



Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department



TA'AM

July 2024



Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department



TA'AM: Used-clothes Application

Under Supervision of:

Dr. Salsabil Amin [BS]

TA. Ahmed Hatem

By:

Saleh Adel Saleh

Shehab Mostafa Fahmy

Abdelaziz Hossam Abdelaziz

Abdulrhman Hosny Muhammed

Muhammed Mahros Muhammed

Acknowledgment

First and foremost, we express our deepest gratitude to Allah, the Most Merciful and Compassionate, for granting us strength, wisdom, and perseverance throughout this journey. His guidance has been the cornerstone of our efforts, and we are humbled by His blessings.

We would like to extend our heartfelt appreciation to our families, whose unwavering support and encouragement have been invaluable. Their patience, understanding, and belief in our abilities have been a constant source of motivation.

A special thank you to our supervisors, *Dr. Salsabil Amin* and *TA. Ahmed Hatem*, for their mentorship, expertise, and invaluable guidance. Their insightful feedback and dedication have played a crucial role in shaping this documentation.

Finally, we would like to acknowledge our friends and everyone who supported us along the way. Whether through their videos, papers, or contributions on social media, their knowledge and resources have been instrumental in our research and documentation process.

Abstract

The clothing industry contributes heavily to global carbon emissions and wastewater. This project introduces *Ta'am*, a platform promoting the resale and reuse of clothes to reduce environmental impact. *Ta'am* aims to empower users to buy and sell used garments, support local traders by showcasing their products, and improve user experience using advanced machine learning models.

This application simplifies selling items by using machine learning to extract details from uploaded images, eliminating the need for users to manually input information. It ensures image quality and offers various search options. When a user uploads a product image, the system automatically creates a post with extracted attributes like category, color, and size, if the image meets quality standards. This process involves assessing image quality, detecting objects in the image, validating the content, estimating key points, calculating size, and detecting colors. Additionally, users can search for items using an image, with the system returning the most relevant matches based on image quality and similarity metrics.

The app employs the [BRISQUE] model to assess image quality and uses image processing techniques to extract relevant details. For classification and validation tasks, [ResNet34] is employed, while [ResNet50] is used to estimate key points. Color detection is achieved using [K-means] clustering. Features are extracted using [MobileNet], and Cosine Similarity is employed to find the closest matches during searches. These technologies are trained on datasets like [DeepFashion2] and [Agrigorev's Custom Dataset] to improve accuracy and functionality.

Table of Contents

Acknowledgment	I
Abstract	II
Table of Contents	III
List of Figures	V
List of Tables	VII
List of Abbreviations	VIII
1- Introduction	1
1.1 Motivation	1
1.2 Problem Definition	2
1.3 Objective	2
1.4 Time Plan	2
1.5 Document Organization	3
2- Literature Review	4
2.1 Project Overview	4
2.2 Theoretical Background	6
2.3 Related Work	9
2.3.1 Clothes Category Classification	9
2.3.2 Clothes Landmark Estimation	13
2.3.3 Size Calculations using a reference object	14
3- Analysis and Design	16
3.1 System Overview	16
3.1.1 System Architecture	16
1. Presentation Layer:	16
2. Logic Layer:	17
3. Data Layer:	23
3.1.2 System Users	24

3.2 System Analysis and Design	25
3.2.1 Use Case Diagram	25
3.2.2 Class Diagram	26
3.2.3 Database Diagrams	27
4- Implementation and Results	29
4.1 Datasets	29
4.1.1 DeepFashion2 Dataset	29
4.1.2 Agrigorev's Custom Dataset	30
4.2 Software Tools	31
4.3 Setup Configurations	32
4.4 Experiments and Results	33
4.4.1 Dataset Selection	33
4.4.2 Category Classification Experiments and Results	34
4.4.3 Key-points Estimation Experiments and Results	35
4.4.4 Color Detection Experiments and Results	37
4.4.5 Feature Extraction Experiments and Results	37
4.4.6 Real-life Scenarios Experiments and Results	38
5- User Manual	39
6- Conclusion and Future Work	52
6.1 Conclusion	52
6.2 Future Work	53
References	54

List of Figures

Figure 1.1 : Garments Operations Chart.....	1
Figure 1.2 : Time Plan Chart.....	2
Figure 2.1 : Quality Assessment Chart.....	4
Figure 2.2 : Attributes Extraction Chart.....	5
Figure 2.3 : Search Capabilities Chart.....	5
Figure 2.4 : Convolutional Layer.....	7
Figure 2.5 : Data Augmentation.....	8
Figure 2.6 : Category Classification and Validation.....	9
Figure 2.7 : CAD full architecture [16].....	10
Figure 2.8 : Basic key points for different types of garment [13].....	13
Figure 2.9 : Estimate size using reference object [19].....	14
Figure 2.10 : Manual measurements [13].....	15
Figure 3.1 : System Architecture [13].....	16
Figure 3.2 : Category Classification and Validation example.....	19
Figure 3.3 : Waist Equation example.....	21
Figure 3.4 : Color Detection.....	22
Figure 3.5 : Use Case Diagram.....	25
Figure 3.6 : Class Diagram.....	26
Figure 3.7 : ERD Diagram.....	27
Figure 3.8 : Schema Diagram.....	28
Figure 4.1 : DeepFashion2.....	29
Figure 4.2 : Agrigorev's custom dataset.....	30
Figure 4.3 : Features extraction result.....	38
Figure 4.4 : Search capabilities result.....	38
Figure 5.1 : Log In Page.....	39
Figure 5.2 : Register Page.....	40
Figure 5.3 : Forgot Password Page.....	41

Figure 5.4 : Home Page	42
Figure 5.5 : Filter Page	43
Figure 5.6 : Sort Page	44
Figure 5.7 : Product Details Page	45
Figure 5.8 : Chats Page	46
Figure 5.9 : Chat Room Page	47
Figure 5.10 : My Ads Page	48
Figure 5.11 : Account Page	49
Figure 5.12 : Sell Page	50
Figure 5.13 : Search Page	51

List of Tables

Table 2.1 : ACS dataset properties	9
Table 2.2 : CNN with CAD and Fashion Dataset comparison	11
Table 2.3 : Performance of category classification using FashionNet [17]	12
Table 2.4 : Garment measurement error in cm [13]	15
Table 3.1 : mAP and MSE for each category	20
Table 3.2 : Features extraction models comparison	23
Table 4.1 : Classification models comparison	34
Table 4.2 : Key-points Estimation model results	36
Table 4.3 : Key-points Estimation trials summary	36
Table 4.4 : Color Detection models comparison	37
Table 4.5 : Features Extraction models comparison	37

List of Abbreviations

Abbreviation	Description
CNN	Convolution Neural Networks
SVR	Support Vector Regression
CRF	Conditional Random Fields
mIoU	Mean Intersection Over Union
ReLU	Rectified Linear Unit
GSCNN	Global Semantic Context CNN
ANOVA	Analysis of Variance
ACS	Apparel Classification with Style
HOG	Histogram of Oriented Gradients
SURF	Speeded Robust Features
LBP	Local Binary Patterns
CAD	Clothing Attributes Dataset

1- Introduction

The clothing industry ranks **fourth** in environmental impact, responsible for **10%** of global carbon emissions and **20%** of wastewater. To address this, extending the lifespan of clothes and choosing second-hand over new ones can significantly reduce the industry's carbon footprint and wastewater contributions.

1.1 Motivation

Creating a platform for users to sell, and buy clothes, particularly used garments, and providing local traders with a dedicated marketplace for showcasing their products is a compelling and innovative concept. **Ta'am** addresses two vital aspects. Firstly, it empowers individuals to extend the lifespan of their clothing, contributing to a more sustainable fashion industry by promoting the resale and reuse of garments. Secondly, it supports local traders who often lack a dedicated space to showcase their products. By doing so, **Ta'am** not only fosters economic growth but also strengthens local communities by bringing people together.

Users will have the flexibility to buy and sell their clothing and engage in sustainable practices such as exchange, rental, and even donation. The significance of these features is underscored by a recent survey **Figure 1.1.**, which revealed that a substantial number of people were not yet participating in these sustainability-driven activities. This survey data underscores the urgency for our project to provide awareness about the importance of sustainability in the fashion industry.

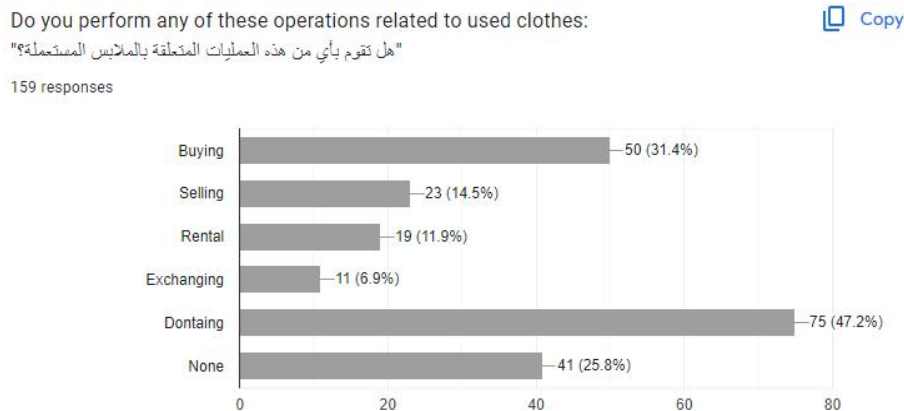


Figure 1.1: Garments Operations Chart

1.2 Problem Definition

Enhancing the user experience by analyzing and optimizing an image captured by the user to showcase their product.

1.3 Objective

- Offer a specialized platform for users to allow them to showcase their surplus clothing and buy suitable ones at a relatively low cost.
- Enhance the user experience by implementing multiple machine learning models to assess image suitability, extract various attributes, and offer diverse search capabilities.

1.4 Time Plan

Figure 1.2 shows the overall project time plan. The time plan manifests the main stages of the project and their corresponding period. The project process starts with the literature review and continues until the final product. As seen, some tasks overlap. Finally, the project documentation is an ongoing process.

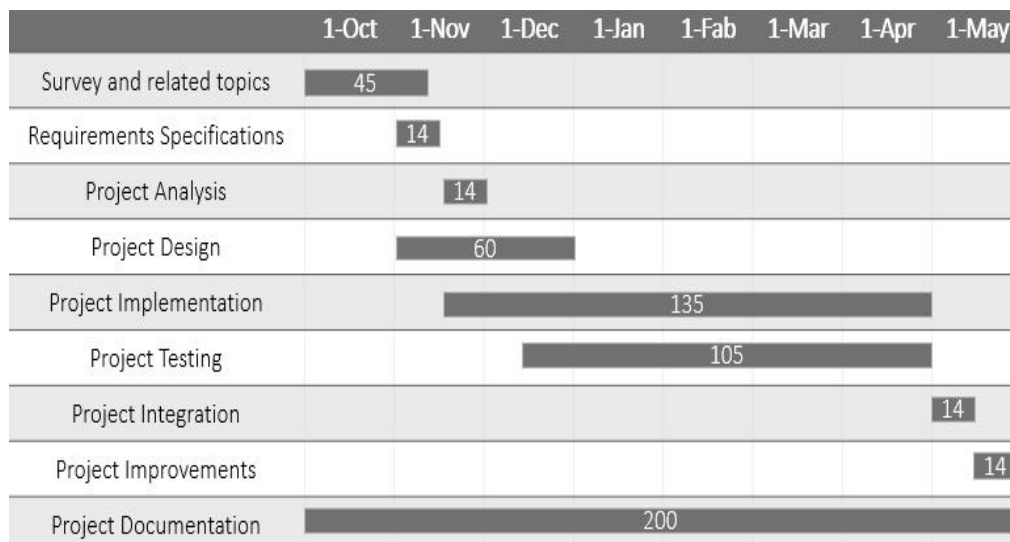


Figure 1.2: Time Plan Chart

1.5 Document Organization

- **Chapter 2:** This chapter delves into the project's foundation in science, and its intended application, and provides a concise overview of comparable initiatives.
- **Chapter 3:** This chapter provides an overview of the system, including its architecture and users. It also covers the analysis and design of the system, featuring diagrams such as the use case diagram, class diagram, ERD diagram, and Schema diagram.
- **Chapter 4:** This chapter focuses on the implementation of the system and presents results. It covers aspects like datasets, software tools used, software setup configuration, and details of experiments conducted. The experiments include real-life scenarios with corresponding results.
- **Chapter 5:** In this chapter, you'll find a comprehensive user's guide detailing step-by-step instructions on how to navigate the system, accompanied by screenshots for clarity.
- **Chapter 6:** In closing, this chapter wraps up the documentation and proposes potential future enhancements to maximize the project's capabilities.

2- Literature Review

2.1 Project Overview

Nowadays, everyone has smartphones, and lots of people buy stuff online. But even though many online stores sell all kinds of things, there's no special app just for clothes. This is surprising because the clothing industry is huge. Seeing this gap in the market gives us a chance to make an app that focuses only on clothes. It would be perfect for people who love fashion and want an easy way to shop for clothes online.

Our project relies on cutting-edge Convolutional Neural Networks (CNNs), which are smart computer models. They change how people use our platform to improve their experience. Let's talk about three important things that make our project special: checking image quality, picking out details, and making it easy to search for stuff.

Figure 2.1, Here are the survey results showing how much users value our feature for checking image quality. The chart shows that most people agree it's important to have high-quality images for a better experience.



Figure 2.1: Quality Assessment Chart

Turning to **Figure 2.2**, A graph showing what users think about using CNNs to automatically pick out details like size and price. People agree it's important to make things easier for sellers and improve how well the platform works overall.



Figure 2.2: Attributes Extraction Chart

Figure 2.3, Here's a visual representation of what users think about our advanced search features. It shows how easy and efficient users find it to search for specific clothes by uploading images. This highlights a big step forward in making online clothes shopping easier for everyone.



Figure 2.3: Search Capabilities Chart

2.2 Theoretical Background

Computer Vision:

In the garment industry, computer vision plays a pivotal role in revolutionizing various processes, from design to production and retail. Leveraging advanced algorithms and image processing techniques, computer vision enables the automatic inspection of fabric quality, pattern recognition, and even the precise measurement of garments. This technology not only enhances efficiency in quality control but also aids in the creation of virtual prototypes, streamlining the design process and reducing time to market [1].

Convolutional Neural Network:

A Convolutional Neural Network (CNN) is a type of artificial neural network that has proven powerful in computer vision tasks, by having higher accuracy and in many cases higher efficiency compared to traditional computer vision techniques. CNNs are built with different building components, the three most common ones being **convolution layers**, **pooling layers**, and **fully connected layers**. The convolution and pooling layers tasks are to extract features while the fully connected layers' task is to map features to the final output [2].

Convolutional Layer:

CNN's high performance in computer vision is dependent on how they extract features, which is done through a linear operation called convolution. In image processing, the linear operation will analyze each pixel of the image to be able to extract features, and since features in images can appear anywhere this convolution is a very efficient way of analyzing images. This thesis makes use of CNNs when conducting semantic image segmentation and thus all examples from here on will be of images and how CNNs can be used to process them [2].

Figure 2.4 visualizes the linear operation convolution, where the input tensor is an image and the kernel is the matrix of weights. The input tensor

and the kernel are multiplied element-wise, and the resulting matrix is summed, giving the value of the corresponding position in the feature map.

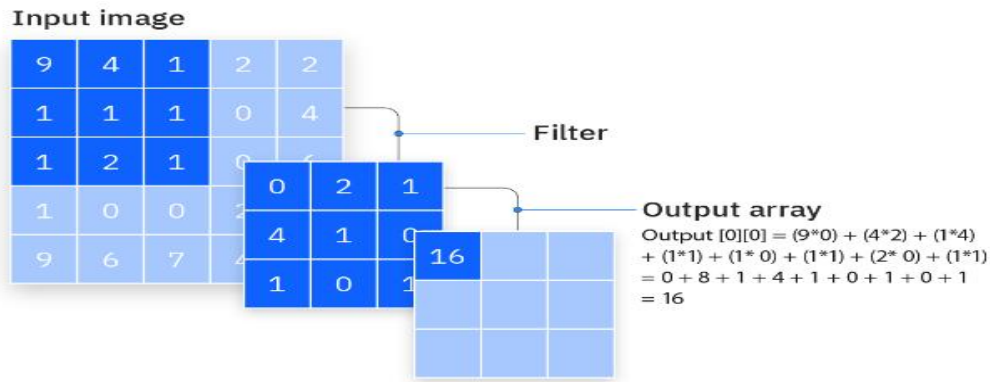


Figure 2.4: Convolutional Layer

Pooling Layer:

Convolutional layers scrutinize individual pixels to extract an image's feature maps, a process heavily influenced by the image's structure and dimensions. Pooling layers then down-sample these maps, retaining vital features while reducing their in-plane dimensionality, thus cutting down on network parameters and computational complexity [2].

Fully Connected Layer:

The fully connected layer directly links every node from its input to the output layer, unlike partially connected layers. In a neural network, this layer processes extracted features for classification. While preceding layers often use **ReLU** functions, fully connected layers typically employ **Softmax** activation, generating probabilities between 0 and 1 for accurate input classification [2].

Object Classification:

Object classification refers to the task of assigning a predefined category or label to an object based on its characteristics or features. In the context of machine learning and computer vision, object classification involves training a model to recognize and categorize objects in images or videos [3].

Key-Points Detection:

Key-point detection is a computer vision task that aims to identify the location of an object – often a person – and key points within the identified area (i.e. legs, arms, head). Key-point detection is at the heart of many cutting-edge technologies, enabling applications from facial recognition in smartphones, assisting in object tracking for autonomous vehicles, or aiding in medical image analysis [4].

Data Augmentation:

In machine learning in general and deep learning in particular, data augmentation is the process of transforming training data in different ways to create better training data, to either speed up convergence or minimize the risk of overfitting and creating a more general model. The traditional way to transform data within image processing is to change the geometry or color of images, for instance by rotating images, changing color space, or cropping the image **Figure 2.5**. These transformations are all affine transformations with regard to the original image, meaning that all points and lines in the image are kept intact in relation to each other [5].



Figure 2.5: Data Augmentation

Transfer Learning:

The intricacy of deep neural networks often makes full training impractical. Instead, utilizing pre-trained models with subsequent fine-tuning using relevant data is a common approach [6]. This practice, known as transfer learning, has demonstrated that even if the pre-trained models were initially trained for disparate tasks or datasets, they can outperform initializing the model with random weights.

2.3 Related Work

This section discusses the commonly used approaches in Classification, and Key-points Estimation.

2.3.1 Clothes Category Classification

Clothing type classification is the multi-class classification problem of predicting a single label that describes the type of clothing within an image. Thus, clothing-type datasets will include images of clothing annotated with a label such as a hat, jacket, or shoe **Figure 2.6**.



Figure 2.6: Category Classification and Validation

- 1) Apparel Classification with Style (ACS) dataset, which contains **89,484 images** that have been cropped based on bounding boxes aimed at encapsulating the clothing on an individual's upper body. Each image is labeled with one of **15** hand-picked clothing categories **Table 2.1**.

Table 2.1: ACS dataset properties

Category	Images	Boxes	Category	Images	Boxes	Category	Images	Boxes
Long dress	22,372	12,622	Suit	12,971	7,573	Shirt	3,140	1,784
Coat	18,782	11,338	Undergarment	10,881	6,927	T-shirt	2,339	1,784
Jacket	17,848	11,719	Uniform	8,830	4,194	Blouses	1,344	1,121
Cloak	15,444	9,371	Sweater	8,393	6,515	Vest	1,261	938
Robe	13,327	7,262	Short dress	7,547	5,360	Polo shirt	1,239	976
						Total	145,718	89,484

Bossard et al. (2012) [9] used the **ACS** dataset to extract features including Histogram of Oriented Gradients (**HOG**), Speeded Robust Features (**SURF**), Local Binary Patterns (**LBP**), and color information. Bossard then used these features to perform multiclass classification with **One vs. All SVM**, **random forests**, and **transfer forests**, achieving average accuracies of **35.03%**, **38.29%**, and **41.36%**, respectively. CNN exceeded these accuracy baselines on the ACS dataset when Lao, Brian, and Karthik A. Jagadeesh (2015) [10] used the standard **AlexNet** Convolutional network which had been trained using ImageNet and achieved a **50.2%** accuracy on the test set.

- 2) Fashion dataset and Clothing Attribute dataset (CAD), which contain (5400 images, 5 classes) and (1575 images, 6 classes) respectively. Another CNN model was proposed by S. S. Islam, E. K. Dey, M. N. A. Tawhid, and B. M. M. Hossain (2017) [11], based on **AlexNet**, to observe the performance and effectiveness of deep features by a total of nine learned layers. Among these layers, five of these layers are Convolutional layers and the remaining four are Fully Connected layers as shown in **Figure 2.7**.

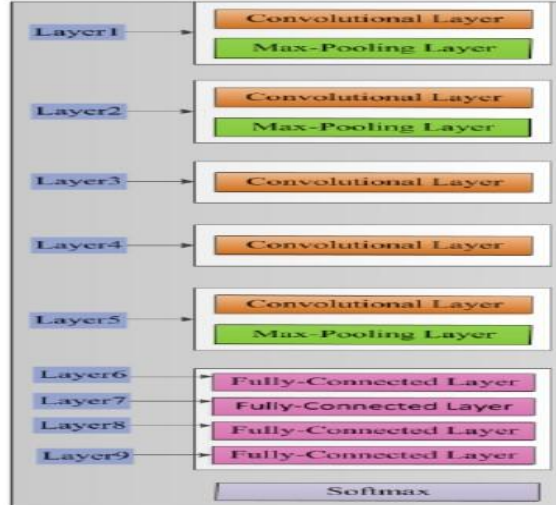


Figure 2.7: CAD full architecture [16]

This research mainly experimented on two existing deep convolutional neural network models alongside the proposed model on the Fashion dataset and Clothing Attribute dataset (CAD). The performance of the proposed deep learning model has been compared with the existing models and also with some existing well-known Hand-Engineering feature extraction approaches for garment design class identification. **Table 2.2** shows the experimental results of different methods for CAD and the Fashion dataset.

Table 2.2: CNN with CAD and Fashion Dataset comparison

Method	Accuracy	
	CAD	Fashion Dataset
HOG	63.76%	79.15%
GIST	72.31%	81.67%
LGP	65.55%	79.79%
CENTRIST	71.97%	79.72%
tCENTRIST	74.48%	84.07%
cCENTRIST	74.97%	84.23%
NABP	74.18%	83.22%
ALEXNET	75.6%	81.8%
VGG_S	76.8%	82.9%
Proposed Model	77.8%	84.5%

- 3) **DeepFashion** dataset, which contains over 800,000 images, richly annotated with massive attributes, clothing landmarks, and correspondence of images taken under different scenarios including store, street snapshot, and consumer. To demonstrate the advantages of **DeepFashion**, Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang (2016) [12] proposed a new deep model, named **FashionNet** using VGG16 architecture as a backbone. Performance of category classification using **FashionNet** **Table 2.3**.

Table 2.3: Performance of category classification using FashionNet [17]

	Category	
	top-3	top-5
WTBI [3]	43.73	66.26
DARN [10]	59.48	79.58
FashionNet+100	47.38	70.57
FashionNet+500	57.44	77.39
FashionNet+Joints [34]	72.30	81.52
FashionNet+Poselets [34]	75.34	84.87
FashionNet (Ours)	82.58	90.17

Top-3 Accuracy:

- For each test sample, the model generates a list of probabilities for each class.
- The predicted class is considered correct if the true class label is among the top three classes with the highest probabilities.
- The top-3 accuracy is the percentage of test samples for which the correct label is within the top three predicted labels.

Formula: (Number of test samples where the correct label is in the top 3 predictions) / (Total number of test samples)

Top-5 Accuracy:

Similar to top-3 accuracy, it considers the top five predicted labels.

Tatiana Sennikova (2021) [13] trained a model using the DeepFashion dataset and pre-trained **ResNet34** model and achieved a **Top-3** Accuracy of **88.6%**, which is 6% higher than the benchmark accuracy and a **Top-5** Accuracy of **94.1%**, which is 4% higher than the benchmark accuracy. This should not come as a surprise as the authors of the original paper used VGG16 architecture as a backbone, which is a less powerful model. On loading 98 user-specified images, the **Top-1** Accuracy of the model is **62.4%**.

2.3.2 Clothes Landmark Estimation

Paulauskaite-Taraseviciene et al. [8] show that the most challenging task is key point detection as it directly depends on segmentation results and different garment types have different key points as shown in **Figure 2.8**.

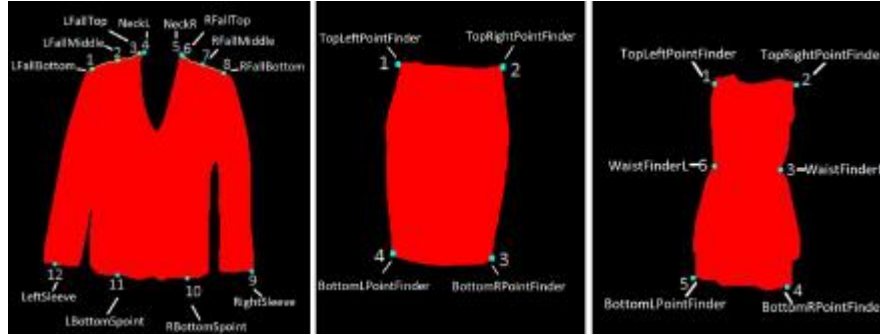


Figure 2.8: Basic key points for different types of garment [13]

the study outlines a systematic approach to garment key-point estimation, leveraging a set of algorithms that are adaptable to different types of garments through parameter tuning and sensitivity adjustments.

The overall process of the algorithms for key-point estimation includes:

- Identification of the bounding box and outermost points of the garment contour.
- Detection of garment angles, shapes, and changes in (x, y) coordinates.
- Key points prediction based on auxiliary algorithms for desired locations.
- Final prediction that sets the sensitivity factor for the algorithms to adapt to the specific garment type.

2.3.3 Size Calculations using a reference object

Adrian Rosebrock. [14] Published an article that presents an approach to automatic size detection using the dimensions of a known-size object at the same image and the same surface **Figure 2.9**.

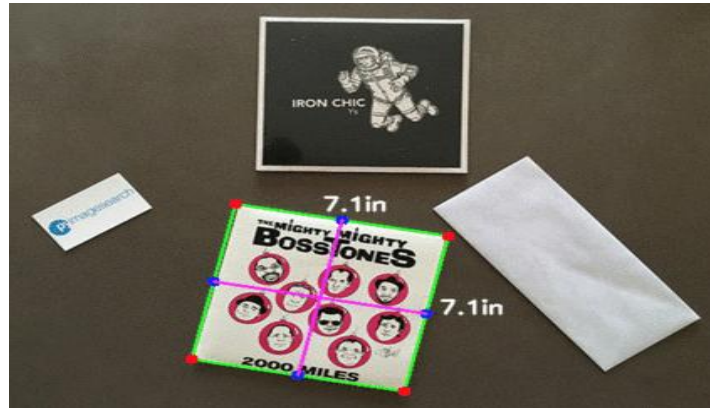


Figure 2.9: Estimate size using reference object [19]

The article shows that measuring the size of objects is similar to computing the distance from our camera to an object in both cases, we need to define a ratio that measures the number of pixels per a given metric we call it pixel per metric.

In order to determine the size of an object in an image we first need to perform a calibration using a reference object and it should have two important properties. The first property the dimensions of the object in a measurable unit. The second property is the reference object should be uniquely identified either by the placement of the object or the appearance of the object.

Manual garment measurements can be time-consuming and prone to human error, with measurements varying by up to 3 cm in different areas of clothing. An empirical experiment involving 20 participants aged 22-58 found that even with clear instructions, errors occurred in measuring waist and skirt length for skirts and shoulder width, overall jacket length, and sleeve length for men's jackets **Figure 2.10**. This highlights the need for automated segmentation and measuring systems to improve accuracy and efficiency in garment measurement.

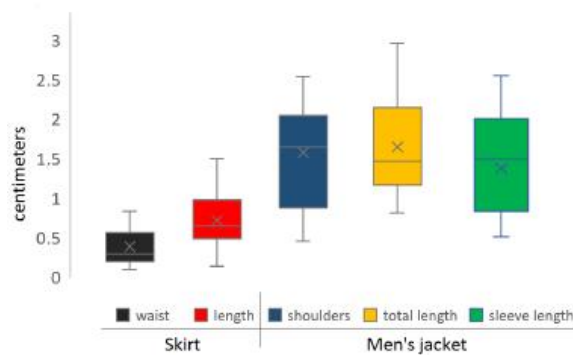


Figure 2.10: Manual measurements [13]

By doing the previous approaches In automated measurement experiments involving blazers, skirts, and dresses Table 2.2, the average measurement errors were found to be 1.27 cm for blazers, 0.747 cm for dresses, and 1.012 cm for skirts. These results are promising because they fall within industry-acceptable measurement errors of up to 2 cm **Table 2.4**, and they represent a significant improvement over the potential 3.02 cm human measurement error.

Table 2.4: Garment measurement error in cm [13]

	Total Length	Waist	Shoulders	Sleeves	Average Error
Dresses	1.113	0.343	0.783	-	0.747
Blazers	0.903	-	1.826	0.652	1.127
Skirts	1.650	0.421	-	-	1.012

3- Analysis and Design

3.1 System Overview

3.1.1 System Architecture

Figure 3.1 visualizes the system architecture of the project. This system architecture is composed of three layers. They are divided into the Presentation layer, Logic layer, and Data layer.

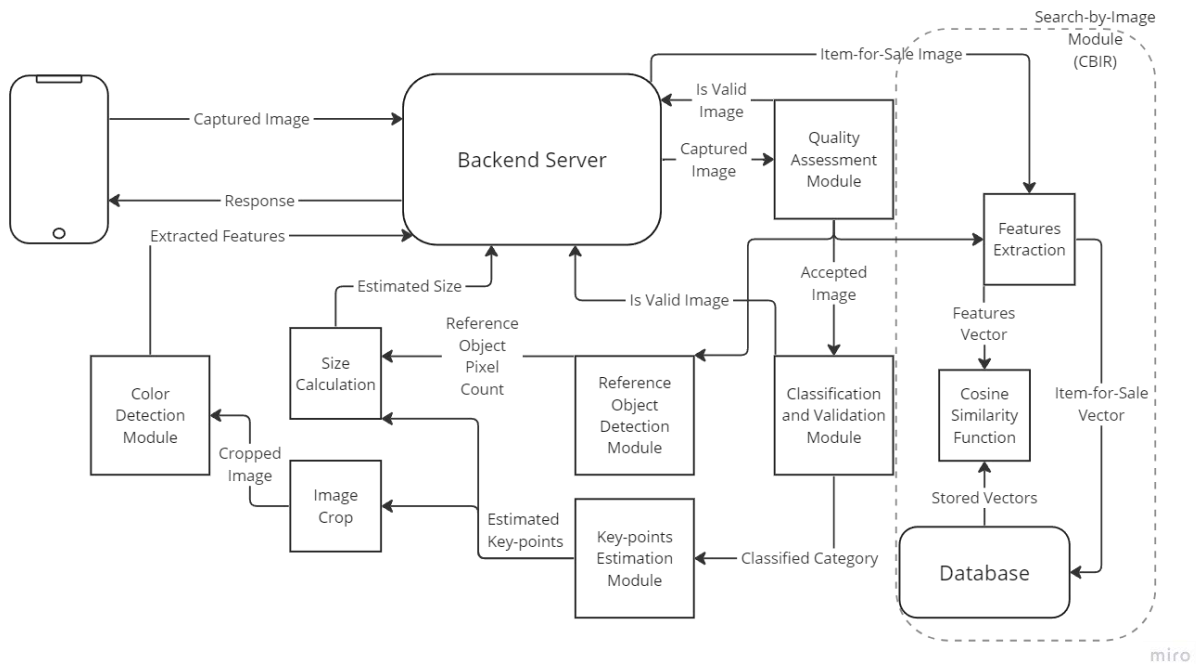


Figure 3.1: System Architecture [13]

1. Presentation Layer:

User Interface and communication layer of the application where the end-user interacts with the application.

The system functions by receiving images for product posting or image-based search from users. It then transmits this data to the logic layer (server) for analysis. After processing, the system retrieves and presents the analyzed data to the user.

2. Logic Layer:

Information collected from the presentation layer is processed by the included models to apply attribute extraction and facilitation of search functionalities.

Scenario One:

When the user uploads a product image the system takes the image as input and produces a post of the product with the attributes extracted “Category, Color, Size” in case the image satisfies the image quality assessment criteria.

Pipeline:

- Quality Assessment
- Reference Object Detection
- Classification and Validation
- Key-points Estimation
- Size Calculation Equation
- Color Detection

Scenario Two:

Where users initiate searches using images, the system returns the closest images to the query in case the image satisfies the image quality assessment criteria.

Pipeline:

- Quality Assessment
- Classification and validation
- Feature Extraction
- Cosine Similarity

Quality Assessment:

- Using the **BRISQUE** model, the image quality score was computed, and a specific threshold of 30% was applied for image assessment.
- The **Brisque** model is a no-reference image quality assessment algorithm designed to predict the perceived quality of images without comparing them to reference images.
- The **Brisque** model extracts statistical features from images, such as contrast, brightness, and texture. It then utilizes a machine learning algorithm, typically based on Support Vector Regression (SVR), to map these features to perceived image quality scores. This mapping is learned from a large dataset of images manually rated for quality.

Reference Object Detection:

- The process of extracting the ID card involves contour analysis, a vital technique in image processing and computer vision. Contour analysis entails enhancing the image using methods like thresholding, edge detection, and morphological operations to isolate regions of interest and reduce noise.
- Key steps include:
 - **Edge Detection:** Utilizing methods such as Canny or Sobel to detect edges within the image.
 - **Morphological Operations (Dilation):** Expanding object boundaries by adding pixels, useful for tasks like filling gaps or enlarging objects.
 - **Gaussian Blur:** Reducing image noise and smoothing details by applying a Gaussian function.
 - **Conversion to HSV Color Space:** Representing the image based on Hue, Saturation, and Value components.
 - **Processing on S Channel:** Applying Gaussian Blur and Sobel edge detection on the Saturation channel.

- **Dilation:** Expanding the edges obtained from the Sobel operation.
 - **Thresholding:** Applying threshold values to the grayscale channel and performing contour analysis to identify regions resembling the ID card.
 - **Masking:** Obtaining the mask of the largest contours, which represents the ID card.
- This methodology effectively isolates the ID card from the image by focusing on its contours and relevant features.

Classification and Validation:

- This model validates whether the image contains clothes or not, based on that, it returns the classified category or “Other”.
- In the case of “Other”, the image is rejected. **Figure 3.2**
- We utilized the **ResNet34** architecture on **Agrigorev’s custom dataset** for category classification.
- We got an accuracy of 88.5% after applying 50 epochs.



Figure 3.2: Category Classification and Validation example

Key-points Estimation:

- **13 pre-trained ResNet50** models were employed to extract key points **for each category**.
- The height and width of the product were determined by calculating the distances between specific key points.
- **Table 3.1** shows mAP and MSE for each category.

Table 3.1: mAP and MSE for each category

Category	mAP	MSE
short sleeve top	0.8111	0.0588
skirt	0.8646	0.0652
vest	0.7933	0.0622
vest dress	0.8271	0.0570
short sleeve dress	0.7920	0.0630
short sleeve outerwear	0.6566	0.0989
trousers	0.8624	0.0605
shorts	0.8936	0.0578
sling dress	0.7215	0.0814
long sleeve dress	0.6278	0.1069
long sleeve outerwear	0.6930	0.0949
sling	0.6897	0.0840
long sleeve top	0.6906	0.0932
short sleeve top	0.8111	0.0588

Size Calculation Equation:

- An EN 13402 Standard for clothing size designation was used to map the measurements from centimeters to real-world measurements “S, M, L, XL”.
- **Figure 3.3** shows an example of the equation applied to get the waist of trousers.

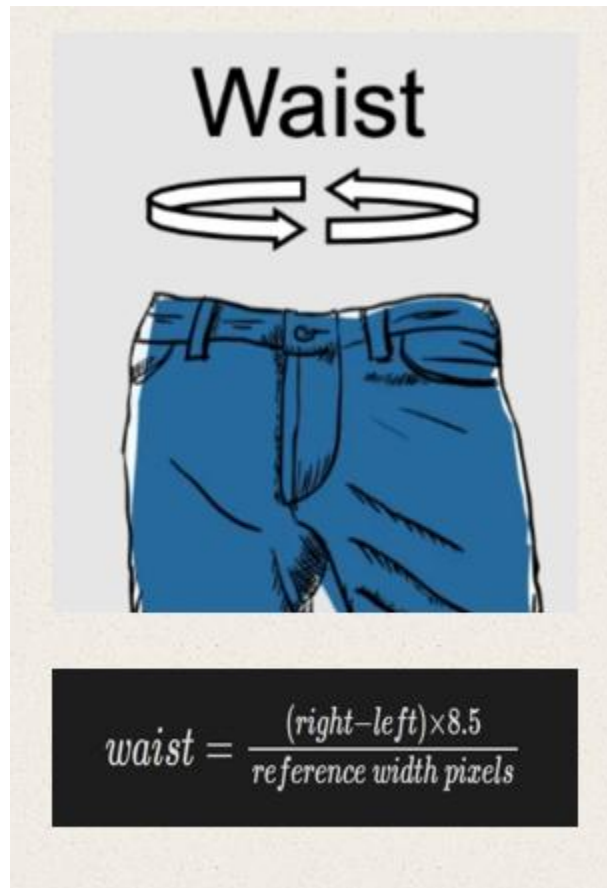


Figure 3.3: Waist Equation example

Color Detection:

- The image was cropped using the detected key points to **ignore background colors**.
- **K-Means** is used to segment the image into regions of similar colors and identify representative colors for each region, forming the color palette.
- The execution time to extract the most three dominant colors is approximately **0.1 seconds**.
- **Figure 3.4** shows an example of the results.

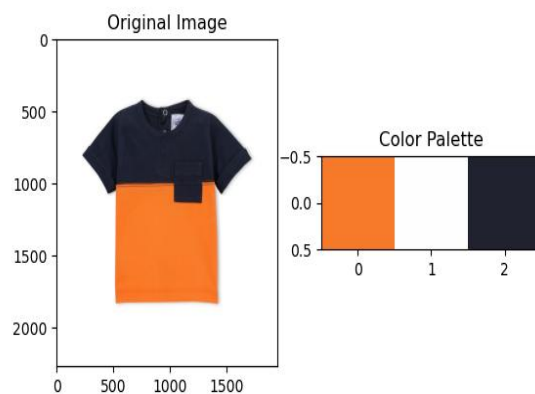


Figure 3.4: Color Detection

Features Extraction:

- The **MobileNet** model extracts features from the query image, generating a **1 x 50176-dimensional vector** within a time frame of **20ms to 50ms**.
- Utilizing **cosine-similarity metrics**, the closest 5 products to the query image are calculated. Comparing the query image with **5000 images**, from the closest to the farthest, typically takes approximately **0.7s to 1s**.
- The **vector dimensionality is reduced to 5000** using an AvgPooling Layer followed by a Dense Layer, resulting in the model being **10 times faster**.
- Comparison is **limited to the same category** rather than all categories, making the model **13 times faster**.
- Overall, the model now takes around **30ms to 50ms** to execute.
- **Table 3.2:** Shows the comparison between the models used.

Table 3.2: Features extraction models comparison

	VGG16	ResNet50	MobileNet
Feature Extraction per Image	1s to 2s	20ms to 50ms	20ms to 50ms
Vector Dimension	25088	100352	50176
Cosine-similarity with 5000 Images	0.4s to 0.7s	1.5s to 1.8s	0.7s to 1s

3. Data Layer:

The data access is where the information processed by the application is stored and managed. The users' data (username, password, history, etc, are stored using Firebase. There is a local database used as well to store the post details and work with the machine-learning models.

3.1.2 System Users

- **Intended Users:**

Ensuring the quality and validity of the products being posted for sale, auto-filling in the attributes of the product, and easing the process of searching for a specific item is done in **Ta'am** to make these processes easier for the intended users who are:

- **Local Traders:**

Local traders often struggle to find a suitable platform to showcase their products and enhance their income via social media channels. **Ta'am** offers them a seamless avenue to display their offerings effectively, attracting potential buyers and thereby fostering the growth of their businesses.

- **Casual Users:**

This category comprises individuals who find themselves with surplus clothing items that they wish to monetize by selling. **Ta'am** offers a convenient solution for them to easily list their items for sale, maximizing their earning potential from their excess inventory.

- **Casual buyers:**

These buyers typically hunt for specific items at lower prices, often preferring pre-owned or slightly flawed goods. **Ta'am** simplifies their search process, enabling them to swiftly find desired items and potentially save money on their purchases.

- **User Characteristics:**

- Basic knowledge of smart-phone use
 - Internet connection

3.2 System Analysis and Design

3.2.1 Use Case Diagram

Figure 3.5 shows The use case diagram illustrates the functionality and interactions within the **Ta'am** application.

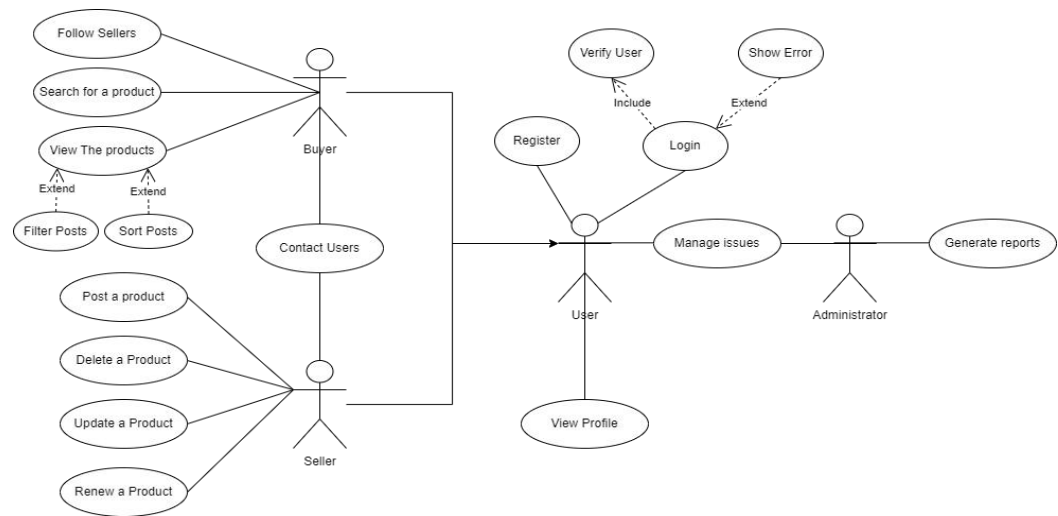


Figure 3.5: Use Case Diagram

3.2.2 Class Diagram

In **Figure 3.6**, we present the comprehensive class diagram of the system, offering a structured view of its architecture and components.

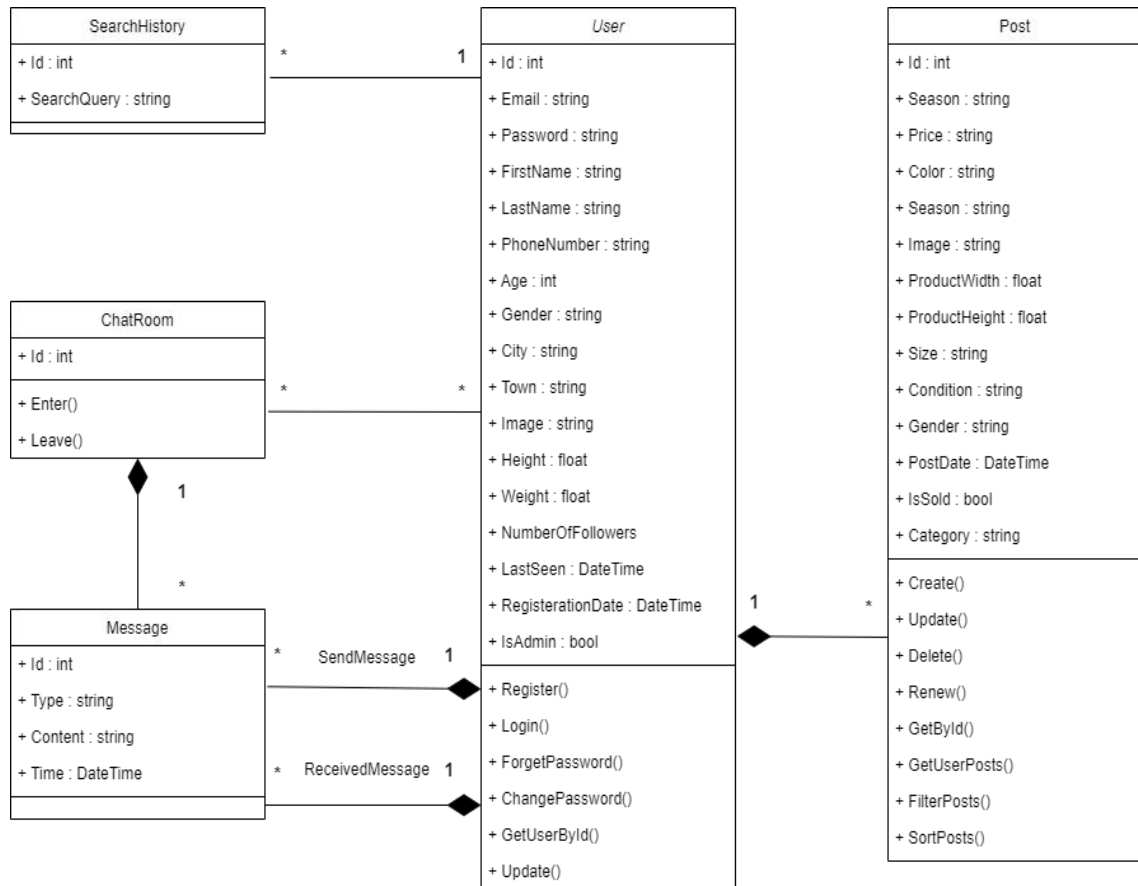


Figure 3.6: Class Diagram

3.2.3 Database Diagrams

Figure 3.7 presents the Entity-Relationship Diagram (ERD) of the system, providing a comprehensive visual representation of its underlying data model.

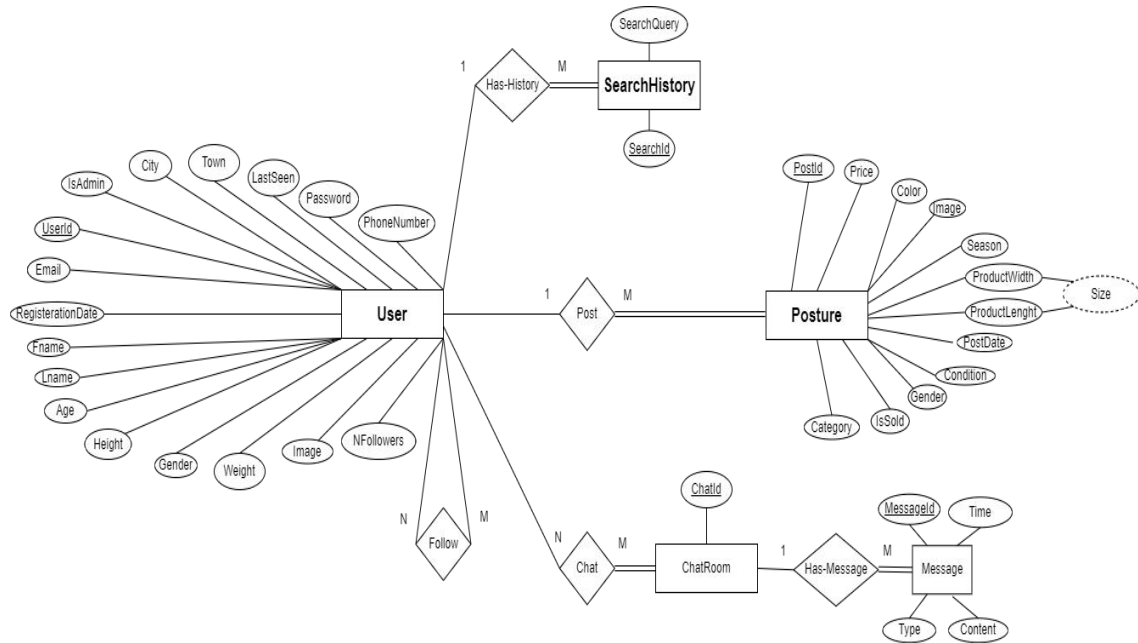


Figure 3.7: ERD Diagram

In **Figure 3.8**, the schema diagram is presented, offering a structured visualization of the database schema utilized within the system.

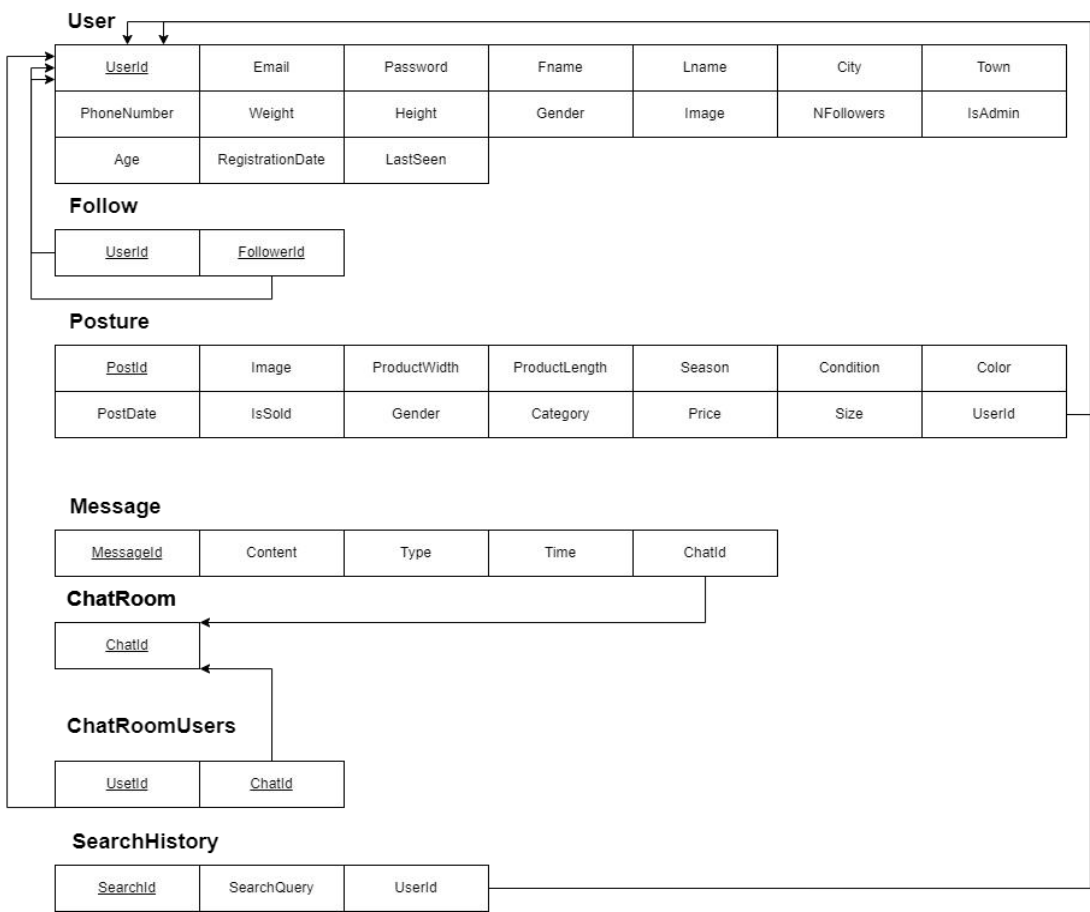


Figure 3.8: Schema Diagram

4- Implementation and Results

4.1 Datasets

Two datasets are used to train and evaluate the different models implemented in this project. Each one is explained below. The datasets are **DeepFashion2** Dataset and **Agrigorev's Custom Kaggle** Dataset.

4.1.1 DeepFashion2 Dataset

DeepFashion2 [16] is a comprehensive fashion dataset. It contains **491K diverse images of 13 popular clothing categories** from both commercial shopping stores and consumers. It totally has **801K clothing items**, where each item in an image is labeled with scale, occlusion, zoom-in, viewpoint, category, style, bounding box, dense **landmarks**, and per-pixel mask. There are also 873K Commercial-Consumer clothes pairs. The dataset is split into **a training set (391K images)**, **a validation set (34k images)**, and **a test set (67k images)**. Examples of DeepFashion2 are shown in **Figure 4.1**.

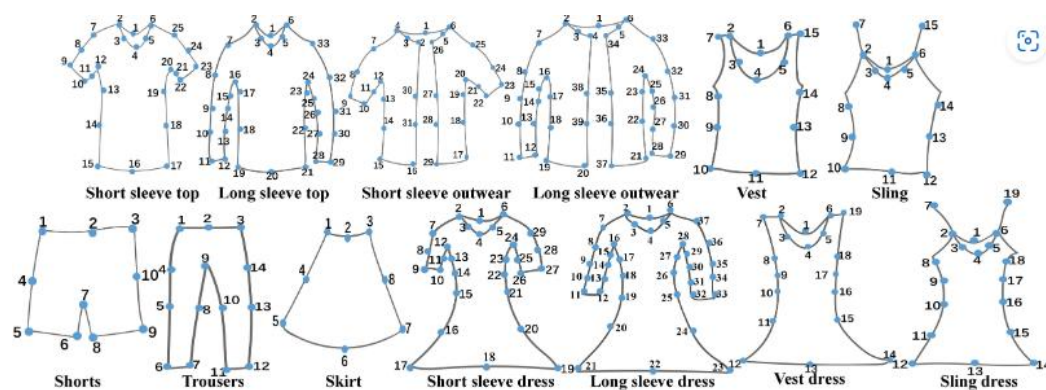


Figure 4.1: DeepFashion2

To solve the problem of **data imbalance**:

- Each category has a **separate** model

This dataset is used in the key points detection Model with **13 pre-trained ResNet50** models employed to extract key points **for each category**.

Challenges:

- Variability in the number of items per image:
 - Solutions:
 - Ignore images with more than one item.
 - Crop images to ensure each one contains only one category.
- Differences in the number of key points per category:
 - Solutions:
 - Train a separate model for each category.
 - Standardize the number of landmarks across all categories.
 - Choose some specific points.
 - Pad and Trim.

4.1.2 Agrigorev's Custom Dataset

Agrigorev's Custom [17] is **7GB** it has **5k product images** of **20 different classes** taken by the user **80%** of the data is used for **training** and **20%** for **testing** **Figure 4.2**



Figure 4.2: Agrigorev's custom dataset

To solve the problem of **data imbalance**:

- **Data Augmentation** using the **FastAI** library was applied.
- **A pre-trained** model was used.

The dataset used to train and evaluate the ResNet34 classification model is comprehensive and well-labeled, providing diverse and representative samples. After 50 epochs of training, the model achieved an accuracy of 88.5%, demonstrating the dataset's robustness in enabling effective learning and precise classification.

4.2 Software Tools

Flutter: Flutter is an open-source framework by Google for building beautiful, natively compiled, multi-platform applications from a single codebase which is used in building the user interface.

Python: Python is a programming language that lets you work quickly and integrate systems more effectively

Kaggle: Kaggle is a machine learning and data science community that provides code, data, notebooks, courses, and competitions for data enthusiasts which are used in training the models. The main reason why Kaggle was used is that the model development environment is that it provided the resources needed to process a huge amount of data especially the GPU resources it offered as our project's data consisted of images which was difficult to do on standard PC GPUs.

Google Colab: is a free cloud-based platform that allows users to write and execute Python code in a Jupyter Notebook environment. It is especially popular for machine learning, data analysis, and educational purposes due to its ease of use and accessibility. Colab provides access to powerful computing resources, including GPUs and TPUs, which are essential for training complex models. Additionally, it offers seamless integration with Google Drive, facilitating easy storage and sharing of projects and datasets.

4.3 Setup Configurations

Git: Git is a distributed version control system that tracks source code changes during software development. It allows multiple developers to collaborate on a project, manage different versions of the code, and facilitate code merging.

Torch: Torch is an open-source machine learning library that provides a flexible environment for building and training neural networks. Known for its efficiency and speed, Torch is particularly popular in the research community for deep learning applications. It offers a wide range of algorithms for deep learning and is highly modular, allowing users to experiment with different neural network architectures easily.

OpenCV: OpenCV (Open Source Computer Vision Library) is an open-source library for computer vision and image processing. It provides tools and functions for tasks like object detection, face recognition, image transformation, and video analysis. Widely used in real-time applications, OpenCV supports multiple programming languages and is highly efficient, making it a popular choice for both academic research and industrial projects.

FastAPI: FastAPI is a high-performance web framework for building APIs with Python. It emphasizes speed and efficiency, using Python-type hints for automatic validation and documentation. Its asynchronous capabilities support many simultaneous connections, making it ideal for scalable web APIs.

Postman: Postman is a popular collaboration platform for API development. It is used by developers to design, test, and document APIs.

4.4 Experiments and Results

4.4.1 Dataset Selection

- **Fashion-MNIST:** is a dataset of Zalando's article images—consisting of a **training set of 60,000** examples and a **test set of 10,000** examples. Each example is a **28x28 grayscale** image, associated with a label from **10 classes**.

This data was rejected because **it doesn't align with our criteria**.

- **Fashion product images Dataset:** is 25GB with a total number of **44k Images** high-resolution product images with **11 attributes** (id, gender, masterCategory, subcategory, articleType, baseColor, season, year, usage, productDisplayName).

This data was rejected because it was **in excellent condition**, which **doesn't align with the criteria** of a model **focused on used clothes images taken by users**.

- **DeepFashion2:** is a comprehensive fashion dataset. It contains 286K diverse images of 13 categories from both commercial shopping stores and consumers.

It was used in the Key-points detection Model with **13 pre-trained ResNet50** models employed to extract key points **for each category**
More details in **section 4.1.1**

- **Agrigorev's Custom Dataset:** is **7GB** and has **5k product images** of **20 different classes**.

It was used to develop the ResNet34 classification model
More details in **section 4.1.2**

4.4.2 Category Classification Experiments and Results

- **First Trial:**

In the initial attempt, a **custom CNN architecture** was employed on the **FashionMNIST** dataset to classify categories. Despite achieving a notable accuracy of approximately **92.4%**, the approach was deemed inadequate due to its reliance on **28x28 grayscale images**, which **did not meet our specified criteria**.

- **Second Trial:**

Next, we utilized **Agrigorev's custom Kaggle dataset** with a **pre-trained ResNet34** architecture which achieved an **accuracy of 88.5%** after **50 epochs**. This dataset, sourced directly from users, is closely aligned with our application's criteria, making it a **suitable choice for our purposes**.

- **Third Trial:**

The **DeepFashion2** dataset was employed with a **pre-trained ResNet50** architecture. However, **it was not accepted**, as Agrigorev's dataset proved to be more compatible with the images likely to be submitted, given that a significant portion of the DeepFashion2 dataset comprised images of clothing being worn.

- **Results:**

Here's a summary table outlining the datasets and architectures employed for category classification **Table 4.1**

Table 4.1: Classification models comparison

Dataset	Architecture	Accuracy
FashionMNIST	Custom CNN Architecture	92.4% (e=10)
Agrigorev's Custom Kaggle Dataset	ResNet34	88.5% (e=50) 82.3% (e=100)
DeepFashion2	ResNet50	90.5% (e=40)

4.4.3 Key-points Estimation Experiments and Results

- **First Trial:**

A **basic CNN model** was employed for key-point detection, but it yielded unsatisfactory results with a **loss of 7000** after **5 epochs**, indicating its **poor performance**.

The model utilized here **focused on single-category** images.

- **Second Trial:**

MobileNetV2 was utilized for key-point detection, achieving a **loss of 2000** over **10 epochs**.

The model utilized here **focused on single-category** images.

- **Third Trial:**

ResNet50 was employed for some specific key-point detection with the following results:

- With a **learning rate of 0.001** and **5 epochs**, the **validation loss was 794.31640625**.
- With a **learning rate of 0.001** and **15 epochs**, the **validation loss decreased to 598.56103515625**.
- Utilizing a **learning rate of 0.003019** and training for **10 epochs**, the **validation loss increased to 912.736328125**.

- **Fourth Trial:**

13 pre-trained ResNet50 models were employed to extract key points for each category **Table 4.2**

Table 4.2: Key-points Estimation model results

Category	mAP	MSE
short sleeve top	0.8111	0.0588
skirt	0.8646	0.0652
vest	0.7933	0.0622
vest dress	0.8271	0.0570
short sleeve dress	0.7920	0.0630
short sleeve outerwear	0.6566	0.0989
trousers	0.8624	0.0605
shorts	0.8936	0.0578
sling dress	0.7215	0.0814
long sleeve dress	0.6278	0.1069
long sleeve outerwear	0.6930	0.0949
sling	0.6897	0.0840
long sleeve top	0.6906	0.0932
short sleeve top	0.8111	0.0588

- **Fifth Trial:**

A pre-trained ResNet50 model was employed to extract key points and classify categories simultaneously. The best results were obtained with a loss of 0.085964 and a category accuracy of 88.3%, trained for 40 epochs with a learning rate of 0.0001.

- **Results:**

Here's a summary table outlining the datasets and architectures employed for key-points detection. **Table 4.3**

Table 4.3: Key-points Estimation trials summary

Target	Architecture	Validation MSE Loss
Single Category	Basic CNN	7000+ (e=5)
Single Category	MobileNetV2	2000+ (e=10)
Single Category	ResNet50	598+ (e=15)
Single Category	13 pre-trained ResNet50	-
All Categories	Multi-task pre-trained ResNet50	0.0859 (e=40)

4.4.4 Color Detection Experiments and Results

This table shows the experiments and results applied in the color detection model. **Table 4.4**

Table 4.4: Color Detection models comparison

	ColorThief	Mini-Batch K-Means
Published In	2017	2018
Execution Time	3 colors: 0.34 sec 10 colors: 0.37 sec 100 colors: 0.4 sec	3 colors: 0.1 sec 10 colors: 0.3 sec 100 colors: 0.4 sec
Accuracy	Accurate	More Accurate

4.4.5 Feature Extraction Experiments and Results

This table shows the experiments and results applied in the feature extraction model. **Table 4.5**

Table 4.5: Features Extraction models comparison

	VGG16	ResNet50	MobileNet
Feature Extraction per Image	1s to 2s	20ms to 50ms	20ms to 50ms
Vector Dimension	25088	100352	50176
Cosine-similarity with 5000 Images	0.4s to 0.7s	1.5s to 1.8s	0.7s to 1s

4.4.6 Real-life Scenarios Experiments and Results

After the models had been applied, an experiment of extracting the attributes of a product image was applied (as mentioned in Chapter 3) and the results were promising as shown in **Figure 4.3**.

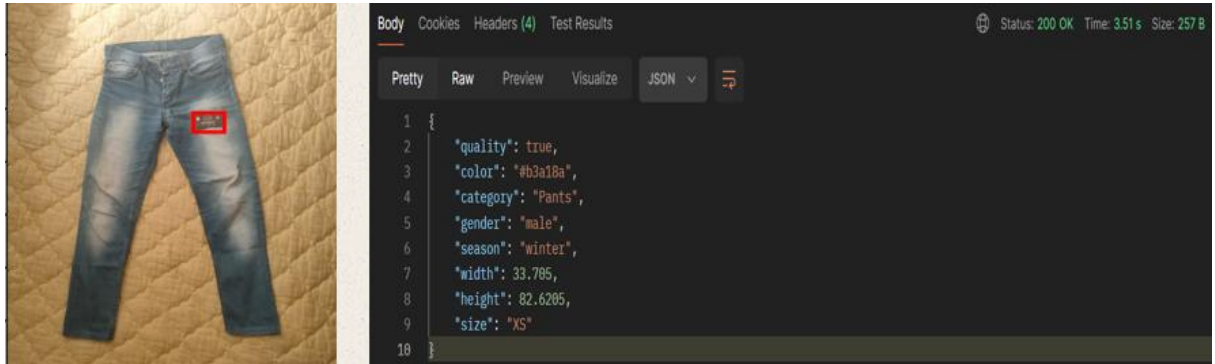


Figure 4.3: Features extraction result

Another experiment was applied to test the search capabilities and the results were promising as shown in **Figure 4.4**.



Figure 4.4: Search capabilities result

To sum up, the techniques used for each of the stages:

- **BRISQUE** for Quality Assessment
- **Image processing** for Card Detection
- **ResNet34** for Classification and Validation
- **ResNet50** for Key Points Detection
- **KMeans** for Color Detection

5- User Manual

This chapter is a walk through the application.

Login Page:

- The first page to be shown is the login page after launching the application.

Figure 5.1

- It contains fields to enter his credentials to login.
- User can also log in via Gmail.
- It also contains two links for the Forgot-Password page and the Register Page.



SIGN IN TO YOUR ACCOUNT

Email

Password

Log In

Log In with Gmail


[Forgot your password?](#)

Do you need an account? [Register](#)

Figure 5.1: Log In Page

Register Page:

- The user can click on the Register button on the login page to reach this page so he can create a new account.
- The user should fill in the needed data (Email, Password, First Name, Last Name). **Figure 5.2**
- User should also accept the terms of use.
- Then he can click on the register button.
- Finally the user should verify his account, and then he will be directed to the home page.



←

PERSONAL DETAILS

Email

Password

First Name

Last Name

We will send you an email to verify E-mail

☐ I accept the terms of use

REGISTER

Figure 5.2: Register Page

Forgot Password Page:

- The user can click on the Forgot Password button on the login page to reach this page so he can recover the forgotten password of his account.
- The user should fill in the Email and a link will be sent to assign a new password.

Figure 5.3

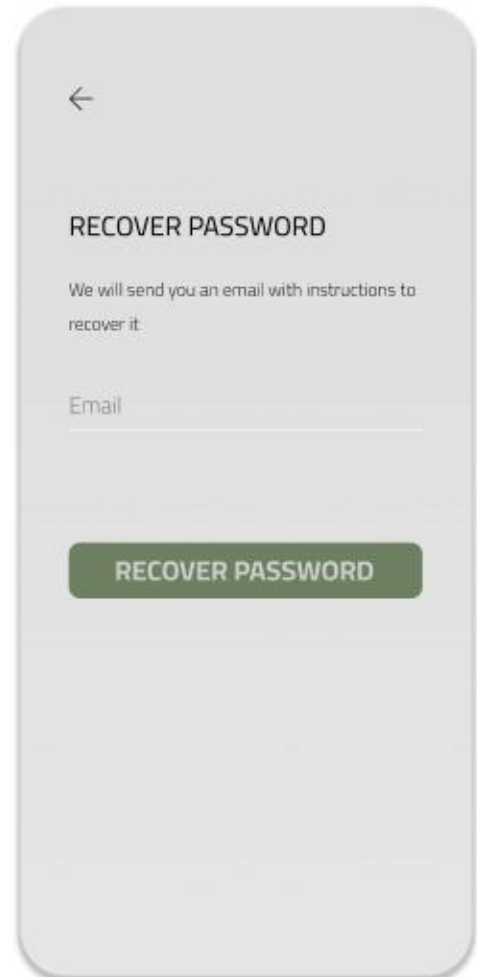
A mobile app interface for the 'Forgot Password' page. At the top left is a back arrow icon. The title 'RECOVER PASSWORD' is centered. Below it, a message states: 'We will send you an email with instructions to recover it.' There is an input field labeled 'Email'. At the bottom is a green button with the text 'RECOVER PASSWORD' in white capital letters.

Figure 5.3: Forgot Password Page

Home Page:

- The user reaches this page after login into the application.
- The user can navigate through the posts, filter, sort, chat with sellers, check product details, navigate to (Chats, Sell, My Ads, and Account) pages, and Search for products. **Figure 5.4**

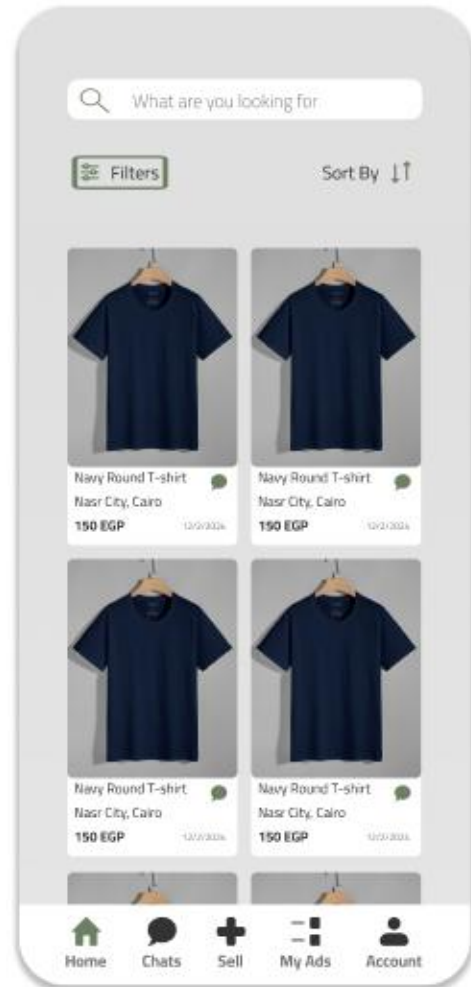
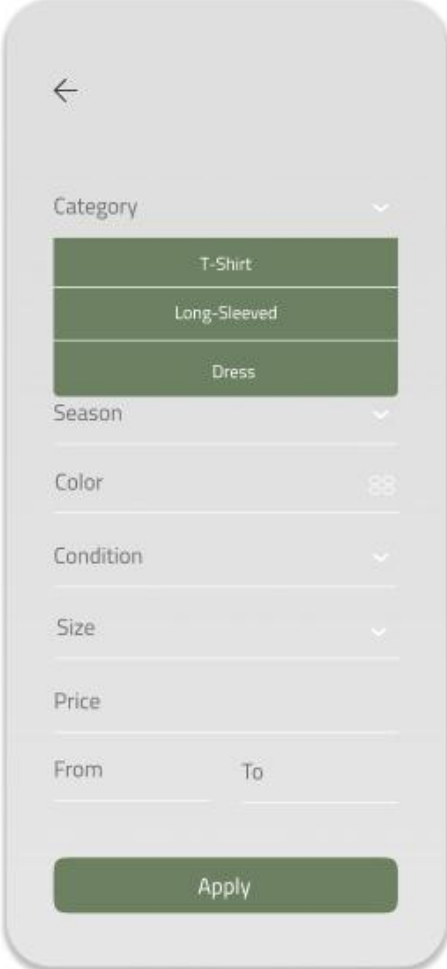


Figure 5.4: Home Page

Filter Page:

- The user reaches this page through the Home page.
- The user can filter the products by (Category, Gender, Color, Condition, Size, Price, Width, and Height).

5.5



Category

- T-Shirt
- Long-Sleeved
- Dress

Season

Color

Condition

Size

Price

From To

Apply

Figure 5.5: Filter Page

Sort Page:

- The user reaches this page through the Home page.
- The user can sort the products by (Most relevant, Newly listed, Lowest Price, and Highest Price). **Figure 5.6**

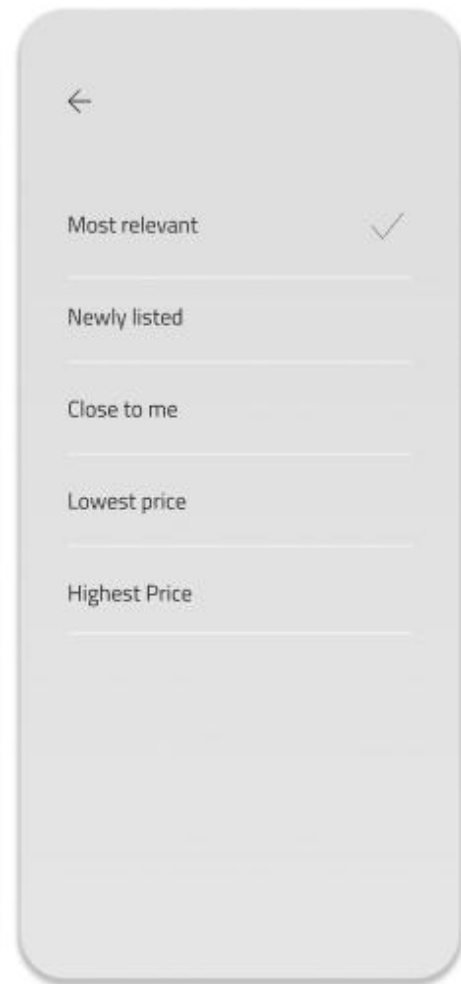


Figure 5.6: Sort Page

Product Details Page:

- This page shows the details of the product (Description, Price, Location of the seller, Category, Gender, Season, Color, Condition, Size, Width, Height, User profile). **Figure 5.7**

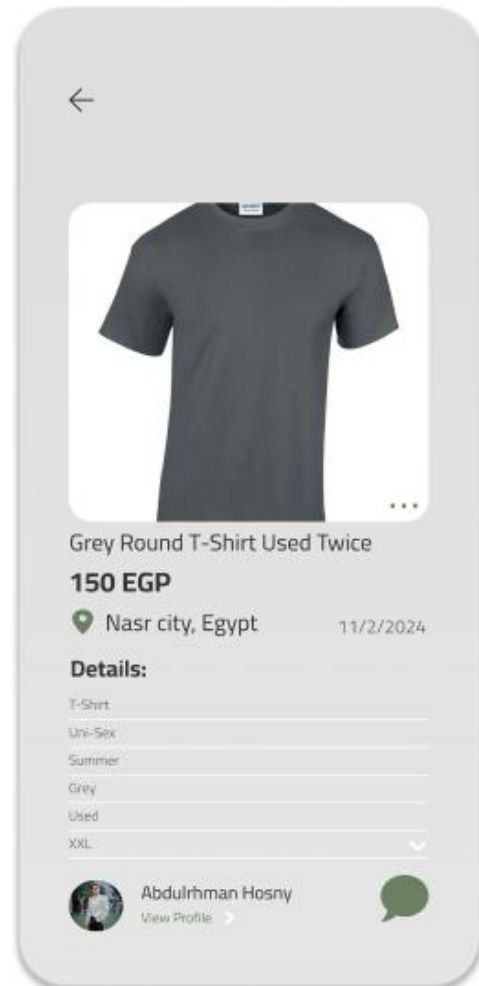


Figure 5.7: Product Details Page

Chats Page:

- This page contains user's chat rooms.

Figure 5.8



Figure 5.8: Chats Page

Chat Room Page:

- This page contains user's conversation.

Figure 5.9

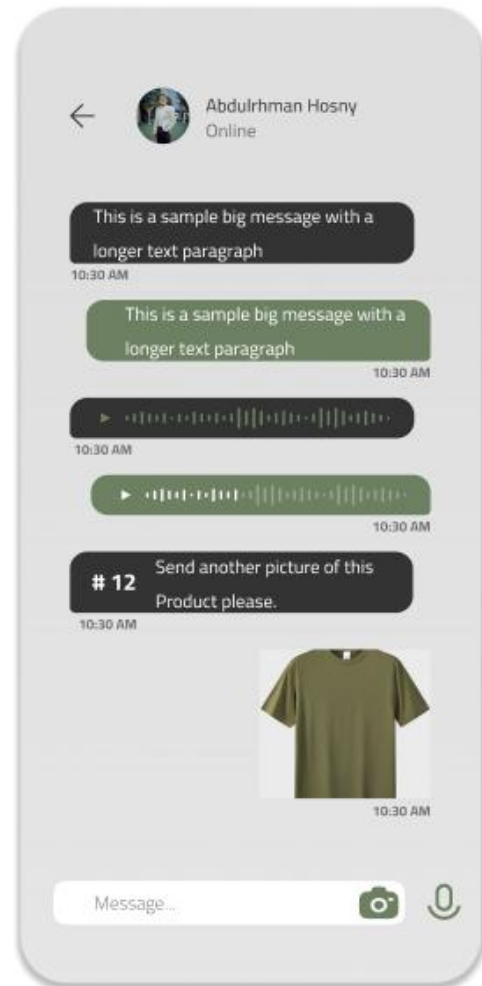


Figure 5.9: Chat Room Page

My Ads Page:

- This page contains user's Posts. **Figure 5.10**

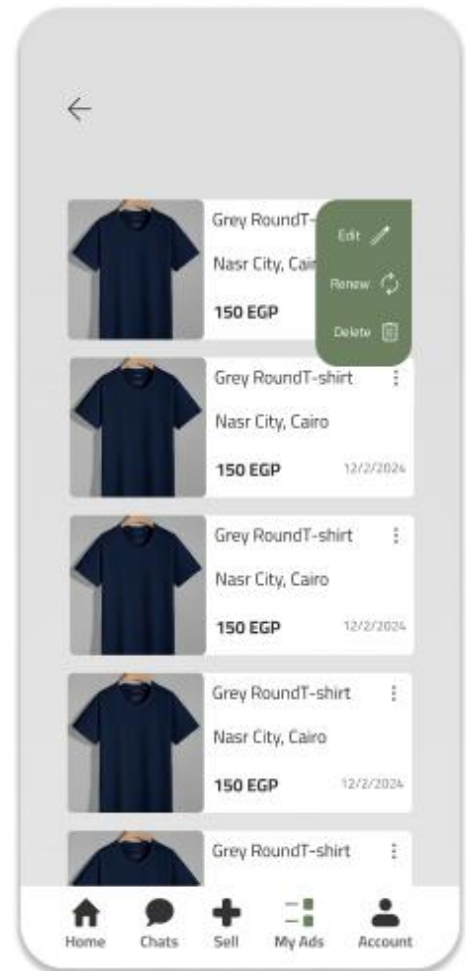


Figure 5.10: My Ads Page

Account Page:

- This page contains the user's account details. **Figure 5.11**

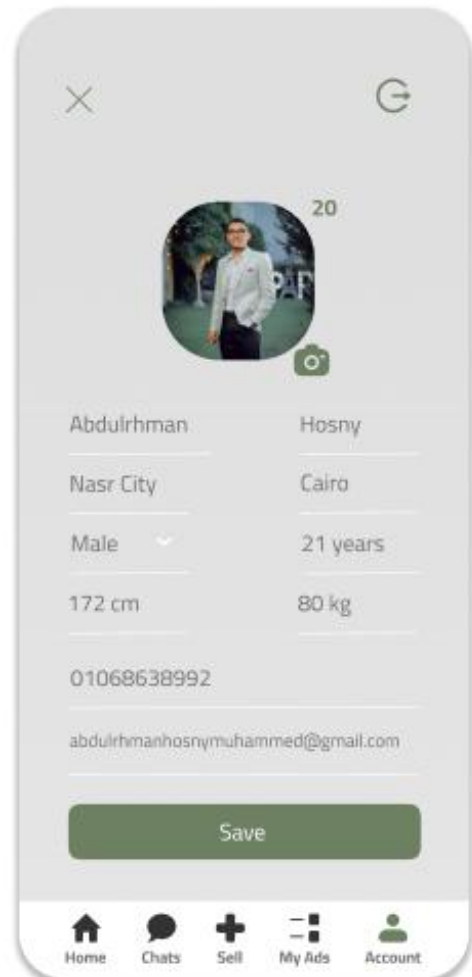


Figure 5.11: Account Page

Sell Page:

- These pages simulate the selling process of the product. **Figure 5.12**



Figure 5.12: Sell Page

Search Page:

- These pages simulate the search process in the application. **Figure 5.13**



Figure 5.13: Search Page

6- Conclusion and Future Work

6.1 Conclusion

This project introduces innovative models designed for **extracting features**, **evaluating image quality**, and **enhancing search capabilities** using images. The feature extraction model, powered by deep learning techniques, aims to accurately identify and classify clothing attributes like category, size, and color. Quality assessment involves using the **BRISQUE** model to ensure uploaded images meet specified standards before processing.

Additionally, the system leverages advanced deep learning techniques to detect and analyze key points critical for tasks such as **estimating garment sizes** based on a **reference object**. This approach enhances accuracy and improves user experience by automating these processes through sophisticated machine-learning models.

The models implemented in this project were trained on comprehensive datasets: **ResNet34**, trained on **Agrigorev's Custom dataset**, focuses on **classification and validation** tasks, while **ResNet50**, trained on the **DeepFashion2** dataset, specializes in **key-point estimation**. These datasets provide a robust foundation for training models that effectively handle diverse clothing items and scenarios.

Overall, this research contributes to advancing automated systems for image-based clothing monitoring, offering practical benefits to local traders and users by facilitating the sale of surplus clothing items with greater ease and accuracy.

6.2 Future Work

Future directions for this project involve enhancing model accuracy through fine-tuning various models and scaling up training with larger datasets encompassing diverse clothing categories from global cultures. Furthermore, the project can expand into creating a comprehensive immersive system that includes weekly progress reports and goal-setting features, fostering user engagement through gamification elements.

Moreover, there is potential to integrate depth sensors into the application to eliminate the reliance on reference objects for garment size estimation. However, this approach may limit the accessibility of the application based on the availability of depth-sensing capabilities in smartphones. These advancements aim to further optimize the user experience and broaden the utility of the application in the future.

Looking ahead, another exciting direction for this project is to incorporate tools that help users understand the environmental impact of their clothing choices. By integrating features like carbon footprint calculators or information on materials sourcing, the app can empower users to make more sustainable decisions when buying and selling clothes. This effort supports global sustainability goals and enhances the app's value by promoting eco-friendly practices among its users.

References

- [1] “What is Computer Vision?” IBM, [Online]. Available: <https://www.ibm.com/topics/computer-vision>
- [2] “What are Convolutional Neural Networks Vision?” IBM, [Online]. Available: <https://www.ibm.com/topics/convolutional-neural-networks>
- [3] Kristina Libby, “What is Object Classification and How Can We Use it” Shutterstock, [Online]. Available: <https://www.shutterstock.com/blog/object-classification-in-computer-vision>
- [4] Petru Potrimba, “What is Key-Point Detection” roboFlow, [Online]. Available: <https://www.shutterstock.com/blog/object-classification-in-computer-vision>
- [5] “A Complete Guide to Data Augmentation” DataCamp, [Online]. Available: <https://www.datacamp.com/tutorial/complete-guide-data-augmentation>
- [6] “What Is Transfer Learning” BuiltIn, [Online]. Available: <https://builtin.com/data-science/transfer-learning>
- [7] Alinder, Helena. "Semantic Image Segmentation on Clothing Imagery with Deep Neural Networks." (2020).
- [8] Paulauskaite-Taraseviciene, Agne, et al. "An intelligent solution for automatic garment measurement using image recognition technologies." *Applied Sciences* 12.9 (2022): 4470.
- [9] Bossard, Lukas, et al. “Apparel classification with style.” Computer VisionACCV 2012. Springer Berlin Heidelberg, 2013. 321-335.
- [10] Lao, Brian and Karthik A. Jagadeesh. “Convolutional Neural Networks for Fashion Classification and Object Detection.” (2015).
- [11] S. S. Islam, E. K. Dey, M. N. A. Tawhid, and B. M. M. Hossain, “A CNN Based Approach for Garments Texture Design Classification”, *Adv. technol. innov.*, vol. 2, no. 4, pp. 119–125, May 2017.
- [12] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, "DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

- [13] Tatiana Sennikova. “Clothes Classification with the DeepFashion Dataset and Fastai” towardsdatascience, [Online]. Available: <https://towardsdatascience.com/clothes-classification-with-the-deepfashion-dataset-and-fast-ai-1e174cbf0cdc>
- [14] Adrian Rosebrock. “Measuring size of objects in an image with OpenCV”. PyImageSearch. [Online]. Available: <https://pyimagesearch.com/2016/03/28/measuring-size-of-objects-in-an-image-with-opencv/>
- [15] Calorie Me, “Image-based Calorie Estimator System.” FCIS-GP 2023.
- [16] Ge, Y., Zhang, R., Wu, L., Wang, X., Tang, X., & Luo, P. (2019). A Versatile Benchmark for Detection, Pose Estimation, Segmentation, and Re-Identification of Clothing Images. CVPR.
- [17] OLOLO. (2020). “Clothing dataset (full, high resolution)”. Kaggle. Available: [Clothing dataset \(full, high resolution\) \(kaggle.com\)](https://kaggle.com/datasets/olololol/clothing-dataset)