|  |
| --- |
| Mercer university |
| Project 2 |
| SSE 657 - Object Oriented Project Methods |
|  |
| **Tanya Do & John Robison** |
| **10/27/2014** |

|  |
| --- |
|  |

Table of Contents

[Introduction 1](#_Toc402107926)

[Problem Statement 2](#_Toc402107927)

[Discussion With Customer 2](#_Toc402107928)

[Commonality and Variability Analysis of System 4](#_Toc402107929)

[Features List 5](#_Toc402107930)

[Use Case Diagram 8](#_Toc402107931)

[Domain Analysis 9](#_Toc402107932)

[Design Pattern Selection 10](#_Toc402107933)

[Determining the Architectural Significance of Components 11](#_Toc402107934)

[Risk Analysis and Reduction 13](#_Toc402107935)

[Additional Discussion With Customer 14](#_Toc402107936)

[Design of Core Features and Use of Design Principles 14](#_Toc402107937)

[Home Page 15](#_Toc402107938)

[User Accounts 15](#_Toc402107939)

[Product Listings 15](#_Toc402107940)

[Mock Scenarios of System Interaction 15](#_Toc402107941)

[Account Registration 16](#_Toc402107942)

[Editing Account Information 20](#_Toc402107943)

[Creating a Product Listing 25](#_Toc402107944)

[Editing a Product Listing 29](#_Toc402107945)

[Purchasing a Product With an Account 33](#_Toc402107946)

[Purchasing a Product Without an Account 41](#_Toc402107947)

[Appendix A: References 43](#_Toc402107948)

**Table of Figures**

[Figure 1: The SparkMacon Logo 2](#_Toc402107949)

[Figure 2: MakerStore Use Case Diagram 9](#_Toc402107950)

[Figure 3: MVC Interaction Diagram 10](#_Toc402107951)

[Figure 4: Home Page of the MakerStore 17](#_Toc402107952)

[Figure 5: The Register Button 18](#_Toc402107953)

[Figure 6: The Registration Page 19](#_Toc402107954)

[Figure 7: The Username After Creating an Account 20](#_Toc402107955)

[Figure 8: The Log In Button 21](#_Toc402107956)

[Figure 9: The Log In Screen 22](#_Toc402107957)

[Figure 10: The User Account Page 23](#_Toc402107958)

[Figure 11: The Account Information Edit Form 24](#_Toc402107959)

[Figure 12: Updated User Account Page 25](#_Toc402107960)

[Figure 13: The Add Product Button 26](#_Toc402107961)

[Figure 14: The Add Product Page 27](#_Toc402107962)

[Figure 15: The Completed Add Product Form 28](#_Toc402107963)

[Figure 16: The Added Product Listing 29](#_Toc402107964)

[Figure 17: The Edit Product Button 30](#_Toc402107965)

[Figure 18: The Edit Product Page 31](#_Toc402107966)

[Figure 19: The Completed Edit Product Form 32](#_Toc402107967)

[Figure 20: The Updated Product Listing 33](#_Toc402107968)

[Figure 21: The Shop Button 34](#_Toc402107969)

[Figure 22: The Shop Page 35](#_Toc402107970)

[Figure 23: The Product Listing Page 36](#_Toc402107971)

[Figure 24: The Add To Cart Button 37](#_Toc402107972)

[Figure 25: The Shopping Cart Page With Updated Cart Icon 38](#_Toc402107973)

[Figure 26: The Checkout Button 39](#_Toc402107974)

[Figure 27: The Checkout Screen 40](#_Toc402107975)

[Figure 28: The Checkout Confirmation Screen 41](#_Toc402107976)

[Figure 29: The Registration Page With the Updated Cart Icon 42](#_Toc402107977)

# Introduction

This report and the project that it corresponds to are intended to display mastery of the concepts contained in Chapters 6 - 8 of the book Head First Object-Oriented Analysis and Design by McLaughlin, Pollice, and West. These chapters build on the previous chapters by explaining how those Object-Oriented Analysis and Design (OOAD) concepts can be applied to larger, real-world problems to make them more manageable to design.

The first step offered by the text to make a large problem less intimidating to approach is to break the project into smaller, logical pieces that can be developed using the OOAD concepts provided in the previous chapters. To determine these logical pieces, techniques such as conversations with the customer and commonality and variability analysis can be used to determine the features of the system. Once the features are determined, the most architecturally significant features are determined using the three Q's of architecture and are then designed first to reduce risk. After the architectural design of the system is completed, design principles are applied to ensure that the system is not rigid, fragile, or immutable.

# Problem Statement

With the opening of SparkMacon, Macon's own Maker Space, there is now a need for a web service that will allow makers of all trades and skill sets to advertise their products and promote their business. This service will allow makers to create a profile that contains: a biography describing their craft, previously completed or sold projects, products currently for sale, user reviews of the maker's products, and a place for users to request unlisted or new products. The people behind SparkMacon are devoted to building a strong maker community in the Macon area and want a system that helps to push their makers further. To build this community that SparkMacon aspires to, the web service will need features that promote community involvement: featured makers (i.e. Maker of the Week), community forums, the ability to like or follow products and users, etc.

# Discussion With Customer



Figure : The SparkMacon Logo

Spark Macon reached out to us, J & T Development, Inc., to create a fully functional web store and market place web application. They sent us an email expressing their need for a web application that will contribute to and serve the Spark Macon community. In order to fully understand the needs of Spark Macon, J & T Development decided we needed to sit down with them to talk about the minutia of the project. We scheduled a meeting with Spark Macon representatives, Michael Rose and Bob Martin. We discussed the scope of the project and the features that the project would have.

**J & T Development, Inc.:** Hey guys. We saw your email and are really interested in what you are doing. Can you tell us more about your company and what it is that you want us to do for you?

**Michael Rose:** So Spark Macon is a makerspace and will have different types of people creating things, and innovating.

**Bob Martin:** Like artists and tech people.

**Michael Rose:** Yeah, and we want to help expand the local economy by setting up a place for our members to sell their products.

**Bob Martin:** But since we are a makerspace and not a flea market, we want this to take place online. But we still want to support our members.

**J & T Development, Inc.:** Okay, cool. So like, you want a web application that works like EBay?

**Bob Martin:** Sort of. We want people to be able to buy and sell. But the things must be handmade, like created through Spark Macon.

**Michael Rose:** Yes. More similar to the Etsy web application. Which stipulates that all items sold must be handmade.

**Bob Martin:** We also want this site to be specific to Macon, though.

**J & T Development, Inc.:** Okay, great. So what type of features does this application need?

**Bob Martin:** It definitely needs to have individual user accounts. We want for each user to essentially have their own mini store on the application.

**Michael Rose:** Right, I agree. Additionally, the user profile should have an "about me" section. Where they can describe themselves and their philosophies to potential buyers.

**Bob Martin:** Also, we'd like for users to have to capability of rating users and their products.

**Michael Rose:** Yes. We'd like to have some standard of integrity for sellers and buyers.

**Bob Martin:** Also, we should be able to allow potential customers to request types of items they'd like to see in the future. But I don't know. Maybe that is a bonus feature. And not something we absolutely need. But something that would be pretty cool.

**Michael Rose:** I think another feature we'd like is a products page for each profile. Where the user can display what they are selling. With pictures and the listed price. And we can also have a product review on there as well.

**Bob Martin:** And we also want a home page where we can have feature sellers.

**Michael Rose:** And products too!

**Bob Martin:** Yes. We want to create an online community as well. So we'd like to have an about me section describing our mission as Spark Macon as well as a blog type section where we keep our community updated on things going on. Like a news section.

**J & T Development, Inc.:** Yeah, that is definitely do-able. We will have to do some internal evaluation about how to approach the project from a technical standpoint. We will get back to you after our analysis.

**Bob Martin:** Thank you so much for your time. We are excited to work with you on our new venture.

**J & T Development, Inc.:**  Yeah, definitely. We are so glad that you chose us to join you on your project.

# Commonality and Variability Analysis of System

In order to ensure that this web service meets all of the customer's expectations, our team performed a commonality and variability analysis on the system. This means that we compared the proposed systems to preexisting web services and described what these systems offered or did that the customer either wanted their system to be like or not be like. In addition to ensuring that the designed system meets the customer's expectations, this commonality and variability analysis will aid our team in determining the major features of the system which will allow us to determine the most architecturally significant components.

Based on the discussion with the customer, the core functionality of the web service should be much like Etsy, an e-commerce marketplace for creators of unique items to sell their goods. The customer stressed that they wanted posted items to be actually made by the users that posted the listings, so the web service should not be like Amazon or Ebay. Since the customer is interested in strengthening the Macon community, the system should not be international or even open to anyone like Etsy, Amazon, and Ebay are. The area-based product listing system used by web services such as Craigslist is ideal for building this strengthened community of Macon makers.

# Features List

After our domain analysis we have compiled a list which conveys the complete application in a succinct and easy to digest manner. Listed below are the features that were determined as well as the requirements that make up each feature:

* User Accounts
  + User biography
  + List of products available
  + List of previously sold products
  + Product Requests
  + User Rating/ Review
* Product Listings
  + Product description
  + Product Category
  + Price
  + Product Rating/ Review
* Home Page
  + Maker of the Week
  + Featured Products
  + News
  + About
  + Link to Spark Macon
* Community Involvement
  + Community forum
  + Event coordination/ planning
  + Liking or following products or users

The User Account feature will be responsible for storing all of a user's information: biography, products, user rating, and requests. A user will be able to create an account at any time and begin selling and marketing their products. Once an account is created, the user will be able to login to the system and edit/ update their account information as well as create postings for products.

* User biography: The user biography is a place on the user’s profile where the user can talk about themselves and/or their store. The purpose of this section is so the members of SparkMacon can express themselves and connect with their fellow members.
* List of products available/Webstore: Each profile will have the option to be associated with a list of products. Those members who want to sell as well as buy, will have the option to open a personal store. The products can be categorized and sorted by price.
* List of previously sold products: There will be a section where a seller’s sold products will be listed. Unlike the list of products, previously sold products will be organized chronologically by date sold.
* Product Request: This feature allows customers to submit request for products they want to see in the future. This will allow for the sellers to have a better grasp of what their customers want, and what to produce. And this will allow for interaction between sellers and customers.
* User Rating/Review: The application will allow for users to rate users as a seller or buyer, and provide a review of the person’s habits. The purpose of this is to provide transparency between members of the community.

The Product Listing will contain all of the information for a user's product: product description, category, price, and product reviews. A user with a registered account will be able to post listings for their products. The use of the description and category will allow other users to search for products that interest them, such as robotics or art.

* Product description: Each product will have its own section or page where the seller can upload a picture of the product and fully describe the product offered.
* Product category: Each product can be assigned to a product category. On the seller’s webstore page, the products offered can be sorted by product category. Additionally, the products can be filtered by product category.
* Price: Each product will have a price associate with them.
* Product Rating/Review: Users will be able to provide ratings and reviews of products they buy from a maker. The purpose of this is to encourage sellers to produce quality products and accurate description of the product sold.

The Home Page will be the main access point of the web service. It will be used to display important pieces of information, such as the featured Maker of the Week, upcoming events, and news about the makerspace. This page will also contain information about what SparkMacon's mission is as well as provide links to their website.

* Maker of the Week: We would like to feature a new Maker each week on the SparkMacon front page. This feature is to foster a community environment. And to help Makers who are talented in their chosen art, but less schooled in marketing and business.
* Feature Products: Additionally, the SparkMacon mission aims to bring light to products that are of high quality but not easily access via any search function. This feature also serves to foster a community environment.
* News: This section of the application will be similar to a blog. All news items will be organized chronologically. News items will be news about the SparkMacon events or announcements.
* About: This section will be explain the purpose of the SparkMacon Marketplace application and how it functions.
* Link to SparkMacon: There will be a fixed section that will link to the main SparkMacon website.

There will be several components that will aim to build community involvement, including user forums, announcing local makerspace events, and the ability to like or follow other users' accounts and products.

* Community forum: There will be a place for discussion between members of SparkMacon. The forum will allow members to communicate with each other about anything maker related. Additionally, users can use this space to discuss marketing techniques.
* Event coordination/planning: This section will be open for users to plan SparkMacon events remotely.
* Liking or following products or users: SparkMacon users have the option to like or follow products. This allows for users to create a wishlist of sorts and also get notifications whenever a seller puts up a new product.

# Use Case Diagram

The users should be able to interact with the system in many different ways. These interactions include:

* Creating and editing accounts
* Creating and editing product listings
* Purchasing products
* Contributing on forums
* Navigating to SparkMacon's website
* Rating users and products.

The account interactions will be handled through registration and log in processes. A new user will have to register with the system in order to have a username and password that they will be able to log in with. Once they are registered, they will be given the ability to edit their account information, such as their biography and available products.

Only registered users will be able to create and edit product listings. Once a user is registered, they will be able to post as many products as they have to sell. Once a product is listed, only the user who posted it will be able to make edits to the listing. These edits include the product's description and price.

Registered users will also be able to contribute to the community by rating other users and their products as well as commenting on forums. If a user is interested in learning more about the SparkMacon makerspace or learning about upcoming events, they will be able to click a link that navigates to the SparkMacon site regardless of if the user is registered.

Figure 2 below visualizes the users' interactions with the system in a UML Use Case Diagram.

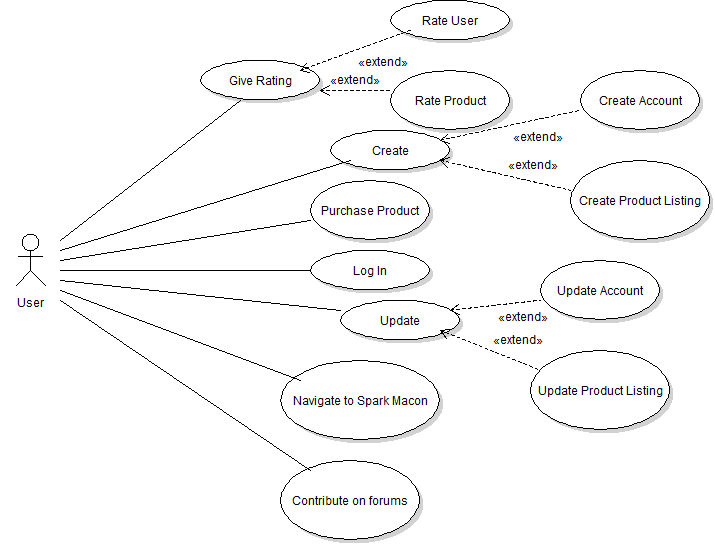


Figure : MakerStore Use Case Diagram

# Domain Analysis

We at J & T Development, Inc. want to be as transparent as possible with our clients. So while we have a detailed plan, use case and design pattern selection in place, we want to reframe the requirements of the web application in a way that our client, Spark Macon will clearly understand. J & T Development, Inc. pride ourselves on not only producing high quality products in an efficient manner, but also on our customer service and communication skills. We realize and understand, unlike our competitors, that our clients may not be able to fully digest documents such as use-cases and UML diagrams. So we go through a process of domain analysis where we study and analyze the system, and collect and organize the relevant information to the system. Then we represent it in a way that is easy to understand and digest for our clients. Our clients have on numerous occasion express appreciation for the way we present information to them. And it reduces confusion as well as ensure that both we and our clients are on the same page. And are talking about the same things.

We created a features list which allowed us to outline the application in its entirety, while expressing the functionalities of the application in a succinct manner.

Then we provide a detailed description of what each feature and sub-feature.

# Design Pattern Selection

The designed system will be built on the ASP.NET MVC Framework. The use of this framework utilizes the MVC (Model-View-Controller) design pattern which encourages designs to be loosely coupled through a separation of concerns. This separation is achieved by using the three main components: Models which handle the data and logic of the system, Views which display the data in a way that makes sense to the user, and Controllers which are responsible for dealing with user inputs by updating the models and view. Below in Figure 3 is a diagram that visualizes the interactions between these components.



Figure : MVC Interaction Diagram

The ASP.NET framework builds on the MVC design pattern by relying on a "convention over configuration" approach which reduces the amount of code required for a project. This approach requires certain design decisions, such that items are placed in the correct locations (i.e. Controllers must be in the Controller folder) and that they are named appropriately (i.e. Controllers will be named ending with "Controller"). Not only does this convention based approach reduce code requirements, it also aids in overall readability of the system.

Due to the loose coupling offered by the MVC design pattern, our team chose ASP.NET MVC as the framework for this system because it allowed for code reuse and parallel development. Since the models, views, and controllers are separated, they can easily be implemented into another application. An example of this would be if the customer requested a mobile application since the model and controller classes could be reused. The only major change in this example is that new views would have to be created specifically for the mobile platform. This separation also allows components to be designed and built separately. One developer can build the store user interface in the views while another developer designs the underlying business logic in the models.

In addition to all of the benefits of MVC's separation of concerns listed above, our team decided to develop with the ASP.NET MVC Framework for its use of .NET languages (C# and Visual Basic) which we are familiar with as well as the use of its easy to learn view engine, Razor. C#, our .NET language of choice, is extremely powerful and easy to develop with due to its number of available libraries and the use of Microsoft's Intellisense, which can auto-fill code for the developer such as variable and method names. The Razor view engine is a combination of HTML and either C# or Visual Basic, which makes it very natural for .NET developers to write and learn. The use of C# in the markup allows for very powerful dynamic web pages. Razor also allows the use of layouts which enables developers to have a single Razor file act as a template for all other views. These layouts reduce duplicate code by encapsulating common view elements into the layout. The final advantage of Razor that will aid in development is the ability to use Intellisense to quickly write the markup and code contained in the Razor files.

# Determining the Architectural Significance of Components

To select which feature to begin with, our team looked at each feature and asked the Three Q's of architecture:

* Is it the essence of the system?
* What does the feature mean?
* How do you implement the feature?

A feature is deemed architecturally significant if it is either the essence of the system, the meaning of the feature is unclear to the developers, the developers are unsure of how to implement the feature, or any combination of the previous three. Neither of the Three Q's carries more significance than the other and thus asking these questions only gives us an unweighted list of the most architecturally significant features.

The features that have been determined to be the essence of the system are: the user accounts, the product listings, and the home page. User accounts are the core of the system. Without users, new content cannot be generated and Macon area makers will not be able to display their crafts. Just like a makerspace without makers is just an empty building, a makerstore without makers is just a blank website. Adding to this point, a store without items is just an empty building. This means that the product listings are also essential to the system. However, the system can still operate to a certain extent with only user accounts. Makers will still be able to describe their craft and promote themselves without the ability to post listing for their products. Finally, the home page acts as the main access point of the system. It will contain a description of the site, links to all of the other features, and a link to SparkMacon's website. This access point is especially important for users who are not familiar with what the system's goals are or the makerspace that is supporting it.

The product listing was a feature that our development team needed clarification on to get a better understanding of. Some of the components of this feature that our team is unsure of are: product categories, listing formatting, and whether the customer would want similar products to be listed. Since product categories should not be something that a user can add or remove, there needs to be a set list of available categories that a product can be. Our team will need to be given this list in order to develop the database that will store product information. Once the categories are determined, our team will need to know what format the product information should be presented in. This includes the layout of the web page as well as what information should be presented to users (such as description, posting user, or price). When performing the commonality and variability analysis of the system, our team noticed that almost every online store service implemented some form of a related products function. We will need to discuss this idea with the client to see if it would be an addition to the product feature that they would be interested in adopting.

The user account was a feature that our development team was unsure of how to implement. Since our team does not have experience with handling user authentication and protection, it is a feature that we will need to thoroughly research, experiment with, and test. User security is very important because user accounts will contain personal information, such as credit card information. Aside from the security aspects, the process for creating an account must be intuitive and not overly intrusive. Our team will research the most effective ways to facilitate the process of creating and managing accounts and account information.

After analyzing these Three Q's, our team determined that the most architecturally significant features of the system were: the user account , the product listing, and the home page. The community involvement functionalities were not deemed architecturally significant for the following reasons:

* These functionalities were regarded as nice-to-haves and thus non-essential to the system
* The meaning of the functionalities (such as the forums) were clear and obvious
* The implementation was well within our team's capabilities

Determining these architecturally significant components narrowed down the list of features to help our team find an appropriate starting place that would help to reduce the overall risk of the development.

# Risk Analysis and Reduction

With the architecturally significant features determined, our team needed to determine the order in which we developed each feature in such a way that will reduce risk. Since it was considered to be the main access point of the system, we decided to develop the home page first. Having this feature developed will allow to have a place to build off of for the remaining features. This main access point will also act as the "glue" that holds all of the remaining features together, so having this feature in place first will reduce the risk of subsequent features not merging together properly.

After the home page is set up, our team will work on the user account functionality. This will include the registration, log in, and account information functions. This feature was selected to be the next task because it enables users of the system to be able to fully interact with all of the components inherent in the system. It is also is required to have a user account system set up before setting up the product listing system because a user account will be required to post a product listing. Once the account system is developed, it will need to be integrated into the home page in such a way that a user can easily find and access it's functionalities. This feature was also selected to be the second task because of the research and experimentation required by our team to ensure a secure system. Keeping this task as close to the top of the priority list as possible will enable us to better estimate time requirements for this system which will allow us to stay on time and on budget.

Once the home page and user accounts have been developed, the product listings can be created. These listings must be available to view from both the home page and user account pages. Listings must also be creatable only by registered users and must only be editable by the registered user who created the product listing. This feature will require our team to further discuss with the client about the layout and format of the listings, so we will want to keep this task as a priority to mitigate any risk of delays or discrepancies due to miscommunications between our team and the customer.

Following these steps will ensure that risk is kept to a minimum in both design and implementation. The ordering of the tasks will be as follows: developing and designing the home page, developing and implementing the user accounts, and developing and implementing the product listings. For the reasons listed above, this task ordering will mitigate much of the risk inherent in the development and design of this system and help to ensure that our team remains on time and on budget. This risk mitigation will also allow our team to determine any potential bottlenecks or complications that we may run into in the development timeline which will allow us to plan in advance for how to deal with them.

# Additional Discussion With Customer

# Design of Core Features and Use of Design Principles

While developing this system, our team will implement the use of design principles such as the DRY (Don't Repeat Yourself) Principle to ensure that our system is maintainable and flexible. The main principles that will be implemented are:

* The Open-Closed Principle (OCP)
* The Don't Repeat Yourself Principle (DRY)
* The Single Responsibility Principle (SRP)
* The Liskov Substitution Principle (LSP)

The Open-Closed Principle is about the changeability of the system. The definition of the Open-Closed Principle in the words of McLaughlin, Pollice, and West is that "classes should be open for extension, and closed for modification". Simply, this means that the components of the system should be designed in such as way that the system is flexible without needing to be changed. Example implementations of this principle include inheritance of an abstract class and the use of private methods.

The Don't Repeat Yourself Principle is about avoiding repeated code. The definition provided by McLaughlin, Pollice, and West is that "avoid duplicate code by abstracting out things that are common and placing those things in a single location". In other words, the DRY Principle is about ensuring that functionality is placed in a single, logical place so that changes to this functionality will not require changes in multiple places of the system. The most common implementation of this design principle is through the use of encapsulation. However, this principle is not limited to software design and can be used in gathering requirements to ensure that no two requirements address the same topic. The use of this principle ensures that a system is easily modifiable and flexible.

The Single Responsibility Principle is closely related to the Don't Repeat Yourself Principle in that it deals with keeping functionality in a single place. McLaughlin, Pollice, and West define this principle by saying that "every object in your system should have a single responsibility, and all the object's services should be focused on carrying out that single responsibility". This means that each class in a software system should be directly related to only one task. This makes each class only have one reason to change, which reduces the effects of a change to the system. Cohesion is an example implementation of this principle, therefore software that is highly cohesive (and thus loosely coupled) is following the SRP principle.

The Liskov Substitution Principle is about the appropriate uses of inheritance and knowing when not to use inheritance. The definition of the Liskov Substitution Principle given by McLaughlin, Pollice, and West is that "subtypes must be substitutable for their base types". This means that any class that inherits from another class should be able to use the base classes methods without causing any problems. Inheritance that doesn't follow the Liskov Substitution Principle becomes hard to understand which can cause issues with implementation of the subclasses. Some alternatives that can be used when inheritance is not appropriate are: delegation, composition, and aggregation. Delegation is where functionality from another class is used to accomplish a task as opposed to extending the used class. This is useful for when the needed functionality does not need to be changed to meet the designed goals. Composition is where your class is made up of other families of classes. This is useful for when the implementation of a class may change at runtime. A side effect of this alternative is that once the composing class is destroyed, all of the composite classes that it owns are also destroyed. The last alternative takes care of this side effect. Aggregation is like composition in that it uses other families of classes to have a dynamic implementation at runtime but the composite classes will still exist outside of the context of the composing class.

## Home Page

## User Accounts

## Product Listings

# Mock Scenarios of System Interaction

To ensure that the system operates as intended, out team created a set of common scenarios that the average user would go through. We then stepped through these scenarios and paid great attention to the results to compare the actual results to the expected results. If there were any discrepancies, we quickly remedied the situation and corrected the interaction. The common scenarios that we tested were:

* Account Registration
* Editing Account Information
* Creating a Product Listing
* Editing a Product Listing
* Purchasing a Product

To test these interactions, our team created step by step instructions on how to accomplish the tasks that were targeted by each scenario. In the following subsections are the scenarios, the steps to accomplish them, and the expected results of each scenario.

## Account Registration

In order to begin using the web service, a user must create an account. This scenario steps through the process of creating this account.

1. Navigate to the home page URL of the website through an internet browser. (Note: the domain hasn't been set yet, so when running these simulations from Visual Studio the URL of the home page is localhost:[port] where port is the port number assigned by Visual Studio at build time.)



Figure : Home Page of the MakerStore

1. Click the Register button at the top right of the screen.



Figure : The Register Button

1. From the registration page, fill in the requested information.



Figure : The Registration Page

1. Select Continue. You will be returned to the home page and the selected username will appear in the top right of the screen.



Figure : The Username After Creating an Account

## Editing Account Information

After an account has been created, a user will have the ability to edit their account's information. This will include details such as their biography and list of available products. This scenario can only be completed after the previous scenario has been completed.

1. Navigate to the home page of the website.
2. Select Log In from the top right of the screen.



Figure : The Log In Button

1. Enter the username and password associated with the account created in the first scenario.



Figure : The Log In Screen

1. Press Continue. You will be returned to the home page. Select your username from the top right of the screen.
2. Click on the username. From the user account page, select edit.



Figure : The User Account Page

1. Input the updated information. Click Continue.



Figure : The Account Information Edit Form

1. The website will redirect you to an updated user account page.



Figure : Updated User Account Page

## Creating a Product Listing

After an account has been created, a user will have the ability to post a listing for their products. The user will be able to specify the details of the product, such as the category and the price. This scenario can only be completed after the previous scenario has been completed.

1. Navigate to the home page of the web site.
2. Select Log In from the top right of the screen.
3. Enter the username and password associated with the account created in the first scenario.
4. After you are redirected to the home page, click your username in the top right of the screen.
5. From the user account page, click Add Product.



Figure : The Add Product Button

1. You will be redirected to the Add Product Page.



Figure : The Add Product Page

1. From this page, input all of the relevant information about your product.



Figure : The Completed Add Product Form

1. Press Continue. You will be redirected to your user account page where your listing will appear.



Figure : The Added Product Listing

## Editing a Product Listing

After an account has been created, a user will have the ability to edit a listing for their products. The user will be able to update the details of the product, such as the category and the price. This scenario can only be completed after the previous scenario has been completed.

1. Navigate to the home page of the web site.
2. Select Log In from the top right of the screen.
3. Enter the username and password associated with the account created in the first scenario.
4. After you are redirected to the home page, click your username in the top right of the screen.
5. From the user account page, click Edit Product for a previously created product.



Figure : The Edit Product Button

1. You will be redirected to the Edit Product page.



Figure : The Edit Product Page

1. Input updated information in the appropriate fields.



Figure : The Completed Edit Product Form

1. Click Save. You will be redirected to the user account page where the product listing will be updated.



Figure : The Updated Product Listing

## Purchasing a Product With an Account

After an account has been created, a user will have the ability to edit purchase a product. This scenario can only be completed after the first scenario has been completed.

1. Navigate to the home page of the web site.
2. Select Log In from the top right of the screen.
3. Enter the username and password associated with the account created in the first scenario.
4. You will be redirected to the home page. Click the Shop button in the top right.



Figure : The Shop Button

1. You will be redirected to the Shop Page.



Figure : The Shop Page

1. From the Shop Page, select a product listing.
2. You will be redirected to the listing page of the product.



Figure : The Product Listing Page

1. From the Product Listing Page, click Add To Cart.



Figure : The Add To Cart Button

1. You will be redirected to the Shopping Cart Page. The Cart icon in the top right of the screen will be updated to display the number of items in the cart.



Figure : The Shopping Cart Page With Updated Cart Icon

1. Click Checkout.



Figure : The Checkout Button

1. Verify that the billing and shipping information is correct. Click Continue.



Figure : The Checkout Screen

1. You will be redirected to a Checkout Confirmation Screen. The Cart icon will be reset to the original configuration.



Figure : The Checkout Confirmation Screen

## Purchasing a Product Without an Account

Products can be purchased by users who do not currently have an account. Products can be added to a temporary Shopping Cart until the user is ready to check out and then they will be required to register an account. This scenario does not require any previous scenarios to have been completed.

1. Navigate to the home page of the web site.
2. Click the Shop button in the top right.
3. You will be redirected to the Shop Page. From the Shop Page, select a product listing.
4. You will be redirected to the listing page of the product.
5. From the Product Listing Page, click Add To Cart.
6. You will be redirected to the Shopping Cart Page. The Cart icon in the top right of the screen will be updated to display the number of items in the cart.
7. Click Checkout.
8. You will be redirected to the Registration Page. The Registration Page will still contain the updated Cart icon.



Figure : The Registration Page With the Updated Cart Icon

1. Fill in the required information and click Continue. You will be redirected to the Checkout Page.
2. Verify that the billing and shipping information are correct. Click Continue.
3. You will be redirected to a Checkout Confirmation Screen. The Cart icon will be reset to the original configuration.

# Appendix A: References

* McLaughlin, Brett, Gary Pollice, and David West. *Head First Object-oriented Analysis and Design*. Sebastopol, CA: O'Reilly, 2007. Print.
* "Easy Intro to ASP.NET MVC." *Easy Intro to ASP.NET MVC*. N.p., n.d. Web. 21 Oct. 2014. <http://www.beansoftware.com/ASP.NET-Tutorials/Intro-ASP.NET-MVC.aspx>.