Proceedings of the 20th Conference on Computer Science and Intelligence Systems (FedCSIS) pp. 417–422 DOI: 10.15439/2025F4362 ISSN 2300-5963 ACSIS, Vol. 43

Encouraging Sustainable Fashion Choices Through a Mobile Application

Bianka Šimková
Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Slovakia
Email: bianka.simkova.2@student.tuke.sk

***Abstract*—This work presents the design and implementation of a mobile application for home wardrobe management, aimed at promoting sustainability in fashion by encouraging more conscious clothing choices. The theoretical part of the article deals with the concept of "fast fashion" and its impact on different areas of the fashion industry and defines the problem with existing apps in this field. The practical section includes the design of an information system based on a user preferences survey, and includes a visualization of the system architecture. This section also describes the implementation process of two parts of the system, namely the creation of a clothing classification application using the CRISP-DM methodology and the development of a mobile app for the Android operating system, as well as the testing process. The conclusion evaluates the outcomes of the project and outlines potential directions for future enhancements.**

I. Introduction

THE motivation for writing this article is the ever-growing fashion industry, which offers us an overabundance of clothes at very low prices. This results in a excessive number of garments in the wardrobe of an average individual and clutter in the wardrobe. Therefore, the aim of this article is to design and implement a home wardrobe management information system that will contribute to sustainability in fashion.

Consumerism still dominates the world today, and much of this is influenced by the fashion industry. The term "fast fashion" emerged in the 1990s, when retailers began offering clothes to the masses at low prices, but at the expense of quality, in order to keep up with the pace of changing trends and emulate more luxurious garments [1]. However, the large volumes of clothing produced lead to increased resources for disposal and consequently high levels of waste, which has a negative impact on the environment [2]. Extending the lifetime of products is one of the most effective environmental strategies as it has a great potential to prevent waste and reduce production, consumption (both in sourcing and disposal) and transportation. This has been conceptually known, documented, and recognised for a long time - for example, in the waste management hierarchy, where prevention is the highest priority [3]. This is also why terms such as sustainability and "slow fashion" have been frequently mentioned in recent years.

A recent study based on a sample of 2,000 respondents found that the average individual owns approximately 136 items of clothing, indicating a relatively high volume of wardrobe content [4]. Often, we may not have an overview of individual items of clothing, nor how many times a piece of clothing has been used and how long it has been in our wardrobe. This factor, combined with the ongoing marketing strategies employed by retailers, can result in impulsive purchases of clothing items that are redundant, seldom worn, or do not fit properly. This inspired the idea of developing a mobile application — a wardrobe database — that can categorize garments based on photos and assess whether similar items are already present in the wardrobe or were owned in the past, along with data on their frequency of use. This feature can assist during shopping by discouraging unnecessary purchases, helping to keep the closet more organized and less cluttered.

II. Related Work

According to Klepp et al. [3], the lifetime of garments can be described in a way that makes them comparable, using the following categories:

- Number of wears;
- Years (Duration of use);
- Number of users (owners);
- Number of cleaning cycles.

Another important aspect in the care of garments is the way they are cleaned, which in addition to having an impact on the lifetime of the garment, also has an impact on the environment. It has been shown that the correct washing machine setting for garments made of polyester will reduce the release of microplastics by 12 to 16% [6]. Accordingly, effective garment care also requires attention to the material's origin, particularly the fibre composition of the fabric, such as:

- Synthetic fibres - such as polyester, nylon or acrylic.
- Natural fibres - such as cotton, linen or wool.

Also, washing synthetic materials with hard surfaces, such as jeans, increases the risk of damaging the material. When washing synthetic fabrics, it is also advisable to use less detergent and avoid bleach. Finally, it is recommended not to use a tumble dryer to dry synthetic materials, as heat and mechanical forces can cause the release of microplastic fibres. Last but not least, a person's preferences in choosing and wearing clothes, including on the basis of hair colour, complexion, eyes and body type, are also a major factor.

As Acerbi et al. [5] argue, fashion is not shaped by random decisions; rather, individuals exhibit strong preferences for specific cultural traits and tend to emulate certain styles and patterns. Fast fashion capitalizes on this behavior by employing innovative production and distribution strategies to drastically reduce the time required to bring garments from design to retail — often within weeks rather than months. As a result, the number of fashion seasons has expanded from the traditional two per year (spring/summer and autumn/winter) to as many as 50 to 100 micro-seasons annually [9]. This acceleration contributes to the continual shortening of the fashion cycle — the lifespan of a fashion trend — which is typically divided into five phases: introduction, rise, peak, decline, and obsolescence [10].

Many authors consider the rise of social media as the main cause of shorter and less predictable fashion cycles, as "trend drivers" [7]. A 2015 study confirmed that there is a direct and significant relationship between social media and the fashion industry [8]. There are some existing apps that focus on creating personal virtual wardrobe, such as Open-Wardrobe Outfit Planner ++[1], Acloset - AI Fashion Assistant[2], Whering:Digital Closet Stylist[3], or Style DNA: Fashion AI Stylist [4]. However, the applications are rather commercially tuned, i.e. built on the basis of recommendations of similar products that the user already owns. This consequently leads to unconsidered purchases and an excessive volume of clothing items, as mentioned in the introduction. Therefore, it would be advisable for the application to be able to assess, based on a photography of the garment, whether or not the user already owns a similar item and its frequency of use. Furthermore, existing apps either lack garment usage statistics or provide them in a rather superficial manner.

III. Proposed App Design

Initially, the requirements for the app were derived from survey responses published across various platforms. The survey was created with the intent to give us an insight into people's attitudes and behaviours related to fashion and clothes shopping. It consisted of 7 questions on demographic data, 8 statements related to personal preferences that respondents rated based on a Likert scale, and 1 question to select from the options of the preferred requirements of the wardrobe database application.[5] In total, 18 functional requirements were implemented in the app, namely: adding garments to your wardrobe; filtering and sorting garments by type, colour, pattern, material, recommended care, colour, number of wears, purchase price and date of purchase; adding care information to garments; recording garment washes; creating and saving an outfit; viewing saved outfits; planning an outfit; filtering saved outfits based on occasion and season; viewing wardrobe usage statistics; viewing usage statistics for a specific garment; automatic classification of garments by photo into type and colour; offering and accepting an outfit for exchange; adding personal preferences for style of clothing; estimating whether an outfit fits in the wardrobe. From the functional requirements, a list of ten use cases was created and made along with a UML diagram of the relationships between them to visualize more complex functionalities of the system that users expect and are of value to them.

Furthermore, four non-functional requirements were defined, focusing on security, performance, and usability. These included: mobile app response time of less than two seconds; support for at least 100 users in the database; encryption of stored data; and a simple, user-friendly graphical user interface (GUI).

The application consists of a 3-tier architecture as shown in Fig.1. The presentation layer consists of a user interface in the form of a xml layout and fragments. The business layer consists of an application core written in the Java programming language and a clothing classification API written in the Python programming language and deployed on Microsoft Azure. The data tier is managed by the Firebase platform. Attributes of garments and outfits, as well as the user's personal preferences, are stored in the Cloud Firestore database, and photos of garments and outfits are stored in Cloud Storage.

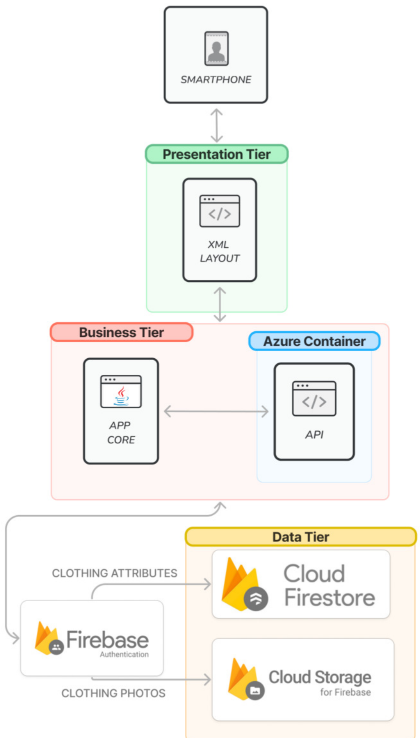

Fig. 1. App Architecture.

[1]https://play.google.com/store/apps/details?id=com.open_wardrobe_mobile
[2]https://play.google.com/store/apps/details?id=com.looko.acloset
[3]https://play.google.com/store/apps/details?id=com.whering.app
[4]https://play.google.com/store/apps/details?id=style.dna.app&hl=sk
[5]Link to the survey: https://docs.google.com/forms/d/1sv9lIR2Uj6OpbMFe94TsRsnGgMJTwuWdhJSrItx7oZ0/

IV. Implementation

A. Methodology

The development took approximately 6 months. In the first part of the implementation, we worked on creating an API for clothing classification using the Python programming language. In the second part, we programmed an Android application in Java programming language and integrated the developed API into it. In modelling the SVM classifier, we followed the steps of the CRISP-DM methodology, which consists of understanding the objective, understanding the data, data preparation, modelling, evaluation and deployment.

B. Clothing Classifier

For training the SVM classifier we utilized the dataset *Clothing dataset (full, high resolution),*[6] which contains over 5,000 photos of clothing items in both original and reduced quality. An analysis of the image orientations — portrait versus landscape — revealed a significantly higher frequency of portrait-format photos, suggesting that the average user tends to capture clothing images in portrait mode. The standard for the aspect ratio of photos taken with a smartphone is 4:3, so we resized all these photos to 160x120 pixels, and converted them to an array of RGB pixel codes. There are 20 categories of clothing, but the number of categories varied considerably, which would have caused us problems in modelling, so we reduced these categories to 8 main ones: short-sleeved top, long-sleeved top, pants, dress, outerwear, shorts, skirt and shoes. After merging and reducing the categories, the frequencies are more balanced.

We used 80% of our preprocessed category and pixel dataset to train the model and 20% to test the model, maintaining the ratio of category frequencies. For training the SVM classifier, we chose the Gaussian Radial Basis Function (RBF) kernel function and the decision function shape One vs. One. We evaluated the model with the metrics of accuracy, precision recall and F1 score. The classification accuracy is 0.74, which provides a good basis for deploying the model in practice. Precision, recall, and F1 score are above 0.7 for most classes. For categorizing and calculating similarity between colours, we used the webcolours library.[7]

C. API Containerization

The SVM classifier was incorporated in an application to process clothing photography using FastAPI, a modern web framework for creating APIs using Python. The application provides two main endpoints that process prediction and classification requests and return JSON responses:

- classify - takes as the image of the garment as an input parameter and outputs the type and colour of the garment;
- predict - takes as the image of the garment, and the name of the user's colour palette, as an input parameters and outputs the type and colour of the garment, and a boolean value that tells whether the colour is in the user's predefined colour palette.

We containerized the application using Docker, with Gunicorn serving the FastAPI app through Uvicorn workers. The resulting Docker image was uploaded to Docker Hub and deployed to Microsoft Azure as a continuously running container.

D. Mobile App

The mobile app was developed in Android Studio in the Java programming language. The frontend of the final version of the mobile application consists of 23 fragments. The bottom of the screen features a navigation menu that is used to switch between the five main modules of the application.

1) Closet: This module can be accessed by the middle button in the bottom menu and will show us the wardrobe contents with options to filter, add new garments, view statistics, and view garments marked for washing. In the process of adding a garment by the "+" button, the aforementioned classifier is also used, which classifies the garment into type and colour. After adding a garment, the detail of the stored garment is displayed, which also contains the statistics of the particular garment (Fig. 2), which are automatically recalculated based on the planning of outfits - sets of garments. Additionally, the user can also access the aggregated statistics for the whole closet (Fig. 3). The "Filter" button will show the options to filter and sort the garments, and after confirming the selection, it will show us the closet with only the garments that meet the required characteristics.

2) Outfits: The first button in the bottom menu will launch the module related to storing, viewing, filtering, and planning outfits. The ability to filter and sort outfits works the same way as for clothing. When adding, all the outfits stored in the wardrobe are displayed, and the user selects the garments that make up the outfit. Scheduled outfits can be viewed by the user in the "OOTD" (Outfit Of The Day) section.

3) Settings: This module is accessed by the far-right button of the bottom menu and displays the login screen or, if the user is logged in, the option to log out and set personal preferences.

4) DoesItFit: The fourth button in the bottom menu gives us the ability to view the saved preferences and also the ability to estimate the fit to the wardrobe, where the user uploads a photo of the garment and the application here uses the classifier to classify the garment into type and colour, verifying that the colour of the garment is within the user's chosen colour palette and displaying similar garments in the wardrobe, i.e. of the same type and colour (Fig. 4).

5) Swap: By clicking on the second button in the bottom menu, we can see the garments offered for exchange. Clicking on a particular garment will show the details and the option to request the garment. Then the user who provided the garment for exchange can see in the details of their garment the email addresses of the users who requested the garment, and after selecting and confirming the selection of one of the email addresses, the garment will be reassigned to that user.

The backend of the application is written in the Java programming language and consists of 53 classes, which

[6]https://www.kaggle.com/datasets/agrigorev/clothing-dataset-full/code
[7]https://webcolours.readthedocs.io/en/latest/

Fig. 2. Clothing item detail.

are divided into five packages: `activities` (as mentioned previously in this section); `api` (classes associated with linking the application to the classifier); `entities` (constructors and methods of `ClothingItem`, `Outfit`, `SwapItem` and `Webcolours` objects); `recycler-view` (classes that modify the view of entities); `tools` (methods for working with the database, data visualization, image editing, calculating statistics, etc.). The remaining two classes are subclasses of the `Activity` class: `WelcomeActivity` and `MainActivity` that runs throughout the application and is responsible for controlling the bottom menu to change fragments.

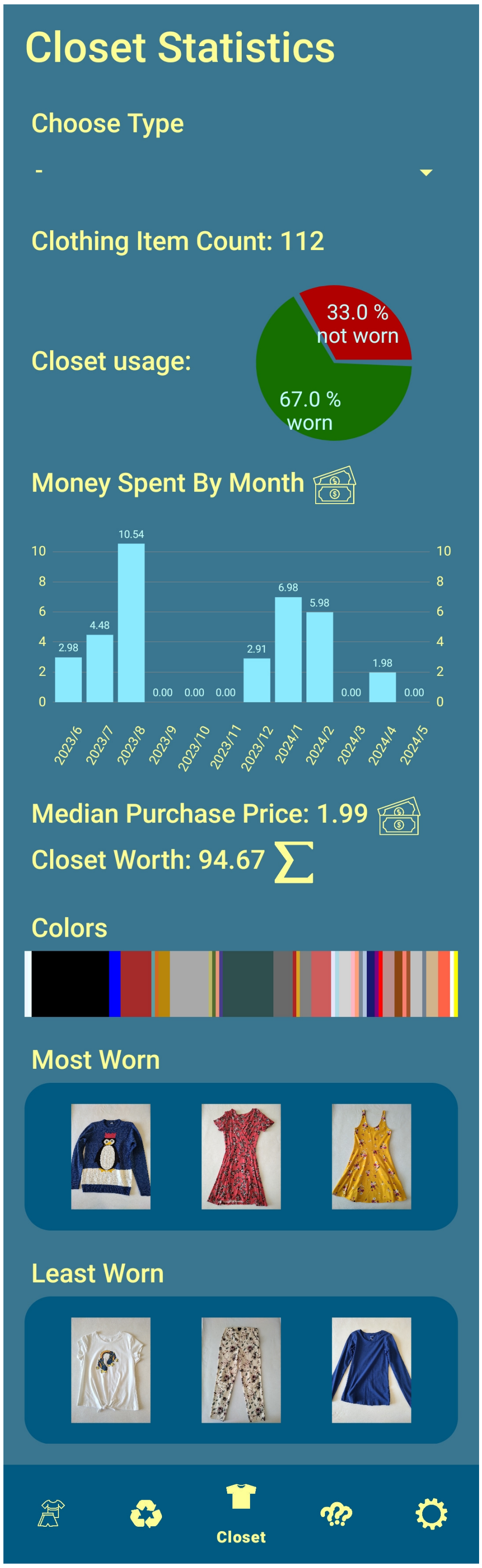

Fig. 3. Closet statistics.

Fig. 4. Application output on finding similar garments based on photo.

E. Database

We used the Firebase Firestore database to store attributes related to clothing items and outfits. The "user_clothes" collection contains documents named according to the unique user ID (UID) generated during registration via Firebase Authentication. Each of these documents includes two subcollections: "clothes" and "outfits", which contain documents named according to the IDs of the outfits. Each document captures 16 features of the corresponding item, including statistical attributes such as `timesWorn`, `timesWashed` and `wearPercentile`. The methods used for calculating statistics are contained in the `StatisticsCalculator` class, which includes, for example, a method to calculate garment usage frequency based on percentiles as shown in Algorithm 1 and 2. The underlying logic for these statistical measures is inspired by the concepts discussed in the article by Klepp et al. [3], as referenced in the introduction.

Algorithm 1 Calculate wear percentiles of clothing items

Require: List of clothing items with `timesWorn`
Ensure: List of wear percentiles (as integers from 0 to 100)

Initialize empty list $percentiles$
for all $calculatedItem \in clothingItems$ **do**
  $countLessOrEqual \leftarrow 0$
  for all $item \in clothingItems$ **do**
    if $item \neq calculatedItem$ **and** $item.timesWorn \leq calculatedItem.timesWorn$ **then**
      $countLessOrEqual \leftarrow countLessOrEqual + 1$
    end if
  end for
  Append $countLessOrEqual$ to $percentiles$
end for
$countAll \leftarrow$ size of $clothingItems - 1$
for $i = 0$ to $percentiles.size() - 1$ **do**
  $percentileFloat \leftarrow \frac{percentiles[i]}{countAll} \times 100$
  $percentiles[i] \leftarrow \lfloor percentileFloat \rfloor$
end for
return $percentiles$

Algorithm 2 Get wear frequency label

Require: A clothing item with computed percentile
Ensure: A frequency label (string)

$percentile \leftarrow clothingItem.getWearPercentile()$
if $percentile \leq 25$ **then**
  return "very low"
else if $percentile \leq 50$ **then**
  return "low"
else if $percentile \leq 75$ **then**
  return "medium"
else if $percentile \leq 100$ **then**
  return "high"
else
  return null
end if

V. Evaluation

Functional and integration testing of the application was also carried out in the implementation phase. At the end of the implementation, we evaluated the app using Firebase Test Lab. After evaluation by the Test Lab, we took the results of this testing into account and patched the version. We conducted acceptance testing in person with five target users who were asked to complete 13 tasks such as: use the app to add a garment to the wardrobe; use the app to create an outfit; use the app to view statistics about your wardrobe usage; use the app to check if the garment fits into your wardrobe. Immediately after testing, we had each participant complete a System Usability Scale (SUS) survey. All users rated the application rather positively, which resulted in the SUS score of 93.5. In addition, the survey included a question about which functionalities users found most or least useful. The most useful features were rated by users as the garment care record, viewing statistics, filtering garments, saving outfits, and estimating whether a garment fits them.

Among all features, testers found the clothing swap option to be the least beneficial.

In order to validate the non-functional requirement "mobile app response time of less than two seconds", we created a `StartupAndFrameBenchmark` class in the Android Studio environment, in which we defined tests to detect the duration of the app launch and the duration of the screen rendering after some action, i.e. the app response time. The results of this testing show that 99% of the measured responses were less than or equal to 205.0 milliseconds, which is approximately 0.2 seconds. The remaining non-functional requirements related to the database. Since we store data in Firestore database, we have fulfilled these requirements, since the capacity of Firestore database is 50 000 active users per month, Firebase automatically encrypts data in database. We also enhanced security by setting the Firebase rules so that each logged in user can only read and write data under the document that is identical to their UID or under the "swap" collection as mentioned before.

VI. Conclusion and future work

The aim of this work was to discuss the influence of different factors on fashion and propose a design and implementation of a mobile application with the intent of promoting sustainability in fashion. The implemented app lays the groundwork for encouraging more conscious consumption habits by helping users manage their wardrobes more efficiently and make sustainable fashion choices. The app also allows for possible future extensions, primarily focused on improving the user experience, such as: adding wardrobe sharing; adding automatic outfit suggestion; automatic background removal on photos of clothes; improving the accuracy of the classifier, or using neural networks in classification. The App and API are available on github.[8]

References

[1] A. Joy, J. Sherry Jr., A. Venkatesh, J. Wang, and R. Chan, "Fast fashion, sustainability, and the ethical appeal of luxury brands," *Fashion Theory, vol. 16, no. 3, pp. 273–295, 2012,* https://doi.org/10.2752/175174112X13340749707123.

[2] L. Cavusoglu and M. Dakhli, "The impact of ethical concerns on fashion consumerism: a review," *Markets, Globalization & Development review, vol. 1, no. 2, art. no. 5, 2016,* https://doi.org/10.23860/MGDR-2016-01-02-05.

[3] I. G. Klepp, K. Laitala, and S. Wiedemann, "Clothing lifespans: what should be measured and how," *Sustainability, vol. 12, no. 15, art. no. 6219, 2020,* https://doi.org/10.3390/su12156219.

[4] Search Department Statista, "Average number of apparel items owned by consumers worldwide from 2017 to 2019 [Infographic]," Statista, 2019, https://www.statista.com/statistics/1008465/average-number-of-apparel-items-owned-by-consumers-worldwide/.

[5] A. Acerbi, S. Ghirlanda, and M. Enquist, "The logic of fashion cycles," *PLOS ONE, vol. 7, no. 3, 2012,* https://doi.org/10.1371/journal.pone.0032541.

[6] G. Dalla Fontana, R. Mossotti, and A. Montarsolo, "Assessment of microplastics release from polyester fabrics: The impact of different washing conditions," *Environmental Pollution, vol. 264, 2020,* https://doi.org/10.1016/j.envpol.2020.113960.

[7] G. G. Berk and M. R. DeLong, "Communicating Fashion: Trend Research and Forecasting," University of Minnesota Libraries Publishing, 2023, https://open.lib.umn.edu/communicatingfashion/chapter/chapter-6-trend-lifecycles-and-adopter-categories/.

[8] N. Ahmad, A. Salman, and R. Ashiq, "The impact of social media on fashion industry: Empirical investigation from Karachiites," *Journal of Resources Development and Management, vol. 7, 2015,* https://ssrn.com/abstract=2603275.

[9] D. Drew and G. Yehounme, "The apparel industry's environmental impact in 6 graphics," World Resources Institute, 2017, https://www.wri.org/insights/apparel-industrys-environmental-impact-6-graphics.

[10] M. Pillai, "Understanding cycles of fashion," *International Journal of Advances in Engineering and Management (IJAEM), vol. 2, no. 12, pp. 177–191, 2020,* https://doi.org/10.35629/5252-0212177191.

[8]https://github.com/biankasim/