

Software Design Specifications Document

# Project Title

*By*

Areej Razzaq 294658

Huda Fatima 288872

Supervisor: Dr. Adnan Rashid

Co-Supervisor Name: Sir Maajid Maqbool

**Bachelor of Science in Computer Science (2019-2023)**

Department of Computing

School of Electrical Engineering and Computer Science

National University of Sciences & Technology

# 

# Table of Contents

[1. Introduction 1](#_Toc126881686)

[2. Design Methodology and Software Process Model 2](#_Toc126881687)

[3. System Overview 2](#_Toc126881688)

[a. Architectural Design 3](#_Toc126881689)

[4. Design Models 4](#_Toc126881690)

[a. Structure and Relationships 4](#_Toc126881691)

[5. Data Design 8](#_Toc126881692)

[6. User Interface Design 10](#_Toc126881693)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

**Application Evaluation History**

|  |  |
| --- | --- |
| **Comments (by committee)**  **\*include the ones given at scope time both in doc and presentation** | **Action Taken** |
|  |  |
|  |  |

**Supervised by**

**Supervisor’s Name**

Signature\_\_\_\_\_\_\_\_\_\_\_

# Introduction

The purpose of the software design document for FlipShop, a web application, is to serve as a blueprint for the development of the product. It provides a detailed description of the application's architecture, functionality, and user interface, along with any technical requirements. The document outlines the design process, including the research and analysis conducted, the design decisions made, and the steps taken to ensure a seamless user experience. Additionally, the document provides a visual representation of the application's interface and navigation structure, making it easier for developers to understand the user's journey and the flow of the application. The software design document serves as a reference for the development team during the implementation phase and provides a comprehensive overview of the application for stakeholders to review. Overall, the purpose of the software design document for FlipShop is to provide a clear, concise, and comprehensive guide for the development of the web application, ensuring a successful outcome.

When designing the FlipShop web application, several design decisions and tradeoffs were made to ensure the optimal functionality and user experience. As a marketplace dependent on third-party APIs, ensuring seamless API integration was a top priority. This led to the decision to use a microservices architecture, allowing for modular and flexible API integration. However, this approach also required careful consideration of data consistency and security. To balance these trade-offs, a combination of caching techniques and real-time data synchronization was implemented. Additionally, user experience was prioritized with the use of a responsive design and intuitive user interface. These design decisions and tradeoffs ensure that FlipShop provides a reliable and user-friendly marketplace experience while utilizing the power of third-party APIs.

# Design Methodology and Software Process Model

Object-Oriented Design (OOD) was chosen as the design methodology for Flip Shop because of its ability to effectively manage complexity in software development. As a marketplace for digital assets, Flip Shop requires a flexible and scalable design that can accommodate a wide range of requirements. OOD provides a well-defined structure for the application, making it easier to manage and maintain. The use of classes and objects in OOD allows for a clear separation of concerns, enabling developers to work on different parts of the application in parallel and reducing the risk of errors. Additionally, OOD encourages the use of reusability, abstraction, and encapsulation, making it easier to reuse code, manage dependencies, and ensure the stability of the application. The application of OOD in Flip Shop results in a well-structured, maintainable, and scalable platform for buying and selling digital assets.

Both Laravel and React make use of Object-Oriented Design (OOD) principles in their development. Laravel, as a PHP framework, follows OOD principles to provide a well-structured and maintainable codebase. The use of classes and objects in Laravel allows for a clear separation of concerns and enables developers to work on different parts of the application in parallel, reducing the risk of errors. Additionally, Laravel makes use of design patterns such as Model-View-Controller (MVC) to promote consistency, reusability, and overall quality of the code.

React, a JavaScript library for building user interfaces, also uses OOD principles. React components are designed as objects, with each component having its own state and behavior. The use of OOD in React allows for a clear separation of concerns between the different parts of the user interface, making it easier to manage and maintain. Additionally, react encourages the use of reusability and abstraction, making it easier to reuse code and manage dependencies.

In conclusion, the use of OOD in both Laravel and React enhances the maintainability, scalability, and overall quality of the Flip Shop platform, making it a great choice for buying and selling digital assets.

# System Overview

Flip Shop is a marketplace web application built with Laravel and React, providing a platform for people to sell their digital assets with ease. The application offers a high level of reliability and automation, ensuring that transactions are secure and efficient. The Laravel framework powers the backend, providing robust security features and efficient data management. The React frontend, on the other hand, offers a smooth and responsive user experience with dynamic updates and intuitive navigation. With Flip Shop, users can easily list their digital assets, manage their sales, and receive payments securely. The platform's automation system ensures that transactions are processed smoothly and accurately, making it a reliable choice for buying and selling digital assets.

## Architectural Design

The selection of the Model-View-Controller (MVC) architectural style for Flip Shop was a deliberate choice due to its proven success in developing web applications. Flip Shop, a marketplace for digital assets built using Laravel and React, leverages the strengths of the MVC architecture to separate the application's data, user interface, and control logic. This separation of concerns allows for a clear division of responsibilities and promotes maintainability, scalability, and ease of development. The Laravel framework implements the MVC pattern, providing a clear structure for the application's backend logic and data storage. Meanwhile, react is used for the front end, providing a dynamic and responsive user interface.

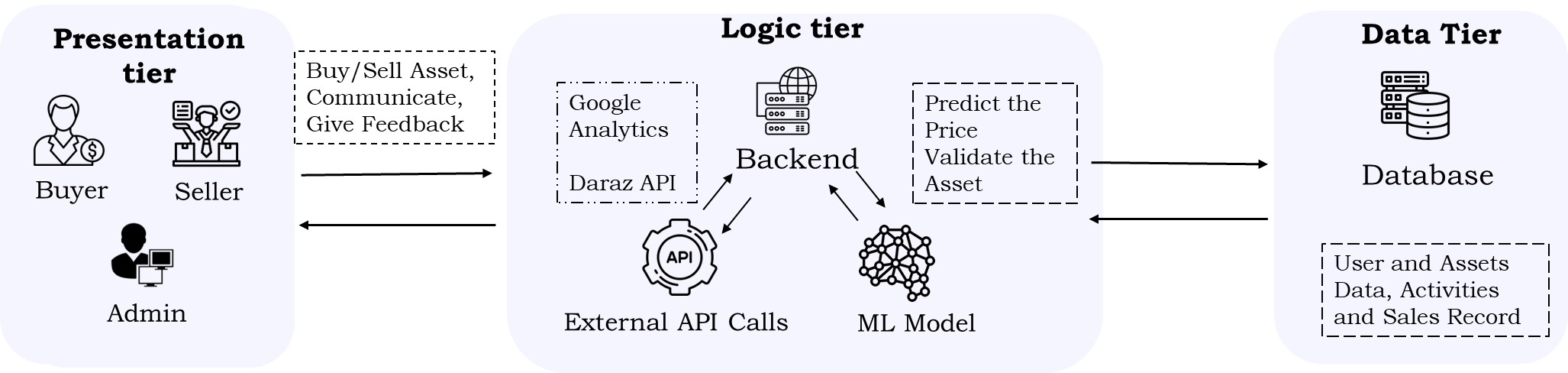


Figure 1- Architectural Design

# Design Models

Process flow diagrams are an essential tool in the development of Flip Shop, a web application for buying and selling digital assets. These diagrams provide a visual representation of the steps involved in a process, such as the purchase of a digital asset on the platform. By outlining the process flow, developers can identify any potential bottlenecks or areas for improvement and ensure a smooth user experience. The process flow diagrams also serve as a basis for creating the code, making it easier to implement the functionality of the platform. The process flow diagrams for the potential stakeholders (buyers[[1]](#footnote-1) and sellers[[2]](#footnote-2)) are represented in the following figures.

## Structure and Relationships

Structure and Relationships are crucial in the software development of any web application, including a marketplace for digital assets. In the case of FlipShop, the architecture is designed to support communication with several third-party APIs to ensure seamless integration and exchange of data. The various components of the platform are structured in a modular manner to promote maintainability, scalability, and ease of development. The relationships between the components are well-defined, and the platform is designed to enforce a strict separation of concerns, ensuring that changes in one module do not affect the functionality of others. Additionally, the platform follows established design patterns and principles to promote consistency, reusability, and overall quality of the codebase. FlipShop communicates with external APIs for fetching data without asking the user to enter the data. This not only helps in getting data for the user but also ensures the reliability of the data. The following diagrams show how FlipShop will handle API calls and how they will interact with the data model of FlipShop. (GetSellerMetrics[[3]](#footnote-3), Get Products[[4]](#footnote-4))

Diagram

Description automatically generated

Figure 2- Buyer Process Flow

Diagram

Description automatically generated

Figure 3- Seller Process Flow

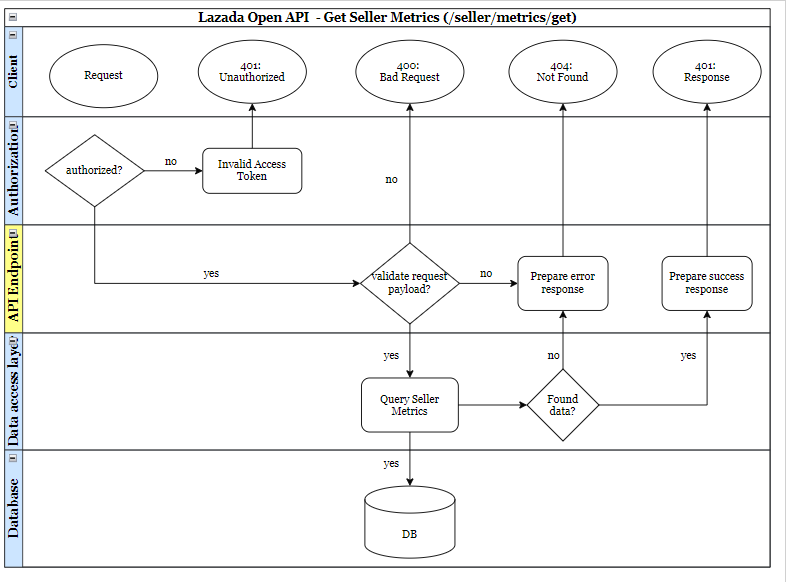


Figure 4- Lazada Open API - Seller Metrics

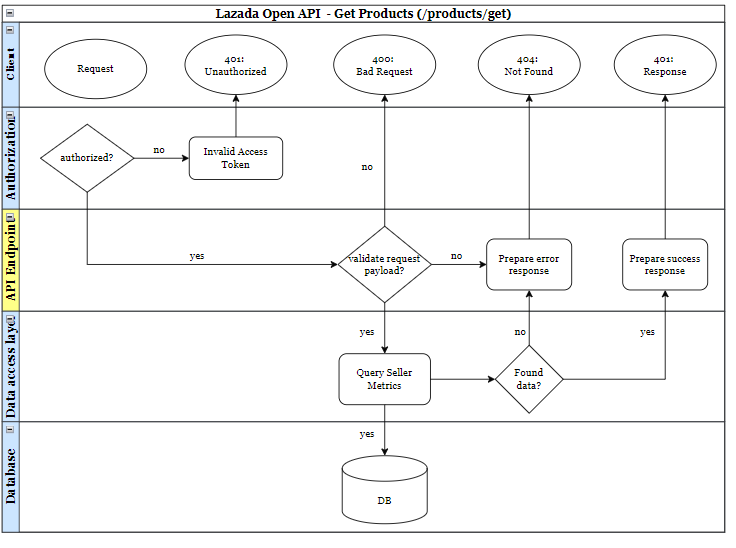


Figure 5- Lazada Open API - Get Products

# Data Design

The use of ERD (Entity Relationship Diagram) in the development of the web application Flip Shop helps to visualize and manage the relationships between entities in the database. By clearly defining entities and their attributes, and the relationships between them, ERD[[5]](#footnote-5) ensures the accuracy and integrity of the data in the system, making it an essential tool in the development process.

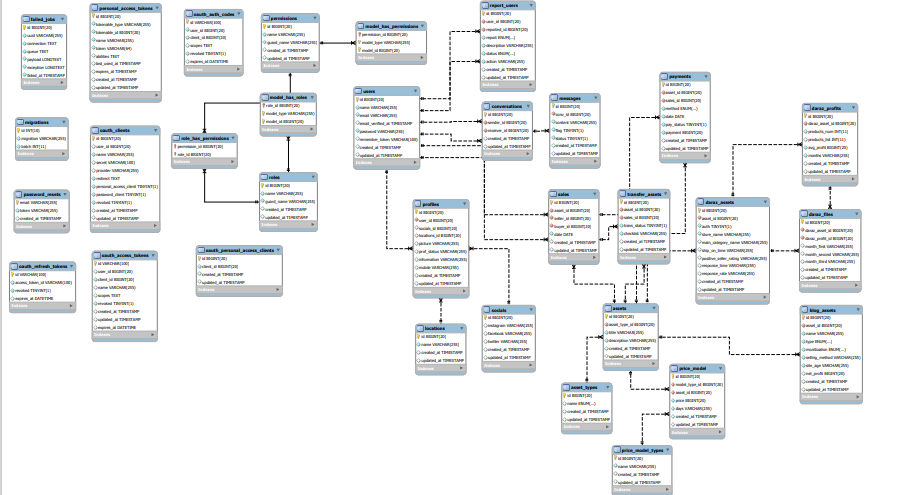


Figure 6 - ERD Diagram

# User Interface Design

The user interface design for Flip Shop, a web application for buying and selling digital assets, is inspired by the design principles of Google. This includes a focus on simplicity, consistency, and accessibility, as well as the use of color, typography, and imagery to create a visually appealing and intuitive user experience. The use of Google's design principles helps to create a cohesive look and feel for the platform and ensure that users can easily navigate and interact with the platform. This means that the interface is uncluttered and easy to navigate, with a focus on functionality and usability. The use of clean and simple design elements, such as clear typography and ample white space, helps to ensure that users can find what they're looking for quickly and easily. The design is also optimized for all devices, so users can access the platform from their desktop or mobile device with the same level of ease.

Graphical user interface, website

Description automatically generated

Figure 7 - UI Figure 1

Graphical user interface, text, application, email, website

Description automatically generated

Figure 8- UI Figure – Listings

Graphical user interface, application, website

Description automatically generated

Figure 9 - UI Figure - Services Page

Graphical user interface, text, application

Description automatically generated

Figure 10- UI Figure - Stats

Graphical user interface, text, application

Description automatically generated

Figure 11 - UI Figure - Testimonials

Graphical user interface, application, website

Description automatically generated

Figure 12 - UI Figure - Team

Graphical user interface, text, application, email

Description automatically generated

Figure 13 - UI Figure - FAQ

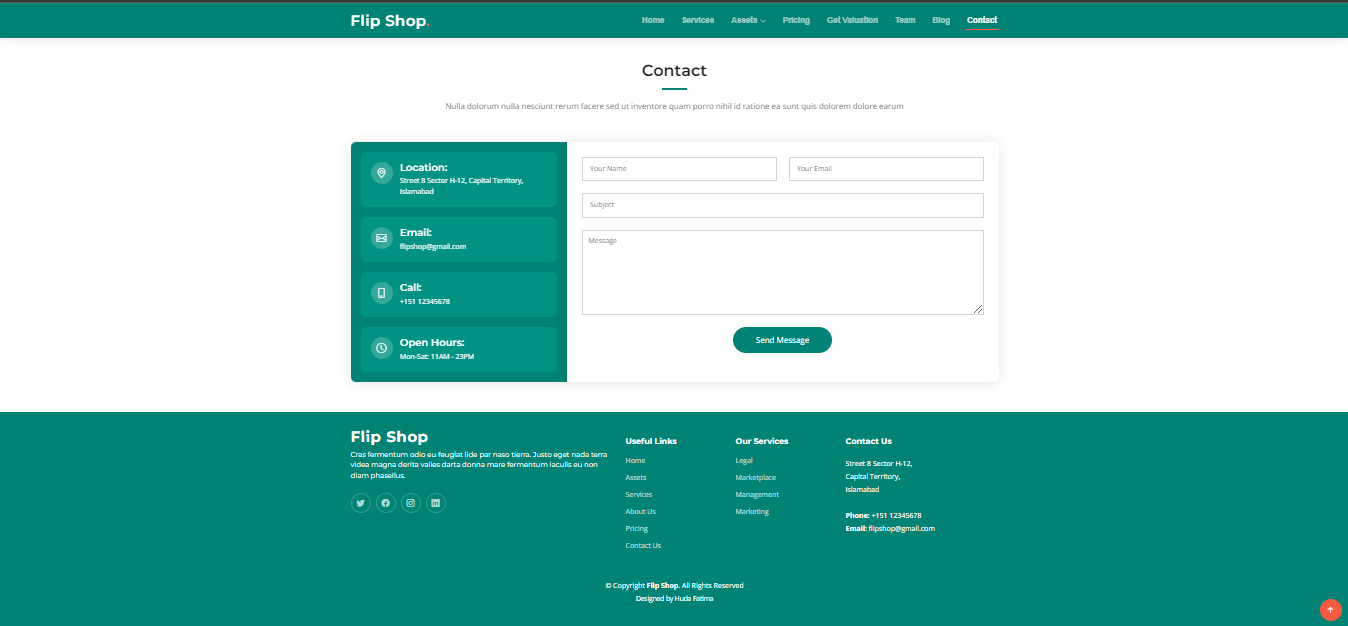


Figure 14 - UI Figure - Contact Us

1. [View Process Flow for Buyer in HTML](file:///C:\Users\trazz\Downloads\ProcessFlow-Buyer.html) (full screen) [↑](#footnote-ref-1)
2. [View Process Flow for Seller in HTML](file:///C:\Users\trazz\Downloads\ProcessFlow-Seller.html) (full screen) [↑](#footnote-ref-2)
3. [View Process Flow for API in HTML](file:///C:\Users\trazz\Downloads\DarazAPI-GetSellerMetrics.html) (full screen) [↑](#footnote-ref-3)
4. [View Process Flow for API in HTML](file:///C:\Users\trazz\Downloads\DarazAPI-GetProducts.html) (full screen [↑](#footnote-ref-4)
5. [View ERD Diagram in PDF](file:///G:\My%20Drive\FYP\flipshop-pdf.pdf) (full view) [↑](#footnote-ref-5)