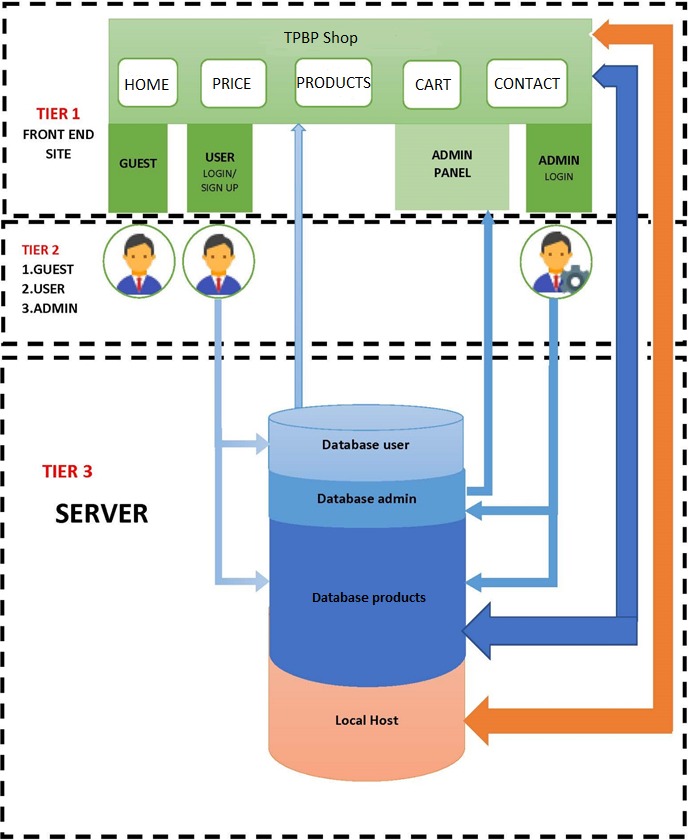
All accounts, all products and all shopping carts, along with the products that they contain, will be stored in the database.

Each operation performed either by the customer, or the administrator, will be automatically updated in the database.

The application is designed to be compatible with modern browsers such as Google Chrome, Firefox, Safari, IE etc. It is easy to use for its customers and it is also intuitive.

**3. SYSTEM ARCHITECTURAL**

**3.1 Architectural Design**



**Relations between**

**Tier 1 – Tier 2**

a. Guests can only visualize the TBPB Shop website without having access to other User-only options such as buying products, customizing profile and making orders.

b. User

- can login through the interface;

- can visualize the TPBP Shop Website;

- is able to buy products, customize their profile, make orders, view his cart;

c. Administrator

- can login through the interface using special admin login button;

- can visualize the TPBP Shop Website;

- can manage many things through the admin panel, such as adding a product, manage orders, etc.;

**Tier 2 – Tier 3**

a. Guest has no relation here. It would be possible to implement a special section in the database for newsletter.

b. User

-the Login interface sends a query to the database that’s on the server;

- as the user performs command such as orders, it gets added to the database;

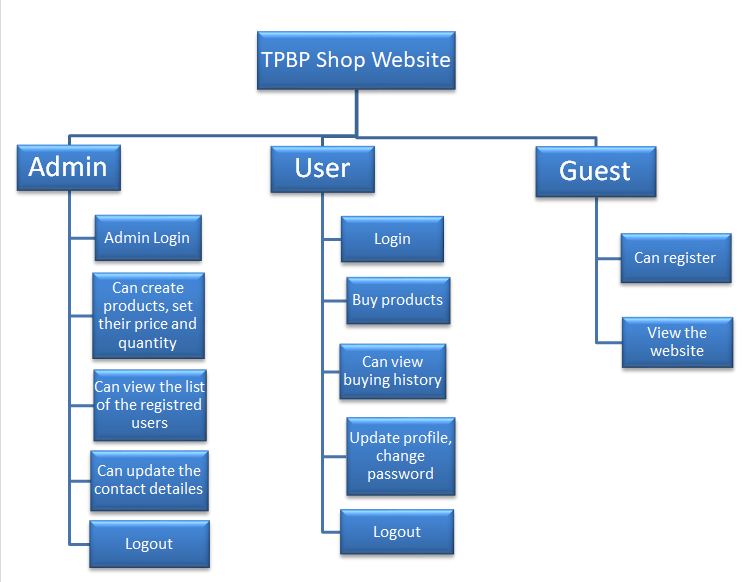
c. Administrator

- admin login interface sends a query to the database;

- whenever the administrator adds new products or changes information about users, these changes get added to the database;

**Tier 1 – Tier 3**

* the data introduced by the user and administrator from the server change the website’s appearance;
* the back-end of this website is located in the local host;
  1. **Decomposition Description**



**3.3 Design Rationale**

The approach for this online shop website architecture has been selected because it is probably the most common; it is usually built around the database, and many applications in business naturally lend themselves to storing information in tables.

The architecture is arranged so the data enters the top layer and works its way down each layer until it reaches the bottom, which is usually a database. Along the way, each layer has a specific task, like checking the data for consistency or reformatting the values to keep them consistent. It is the standard software development approach offered by most of the popular web frameworks and it is a layered architecture. Just above the database is the model layer, which often contains business logic and information about the types of data in the database. At the top is the view layer, which is often CSS, JavaScript, and HTML with dynamic embedded code. In the middle, you have the controller, which has various rules and methods for transforming the data moving between the view and the model.

The advantage of a layered architecture is the separation of concerns, which means that each layer can focus solely on its role. This makes it:

* Maintainable
* Testable
* Easy to assign separate "roles"
* Easy to update and enhance layers separately

Proper layered architectures will have isolated layers that aren’t affected by certain changes in other layers, allowing for easier refactoring. This architecture can also contain additional open layers, like a service layer, that can be used to access shared services only in the business layer but also get bypassed for speed.

**Best for:**

* New applications that need to be built quickly
* Enterprise or business applications that need to mirror traditional IT departments and processes
* Teams with inexperienced developers who don’t understand other architectures yet
* Applications requiring strict maintainability and testability standards

**4. DATA DESIGN**

**4.1 Data Description**

In our code files all the database tables are represented by models. These models are afterwards transformed into tables by a framework. We will use as data types: integer lists, integer variables, integer arrays, string lists and arrays, string variables. This will help us implement the functionalities for the queries, for example: log in, saving the customers’ cart and any other operations. The data structures are to be used in such ways as following tables at 4.2.

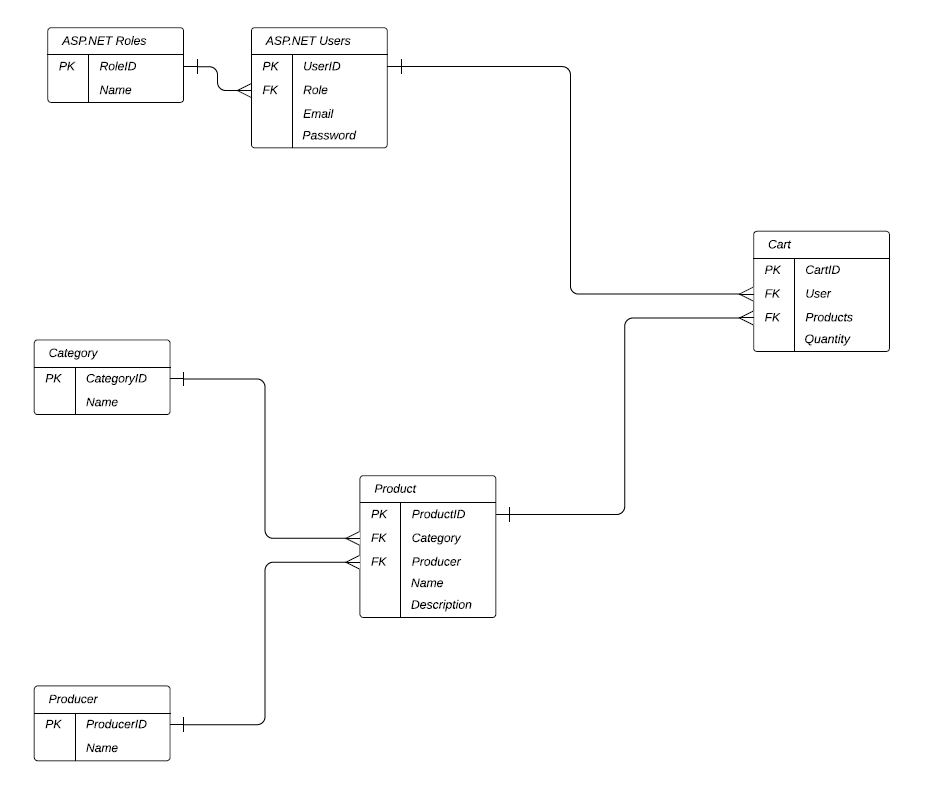
**4.2 Data Dictionary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table for**  **User** | **Column Name** | **Data Type** | **Allow Nulls** |
| ID | Nvarchar(50) |  |
| Email | Nvarchar(50) | X |
| Password | Nvarchar(50) | X |
| Quantity | Int | X |
| Total Price | Decimal(18,2) | X |
| Role | String |  |

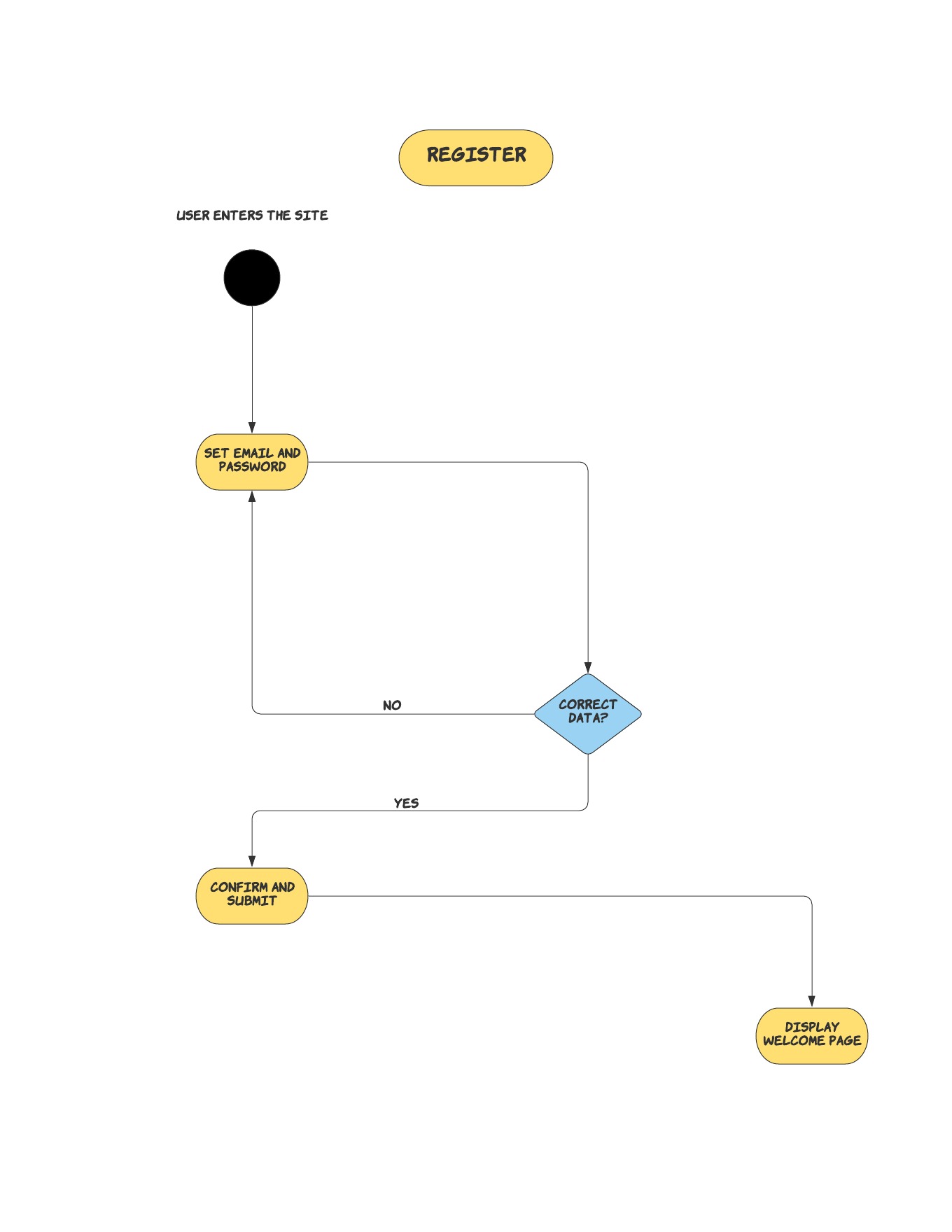
|  |  |  |  |
| --- | --- | --- | --- |
| **Table for**  **Roles** | **Column Name** | **Data Type** | **Allow Nulls** |
| ID | Nvarchar(50) |  |
| Name | String | X |

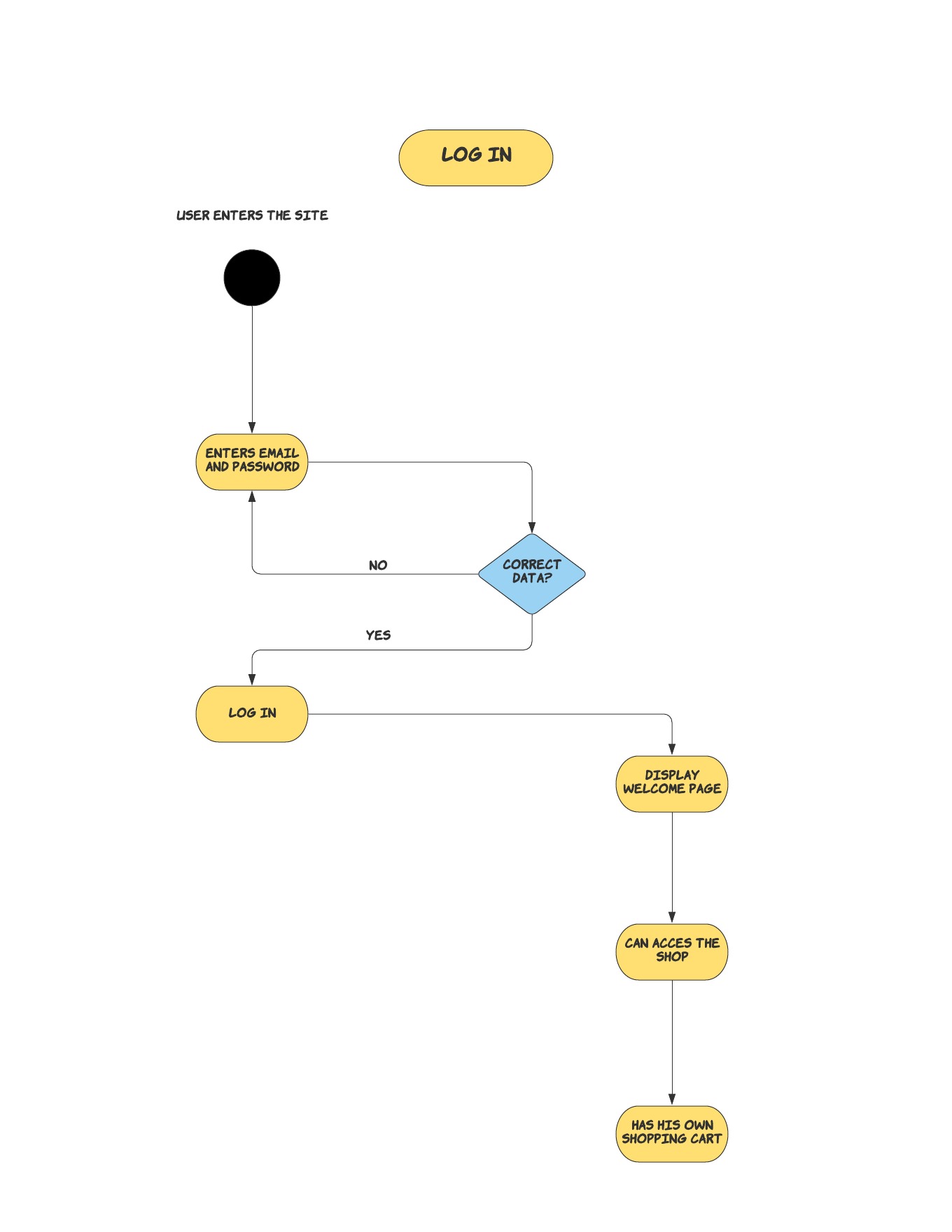
|  |  |  |  |
| --- | --- | --- | --- |
| **Table for**  **Products** | **Column Name** | **Data Type** | **Allow Nulls** |
| ID | Nvarchar(50) |  |
| Name | Nvarchar(50) | X |
| Price | Decimal |  |
| Category | String |  |
| Producer | String |  |

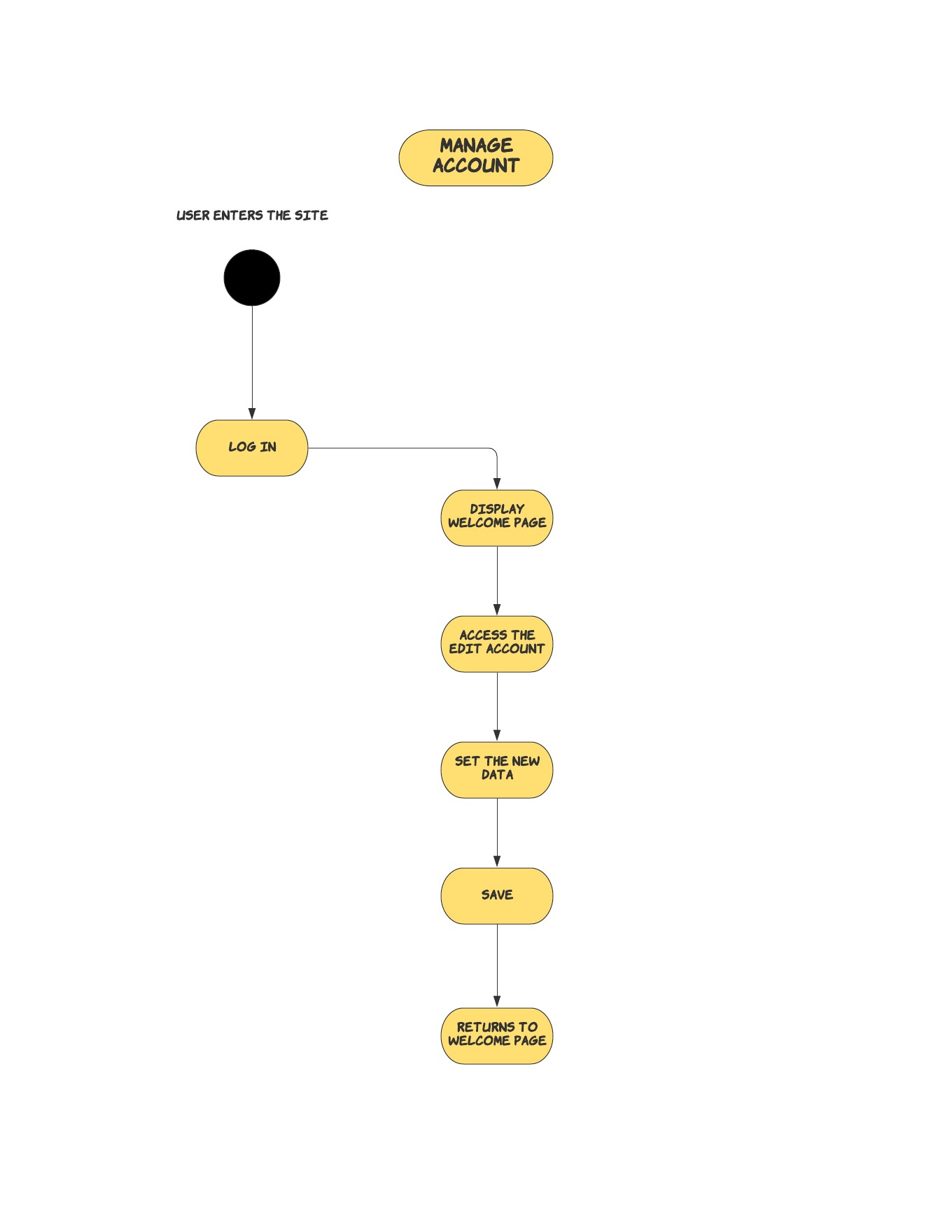
|  |  |  |  |
| --- | --- | --- | --- |
| **Table for**  **Cart** | **Column Name** | **Data Type** | **Allow Nulls** |
| ID | Nvarchar(50) |  |
| User ID | Nvarchar(50) | X |
| Total Price | Decimal |  |
| Quantity | Int |  |

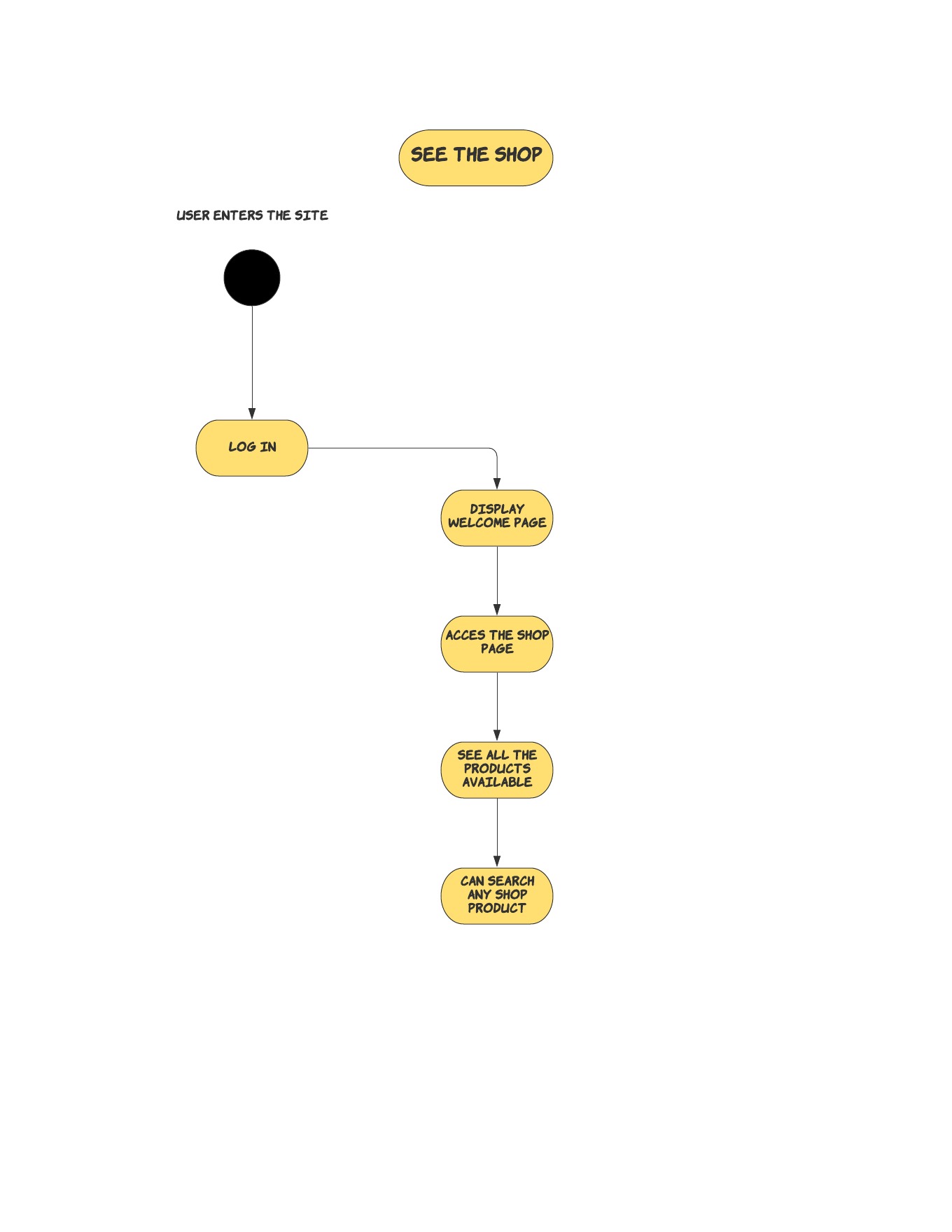
****

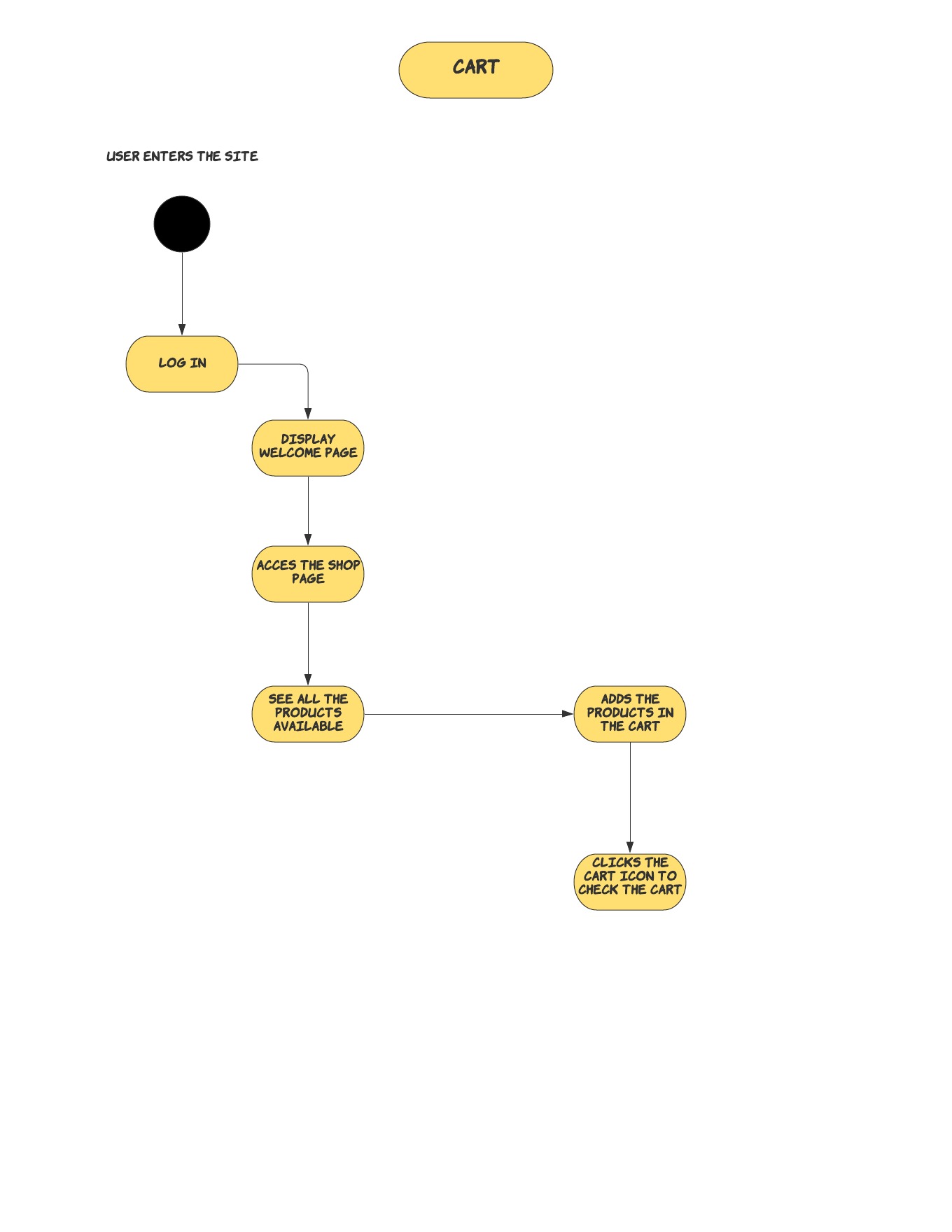
**5. COMPONENT DESIGN**

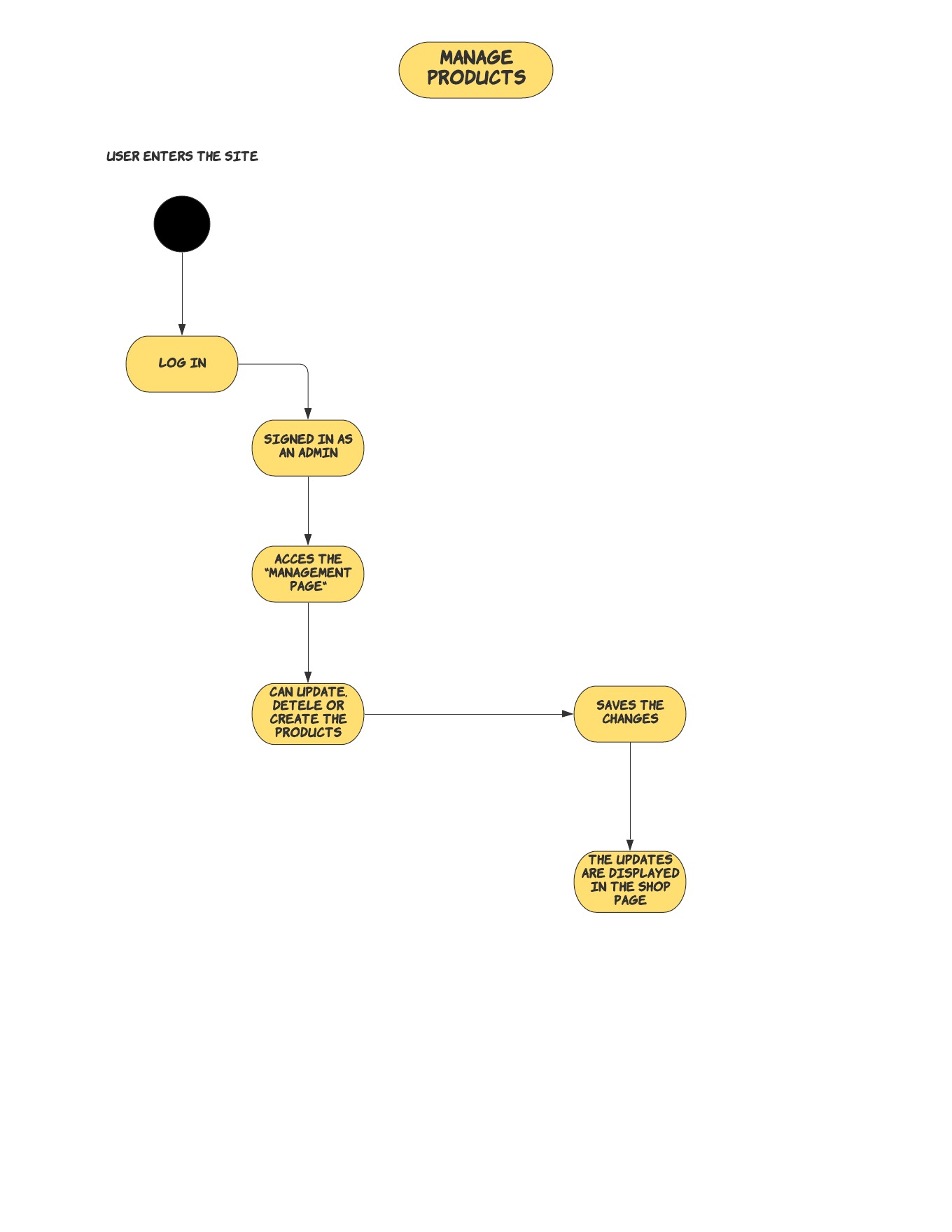












**6. Human Interface Design**

**6.1 Overview of User Interface**

Based on the user’s perspective, upon accessing the website they will see the home page, where they can see information about the TBPBShop. There are two headers, one header for the guest-type user which contains: Home, Products, Login Register and another header for the users that are logged in: Home, Products, Cart, UserInfo, Logout. Everyone starts as a guest-type user, and because this is a ecommerce shop, they must click on the “login/register” button to be able to use the full functionalities of the website. Of course, guest-type users are still able to view the products in the Products page, but they will not be able to buy any of them if they are not logged in.

Unsuccessful registrations/logins will lead to a warning. Failing to complete mandatory fields will also lead to a warning. Once logged in, the client type user will be able view the products and add to basket any products that is listed in the Products page, will be able also to modify his/her account information. Admin user has, in addition to the client user, the ability of creating new products, delete or edit them.

User interface share following qualities or characteristics:

**Clarity**: The interface avoids ambiguity by making everything clear through language, flow, hierarchy and metaphors for visual elements.

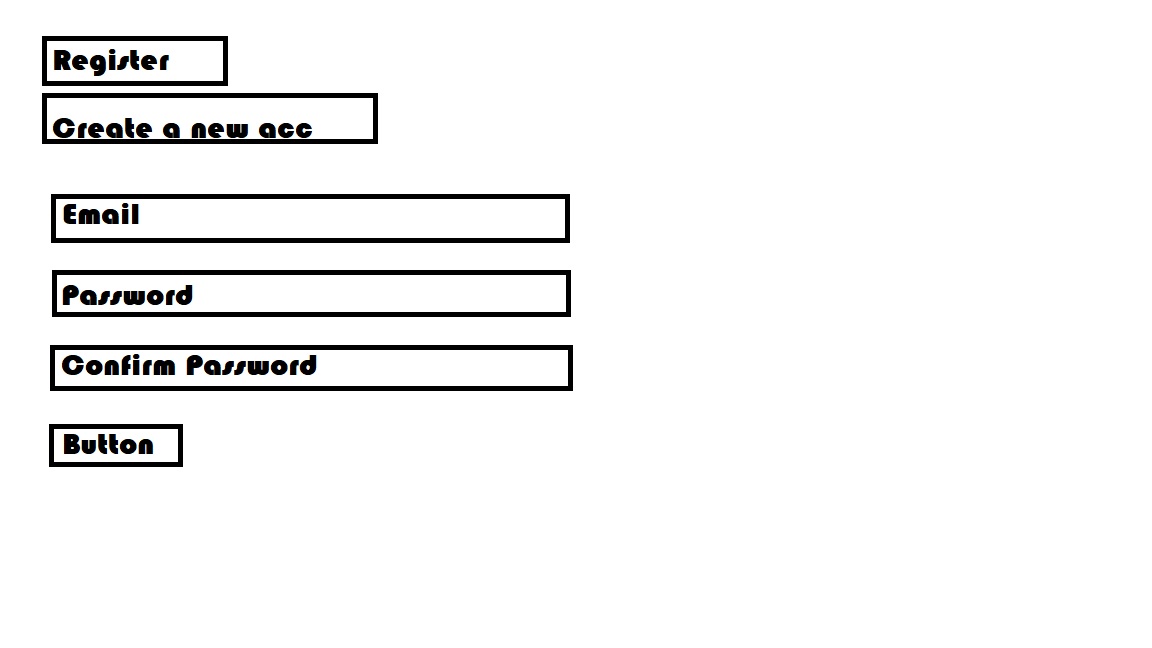
**Responsiveness**: This means a couple of things. First, responsiveness means speed: a good interface should not feel sluggish. Secondly, the interface should provide good feedback to the user about what’s happening and whether the user’s input is being successfully processed. And also the design could adapt to any type of screen(phone, laptop, tablet etc.)

**Aesthetics**: While you don’t need to make an interface attractive for it to do its job, making something look good will make the time your users spend using your application more enjoyable; and happier users can only be a good thing.

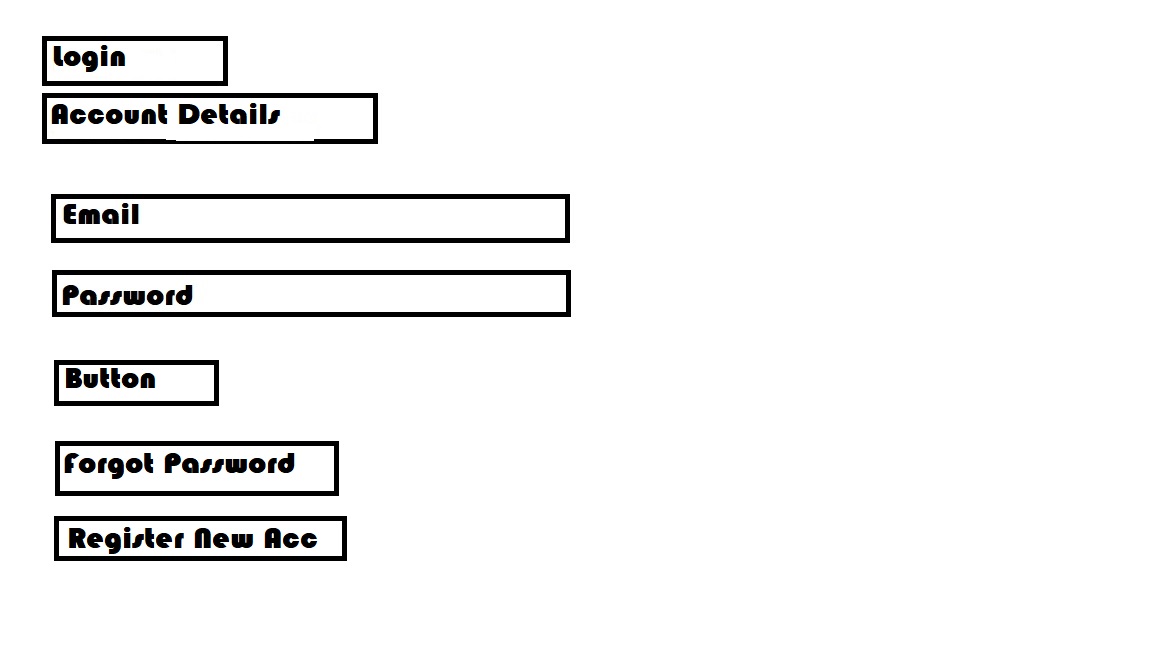
**Efficiency**: Time is money, and a great interface should make the user more productive through shortcuts and good design.

**6.2 Screen Images**

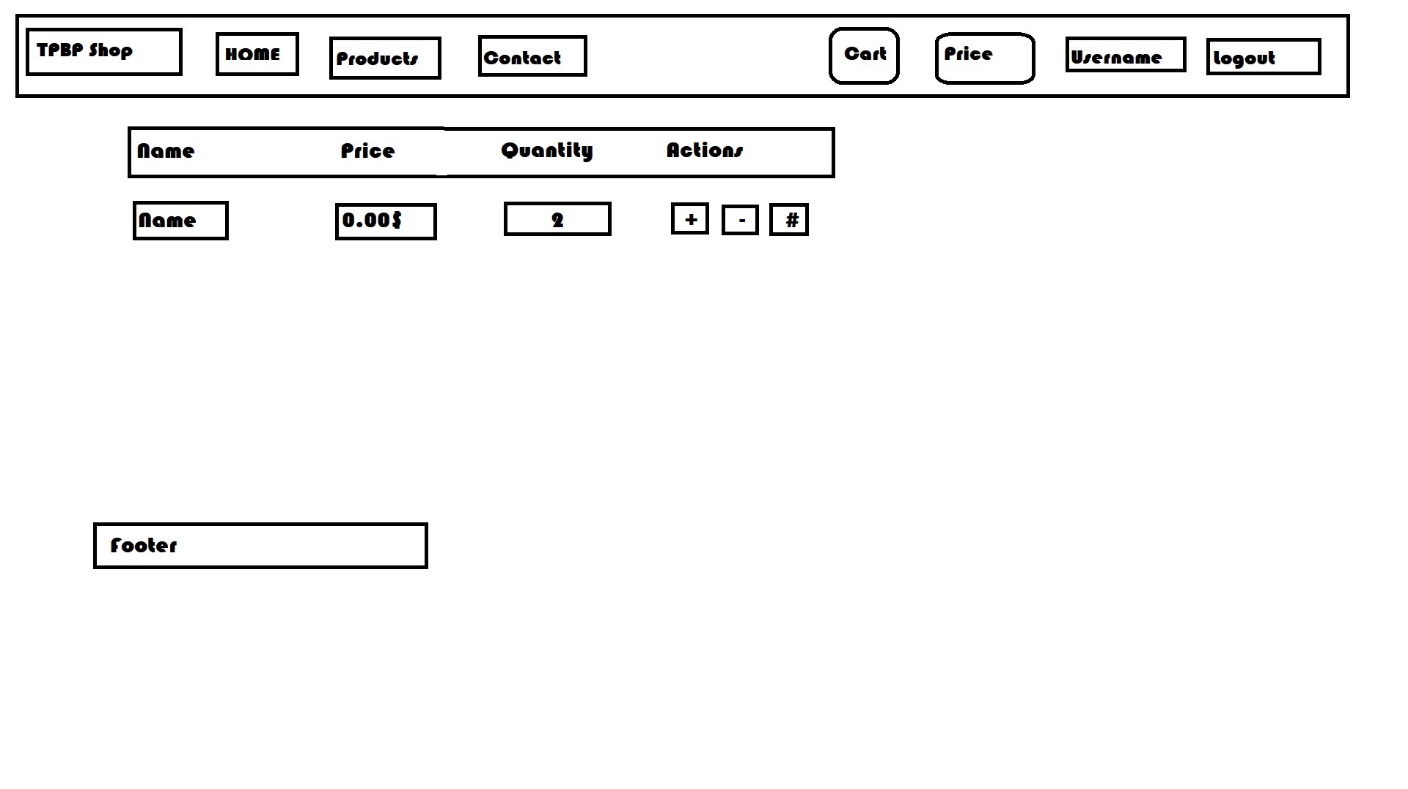
* Register Page



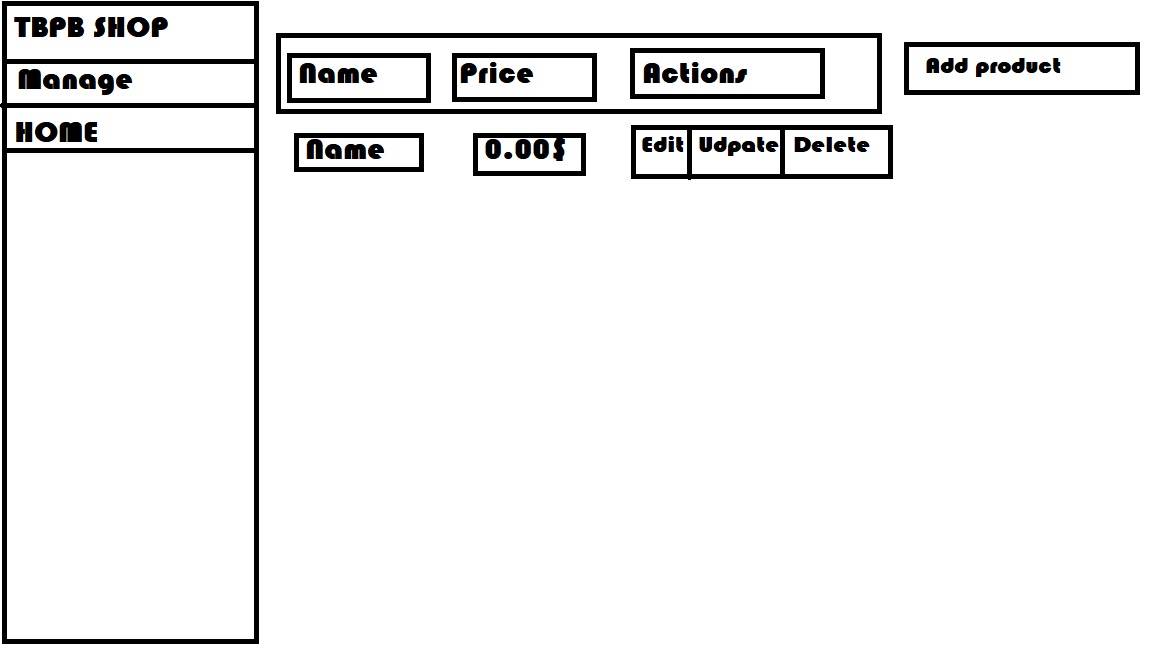
* Login Page



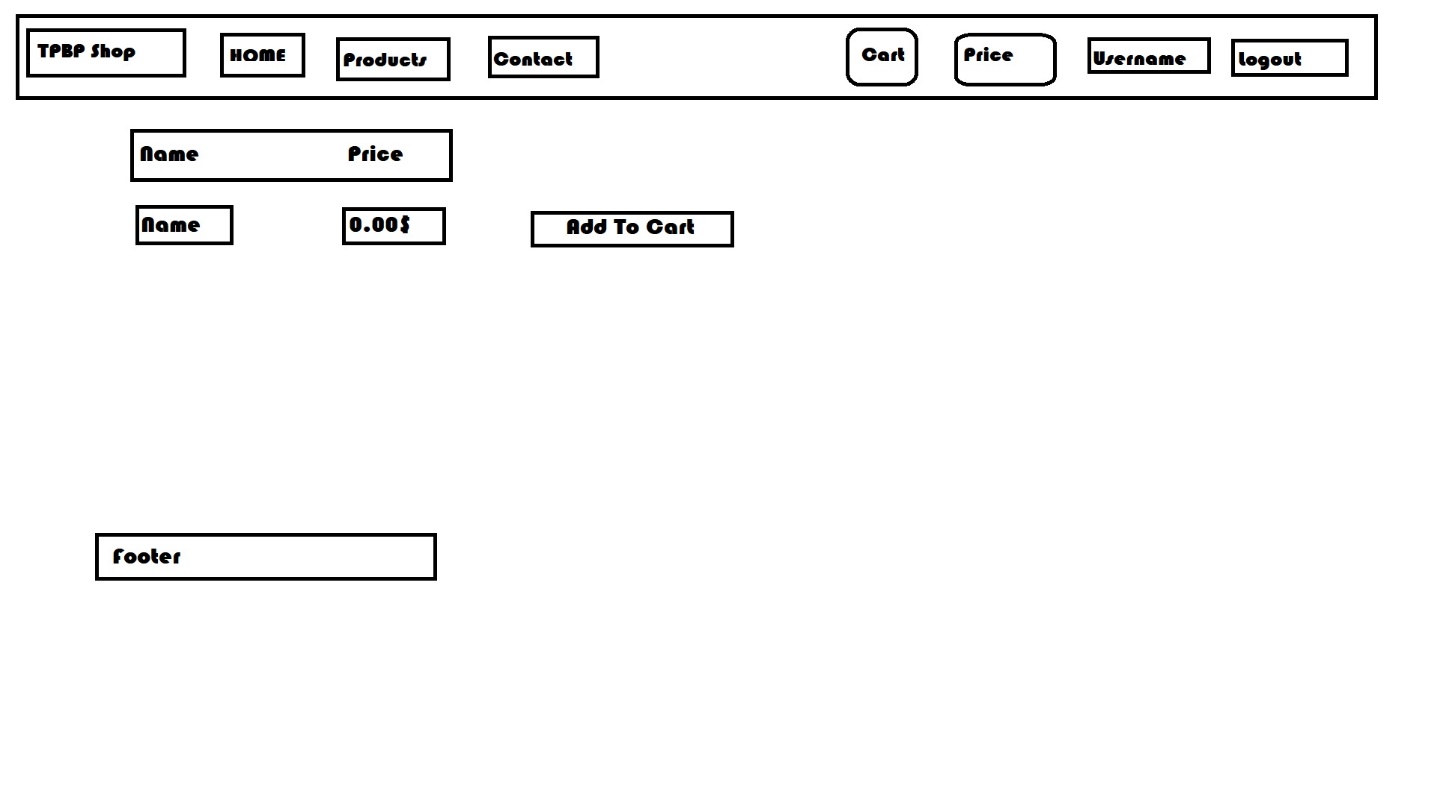
* Cart Page



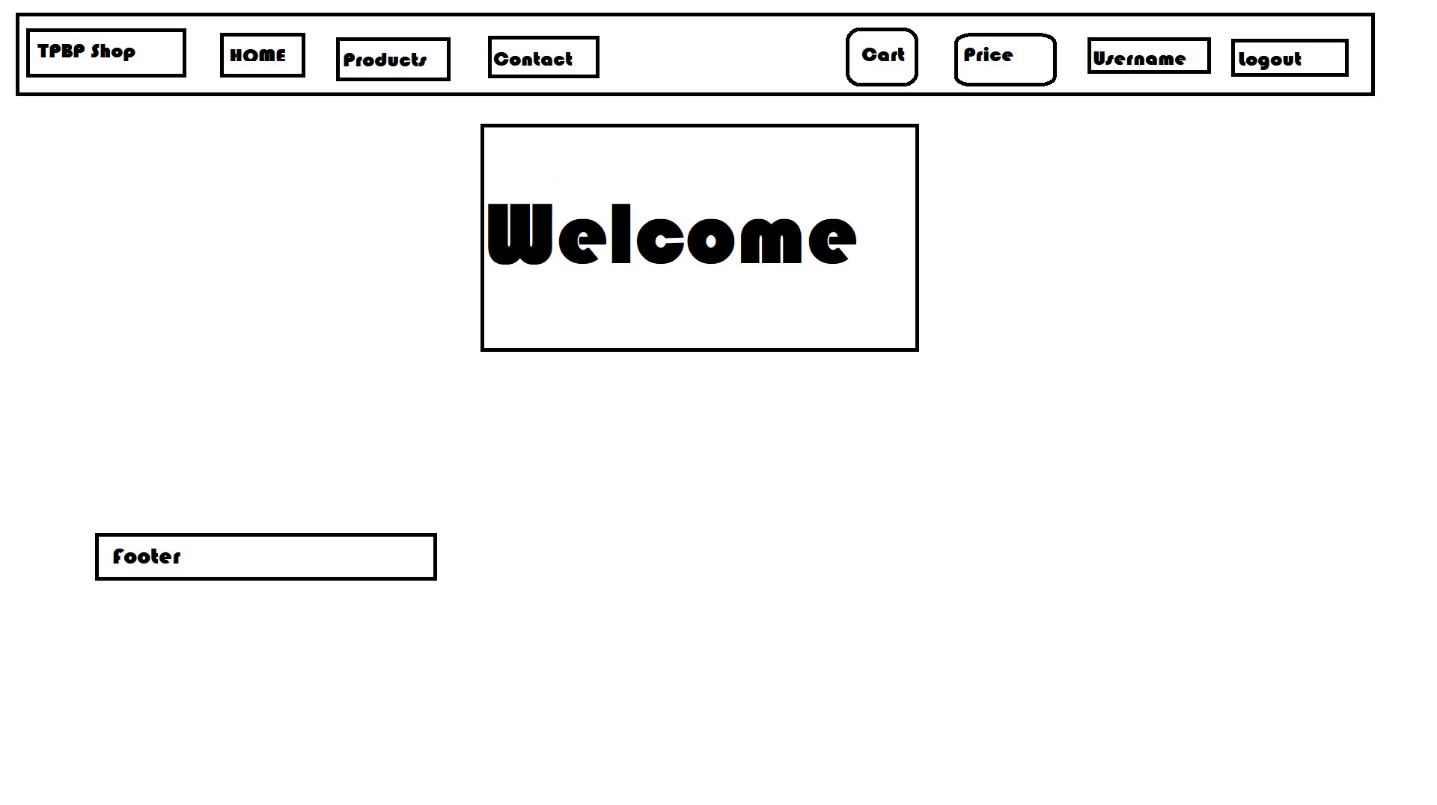
* Products management Page



* Shop Page



* Home Page



**6.3 Screen Objects and Actions**

Interface elements include:

* **Input Controls**: list boxes, buttons, text fields
* **Navigational Components**: breadcrumb, slider, icons
* **Informational Components**: tooltips, modal windows

**7. REQUIREMENTS MATRIX**

|  |  |  |  |
| --- | --- | --- | --- |
| **CLIENT** | **Name** | **Description** | **Reason** |
| **Register** | The new users that access the application has to access the “REGISTER” section, in order to create an individual account and fully enjoy our application.  This section has 3 input labels (email, password, confirm password). All the inputs must respect some constraints. The email must respect the given design, the password must respect the length and the strength.  If any of these constraints are not respected, a pop-up text will be displayed, telling them what they have done wrong. | This feature is a must, because the user must, somehow, belong in this application.  Other reason that this feature is a must, is that every individual register will be stored in the database. |
| **Log** **In** | Every user, no matter the role, must log in, so he can see the products and add them in the cart, as he wishes. Based on the role, he can access different areas of the application. | Through log in, the user can actually use the application. |
| **Search** | Every user has the option to search any product that our shop has, based on the name, or any other product characteristics. | This will make it easy for the user to find the product that he is looking for. |
| **Shop** | Here, everyone can see all the products available in the shop and add them in their own carts. | The shop page will let the user see every product in the shop and decide whether or not, they need it. |
| **Cart** | The cart will allow the user to add the products that he needs in it, so he can have an evidence of the total price and the all the items that he has added. | The cart will help both users, because they have an evidence of what they want to buy, and admin, which can check every user’s cart, since each one will be stored in the database. |
| **My Account** | This will allow the user to modify any characteristic of his own account, either changing the name, the password or anything else. | Using this feature, the user can manage his own account. |
| **Logout** | Every user has the option to log out. | By logging out, they will can log in into other account, or just log out from the account. |

|  |  |  |  |
| --- | --- | --- | --- |
| **ADMIN** | **Name** | **Description** | **Reason** |
| **Log In** | When logged in, this user will be assigned as a role of admin, which is set in the database. | The log in will, somehow, make the difference between the users, because of the roles, which are set in the database. |
| **Manage Products** | This is special future, only for the admin users, which allows them to manage the products and everything in the shop. | This feature will give the power to the specified user to take control over the application. He can perform the CRUD operation over the products. |
| **My Account** | This will allow the admin to update not only his account, but also the other accounts that are registered in the shop. | Through this feature, the user, that has the role of admin, can use the Read, Update and Delete operations over any account in the shop. |
| **Logout** | Every user has the option to log out. | By logging out, they will can log in into other account, or just log out from the account. |