**Software Requirements and Design Document**

**For**

**Group 6**

Version 1.0

**Authors**:

Sola Adebisi

Rachel Andris

Hasan Bazzi

Alyssa Evans

Souhail Marnaoui

# Overview

OOPS, Our Online Personal Stylist, is dedicated to developing a dynamic platform for the trading of clothing items, serving both buyers and sellers. The platform allows users to switch seamlessly between buyer and seller modes, offering the flexibility to explore and buy clothing items or to list their own items for sale. It is designed with a strong emphasis on user experience, ensuring the buying and selling process is not only efficient but also secure and enjoyable.

The platform is engineered with practicality in mind, incorporating core features such as the ability to list items, browse through a selection of clothing, and make purchases for buyers. Sellers, on the other hand, will be equipped with tools to monitor and manage their listings. The project plans to integrate a database module to adeptly manage data related to items and users, and a functionality module that will oversee database operations and user interactions, enhancing the overall effectiveness and user-friendliness of the platform.

# Functional Requirements (10 points)

**Buyer** **Mode**- (high) allows users to browse clothing items, search for items, view item details including photos, sizes, and prices, and add items to their shopping cart.

**Seller** **Mode**- (high) allows users to create new listings for clothing items, including uploading photos, providing detailed descriptions, specifying sizes, and setting prices.

**Purchase**- (high) In buyer mode, the system enables users to securely purchase items from sellers, with payment and shipping pages to process transactions.

**User Profile**- (medium) includes contact details, shipping addresses, purchase and listing history, and favorites.

**Chat**- (low) facilitates communication between buyers and sellers, allowing users to inquire about items, negotiate prices, and coordinate shipping details.

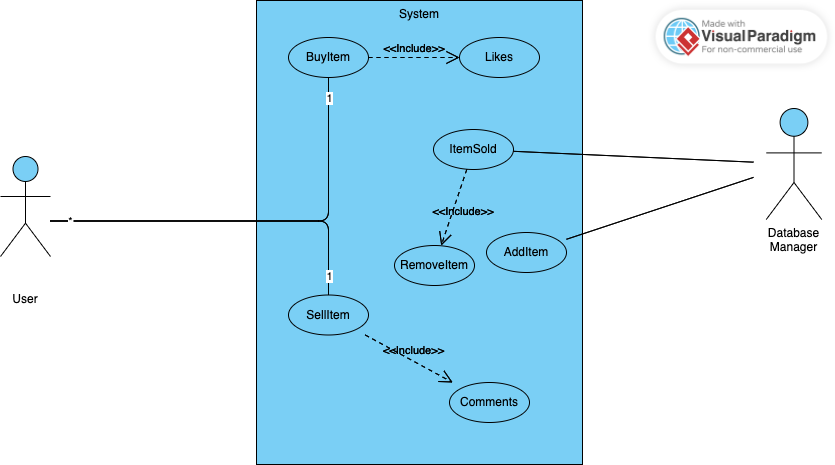
# Non-functional Requirements (10 points)

**Performance:** (high)handles a large number of users and transactions without significant a decline in performance, page loading, and transaction processing.

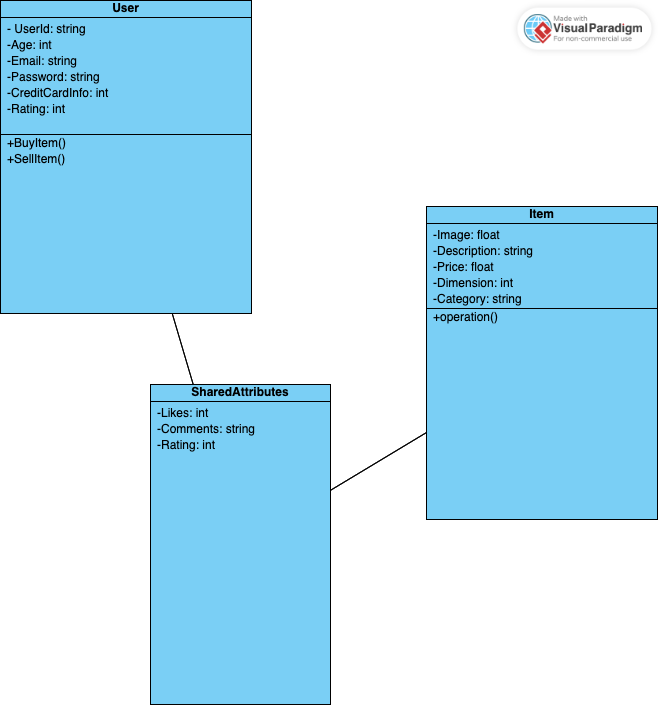
**Accessibility**: (medium) ensures that all users can access and use the platform effectively.

**Usability**: (medium) prioritize user experience, with intuitive interfaces, clear navigation paths, and responsive design across various devices and screen sizes

# Use Case Diagram (10 points)

**

# Class Diagram and/or Sequence Diagrams (15 points)

**

***6. Operating Environment***

*The proposed e-commerce platform is designed to operate in a versatile and dynamic web-based environment, ensuring broad accessibility and ease of use across various devices and user demographics. The following outlines the software's operating environment:*

***Hardware Platform:*** *The platform is accessible on a wide range of devices, including desktop computers, laptops, tablets, and smartphones, without requiring specific hardware capabilities beyond those necessary for running modern web browsers.*

***Operating System and Versions:*** *The software is platform-independent and can be accessed through any operating system capable of running modern web browsers, such as Windows, macOS, Linux, iOS, and Android. This ensures users can access the platform regardless of their device's operating system.*

***Web Browsers:*** *The platform is optimized for compatibility with major web browsers, including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge, ensuring a consistent and responsive user experience across different browsers and versions.*

***Other Software Components:*** *The platform is designed to coexist with various web technologies and frameworks, including*

*Flask (for the web application framework),*

*SQLite (for the database management system),*

*TensorFlow and Pandas (for data processing and machine learning capabilities).*

*Hosting Services: The platform will be hosted on cloud services capable of providing scalable resources and global accessibility, such as Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure, to ensure high availability and performance.*

***7. Assumptions and Dependencies***

***Assumptions:***

*Stable Internet Connection: It is assumed that users will have a stable and reliable internet connection to access the platform, as the application is web-based and requires internet access for all functionalities.*

*Third-Party APIs: The assumption that third-party payment gateways and other APIs (e.g., for secure payments) will remain stable and compatible with our platform over time.*

*Browser Compatibility: We assume that users will access the platform using modern and updated web browsers that support HTML5, CSS3, and JavaScript ES6 standards.*

*Cloud Hosting Performance: It is assumed that the chosen cloud hosting service will provide consistent performance and uptime as required by the platform's operational needs.*

***Dependencies:***

*Third-Party Libraries/Frameworks: The project's development and operation depend on the availability and continued support of Flask, TensorFlow, Pandas, SQLite, and any other third-party libraries or frameworks utilized in the platform.*

*E-commerce Dataset: The dependency on external datasets, such as those available on Kaggle, for initial model training and testing. Any changes or unavailability of these datasets could impact the development and functionality of the recommendation system.*

*Payment Gateway APIs: The platform's ability to process transactions depends on third-party payment gateway APIs. Any disruptions or changes in their service could affect the platform's transaction processing capabilities.*

*Cloud Hosting Services: The project depends on cloud hosting services (AWS, GCP, or Azure) for deployment, scaling, and managing the application. Any changes in their pricing, policies, or service offerings could impact the project's operational costs and availability.*