Computer Retail Shop Web App

ANALYSIS AND DESIGN DOCUMENT

**CS 319 – Software Engineering**

**Group 5**

Alex Chen

Aziz Makda

Melisew Mengisitie

March 25, 2019

Contents

[**Contents** 2](#_Toc4433352)

[**System Architecture** 5](#_Toc4433353)

[**System Overview** 7](#_Toc4433354)

[**Architectural Design** 8](#_Toc4433355)

[**Decomposition Description** 10](#_Toc4433356)

[**Design Rationale** 13](#_Toc4433357)

[**Human Interface Design** 14](#_Toc4433358)

[**FEATURE: Shopping Cart** 21](#_Toc4433359)

[**INPUT & OUTPUT** 21](#_Toc4433360)

[**Actions** 22](#_Toc4433361)

[**Pre and Post Conditions** 23](#_Toc4433362)

[**Validation** 23](#_Toc4433363)

[**INPUT & OUTPUT** 23](#_Toc4433364)

[**Actions** 24](#_Toc4433365)

[**Pre and Post Conditions** 24](#_Toc4433366)

[**Validation** 25](#_Toc4433367)

[**INPUT & OUTPUT** 25](#_Toc4433368)

[**Actions** 26](#_Toc4433369)

[**Pre and Post Conditions** 26](#_Toc4433370)

[**Validation** 27](#_Toc4433371)

[**FEATURE: Item Catalog** 27](#_Toc4433372)

[**INPUT & OUTPUT** 27](#_Toc4433373)

[**Actions** 28](#_Toc4433374)

[**Pre and Post Conditions** 29](#_Toc4433375)

[**Validation** 29](#_Toc4433376)

[**INPUT & OUTPUT** 30](#_Toc4433377)

[**Actions** 31](#_Toc4433378)

[**Pre and Post Conditions** 31](#_Toc4433379)

[**Validation** 31](#_Toc4433380)

[**INPUT & OUTPUT** 32](#_Toc4433381)

[**Actions** 32](#_Toc4433382)

[**Pre and Post Conditions** 33](#_Toc4433383)

[**Validation** 33](#_Toc4433384)

[**FEATURE: Review and Q&A** 33](#_Toc4433385)

[**INPUT & OUTPUT** 33](#_Toc4433386)

[**Actions** 34](#_Toc4433387)

[**Pre and Post Conditions** 35](#_Toc4433388)

[**Validation** 35](#_Toc4433389)

[**INPUT & OUTPUT** 36](#_Toc4433390)

[**Actions** 36](#_Toc4433391)

[**Pre and Post Conditions** 37](#_Toc4433392)

[**Validation** 37](#_Toc4433393)

[**INPUT & OUTPUT** 38](#_Toc4433394)

[**Actions** 39](#_Toc4433395)

[**Pre and Post Conditions** 39](#_Toc4433396)

[**Validation** 39](#_Toc4433397)

[**Component Design** 39](#_Toc4433398)

[**Feature: Shopping Cart** 39](#_Toc4433399)

[**Entities** 40](#_Toc4433400)

[**Feature: Item Catalog** 42](#_Toc4433401)

[**Entities** 42](#_Toc4433402)

[**Feature: Review** 44](#_Toc4433403)

[**Entities** 44](#_Toc4433404)

[**Data Design** 48](#_Toc4433405)

[**Data Description** 48](#_Toc4433406)

[**Shopping Cart Feature** 49](#_Toc4433407)

[**DATABASE Dictionary / SCHEMA** 52](#_Toc4433408)

[**Shopping Cart Feature** 52](#_Toc4433409)

[**Item Catalog Feature** 52](#_Toc4433410)

[**Review Feature** 52](#_Toc4433411)

[**Full Database Model** 57](#_Toc4433412)

[**Requirements Matrix** 58](#_Toc4433413)

[**Appendices** 61](#_Toc4433414)

**Introduction**

* **Purpose**
  + The purpose of the analysis and design document is to provide a detailed overview to assist in the development by providing information on how the software should be constructed. Within the analysis and design document, is an anecdotal and visual specification that will provide a basis on the architecture of the website which includes designs of system architecture, user interface, component, data and a full database model.
* ***Scope:*** 
  + This analysis and design document will be implemented to assist and guide the development team by an architectural basis. This architectural basis contains design components along with database models and how they interact with each other to provide a UI that is user friendly.
* **Overview/References:**
  + The goals and objectives of this website are to provide consumers with a quick and easy way to purchase computer-related products while expanding popularity of the website to become a national brand.  We plan on achieving this by providing an effective website that provides real user experiences of products, a Q&A section and specifications of products to reduce complications of purchasing products. The three main components that are being developed and included are the shopping cart, product catalog and the review/Q&A section of the website.

***The breakdown of work is as follows:***

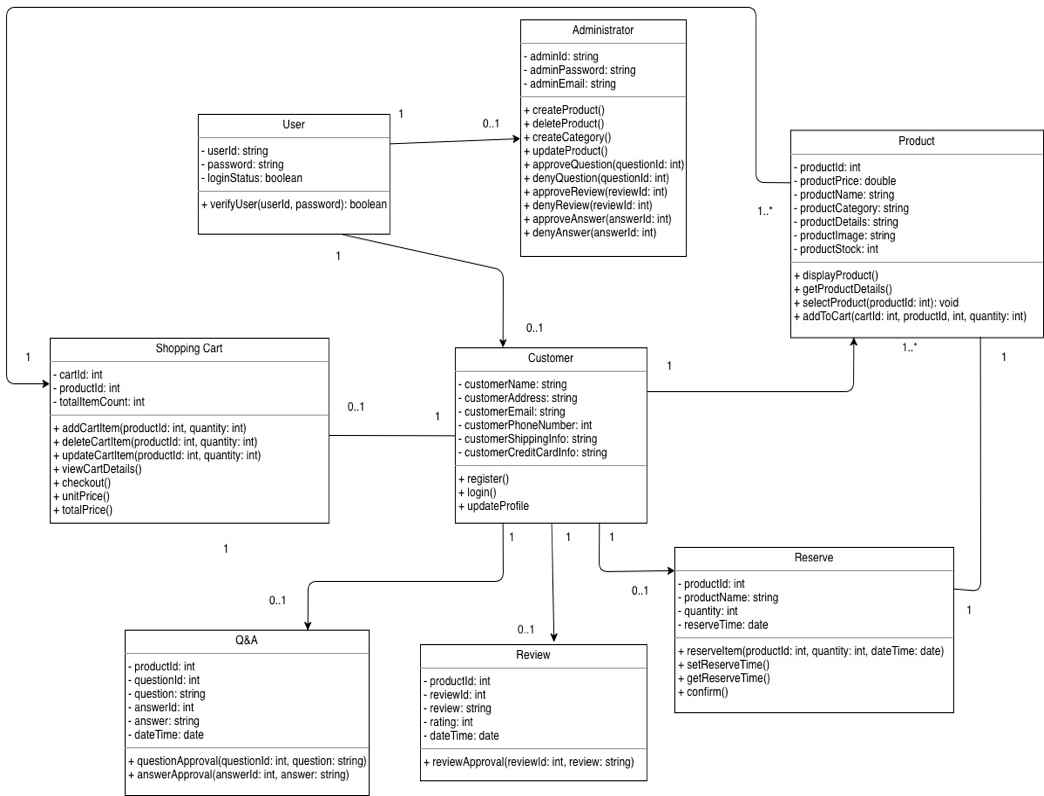
|  |  |
| --- | --- |
| Component | Team Member |
| Shopping Cart | Melisew Mengisitie |
| Product Catalog | Alex Chen |
| Review/Q&A | Aziz Makda |

* **Overview**
  + This document is organized with the main components of the introduction, system architecture, human interface design, component design, data design and the database model. The introduction contains an overview of the document, purpose and why this document exists. The system architecture gives a general description of the functionality, context and design of this computer retail store. The human interface design provides detailed information of our website features and goes in depth on how the features act and how they are connected with the rest of the system. The component design section provides information on the methods in each feature and how they are connected with visual representations provided in Universal Modeling Language representations.  The data design section is a broad view of how all data in the system is interconnected that is represented by entity-relationship model diagrams.
* **Definitions and terms:**
* DBMS- (Database Management System) - A system software used for creating, updating and managing databases.
* IDE- (Integrated Development Environment)- A software suite that contains a code editor to write and edit code.
* Software - A program/set of instructions that is used for operating and executing a particular task.
* Compile - A program that converts a programming language code into machine language for the computer to execute the program.
* Execute - Perform a computer program.
* Debug - Used for detecting errors in programming code, so the programmer can fix/remove the errors.
* Prototype - is a sample or model of the designing product that’s used for testing the process of a product.
* Database - A structured collection of data that is stored, accessed and managed through a computer system.
* Use case - A list of actions that illustrates how the user will interact with the software product.
* Tables (Database) - A collection of data stored in the form of table.
* Rows (Database) - The horizontal structured data in the table.
* Columns (Database) - The vertical structured data in the table.
* Website Host (FTP) - hosts the website to be accessible and viewed in the internet.
* HTTP - Hypertext Transfer protocol - is a client server protocol that is used for exchanging data in the web and client server.
* Domain - is an IP (Internet Protocol) address or identification of the website that a server device hosting a website.
* IP (Internet Protocol) - is a set of principles that manages the format of data that is sent from one device to another.
* System - A set of steps, principles and components that includes hardware and software that are connected together to provide services to other programs.
* User - A person who uses the software product.

**System Architecture**

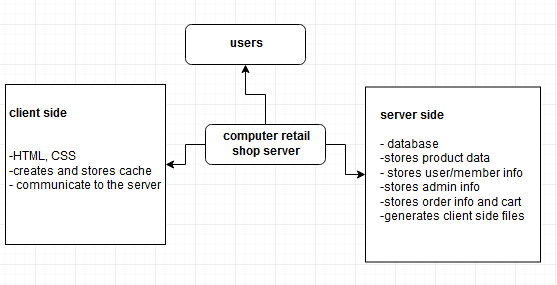
This system will be comprised of a set of major components/classes that include user, administrator, product, reservation, review, Q&A, shopping cart and the customer. These components will work hand in hand with the main modules in the following ways:

* The customer will have 0 or 1 shopping cart that will comprise a unique shopping cart ID, product ID(s), and a total item count for the total items to be displayed on the website. The shopping cart contains actions such as adding items, deleting items, updating the cart, viewing the cart, checkout and unit/total prices. The shopping cart will inherit information from the product class which contains product ID, price, name, category, details, image and the stock count of the item.
* Within the product features, it contains all pertinent product information and passes information to the shopping cart and the product reservation and inherits information from the customer. The actual product information contains product ID, price, name, category, details, image and the stock count of the item. The actions within the product class display the product, get product details, select a product(s) and add an item to the cart.
* The review and Q&A section receive information from the customer class. The review feature contains information like the product review, the review ID, the review contents, rating of the review and a time and date of the review posting, and an administrator can approve or deny the review. Within the Q&A class, there is information like product ID, question ID, the actual question, answer ID, answer and the time and date along with methods to approve or deny both the question and answer.

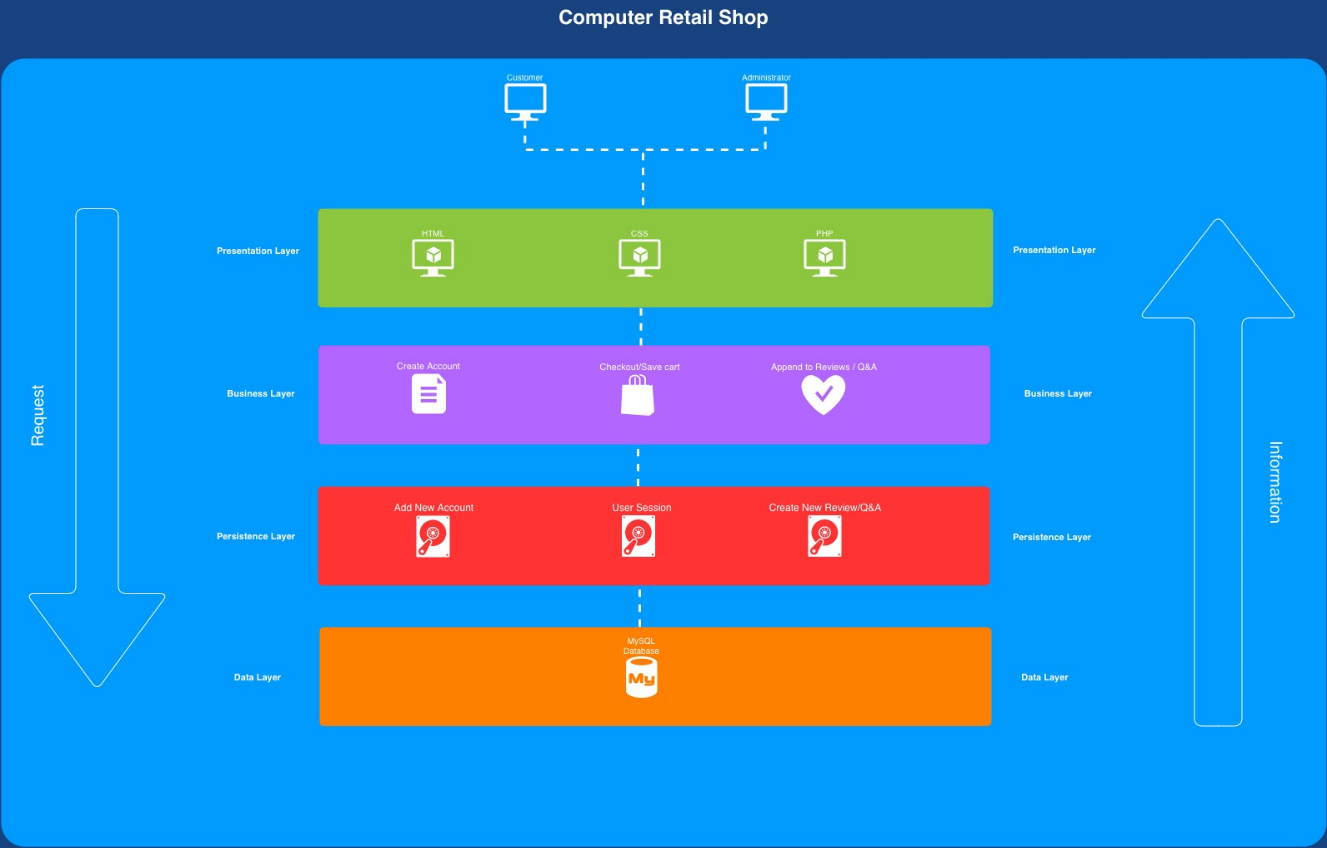
**

**System Overview**

The computer retail shop is an e-commerce website intended for audiences that have knowledge on computer parts. The computer retail shop has two important components, one is the client side, and the other is the server side. The server-side component will store background information such as database information for example: storing a user, product, admin, order and cart tables. An additional feature of the server side is emailing request to the user for a review postage if the user’s product has shipped successfully. The client-side component generates item catalog for the user, creates cache, and communicates and synchronize to the server.



**Architectural Design**



Within the computer retail website, the presentation layer is the actual visual aspect of the website where customers and administrators will view products, look through different computer hardware categories and interact with different visual items on the website. Within the presentation layer, the HTML (Hyper Text Markup Language) portion of the website will provide the look and basic styling of the actual website since the HTML language provides a foundation of how a browser renders that information. The CSS (Cascading Style Sheets) takes the structured content that is initiated within the HTML and enables a high level of customization such as sizing, colors and different fonts. Finally, the PHP (Hypertext Preprocessor) language will be the data manager for the website such as sending customer data for payment processing.

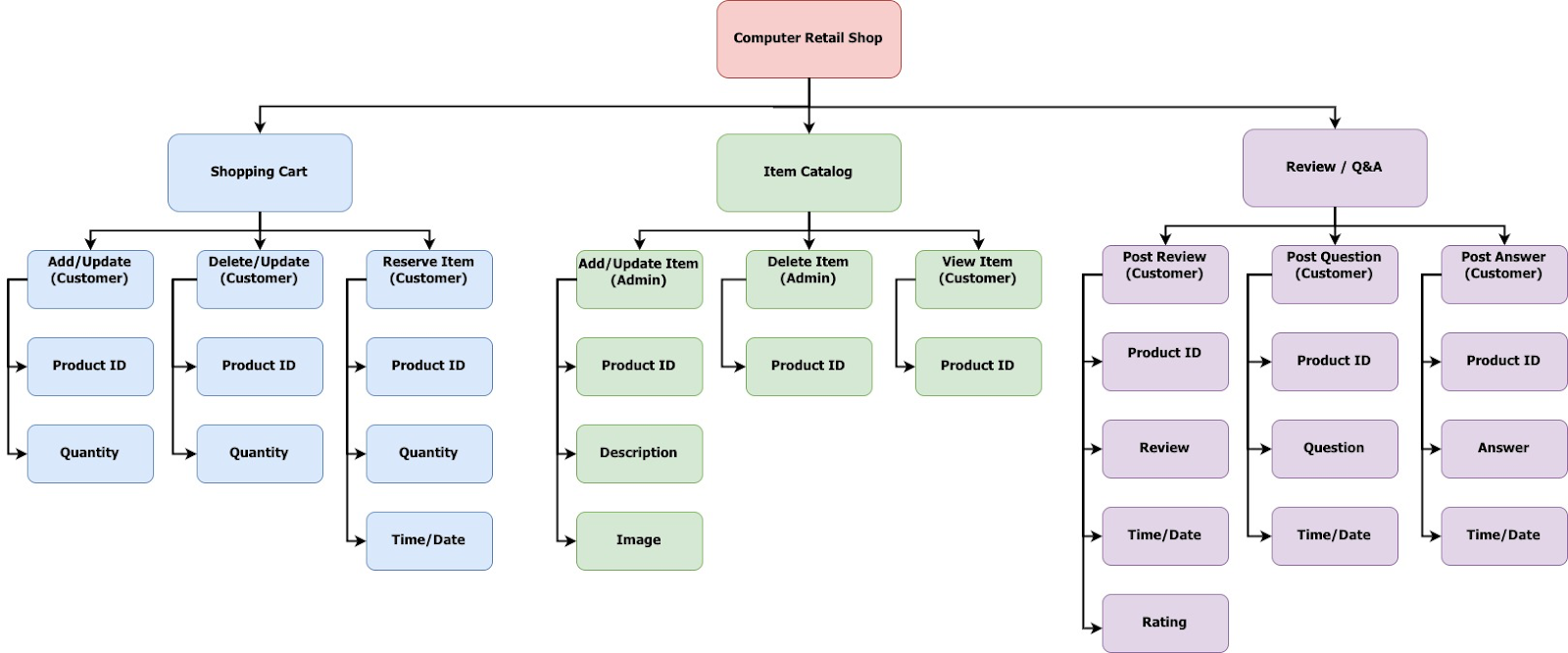
The business layer comprises the website features that pertain to the business functions which are the creation of a new account, checkout of a current shopping cart and appending information to the review/Q&A of the site. The presentation and business layer will work together by customers using the visual aspect of the presentation layer coupled with the computer’s I/O to create an account, checkout and change the review/Q&A part of the site. The information will be sent to and from the presentation and business layer by PHP programming.

The persistence layer is where the customer’s sessions are saved regarding the shopping cart. If a customer adds products to their cart and have to close their web browser, the products within their shopping cart will be restored upon returning to the site. The user session, within the persistence layer, will receive data from the business layer via PHP and save an instance of the cart state for restoration.

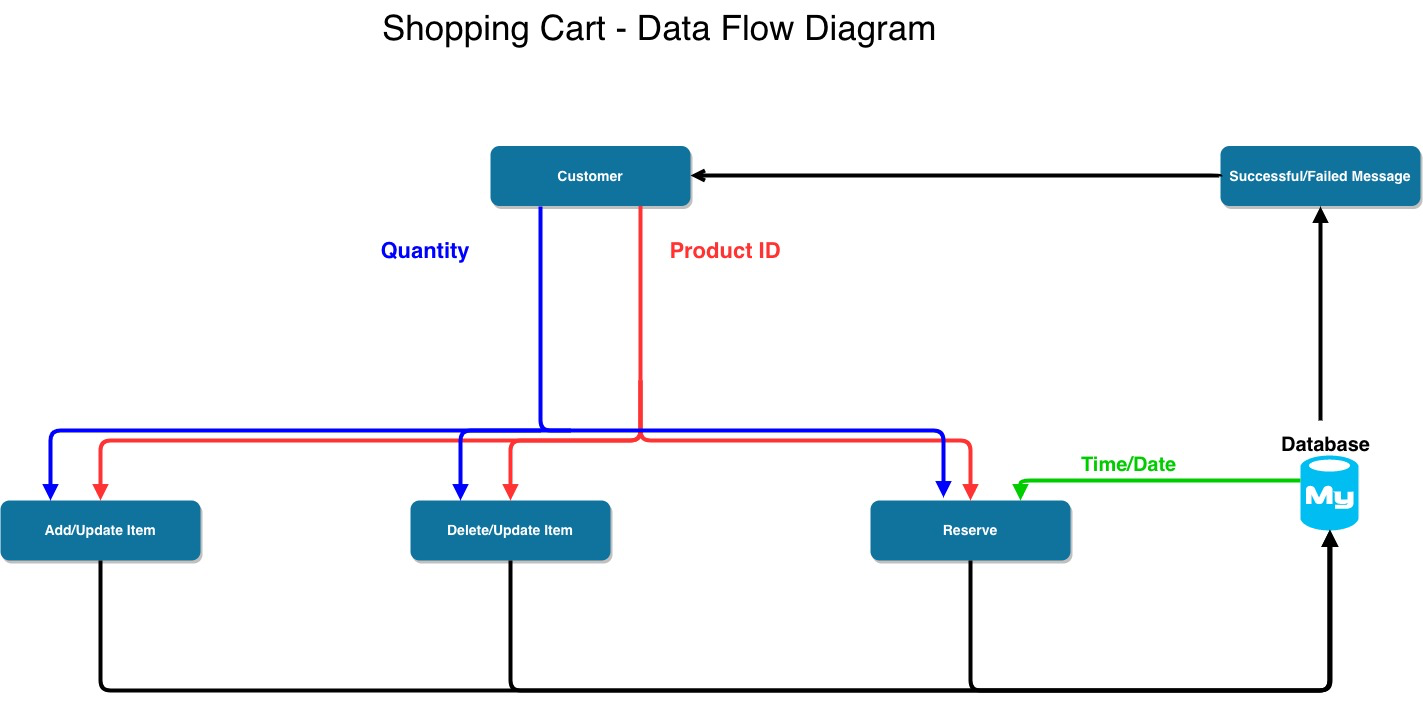
The data layer will contain the actual MySQL based database that stores pertinent website information like the user sessions and product-based information like descriptions, pictures, pricing, stock levels accounts and reviews and Q&A data. The data level and persistence level will work together by the information stored in the user session of the persistence layer will be stored accordingly in the database, so it can be readily restored when the user returns to our retail shop. Information from new accounts and any reviews or Q&A information will be pushed to the database from the persistence layer.

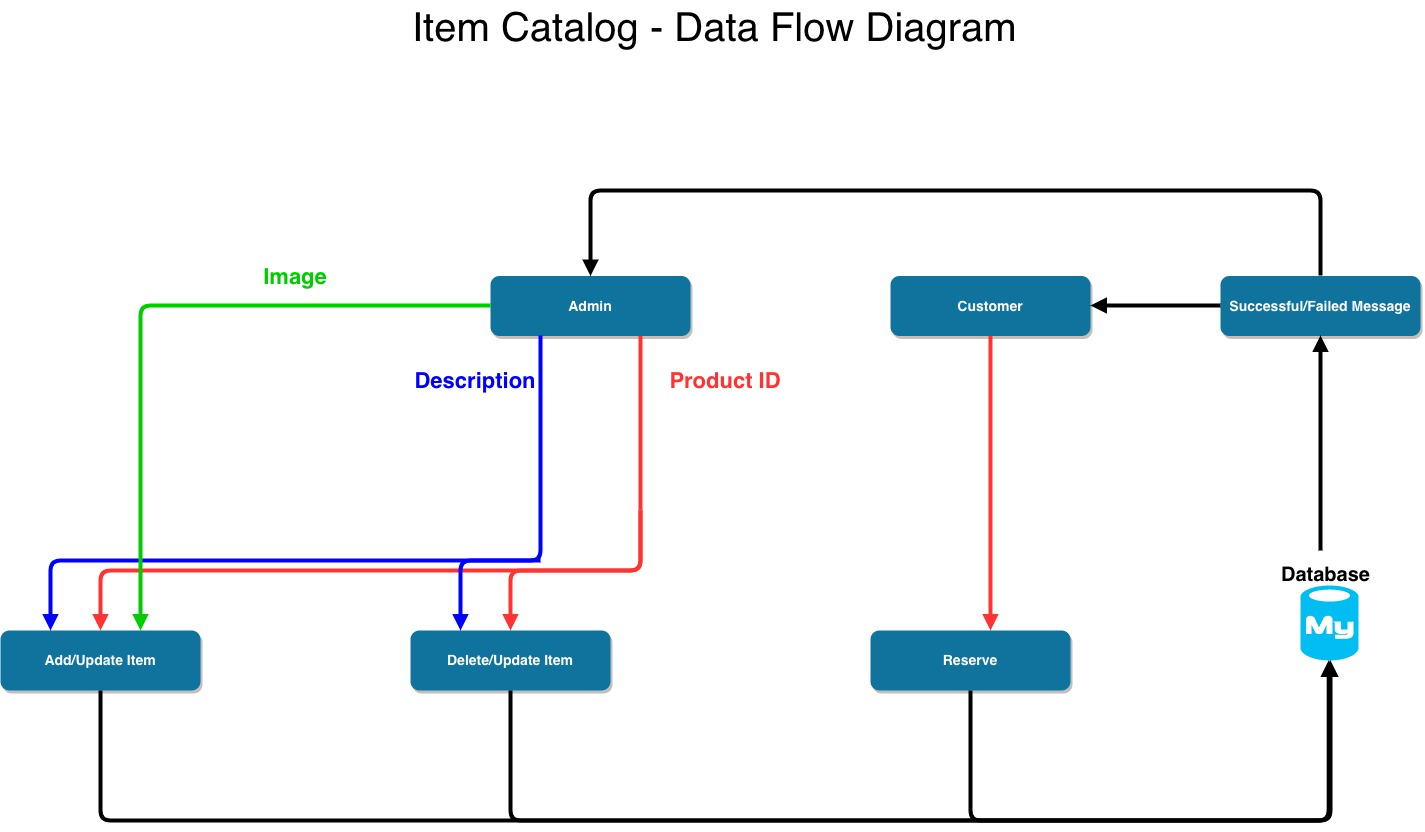
**Decomposition Description**

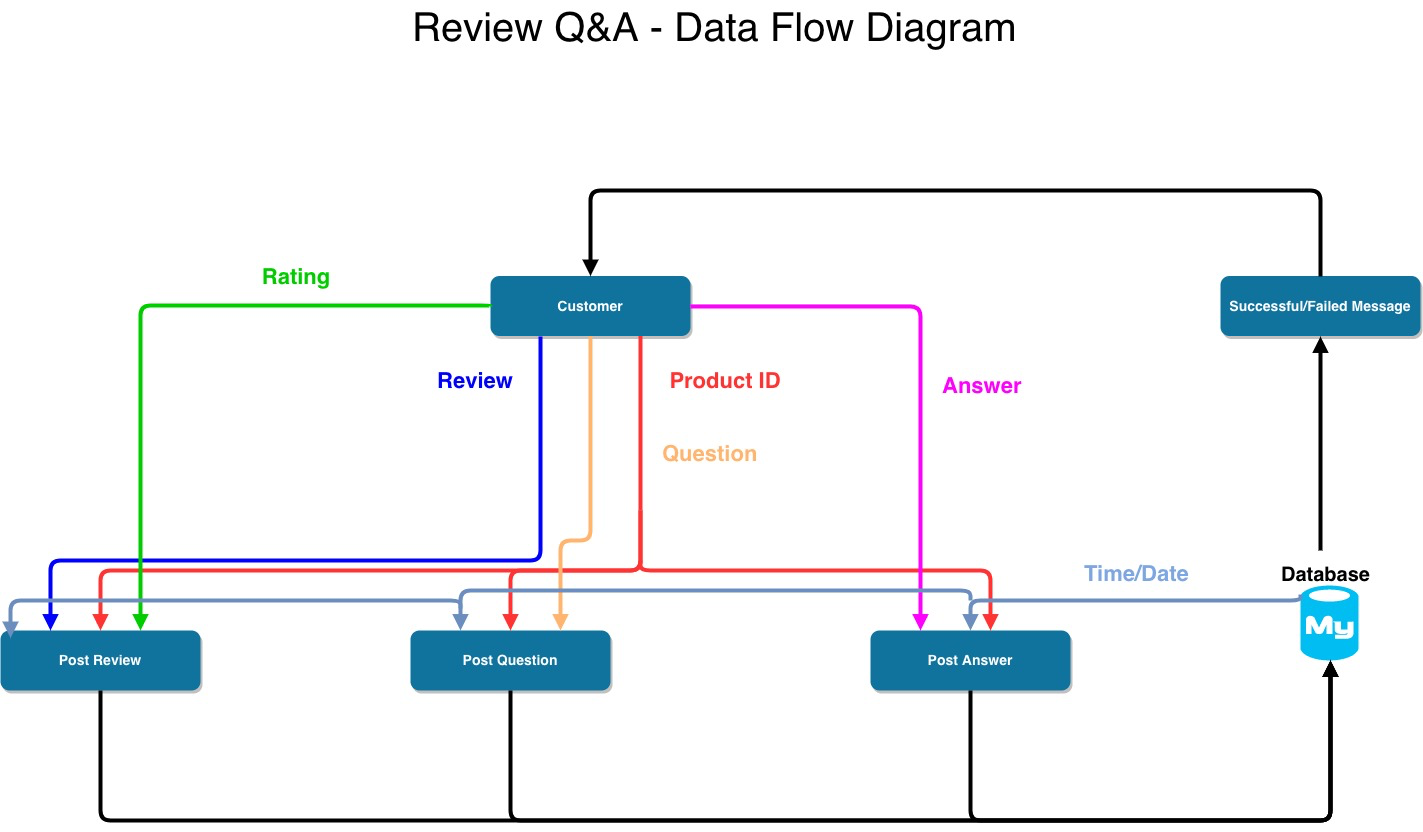
Structural Decomposition Diagram

**

Data Flow Diagram

**

**

**

**Design Rationale**

Our rationale in picking the layered arch is because it keeps all functions inside the layer insulated and isolated from one another. The changes that happen in one layer will not affect any other layer unless for example SQL is used in the persistence/data access layer to affect the business layer by retrieving customer information from the database layer.

Layered arch will keep all the information and code self-contained and organized. It will be easy for the third party to support if need be in the future. We also chose this arch because if future expansions were to be made into the computer retail website it will be easier to see the flow of functions rather than have everything in one place.

Another reason we chose the layered arch is because since we will utilize the database in our website due to our products, the layered arch will be more efficient to implement. It will make the arch also easier to debug since all layers are apart and at the same time we can see the flow of the code itself.

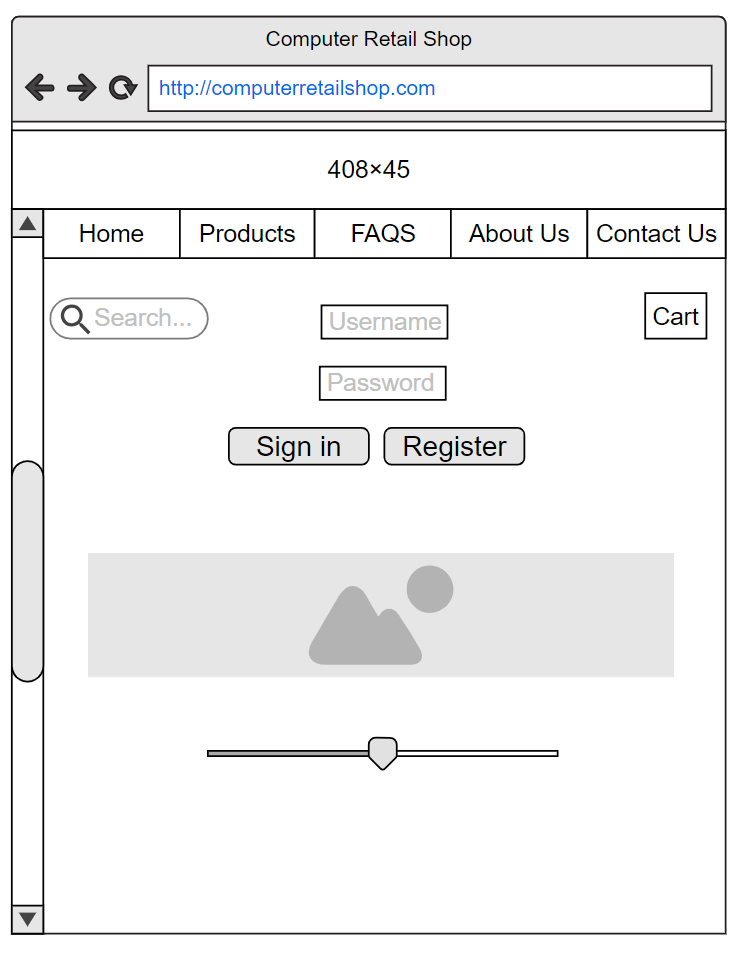
The downside to picking this arch for us will be that it is based on database mainly, therefore going away from database will not be a solution in the future. Another downside is that layers themselves cannot be extended due to the fact that the entire architecture will have to be expanded together which will be useless in some situations and increase cost. Layered architecture has to usually be release at once, so either one team or many other teams working on different sections will have to release the layered architecture together.

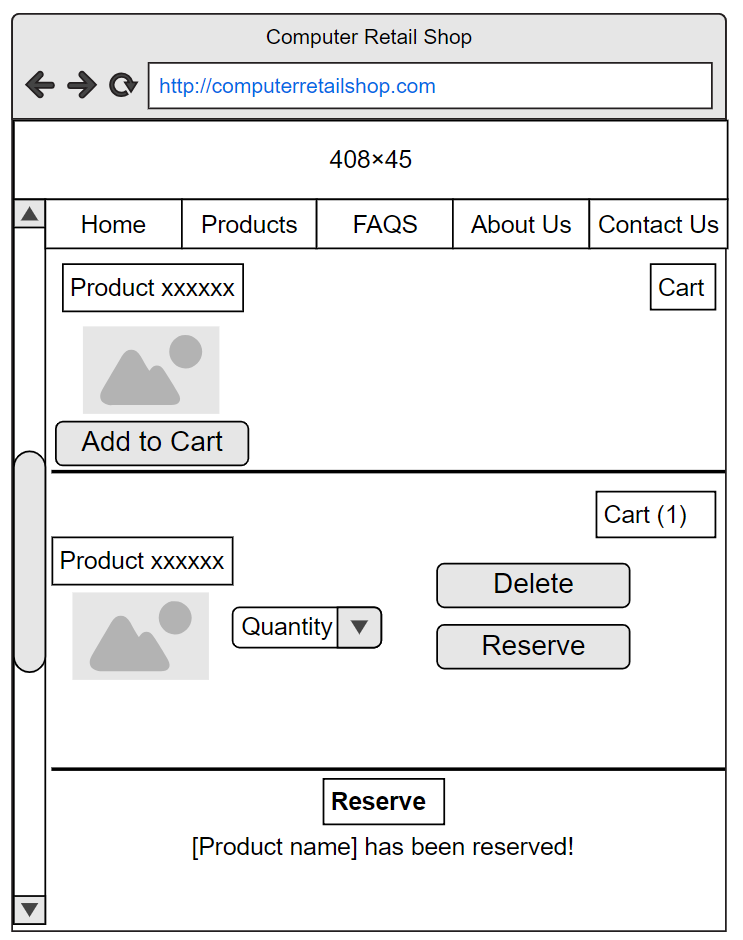
**Human Interface Design**

    User View Design:

This page presents the user to view the banner for advertisements with the slider, log in or register. User has the option to search for products via search bar or navigation bar presented. User can also choose other navigation bar options.

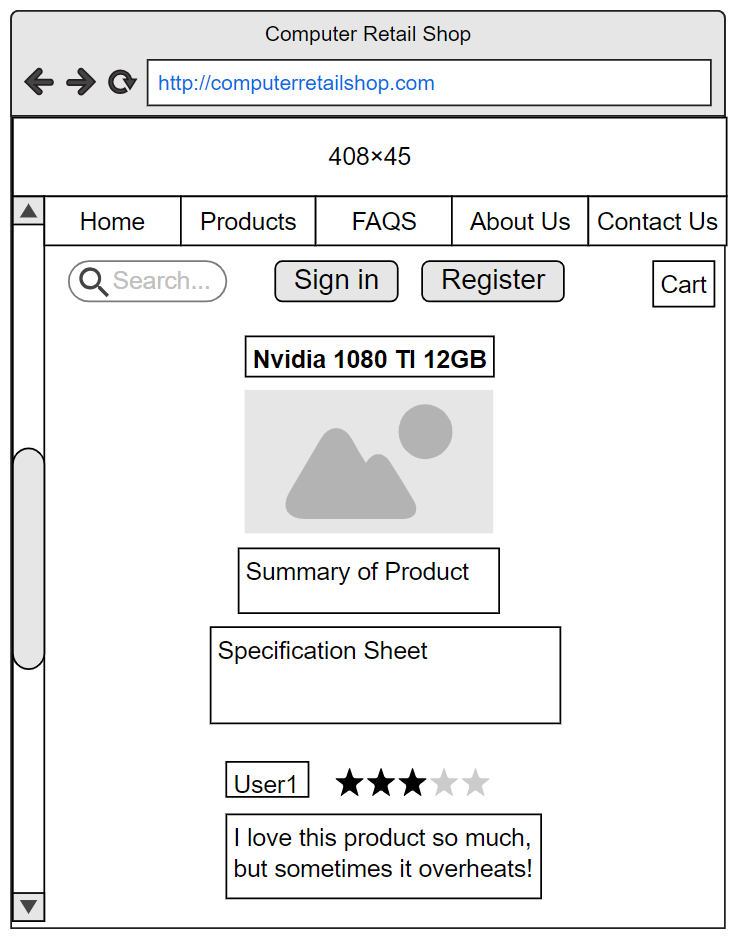
The second page presents three scenarios which are: user adding a product to the cart, the cart page and successful reservation of a product.

**

**

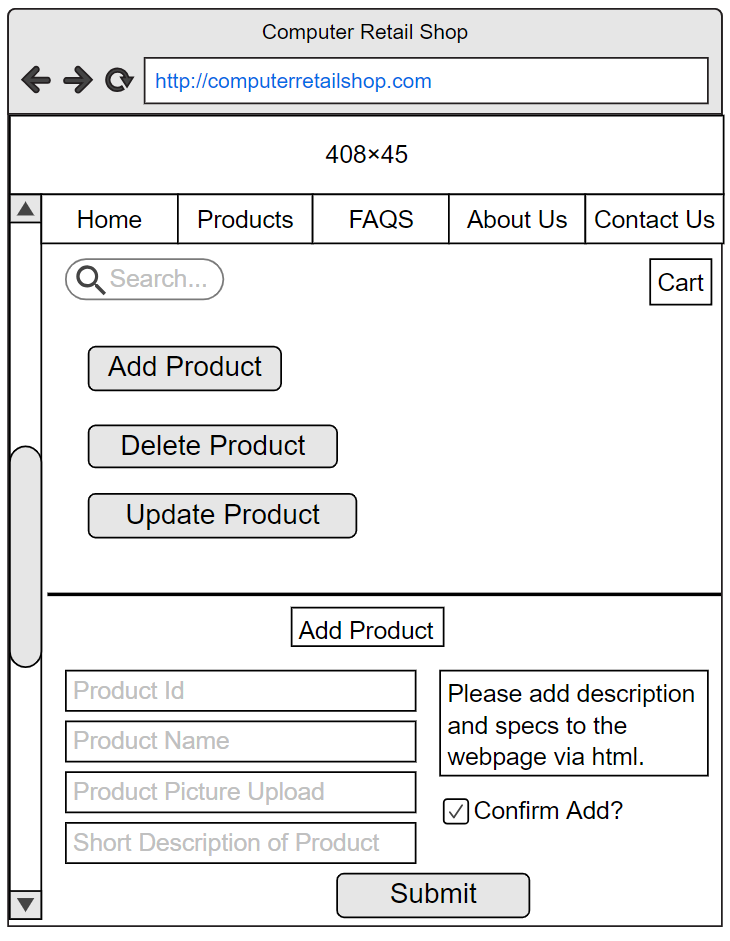
**Product Catalog View (user):**

This page will allow a user to look at the detail description and rating of the product.

**

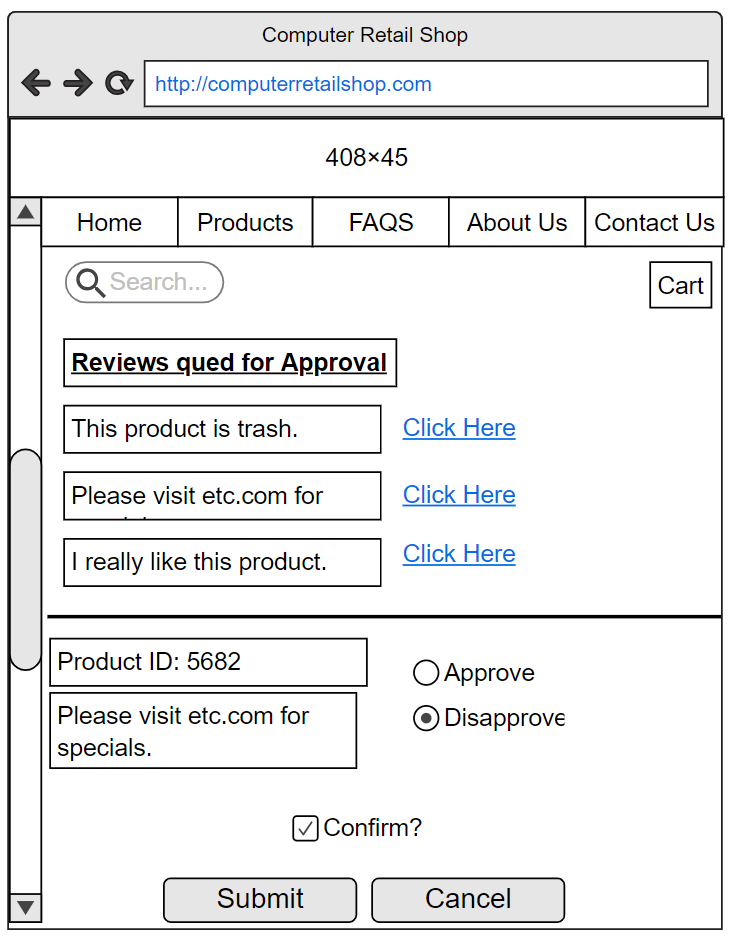
**Admin View of product catalog:**

This page will allow to add, update and delete the product. Underneath the horizontal line is an example of a product being added.

**

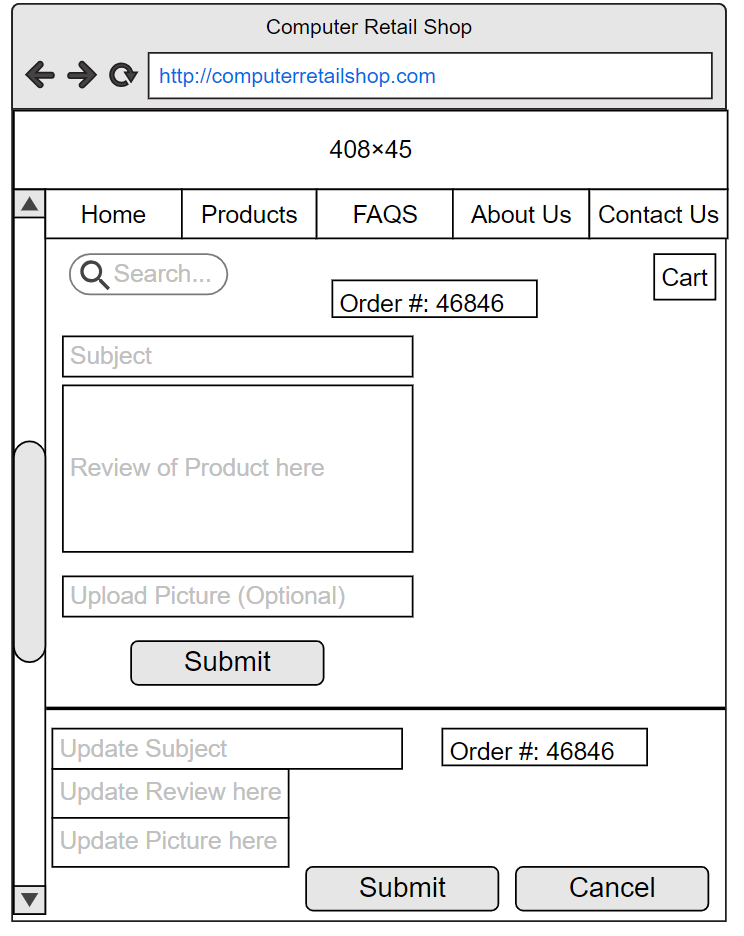
**Admin view of Reviews:**

This page will allow an Admin to approve or delete the review.

**

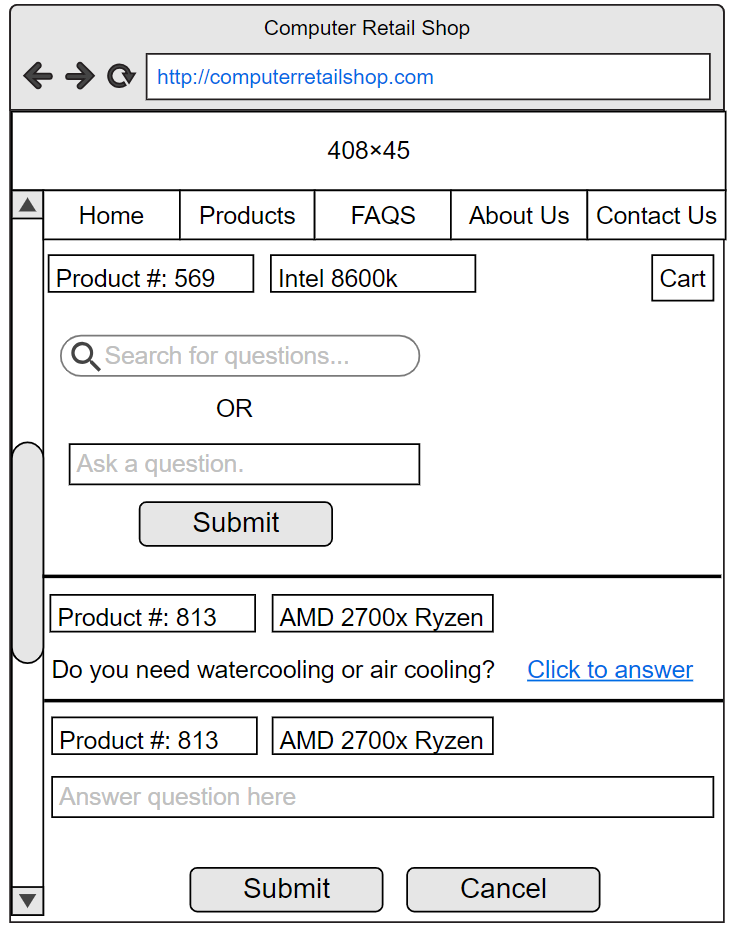
**User view of Reviews:**

This page will allow a user to add or update a review for a product that the user bought. Update scenario is below the horizontal line.

**

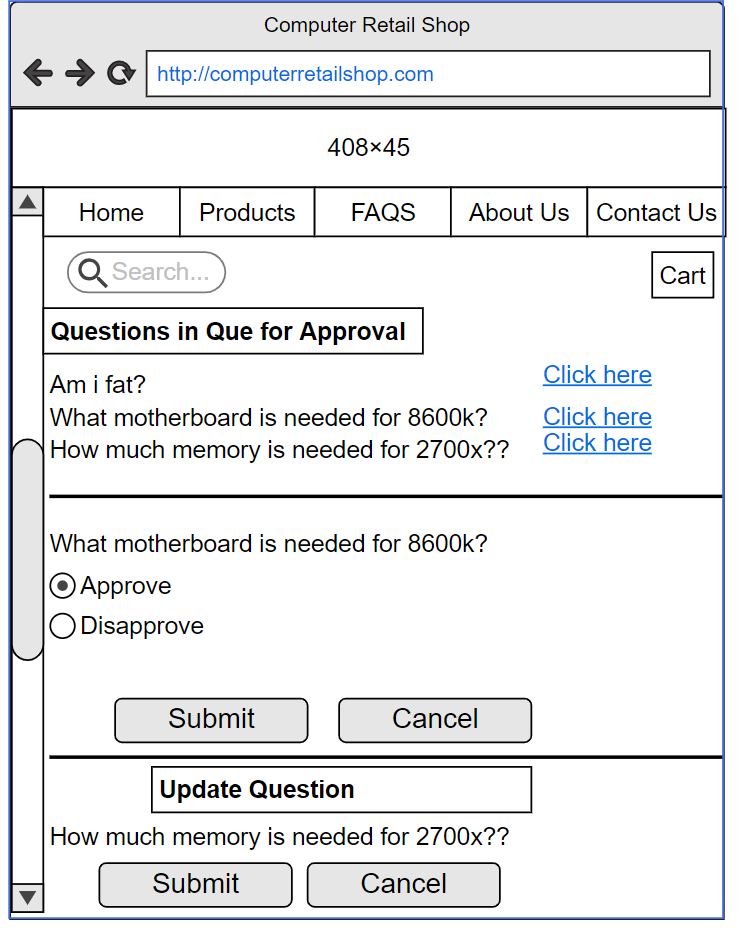
**User View of Q&A:**

This page will allow a user to add a question or answer about the product. The mock up UI presents three scenarios. The first scenario is searching for a question or asking a question. The second and third scenario divided by the horizontal line and contains when a user clicks on a question to answer and the user proceeding to answer the question.

**

**Admin View of Q&A:**

This page will allow an Admin to approve, update and delete a question or answer about the product. The mock up shows three scenarios, the questions for approval, the approve/disapprove process of a question and updating a question. The same could be done for answers.

**

**FEATURE: Shopping Cart**

**USE CASE #1 User Adds/Updates Item to the Shopping Cart**

The user will be able to add or update products to the shopping cart. The user must be logged in to buy the product with the cart.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product Quantity:** Integer representing the number of products available.
* **Product Price**: Decimal representing the cost of a single product.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **User ID:** Unique string representing a user.

**Output:**

Upon successful addition of product in cart, the system will display a success message. If the product is not available, it will return a warning message that the product is out of stock.

**Possible messages:**

Successful: “[Product name] Added to Cart!”

Successful: “[Product name]” quantity updated!”

Failed: “No stock available for [Product name], sorry! Please check back again when we restock.”

**Actions**

Upon adding or updating quantity of the product, the system will validate the stock before adding/updating quantity of the product. System will also decrement stock when an item is added to the cart. System must make sure that item(s) does not fall below zero in the database. When updating cart quantity, the system must update cart quantity and send a confirmation output to the user of success or failure.

**Pre and Post Conditions**

**Pre-conditions:** If user tries to add or update or buy item(s) to the cart, the database checks if item is available.

**Post-conditions:** If the item is in the database, the user can add/update the item into the cart.

**Validation**

The following validations will be performed on the submitted data:

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product Quantity:** Fields must be at least a single character. Field cannot be null.
* **Product Price**: Field is designated a decimal type and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **User ID:** Must be unique and field cannot be null.

**USE CASE #2 User Deletes/Updates Item from the Shopping Cart**

The user will be able to delete products from the shopping cart. The user must be logged in to buy the product with the cart.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product Quantity:** Integer representing the number of products available.
* **Product Price**: Decimal representing the cost of a single product.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **User ID:** Unique string representing a user.

**Output:**

Upon successful deletion of product in cart, the system will display a success message. If the product is not in the cart it will return a warning message to report to the admin about possible errors.

**Possible messages:**

Successful: “[Product name] Deleted from Cart!”

Failed: “[Product name] is not in cart to be deleted. Please contact the System Admin about this problem.”

**Actions**

Upon deleting or updating quantity of the product, the system will validate the stock before deleting/updating quantity of the product. System will also increment stock when an item is deleted from the cart. System must make sure that item(s) does not fall below zero in the database. When updating cart quantity, the system must update cart quantity and send a confirmation output to the user of success or failure.

**Pre and Post Conditions**

**Pre-conditions:** If the user deletes an item in the cart, the item must be in the cart.

**Post-conditions:** If the item is indeed in the cart, then the system will proceed to delete the item from the cart.

**Validation**

Both the product and user will be validated by comparing them to the specific credentials stored in the database.

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product Quantity:** Fields must be at least a single character. Field cannot be null.
* **Product Price**: Field is designated a decimal type and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **User ID:** Must be unique and field cannot be null.

**USE CASE #3 User Reserves Item from the Shopping Cart**

The user will be able to reserve products from the shopping cart. The user must be logged in to buy or reserve products with the cart.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product Quantity:** Integer representing the number of products available.
* **Product Price**: Decimal representing the cost of a single product.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **User ID:** Unique string representing a user.

**Output:**

Upon successful reservation of product in cart, the system will display a success message. If the product is not in stock it will return a warning message.

**Possible messages:**

Successful: “[Product name] Reserved!”

Failed: “No stock available for [Product name], sorry! Please check back again when we restock.”

**Actions**

Upon reserving product, the system will validate the stock before the reservation of the product. System will also decrement stock when an item is added to the cart. System must make sure that item(s) does not fall below zero in the database. When updating cart quantity, the system must update cart quantity and send a confirmation output to the user of success or failure.

**Pre and Post Conditions**

**Pre-conditions:** If user tries to reserve item(s) to the cart, the database checks if item is available.

**Post-conditions:** The item(s) will be reserved, and the quantity will be decremented in the database.

**Validation**

Validation for reservation of product will require the system to validate the product and user fields before reservation can be completed of the product.

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product Quantity:** Fields must be at least a single character. Field cannot be null.
* **Product Price**: Field is designated a decimal type and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **User ID:** Must be unique and field cannot be null.

**FEATURE: Item Catalog**

**USE CASE #1 System Admin Adds/Updates Item to the Product Catalog**

The system admin will be able to add or update products to the product catalog. User(s) are not able to modify the item catalog and can only view it. User(s) do not need to be logged in to access the item catalog.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product Quantity:** Integer representing the amount of products available.
* **Product Price**: Decimal representing the cost of a single product.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **Admin ID:** Unique integer representing the system admin.

**Output:**

Upon successful addition of product to the item catalog, the system will display a success message. If the product already exists in the item catalog, it will return a warning message that the product exists. When product is updated it will generate a successful message, otherwise it will generate a failure message if product does not exist in the database.

**Possible messages:**

Successful: “[Product name] Added to the Item Catalog!”

Successful: “[Product name] updated!”

Failed: “[Product name] already exist!”

Failed: “Could not find [Product name], please try again.”

**Actions**

Upon adding or updating product to the product catalog, the system/system admin will validate

If adding/updating of product is completed. System will also confirm if the system administrator added/updated product information. System is also able to append adding product and returning confirmation message upon having actions completed. System/System Administrator is able to add/update general specification of products and return confirmation message upon having actions completed.

**Pre and Post Conditions**

**Pre-conditions:** Product should not exist in the database/webpage when adding the product. The product must already exist when updating the product and the specification of the product must be correct according to the product.

**Post-conditions:** If product is not in database/web page then product will be displayed, otherwise if the product does exist then the product will be updated. If product specification is correct, then product will be displayed to the web page.

**Validation**

The following validations will be performed on the submitted data:

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product Quantity:** Fields must be at least a single character. Field cannot be null.
* **Product Price**: Field is designated a decimal type and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **Admin ID:** Must be unique and field cannot be null.

**USE CASE #2 System Admin Deletes Item from the Product Catalog**

The system admin will be able to delete products from the product catalog. User(s) are not able to modify the item catalog and can only view it. User(s) do not need to be logged in to access the item catalog.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product Quantity:** Integer representing the number of products available.
* **Product Price**: Decimal representing the cost of a single product.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **Admin ID:** Unique integer representing the system admin.

**Output:**

Upon successful deletion of product to the item catalog, the system will display a success message. If the product does not exist in the item catalog, it will return a warning message that the product does not exist.

**Possible messages:**

Successful: “[Product name] Added to the Item Catalog!”

Successful: “[Product name] updated!”

Failed: “Failed to delete [Product name]! [Product name] does not exist!”

**Actions**

Upon deletion of product to the product catalog, the system/system admin will validate

If deletion of product is completed. System will confirm if deleting of product(s) is completed and if System Administrator deleted product information. System/System Administrator is able to delete general specification of products and return confirmation message upon having actions completed.

**Pre and Post Conditions**

**Pre-conditions:** Check if item is available in database/web page.

**Post-conditions:** If item is available in database/web page, then delete it.

**Validation**

Both the product and admin credentials will be validated by comparing them to the information stored in the database.

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product Quantity:** Fields must be at least a single character. Field cannot be null.
* **Product Price**: Field is designated a decimal type and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **Admin ID:** Must be unique and field cannot be null.

**USE CASE #3 User views Item(s) from the Product Catalog**

Only the system admin will be able to modify products from the product catalog. User(s) are not able to modify the item catalog and can only view it. User(s) do not need to be logged in to access the item catalog.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product Quantity:** Integer representing the number of products available.
* **Product Price**: Decimal representing the cost of a single product.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.

**Output:**

Upon successful input of the product by the user, the product page will be displayed by the system to the user.

**Possible messages:**

None

**Actions**

None

**Pre and Post Conditions**

**Pre-conditions:** Product must be linked to correct web page.

**Post-conditions:** If Product is linked to correct web page, then webpage will be displayed on site.

**Validation**

The following validations will be performed on the submitted data:

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product Quantity:** Fields must be at least a single character. Field cannot be null.
* **Product Price**: Field is designated a decimal type and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.

**FEATURE: Review and Q&A**

**USE CASE #1 User Post Review**

The user will be allowed to post a review about a product after the user has bought the item and it has shipped successfully. It is up to the system admin whether to approve or disapprove the review to prevent spam content. The user must be logged in to post the review.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product Quantity:** Integer representing the number of products available.
* **Product Price**: Decimal representing the cost of a single product.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **Admin ID:** Unique integer representing the system admin.
* **User ID:** Unique integer representing the user.
* **Order ID:** Unique integer representing the order number.

**Output:**

Upon successful posting of product review the user will get a message that the review is sent to the system admin for review. The warning message will be generated if the user has not filled in all the fields.

**Possible messages:**

Successful: “[Product name] Review queued for System Admin approval!”

Successful: “[Product name] review updated! Review queued for System Admin approval!”

Failed: “Please fill out all fields, please try again.”

**Actions**

Upon a user posting a review about a product, the system/system admin will validate if the user’s item has been shipped successfully and then dispatch an email for a review of the product.

**Pre and Post Conditions**

**Pre-conditions:** If user has bought item which has shipped successfully, then user can make review post.

**Post-conditions:** User review post will be accepted and approved by system admin if content is on-topic.

**Validation**

The following validations will be performed on the submitted data:

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product Quantity:** Fields must be at least a single character. Field cannot be null.
* **Product Price**: Field is designated a decimal type and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **Admin ID:** Must be unique and field cannot be null.
* **Order ID:** Must be unique and field cannot be null.
* **User ID:** Must be unique and field cannot be null.

**USE CASE #2 User Post Question**

The user will be allowed to post a question about a product. It is up to the system admin whether to approve or disapprove the question to prevent spam content. The user must be logged in to post the question.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **Admin ID:** Unique integer representing the system admin.
* **User ID:** Unique integer representing the user.

**Output:**

Upon successful posting of product question, the user will get a message that the question is sent to the system admin for review. The warning message will be generated if the user has not filled in all the fields.

**Possible messages:**

Successful: “[Product name] question posted. It has been queued for System Admin approval!”

Successful: “[Product name] question updated! Question has been queued for System Admin approval!”

Failed: “Please fill out all fields, please try again.”

**Actions**

Upon a user posting a question about a product, the system/system admin will validate if the user’s question is on a topic about the product. If the user’s question is on-topic, then the system administrator will approve of the question. If the question is off-topic or deemed to be spam, then the system administrator will disapprove the question.

**Pre and Post Conditions**

**Pre-conditions:** If user’s question is on-topic about product.

**Post-conditions:** User question will be accepted and approved by system admin if content is on-topic.

**Validation**

The following validations will be performed on the submitted data:

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **Admin ID:** Must be unique and field cannot be null.
* **User ID:** Must be unique and field cannot be null.

**USE CASE #3 User Post Answer**

The user will be allowed to answer a question about a product. It is up to the system admin whether to approve or disapprove the answer to prevent spam content and correct product answer. The user must be logged in to post the answer.

**INPUT & OUTPUT**

**Input:**

* **Product Name**: String representing the name of the product. Minimum of 1 and maximum of 50 characters.
* **Product MPN:** String representing manufacturer part number. Minimum of 1 and maximum of 50 characters.
* **Product ID:** Unique integer representing the product.
* **Admin ID:** Unique integer representing the system admin.
* **User ID:** Unique integer representing the user.

**Output:**

Upon successful posting of product answer the user will get a message that the answer is sent to the system admin for review. The warning message will be generated if the user has not filled in all the fields.

**Possible messages:**

Successful: “[Product name] answer posted. It has been queued for System Admin approval!”

Successful: “[Product name] answer updated! Answer has been queued for System Admin approval!”

Failed: “Please fill out all fields, please try again.”

**Actions**

Upon a user posting an answer about a product, the system/system admin will validate if the user’s answer is on a topic about the product and is the correct answer. If the user’s answer is correct and on-topic, then the system administrator will approve of the answer. If the answer is wrong and off-topic or deemed to be spam, then the system administrator will disapprove the answer.

**Pre and Post Conditions**

**Pre-conditions:** If user’s answer is correct and on-topic.

**Post-conditions:** User answer will be accepted and approved by system admin if content is on-topic.

**Validation**

The following validations will be performed on the submitted data:

* **Product Name**: Field must be at least the minimum specified length and cannot be null.
* **Product MPN:** Field must be at least the minimum specified length and no more than the maximum length. Field cannot be null.
* **Product ID:** Must be unique and field cannot be null.
* **Admin ID:** Must be unique and field cannot be null.
* **User ID:** Must be unique and field cannot be null.

**Component Design**

**Feature: Shopping Cart**

**Entities**

**User**: The User class represents only the customer. It is the base class for all user information. This class is responsible for managing the user personal information as well as defining the formatting of names.

User Attributes/Fields:

* User\_id: unique identifier for a single user instance
* user\_username: a unique string chosen by the user to gain access to the system
* User\_pw: a string representing the password selected by the user to gain access to the Web site. It should be no less than 22 characters
* user\_fName: The first name of the user
* user\_lName: The last name of the user
* user\_phone: The phone number of the user
* user\_email: a string representing a valid email address in the form of [account]@[domain].[suffix]
* user\_address: The address of the user
* user\_city: The city the user is located in
* user\_state: The state of origin of the user. It should be no less than 2 characters.
* user\_zipcode: The zipcode of the user

Relationships:

* Product: Each user has a one to many relationships with product entity.
* Cart: Each user has a one to one relationship with the cart entity.

**Product**:  Entity that shows information about the item to the user and system admin of the website. Product entity can interact with the cart entity when stored in the cart.

**Cart**: This entity will be responsible for containing the item that the user has selected to buy, reserve or keep in the cart.

**Product**: The Product class represents products/items in the website to sell. It is the base class for all product information. This class is responsible for managing the product information as well as defining the formatting of names.

Product Attributes/Fields:

* prod\_id: unique identifier for a single product instance
* prod\_name: The product name
* prod\_price: The product price
* prod\_quantity: The number of products available
* prod\_MPN: A unique identifier created by the manufacturer

Relationships:

* User: Each user has a one to many relationships with the product entity.
* Cart: This entity shared a one to one relationship the user entity and the product entity.

**User**: Entity that identifies one the customers of the website. Possible options might be “customer” and, “guest”

**Cart**: This entity will be responsible for containing the item that the user has selected to buy, reserve or keep in the cart.

**Cart**: The Cart class represents the cart entity. It is the base class for all cart information. This class is responsible for managing the cart information as well as defining the formatting of names.

Cart Attributes/Fields:

* prod\_id: unique identifier for a single Product instance
* user\_id: unique identifier for a single User instance

Relationships:

* Product: Has a many to one relationship with the cart entity.
* User: Each user has a one to many relationships with the product entity.

**Product**: Entity that shows information about the item to the user and system admin of the website. Product entity can interact with the cart entity when stored in the cart.

**User**: Entity that identifies one the customers of the website. Possible options might be “customer” and, “guest”.

**Feature: Item Catalog**

**Entities**

**Admin**: The Admin class represents the system admin entity. It is the base class for all admin information. This class is responsible for managing the admin and product personal information as well as defining the formatting of names.

Admin Attributes/Fields:

* admin\_id: unique identifier for a single admin instance
* admin\_username: a unique string chosen by the admin to gain access to the system
* admin\_pw: a string representing the password selected by the user to gain access to the Web site. It should be no less than 22 characters
* admin\_phone: The phone number of the admin
* admin\_email: a string representing a valid email address in the form of [account]@[domain].[suffix]

Relationships:

* Product: Each admin has a one to many relationship with the product entity.

**Product**: Entity that shows information about the item to the user and system admin of the website. Product entity can interact with the System Admin entity when manipulated by getting added, deleted and updated.

**Product**: The Product class represents products/items in the website to sell. It is the base class for all product information. This class is responsible for managing the product information as well as defining the formatting of names.

Product Attributes/Fields:

* prod\_id: unique identifier for a single product instance
* prod\_name: The product name
* prod\_price: The product price
* prod\_quantity: The amount of products available
* prod\_MPN: A unique identifier created by the manufacturer

Relationships:

* Admin: Each product has a many to one relationship with the admin entity.

**Admin**: The entity that identifies the system admin who manages the product entity by adding, deleting and updating products. It also manages the review entity by approving and disapproving reviews.

**Feature: Review**

**Entities**

**User**: The User class represents only the customer. It is the base class for all user information. This class is responsible for managing the user personal information as well as defining the formatting of names.

User Attributes/Fields:

* User\_id: unique identifier for a single user instance
* user\_username: a unique string chosen by the user to gain access to the system
* User\_pw: a string representing the password selected by the user to gain access to the Web site. It should be no less than 22 characters
* user\_fName: The first name of the user
* user\_lName: The last name of the user
* user\_phone: The phone number of the user
* user\_email: a string representing a valid email address in the form of [account]@[domain].[suffix]
* user\_address: The address of the user
* user\_city: The city the user is located in
* user\_state: The state of origin of the user. It should be no less than 2 characters.
* user\_zipcode: The zipcode of the user

Relationships:

* Product: Each user has a one to many relationship with product entity.
* Cart: Each user has a one to one relationship with the cart entity.

**Product**: Entity that shows information about the item to the user and system admin of the website. Product entity can interact with the cart entity when stored in the cart.

**Cart**: This entity will be responsible for containing the item that the user has selected to buy, reserve or keep in the cart.

**Admin**: The Admin class represents the system admin entity. It is the base class for all admin information. This class is responsible for managing the admin and product personal information as well as defining the formatting of names.

Admin Attributes/Fields:

* admin\_id: unique identifier for a single admin instance
* admin\_username: a unique string chosen by the admin to gain access to the system
* admin\_pw: a string representing the password selected by the user to gain access to the Web site. It should be no less than 22 characters
* admin\_phone: The phone number of the admin
* admin\_email: a string representing a valid email address in the form of [account]@[domain].[suffix]

Relationships:

* Product: Each admin has a one to many relationship with the product entity.

**Product**: Entity that shows information about the item to the user and system admin of the website. Product entity can interact with the cart entity when stored in the cart.

**Product**: The Product class represents products/items in the website to sell. It is the base class for all product information. This class is responsible for managing the product information as well as defining the formatting of names.

Product Attributes/Fields:

* prod\_id: unique identifier for a single product instance
* prod\_name: The product name
* prod\_price: The product price
* prod\_quantity: The amount of products available
* prod\_MPN: A unique identifier created by the manufacturer

Relationships:

* Admin: Each product has a many to one relationship with the admin entity.

**Admin**: The entity that identifies the system admin who manages the product entity by adding, deleting and updating products. It also manages the review entity by approving and disapproving reviews.

**Review**: The Review class represents reviews posted by the user entity and approved or disapproved by the admin entity. It is the base class for all review information. This class is responsible for managing the review information as well as defining the formatting of names.

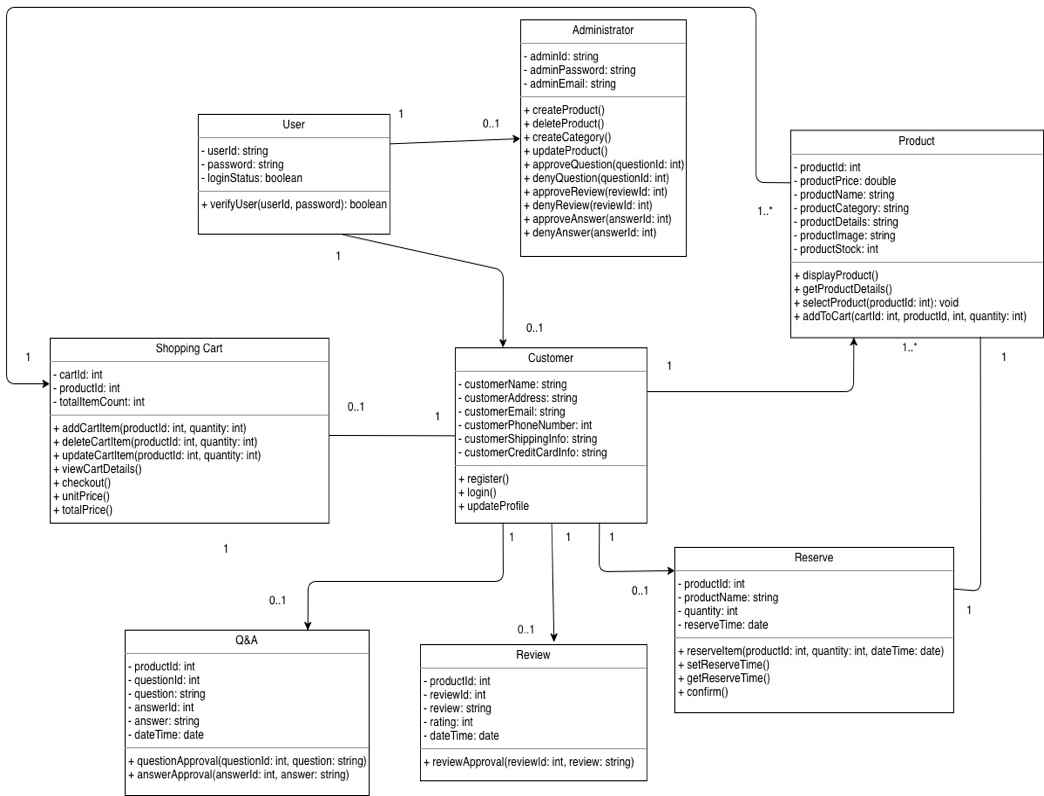
Review Attributes/Fields:

* User\_id: unique identifier for a single user instance
* prod\_id: unique identifier for a single product instance
* order\_id: gives confirmation of item that was bought and shipped successfully. Prerequisite to making a review
* Review\_RatingAvg: average rating of a product entity

Relationships:

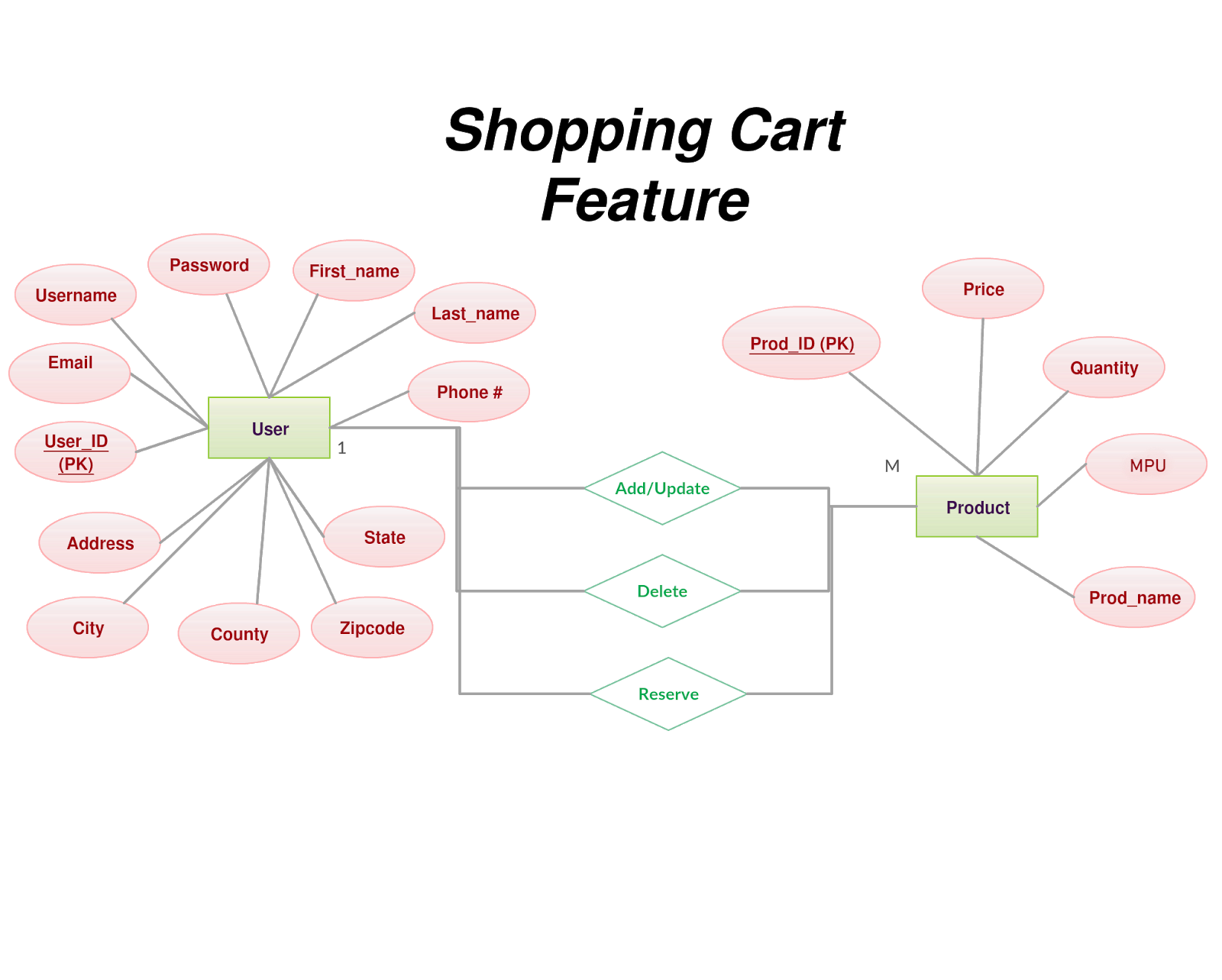
* Admin: Each product has a many to one relationship with the admin entity.

**Admin**: The entity that identifies the system admin who manages the product entity by adding, deleting and updating products. It also manages the review entity by approving and disapproving reviews.

* **

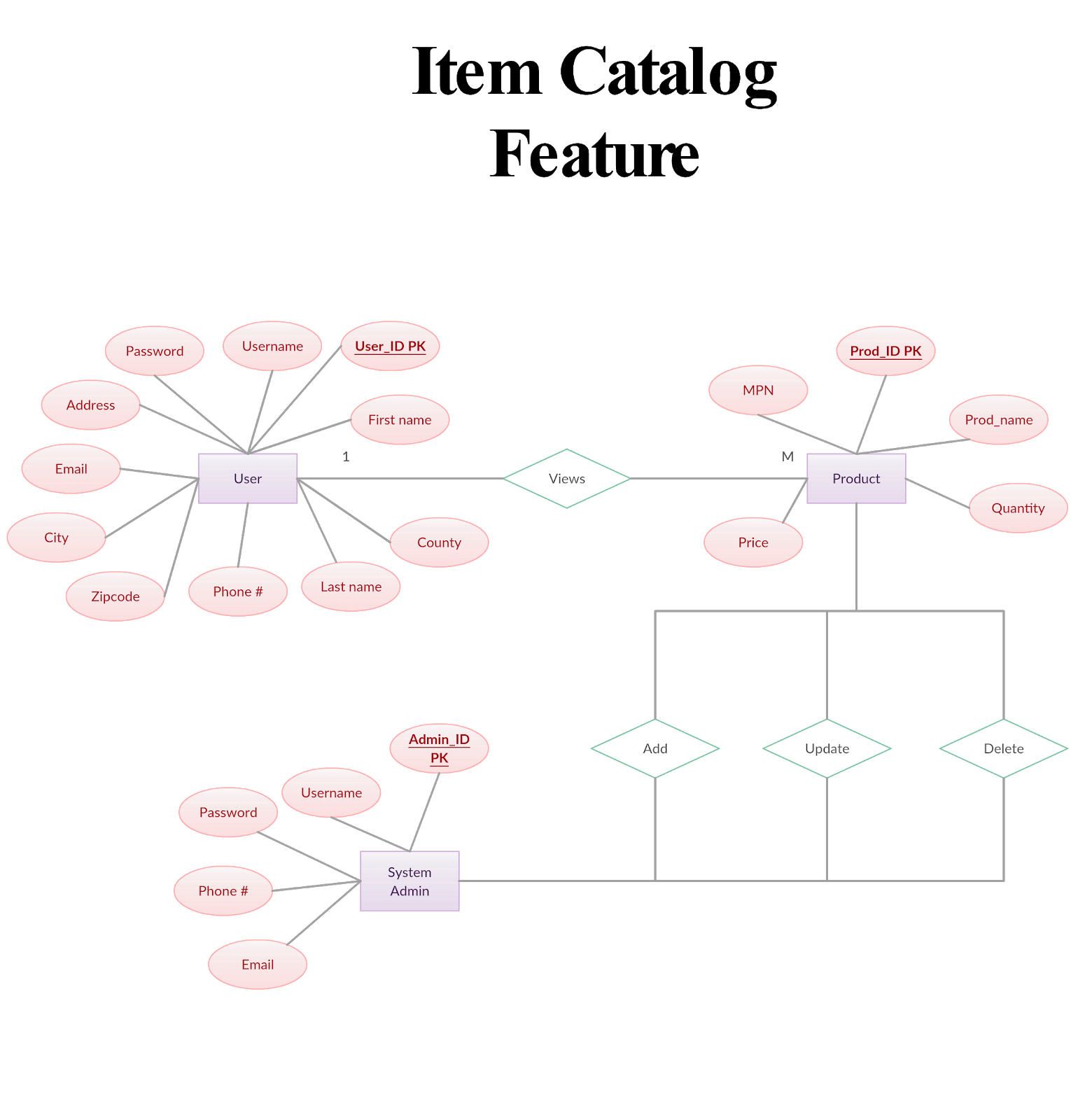
**Data Design**

**Data Description**

**

**Shopping Cart Feature**

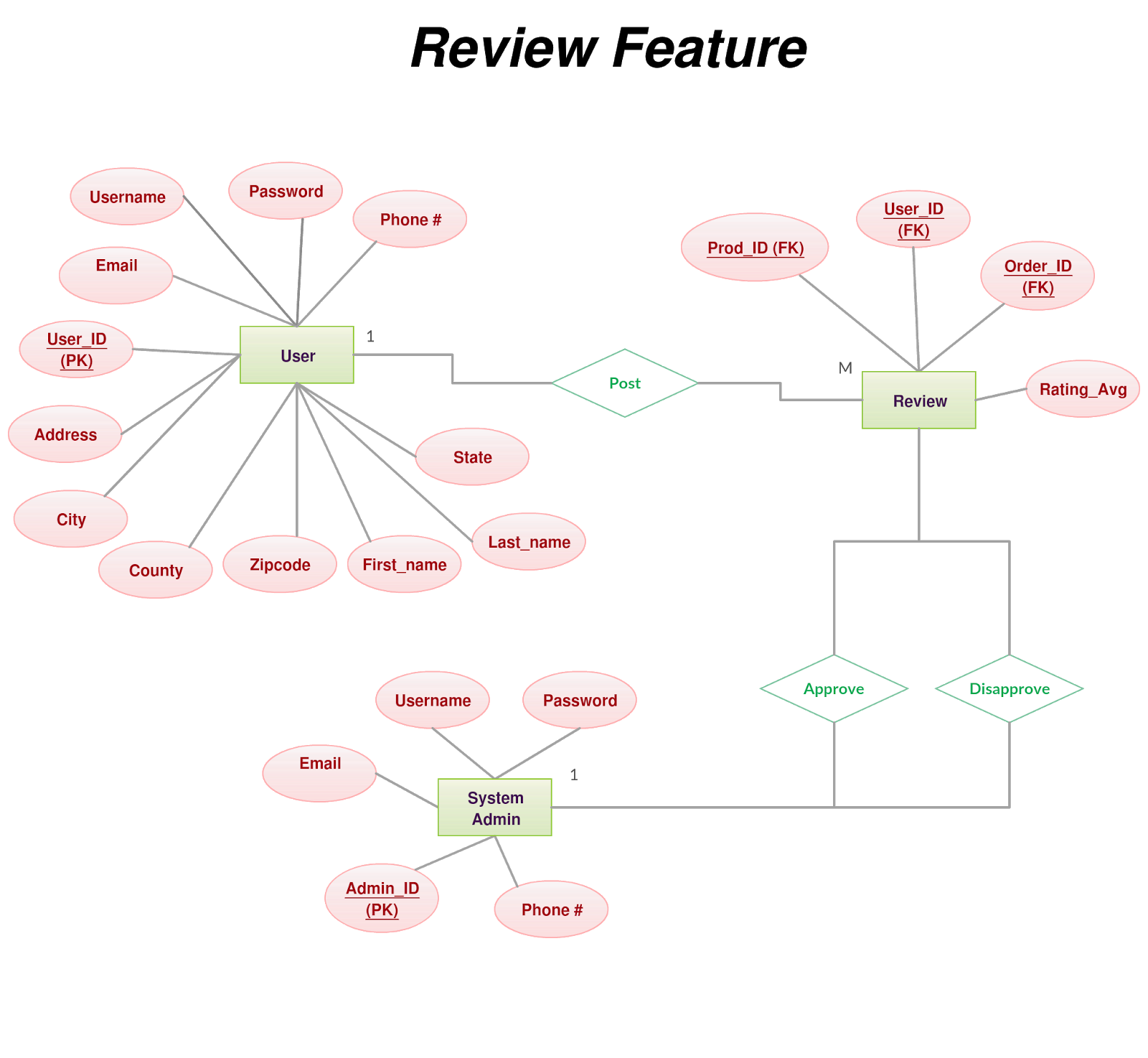
From the above diagram, we identified a relationship between a user and product. The types will be User and Product. The relationships “Add”, “Update” and “Delete” include none specific attributes.



**Item Catalog Feature**

From the above diagram, we identified a relationship between a user and product. The possible types will be User and Product. The relationships “Add”, “Update” and “Delete” do not include any specific attributes.

We also identified a relationship between System Admin and Product. The relationship “Adds” between System Admin and Product does not include any specific attributes.



**Review Feature**

From the above diagram, we identified a relationship between a user and a review. The possible types will be User and Review. The relationships “Post” does not include any specific attributes.

We also identified a relationship between System Admin and Review. The relationship “Approve” and ”Disapprove” between System Admin and Product does not include any specific attributes.

**DATABASE Dictionary / SCHEMA**

**Shopping Cart Feature**

The following tables will be used for the Add/Update, Delete and Reserve Cart module:

Cart, User and Product table as shown below.

**Item Catalog Feature**

The following tables will be used for the Add/Update, Delete module:

Admin and Product table as shown below.

The following tables will be used for the View module:

Product table as shown below.

**Review Feature**

The following tables will be used for the Post Review module:

User, Product, Order and Admin table as shown below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Cart Table** |  |  |  |
| **Entity** | **Type** | **Value** | **Description** |
| **prod\_id** | INT | Not Null | FK. product identifier. Relationship key to product table |
| **user\_ID** | INT | Not null | FK. user identification and relationship key to user table |

|  |  |  |  |
| --- | --- | --- | --- |
| **Product Table** |  |  |  |
| **Entity** | **Type** | **Value** | **Description** |
| **prod\_id** | INT | Not Null | PK. Unique identifier of a product. |
| **prod\_price** | Decimal(10, 2) | Not null | The price of the product |
| **prod\_quantity** | INT | Not null | The quantity of products |
| **prod\_MPN** | varchar(50) | Not null | Manufacturer's identification of the product |
| **prod\_name** | varchar(50) | Not null | The name of the product |

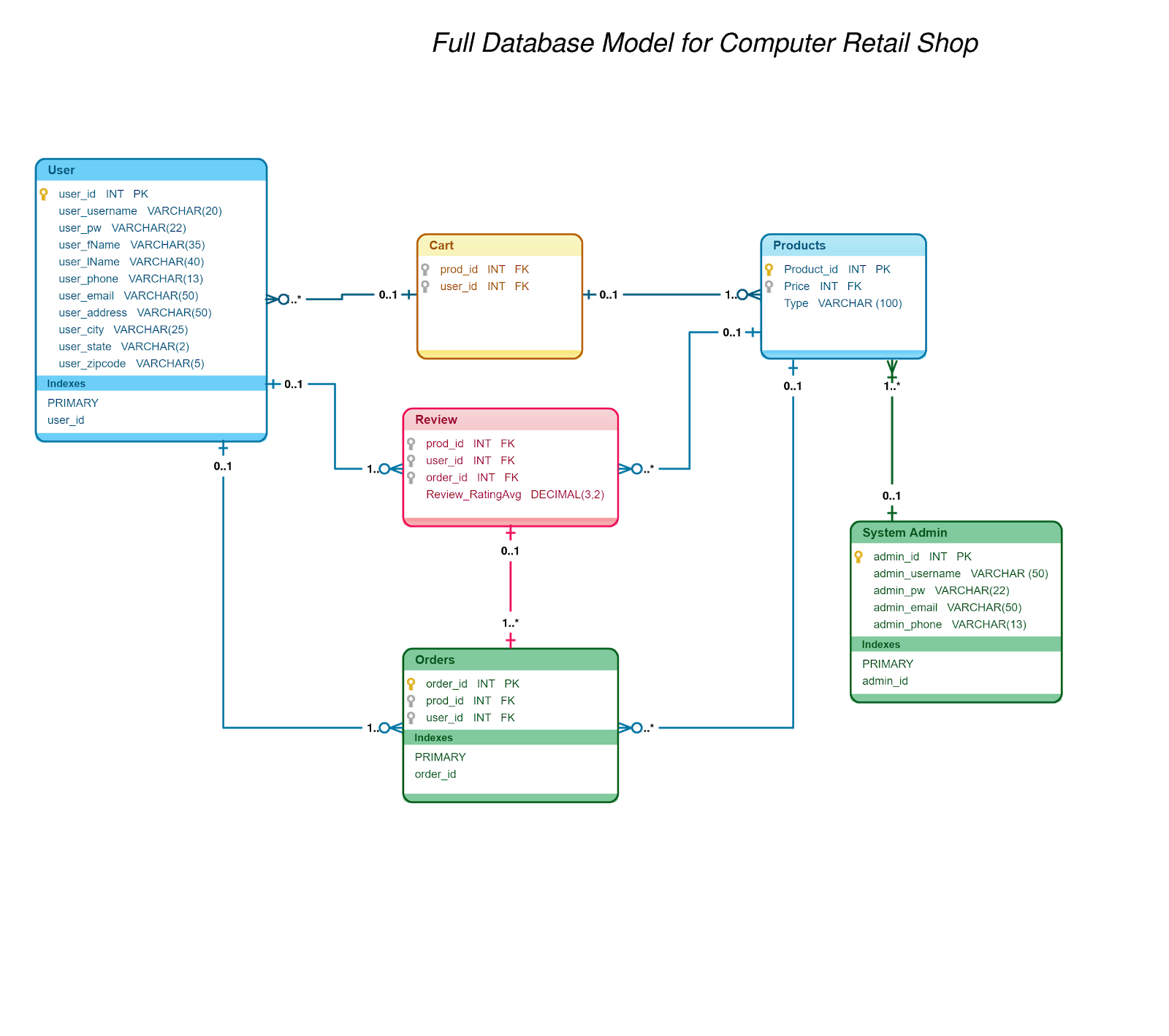
|  |  |  |  |
| --- | --- | --- | --- |
| **User Table** |  |  |  |
| **Entity** | **Type** | **Value** | **Description** |
| **user\_id** | INT | Not null | PK. Unique identifier of a user. |
| **user\_username** | varchar(20) | Not Null | User’s username |
| **user\_pw** | varchar(22) | Not null | User’s password |
| **user\_fName** | varchar(35) | Not null | User’s first name |
| **user\_lName** | varchar(40) | Not null | User’s last name |
| **user\_phone** | varchar(13) | Not null | User’s phone number |
| **user\_email** | varchar(50) | Not null | User’s Email address |
| **user\_address** | varchar(50) | Not null | User’s home address |
| **user\_city** | varchar(25) | Not null | City of the user |
| **user\_state** | varchar(2) | Not null | State |
| **user\_zipcode** | varchar(5) | Not null | Zip code for the area of the user |

|  |  |  |  |
| --- | --- | --- | --- |
| **Review Table** |  |  |  |
| **Entity** | **Type** | **Value** | **Description** |
| **prod\_id** | INT | Not Null | FK. Relationship key to the product table |
| **user\_id** | INT | Not null | FK. Relationship key to the user table |
| **order\_id** | INT | Not null | FK. Relationship key to the order table |
| **Review\_RatingAvg** | DECIMAL(3, 2) | Not null | The average rating of the product |

|  |  |  |  |
| --- | --- | --- | --- |
| **Order Table** |  |  |  |
| **Entity** | **Type** | **Value** | **Description** |
| **order\_id** | INT | Not Null | PK. Unique identifier of a order of the product. |
| **prod\_id** | INT | Not null | FK. relationship key to the product table |
| **user\_id** | INT | Not null | FK. relationship key to the user table |

|  |  |  |  |
| --- | --- | --- | --- |
| **System Admin**  **Table** |  |  |  |
| **Entity** | **Type** | **Value** | **Description** |
| **admin\_id** | INT | Not Null | PK. Unique identifier of an admin. |
| **admin\_username** | varchar(50) | Not null | The name of the admin |
| **admin\_pw** | varchar(22) | Not null | Password for the admin |
| **admin\_email** | varchar(50) | Not null | Admin’s Email Address |
| **admin\_phone** | varchar(13) | Not null | Admin’s phone number |

**Full Database Model**

****

**Requirements Matrix**

|  |  |  |
| --- | --- | --- |
| ***Module*** | ***Use Case*** | ***Design Component*** |
| ***Shopping Cart*** | *User adds/updates Product to cart* | *User Interface: User view of cart ERD Diagram 11*  *Component Model: Shopping Cart*  *DB Schema section7* |
|  | *User deletes/updates Product to cart* | *User Interface: User view of cart ERD Diagram 11*  *Component Model: Shopping Cart*  *DB Schema section7* |
|  | *User reserves Product to cart* | *User Interface: User view of cart ERD Diagram 11*  *Component Model: Shopping Cart*  *DB Schema section7* |

|  |  |  |
| --- | --- | --- |
| ***Module*** | ***Use Case*** | ***Design Component*** |
| ***Item Catalog*** | *System Admin adds/updates Product to product catalog* | *User Interface: Admin view of Product catalog ERD Diagram 13*  *Component Model: Item Catalog*  *DB Schema section7* |
|  | *System admin deletes product from product catalog* | *User Interface: Admin view of Product catalog ERD Diagram 13*  *Component Model: Item Catalog*  *DB Schema section7* |
|  | *User views Product from item catalog* | *User Interface: User view of Product Catalog ERD Diagram 12*  *Component Model: Item Catalog*  *DB Schema section7* |

|  |  |  |
| --- | --- | --- |
| ***Module*** | ***Use Case*** | ***Design Component*** |
| ***Review and Q&A*** | *User posts review about product* | *User Interface: User and system view of review ERD Diagram 13-14*  *Component Model: Review and Q&A*  *DB Schema section7* |
|  | *User post question about product* | *User Interface: User and admin view of question post ERD Diagram 15*  *Component Model: AReview and Q&A*  *DB Schema section7* |
|  | *User post answer about product* | *User Interface: User and admin view of answer post ERD Diagram 16*  *Component Model: Review and Q&A*  *DB Schema section7* |

**Appendices**

Database - An organized collection of data that is stored, accessed, modified and managed through a computer system.

Database Model - structural database design that shows the relationship between entities and attributes.

Flow chart - Represents a flow of data to visually present the process of the software.

System diagram - logical view of the system design.

UI interface - a design or mock ups that shows the user interaction with the system.