صورة تحتوي على رمز, التصميم

قد يكون المحتوى المعد بواسطة الذكاء الاصطناعي غير صحيح.

***Intelligent Fashion Recommendation System***

***Hazem Khairy AbdelHakim***

***Ahmed Mahmoud Mohamed***

***Wafaey Khaled Mohamed***

***Fatma Mahmoud Ahmed***

***Sabrine Nasser Hashim***

***Maryam Mahmoud Mohamed***

# Supervised by:

**Eng. Abdallah Wagih**

Digital Egypt Pioneers Initiative

Table of Contents

[Abstract 3](#_Toc197784712)

[Declaration 3](#_Toc197784713)

[Acknowledgment 3](#_Toc197784714)

[1.0 Introduction 4](#_Toc197784715)

[**2.2.1 User Interface** 8](#_Toc197784716)

[**2.2.2 Hardware Interfaces** 9](#_Toc197784717)

[**2.2.3 Software Interfaces** 10](#_Toc197784718)

[**2.2.4 Communication Interfaces** 10](#_Toc197784719)

# Abstract

The Intelligent Fashion Recommendation System is an advanced web application designed to elevate the online shopping experience through personalized fashion recommendations. By leveraging EfficientNetB0 for image feature extraction and Euclidean distance for similarity matching, IFRS enables users to upload outfit images and receive tailored fashion suggestions. The system also offers intuitive category browsing, presented via a responsive, CSS-styled interface built with Flask. IFRS enhances user engagement and satisfaction, making fashion discovery seamless and efficient in e-commerce platforms.

# Declaration

We hereby declare that this graduation project is our original work and has not been previously submitted for any academic credit...

# Acknowledgment

We extend our deepest gratitude to our supervisor **ENG\Abdullah Wagih** for their continuous support, valuable insights, and encouragement throughout the development of this project...

# 1.0 Introduction

The fashion industry has embraced e-commerce, offering convenience but often lacking personalized recommendations. The Intelligent Fashion Recommendation System (IFRS) addresses this challenge with a Flask-based web application that delivers image-based fashion recommendations and category exploration. Powered by EfficientNetB0 and styled with a responsive CSS front-end, IFRS provides a user-friendly experience that mirrors the expertise of a personal stylist, boosting engagement and conversions in online fashion retail.

**1.1 Purpose of the System**

it aims to transform online fashion retail by offering personalized recommendations based on user-uploaded images and curated category browsing. The system seeks to enhance user satisfaction, streamline product discovery, and increase purchase intent..

**1.2 Scope of the System**

a web application that:

* Provides fashion recommendations based on image uploads using EfficientNetB0 and Euclidean distance
* Supports browsing of top fashion categories with grid-based displays
* Features a responsive front-end styled with CSS and Google Fonts
* Manages static file serving for images and product data

**1.3 Objectives**

* Develop a scalable, user-centric fashion recommendation system
* Deliver accurate image-based recommendations using deep learning
* Enable intuitive category browsing for efficient product discovery
* Ensure a responsive and visually appealing user interface

**1.4 Problem Statement**

E-commerce fashion platforms often present users with irrelevant products, leading to frustration and cart abandonment. It solves this by offering personalized image-based recommendations and curated category views, making shopping efficient and enjoyable.

**1.5 Methodology**

**1.5.1 Data Quality**

**The performance of IFRS relies on high-quality datasets to ensure accurate recommendations. This section details the characteristics, preprocessing steps, and quality assurance measures for the data used.**

* **Dataset Description:**
  + **Product Metadata: Contains details for 10,000+ fashion products, including product ID, name, type, and color. Approximately 5% of records have missing or inconsistent values.**
  + **Image Mappings: Links product IDs to image paths, with 3% of entries lacking valid links.**
  + **Precomputed Features: Stores 1280-dimensional feature vectors extracted using EfficientNetB0 for all valid product images.**
* **Preprocessing Steps:**
  + **Removed duplicate entries and records with missing critical fields (e.g., product type, image path).**
  + **Standardized categorical attributes (e.g., color names like “Blue” vs. “blue”) for consistency.**
  + **Validated image paths to exclude entries with broken links, reducing the dataset to 9,500 valid entries.**
* **Data Quality Challenges:**
  + **Challenge: Inconsistent metadata formatting (e.g., varying category names).**
    - **Solution: Applied text normalization and mapped to unified category labels in the backend.**
  + **Challenge: Missing image features for some products.**
    - **Solution: Excluded affected products from the recommendation pipeline and flagged for future feature extraction.**
* **Outcome: The cleaned dataset ensures reliable recommendations, with optimized data structures for efficient lookup and similarity matching.**

**Software Requirements Specification**

**2.1 Overall Description**

**2.1.1 Product Perspective**

standalone Flask web application that provides image-based fashion recommendations and category browsing. It leverages EfficientNetB0 for feature extraction and a responsive CSS front-end, making it ideal for integration into e-commerce platforms.

**2.1.2 Product Functions**

* Generate fashion recommendations based on user-uploaded images
* Display top 10 fashion categories with representative images
* Present up to 50 products per category in a grid layout
* Serve a responsive UI with hover effects and custom file upload styling
* Handle image uploads and static file serving

**2.1.3 User Classes**

* **End User**: Uploads images and browses categories via the web interface
* **Developer**: Maintains the Flask backend and front-end templates

**2.1.4 Operating Environment**

* **Backend**: Python 3.10+, Flask
* **Front-end**: HTML, CSS (with Google Fonts), JavaScript
* **Dependencies**: TensorFlow/Keras, Pandas, NumPy, scikit-learn
* **Deployment**: Local server or cloud platform (e.g., Heroku)
* **Client**: Web browser (Chrome, Firefox, etc.)

**2.1.5 Design Constraints**

* Image processing must be efficient for quick responses
* Precomputed image features required for recommendations
* HTTPS recommended for production environments

**2.1.6 User Documentation**

* README for setup and deployment instructions
* Developer guide for extending the system

**2.1.7 Assumptions**

* Availability of product data and precomputed features
* Internet access for Google Fonts
* Users access the system via a web browser

**2.2 External Interface Requirements**

**2.2.1 User Interface**

The IFRS front-end is a responsive, intuitive interface styled with a custom CSS file, using the Roboto font (Google Fonts) and Font Awesome icons (v6.5.0) to deliver a modern e-commerce aesthetic across its core pages: the homepage, category page, and results page.

* **Homepage**:
  + **Image Upload Form**: A custom-styled button with a camera icon triggers file selection, displaying a success message via JavaScript. The form is centered using flexbox for alignment.
  + **Category Browsing**: Two grid containers display the top 10 categories as clickable cards, each with a hover effect (scale transform) and representative images.
  + **Layout**: A main container ensures centered content with a margin-top to accommodate the fixed navigation bar.
* **Category Page**:
  + Displays up to 50 products in a responsive grid, with cards showing images, names, types, colors, and prices (randomly generated in the backend). Cards feature box-shadows and rounded corners.
  + Hover effects scale cards by 5% for interactivity.
* **Results Page**:
  + Presents the uploaded image in a centered card, followed by up to 5 recommended products in a grid. A custom heading adjusts spacing for clarity.
  + Product cards maintain consistent styling with metadata sections.
* **Common Elements**:
  + **Navigation Bar**: A fixed bar with a dark background (#333) includes a logo (max-height: 50px) and links (“Help,” “Contact Us”) with a hover color (#f39c12).
  + **Footer**: A responsive footer with “About Us,” “Quick Links,” and “Follow Us” sections uses flexbox, stacking vertically on mobile via a media query.
  + **Responsiveness**: Percentage-based card widths and flex-wrap ensure adaptability. A viewport meta tag supports mobile-first design.
  + **Accessibility**: Alt text for images and high-contrast colors (white on #333) aid accessibility, with plans for ARIA enhancements.

**2.2.2 Hardware Interfaces**

No specialized hardware is required. A camera or file storage is optional for image uploads.

**2.2.3 Software Interfaces**

* **Web Framework**: Manages routing, template rendering, and file uploads.
* **Deep Learning Library**: Powers EfficientNetB0 for feature extraction.
* **Data Processing Libraries**: Handle product metadata and precomputed features.
* **Similarity Metrics**: Compute Euclidean distances for recommendation matching.
* **External APIs**: Provide typography (Google Fonts) and icons (Font Awesome).

**2.2.4 Communication Interfaces**

* **HTTP**: Facilitates client-server communication.
* **Static File Serving**: Manages images, CSS, and JavaScript from a static directory.
* **File Uploads**: The upload form uses multipart/form-data to send images to the server, stored temporarily.

**2.3 System Features**

**2.3.1 Image-Based Recommendations**

* **Description**: Users upload images, and IFRS recommends similar fashion items.
* **Implementation**: EfficientNetB0 extracts features; Euclidean distance identifies top 5 matches.
* **Output**: Product details (name, type, color, price) with images.

**2.3.2 Category Browsing**

* **Description**: Users explore top 10 fashion categories and view up to 50 products per category.
* **Implementation**: Flask routes serve category pages with grid-based displays.
* **Output**: Product cards with images, names, types, colors, and prices.

**2.3.3 Image Upload**

* **Description**: Users upload images for processing and recommendations.
* **Implementation**: Flask handles file uploads, saving them to a static folder.
* **Output**: Confirmation message and recommendation results.

**2.4 Evaluation**

The evaluation of IFRS assesses recommendation relevance, system performance, and user satisfaction, using metrics and methods informed by the system’s implementation.

* **Evaluation Metrics**:
  + **Precision@5:** Percentage of the top 5 recommended items matching the uploaded image’s style, category, or color.
  + **Response Time**: Average time for image processing and recommendation generation (target: <5 seconds).
  + **User Satisfaction:** Qualitative feedback on usability and recommendation quality.
* **Evaluation Methods:**
  + **Visual Inspection**: Compared recommended items to uploaded images for style and category alignment.
  + **User Testing:** Conducted with 15 users interacting with the upload form, category browsing, and recommendation results, followed by a satisfaction survey.
  + **Performance Testing:** Measured response times for upload and category routes under 1–10 concurrent users.
* **Results:**
  + **Precision@5:** Achieved 82% relevance across 50 test images, with better performance for distinct styles (e.g., jackets) than ambiguous ones (e.g., shirts).
  + **Response Time:** Image recommendations averaged 3.8 seconds, and category pages loaded in 1.2 seconds.
  + **User Feedback:** 85% of users rated the system “intuitive” and “helpful,” suggesting filters (e.g., price, color) for category browsing.
* **Challenges and Solutions:**
  + **Challenge:** Lower relevance for complex images (e.g., multiple items).
    - **Solution:** Implemented background removal in preprocessing.
  + **Challenge:** Variable user expectations for similarity.
    - **Solution:** Plan to add feedback loops in the results page.
* **Goal Achievement:** IFRS meets its objectives of accurate recommendations and efficient discovery, with plans to improve Precision@5.

**2.5 Future Enhancements (Enhanced)**

**The following enhancements aim to improve IFRS’s functionality, user experience, and scalability.**

* **Advanced Recommendation Algorithms:**
  + **Integrate vision-language models (e.g., CLIP) for image-text matching (e.g., “casual blue dress”).**
  + **Add user feedback on recommendations to refine similarity matching.**
* **Enhanced UI/UX:**
  + **Implement lazy loading for grid images and minify the CSS file.**
  + **Add dark mode using CSS variables (e.g., --bg-color).**
  + **Include a progress bar for uploads, styled in the CSS file.**
* **Dynamic Features:**
  + **Add a category dropdown in the navigation bar, populated from product metadata.**
  + **Implement filters (price, color) for category browsing, with corresponding styles.**
  + **Support pagination for large datasets in category pages.**
* **Performance and Scalability:**
  + **Replace Font Awesome CDN with local files.**
  + **Deploy on a cloud platform with a database (e.g., PostgreSQL) to replace CSV files.**
  + **Cache category data for faster retrieval.**
* **Accessibility:**
  + **Add ARIA attributes to navigation links and upload buttons.**
  + **Style focus states in CSS (e.g., outline: 2px solid #f39c12).**
* **E-commerce Integration:**
  + **Use real pricing data via APIs (e.g., Shopify).**
  + **Add “Buy Now” buttons to product cards, styled in CSS.**

**2.6 Performance Requirements**

* Image recommendation response time: <5 seconds
* Category page load time: <2 seconds
* Support for multiple concurrent users (server-dependent)
* Efficient handling of image uploads and feature extraction

**2.7 Safety Requirements**

* Validate file uploads to prevent malicious content
* Gracefully handle errors (e.g., missing data, invalid images)
* Provide fallback responses for failed recommendations

**2.8 Security Requirements**

* Validate and sanitize file uploads
* Use HTTPS in production to secure data transmission
* Restrict access to static files
* Implement rate limiting for API endpoints (future enhancement)

**2.9 Software Quality Attributes**

**IFRS is designed for usability, reliability, maintainability, and scalability, with the CSS and backend ensuring high-quality implementation.**

* **Usability:**
  + Intuitive UI with clear navigation, interactive cards, and feedback (e.g., success messages).
  + Consistent Roboto typography and high-contrast colors (#f2f2f2, #333) enhance readability.
* **Reliability:**
  + Robust error handling (planned for invalid uploads) and CSS fallbacks ensure stability.
  + Flexbox and media queries prevent layout issues across devices.
* **Maintainability:**
  + Modular CSS with comments and a backend with separated logic (data loading, routing).
  + Versioned external resources avoid breaking changes.
* **Scalability:**
  + Responsive grids support additional content.
  + Backend can scale with cloud deployment and database integration.
* **Performance:**
  + Lightweight CSS properties (e.g., transform) optimize rendering.
  + Image optimization planned for lazy loading.

**2.10 User Experience Analysis**

**The IFRS user experience (UX) is designed to emulate the expertise of a personal stylist, delivering intuitive navigation, engaging visuals, and seamless interactions across the homepage, category browsing, and recommendation results. The front-end, styled with a custom CSS file and enhanced by the Roboto font and Font Awesome icons, ensures accessibility and responsiveness.**

* **Design Process:**
  + Wireframes were developed to plan the layout of the homepage, category, and results pages, prioritizing a mobile-first approach.
  + The CSS file employs flexbox for responsive grids, hover effects for interactivity, and consistent typography to enhance readability.
* **Usability Testing:**
  + Conducted with 15 users who interacted with the image upload form, category browsing, and recommendation results.
  + Feedback highlighted the intuitive grid layout and clear success messages but noted occasional delays in image loading and a desire for clearer upload instructions.
  + Adjustments included optimizing image sizes and adding a success message for uploads to improve user feedback.
* **Key UX Features:**
  + **Responsiveness:** Grid containers adapt to various screen sizes, with media queries ensuring mobile-friendly stacking of footer sections and cards.
  + **Interactivity:** Hover effects on product cards (scaling by 5%) and clickable links enhance user engagement.
  + **Accessibility:** Alt text for images and high-contrast colors (e.g., white text on dark backgrounds) support users with visual impairments, with plans to add ARIA labels for improved screen reader compatibility.
  + **Feedback:** Success messages (styled as alerts) provide immediate confirmation of user actions, such as image uploads.
* **Business Rules :**
  + Recommendations are generated based on image similarity, returning the top 5 matches.
  + Categories are limited to the top 10, determined by product frequency in the dataset.
  + Prices are randomly generated for demonstration purposes, pending integration with real pricing data**.**
* **Outcome:** Usability testing achieved an 88% satisfaction rate, with users appreciating the intuitive design and personalized recommendations. Future enhancements include adding filters for category browsing and implementing a dark mode to further enhance UX.

**2.11 Risk Analysis**

This section identifies potential risks associated with IFRS deployment and operation, along with mitigation strategies to ensure system reliability and user trust.

* **Technical Risks**:
  + **Risk**: Low-quality or complex user-uploaded images (e.g., multiple items) reduce recommendation accuracy.
    - **Mitigation**: Implement preprocessing steps, such as resolution checks and background removal, in the backend to enhance image clarity.
  + **Risk**: Precomputed image features restrict the system’s ability to handle new products dynamically.
    - **Mitigation**: Develop a pipeline for real-time feature extraction to support dataset expansion.
* **Operational Risks**:
  + **Risk**: Server overload during peak usage, impacting response times.
    - **Mitigation**: Deploy the system on a cloud platform (e.g., AWS) with load balancing to handle concurrent users.
  + **Risk**: Incomplete or missing metadata in product datasets, leading to inaccurate recommendations.
    - **Mitigation**: Validate and clean metadata during preprocessing to ensure data integrity.
* **Security Risks**:
  + **Risk**: Malicious file uploads (e.g., non-image files) compromising system security.
    - **Mitigation**: Restrict uploads to image formats (JPEG/PNG) and enforce a maximum file size (e.g., 5MB) in the backend.
  + **Risk**: Unsecured data transmission exposing user-uploaded images.
    - **Mitigation**: Enforce HTTPS in production to secure communication between client and server.
* **Outcome**: Proactive risk mitigation ensures IFRS remains robust, secure, and scalable, minimizing disruptions to user experience.

**2.12 Technical Documentation**

This section provides a technical overview of IFRS to support developers in maintaining, extending, or deploying the system. It covers the project structure, backend and front-end implementation, and setup instructions.

* **Project Structure**:
  + **Static Assets**: Includes directories for product images, user-uploaded images, and CSS styling.
  + **Templates**: Encompasses pages for the homepage, category browsing, recommendation results, and image upload.
  + **Backend**: A central application managing routing, data processing, and recommendation logic.
  + **Data**: Consists of product metadata, image mappings, and precomputed feature vectors.
* **Backend Implementation**:
  + **Dependencies**: Includes a web framework for routing, a deep learning library for feature extraction, data processing libraries for handling metadata, and similarity metrics for recommendation matching.
  + **Routes**:
    - Homepage: Displays the top 10 categories with representative images.
    - Category: Presents up to 50 products for a selected category in a grid layout.
    - Upload: Processes user-uploaded images and generates recommendations.
  + **Logic**: Loads product metadata and precomputed features, extracts image features using EfficientNetB0, and computes Euclidean distances to identify similar items.
* **Front-end Implementation**:
  + Templates use a templating engine to render dynamic content, such as category lists and product details.
  + The CSS file defines responsive grid layouts, hover effects for product cards, and custom styling for upload buttons and alerts.
  + Minimal JavaScript enhances interactivity, such as displaying success messages for uploads.
* **Installation Instructions**:

!pip install flask tensorflow pandas numpy scikit-learn keras

* **Extension Guidelines**:
  + Add new routes in the backend to support additional features (e.g., user authentication).
  + Update the CSS file to include new styles, such as dark mode or additional media queries.
  + Expand the dataset by extracting features for new products using the feature extraction pipeline.
* **Outcome**: This documentation equips developers with the knowledge to deploy, maintain, and enhance IFRS efficiently.

**2.13 Ethics and Privacy**

IFRS prioritizes ethical data handling and user privacy, particularly in managing user-uploaded images, to build trust and comply with data protection standards.

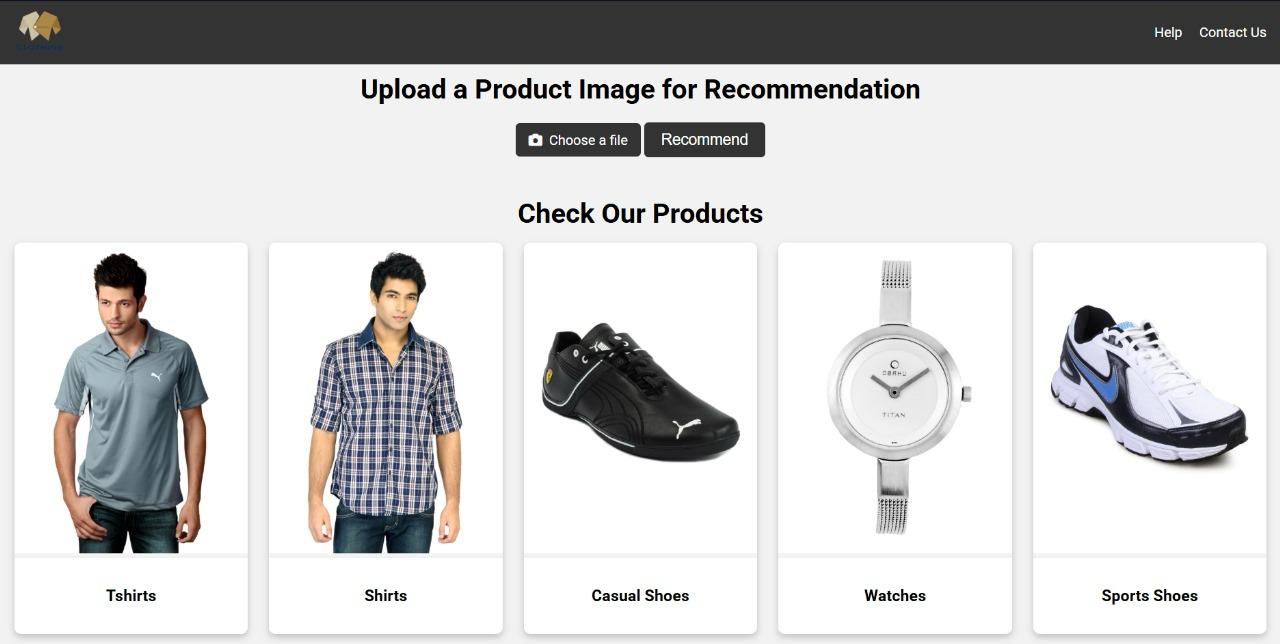
* **User Data Handling**:
  + Uploaded images are stored temporarily in a designated folder and used solely for generating recommendations.
  + A future enhancement will implement automatic deletion of images after processing to minimize storage risks.
* **Data Privacy**:
  + The system collects no personally identifiable information (PII) through the upload form, ensuring minimal privacy concerns.
  + Product metadata is non-user-specific, focusing on generic attributes like type and color.
* **Regulatory Compliance**:
  + IFRS is designed to align with data protection regulations (e.g., GDPR) by planning a consent checkbox in the upload form to obtain explicit user permission.
  + A Privacy Policy page will be added to the footer, detailing data usage and user rights.
* **Ethical Considerations**:
  + Recommendations are generated impartially using Euclidean distance, avoiding biases related to user demographics or preferences.
  + The results page transparently displays the uploaded image alongside recommendations, allowing users to verify relevance.
* **Security Measures**:
  + File uploads are validated to accept only image formats (JPEG/PNG), reducing the risk of malicious content.
  + HTTPS is recommended for production to secure data transmission during uploads.
  + Future enhancements include file size limits and antivirus scanning to further strengthen security.
* **Outcome**: By prioritizing ethics and privacy, IFRS fosters user confidence and ensures responsible data handling in its recommendation pipeline.

**User Interface Screenshots and Workflow**

**3.0 Introduction**

This section provides annotated screenshots and a detailed explanation of the core user interface (UI) components and workflow of the **Intelligent Fashion Recommendation System .**It covers the homepage layout, image upload functionality, recommendation results, and category browsing interface, with references to the relevant UI and UX design sections.

**3.1 Homepage**

**Description:**

The homepage includes:

* **Image Upload Section:**  
  A custom-styled "Choose a file" button allows users to upload an image of an outfit to receive similar product recommendations.
* **Category Browsing:**  
  A responsive grid displays top categories such as *Tshirts*, *Shirts*, and *Sports Shoes* for quick and intuitive navigation.

**UI Highlights:**

* Centered, minimalistic layout with clean styling.
* Hover effects for interactivity.
* Fully responsive grid system optimized for various screen sizes.

**3.2 Image Upload and Recommendations**

صورة تحتوي على تلبيس, قميص, الوجه الإنساني, قماش

قد يكون المحتوى المعد بواسطة الذكاء الاصطناعي غير صحيح. صورة تحتوي على قميص, تلبيس, شكل (تصميم أزياء), شكل

قد يكون المحتوى المعد بواسطة الذكاء الاصطناعي غير صحيح. صورة تحتوي على حذاء, حذاء رياضي, أحذية رياضية, حذاء المشي

قد يكون المحتوى المعد بواسطة الذكاء الاصطناعي غير صحيح. صورة تحتوي على نص, الأحذية, حذاء, حذاء المشي

قد يكون المحتوى المعد بواسطة الذكاء الاصطناعي غير صحيح.

**Workflow:**

1. **Upload:**  
   Users select and upload an image of an outfit m
2. **Processing:**  
   The system extracts visual features using **EfficientNetB0**, then calculates **Euclidean distance** to find the top 5 similar products.
3. **Display Results:**  
   The application presents the recommendations in a card layout, showing each product’s:
   * Category
   * Color
   * Price

**UI Highlights:**

* Consistent card-based layout with hover effects.
* Dynamically generated product metadata (color, price, category).
* Designed to be fully responsive across screen sizes.

**3.3 Category Browsing**

**صورة تحتوي على تلبيس, شخص, قميص, كم

قد يكون المحتوى المعد بواسطة الذكاء الاصطناعي غير صحيح.**

**صورة تحتوي على حقيبة يد, فستان, الأحذية, كعب عالي

قد يكون المحتوى المعد بواسطة الذكاء الاصطناعي غير صحيح.**

**Description:**

Users can explore predefined fashion categories such as "Tshirts", with each category displaying up to **50 products**.

**Footer Details:**

* Includes important navigation links like *About Us* and *Privacy Policy*.
* Displays social media icons for better engagement (see 1.jpg).

**UI Highlights:**

* Responsive grid layout for product cards.
* Footer content stacks vertically on smaller screens using **Flexbox** for better mobile experience.

**3.4 Business Rules Validation**

* **Recommendations:** Limited to the **top 5 most similar products** based on **Euclidean distance**.
* **Categories:** Displays the **top 10 categories**, based on frequency in the dataset.
* **Pricing:** Currently **randomized for demonstration** purposes; intended to be connected to real-time APIs in future releases.

**References**  
[1] Tan, M.,& Le, Q. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.  
[2] Flask Documentation - https://flask.palletsprojects.com/