# **Final Project Report**

Eduardo Valencia

Michael Esan

Larry Alexander

Treylon Wofford

University of Advancing Technology

CSC318

Dr. Jill Coddington

July 14, 2021

# **Abstract**

We formed a team with Eduardo Valencia, Treylon Wofford, Michael Esan, and Larry Alexander to create a shopping application. Our goal was to explore using project management and software engineering methodologies in a team project. At the beginning of our project, we selected a project management methodology and chose our application’s architecture. Over four weeks, our team collaborated to create our application, and we documented our process at the end of each week. In the end, we successfully completed our shopping application according to our requirements. We will discuss our project’s requirements, our project management methodologies, our software engineering methodologies, and how those methodologies contributed to our project’s success.

# **Table of Contents**

Contents

[**Title Page** 1](#_Toc77243765)

[**Abstract** 2](#_Toc77243766)

[**Table of Contents** 3](#_Toc77243767)

[Introduction and Explanation 4](#_Toc77243768)

[Development Methodology 5](#_Toc77243769)

[UML Diagrams 8](#_Toc77243770)

[Design Patterns 9](#_Toc77243771)

[Use Case and State Diagrams 10](#_Toc77243772)

[Requirements 11](#_Toc77243773)

[Requirements Negotiation 12](#_Toc77243774)

[Requirements Validation 12](#_Toc77243775)

[UX Diagram 14](#_Toc77243776)

[Lessons Learned 15](#_Toc77243777)

[References 16](#_Toc77243778)

# Introduction and Explanation

We created an online shopping website for selling clothing. We created three main pages: the product list, product details, and checkout pages. First, the product list page shows the list of products, organized by categories. If users want to purchase a product, they can add it directly to their cart. Alternatively, they can click on each product to see its details page. The product details page shows more information about a product, including its images, description, size options, and color options. Additionally, users can click on an “Add to Cart” button to add the product to their cart. Third, the cart page shows users all the items in their cart, the total, and a “Checkout” button to pay. Finally, the editor page helps administrators add products or categories to the website. By collaborating with our team and dividing this project into sprints, we created an e-commerce application.

# Development Methodology

Scrum is a development process that organizes tasks into sprints. The book *Essentials of software engineering* explains more about sprints. It says, “They are short in duration (one month or less) and time-boxed (their duration is held constant, but scope may be adjusted if needed). Ideally, a potentially shippable product is produced at the end of each sprint" (Tsui, F. F., Karam, O., Bernal, B, 2018). Therefore, Scrum organizes tasks into short sprints, and each one should produce a deliverable product. Since each sprint has a deadline and a clear set of requirements, sprints help organize the team’s tasks. We used Scrum to help our team organize our tasks and build a shopping application.

Scrum defines a process that helps teams communicate and eliminate errors as quickly as possible. This process includes meetings. First, Scrum encourages development teams to meet with the product owner to discuss the requirements. In this meeting, they will negotiate the requirements for the first sprint (Tsui, F. F., Karam, O., Bernal, B, 2018). By involving the entire team in the requirements process, they can understand the owner’s needs and start the project on the right track. Additionally, Scrum encourages daily meetings between team members. The article *Fundamentals of the Scrum Methodology* explains, "During these brief, 15-minute meetings, everyone reports what they accomplished yesterday, what they plan to work on that day, and any current 'impediments' (factors that are keeping them from working more efficiently). This visibility helps uncover problems and bring them to the forefront quickly, so the team can tackle and overcome them together" (Bonnie, 2014). This shows that daily meetings can help the team quickly bring problems to light so that they can fix them. By meeting with the product owner and with other team members, teams will have a clear set of requirements.

Next, Scrum encourages teams to review and retrospect their work. After the team finishes a sprint, they should meet to review their work. The book *Essentials of software engineering* explains, “The development team demonstrates the work it has done, answers questions, and discusses what went well, what problems they faced, and how they solved them. Finally the entire team collaborates on what to do next (which is input for the sprint planning for the next sprint)” (Tsui, F. F., Karam, O., Bernal, B, 2018). Therefore, the review process involves identifying the issues the team faced and planning the next sprint. The next step is the retrospective step, which encourages the team to reflect on the issues that they identified and determine how they can improve (Tsui, F. F., Karam, O., Bernal, B, 2018). The review and retrospective processes help the team prepare for the next sprint and learn from their mistakes.

We used the Scrum development process for building a shopping application. The first step in the Scrum process is to meet to discuss the requirements. We met during the first week to discuss the shopping application’s requirements and decide the tasks for the week’s sprint. Since Scrum encourages daily meetings, we also met constantly through online chat to discuss our progress and disclose any problems we had. By meeting to discuss our requirements and progress, we encouraged communication and helped identify issues quickly (Bonnie, 2014).

Next, Scrum encourages teams to divide their tasks into sprints. In the first week, we organized our tasks into sprints. Since we had three weeks to complete our assignment, we divided the tasks into three sprints. Additionally, we assigned team members to each task. Our project-management tool, Notion, also indicated which tasks were due for each sprint. By dividing our tasks into sprints, we organized our tasks efficiently and completed our shopping application on time.

Finally, we reviewed and reflected on our work after each sprint. At the end of the week, we reflected on our work and discussed any problems we had, such as time constraints or missing requirements. Then, we reviewed the tasks for the upcoming sprint and discussed any new requirements. After reviewing our work, we reflected on the issues we encountered to determine how we can improve. For example, if we did not have enough time to develop the application, we might have decided to push the design’s deadline a day earlier so that we can begin the development process sooner. By reviewing our work and reflecting on the issues we encountered, we prevented problems and improved our work for the next sprints.

# UML Diagrams

Diagram, schematic

Description automatically generated

# Design Patterns

We used the adapter pattern throughout our shopping application. We added a content management system, Strapi, in our application to store the list of products, categories, and orders. We created React components to display this data. However, Strapi data’s format was different from the format our components accepted. Therefore, we used the adapter pattern to make the formats compatible. According to the article *Adapter,* an adapter is "…a special object that converts the interface of one object so that another object can understand it. An adapter wraps one of the objects to hide the complexity of conversion happening behind the scenes" (Refactoring Guru). This shows that adapter patterns make interfaces compatible by converting between different formats. Since our content management system’s and components’ data formats were different, we used an adapter pattern to make the application’s data compatible with our components. By using adapter patterns throughout our application, we made its interfaces compatible and displayed the content management system’s data.

# Use Case and State Diagrams

Diagram, schematic

Description automatically generated

# Requirements

1. Product list page
   1. Users can see a list of products, and each product has a title and price.
   2. Users can click on each product to see the product details page.
2. Product details page
   1. Users can see the product’s name, description, thumbnails, and price.
   2. Users can click on an “Add to Cart” button to add the item to their cart.
      1. Their cart saves which items they want to purchase until they either check out or abandon the page.
3. Cart page
   1. Users can see the list of products they want to purchase.
      1. Each product shows its thumbnail, name, and price.
   2. Users can click a button on each product to remove the item from their cart.
   3. Users can see the total.
   4. Users can click a button to start the checkout process.
4. Checkout
   1. It prompts them for their email and payment information.
   2. Users can click a button to pay for the items and finalize the checkout process.
   3. Once the checkout process succeeds
      1. It clears their cart.
      2. It creates a new order with the user’s information.
      3. It shows them confirmation of their order request.
5. Administrator panel
   1. Administrators can sign in with a username and password.
   2. It can be accessed as a website.
   3. Administrators must be able to add new products with the following information:
      1. Price
      2. Name
      3. Thumbnail
      4. Description
6. Main website
   1. The application must have a product list, product details, and cart page.
   2. All pages are mobile-friendly and work on different screen sizes.
   3. It must show the products from the administrator panel.
7. Each page must take less than 4 days to develop.
8. The design for each page should be completed before Wednesday of each week.
9. Each page must be completed within a sprint.

# Requirements Negotiation

Although Larry wanted to create a search page, Eduardo said he did not have time to develop search functionality. To save time and make the website simpler, we decided to remove the search functionality and create a featured products page instead.

Although Treylon wanted to design the website with Figma, Larry said he was more familiar with Adobe XD. They decided to use Adobe XD to design the page so that we could all work with the same platform.

# Requirements Validation

The requirements were valid. Each requirement helped us achieve our goal of building a shopping application within three sprints. All pages, including the product list, product details, and cart page, were essential for a shopping application. Furthermore, we knew the application would be completed on time because each page would be completed within a sprint.

The requirements did not have any conflicts. Although there could have been a conflict in deciding which design tool to use for our application, Larry and Treylon compromised and decided to use Adobe XD. There could have also been a conflict if the design required us to build search functionality. Since Eduardo may not have been able to add the search functionality in time, the search page requirement would have conflicted with the time constraint. Fortunately, Larry and Eduardo compromised and decided to avoid building a search page.

# UX Diagram

Graphical user interface

Description automatically generated with medium confidence

# Lessons Learned

We learned a few lessons about software engineering by completing a shopping application. First, we learned of the importance of project management for completing applications on time. To create our shopping application, we needed to complete dozens of tasks and subtasks, and we needed to coordinate them between our team members. Additionally, each member needed to know which tasks they needed to complete and when they were due. Fortunately, project management helped us solve these problems. We used a project management tool, Notion, to coordinate tasks between team members. By tracking our tasks in Notion, we were able to monitor our project and see if we were falling behind schedule. Project management is an indispensable tool for building applications with teams.

Additionally, we learned about the importance of planning our code. First, planning our code helped us understand the complexity of our application. After analyzing our requirements, we knew we would have to set up a content management system and create three different pages: a product list, product details, and cart page. After analyzing the requirements for each page, we also knew we had to integrate a payments processor on both the front-end and back-end. In turn, understanding our application’s complexity helped us create estimates for each task. This helped us schedule our project and eliminate any unnecessary tasks that would exceed our project’s deadline. Planning our code helped us understand our application’s complexity and schedule our project.

# References

Bonnie, E. (2014, June 12). *Fundamentals of the Scrum Methodology*. Wrike. https://www.wrike.com/blog/fundamentals-of-the-Scrum-methodology/.

Refactoring Guru. (n.d.). *Adapter*. Refactoring.Guru. https://refactoring.guru/design-patterns/adapter.

Tsui, F. F., Karam, O., & Bernal, B. (2018). *Essentials of software engineering* (4th ed.). Jones & Bartlett Learning.