Bilkent University

Department of Computer Engineering

CS 491: Senior Design Project

Rhapso

High-Level Design Report

Group Members:

Işık Özsoy	21703160	isik.ozsoy@ug.bilkent.edu.tr
Burak Yetiştiren	21802608	burak.yetistiren@ug.bilkent.edu.tr
Defne Betül Çiftci	21802635	betul.ciftci@ug.bilkent.edu.tr
Şebnem Uslu	21802068	sebnem.uslu@ug.bilkent.edu.tr
Melike Fatma Aydoğan	21704043	fatma.aydogan@ug.bilkent.edu.tr

Supervisor : *Prof. Dr.* Halil Altay Güvenir

Jury Members : *Asst. Prof. Dr.* Shervin R. Arashloo and *Asst. Prof. Dr.* Hamdi Dibeklioğlu

Website URL : https://ivorymask.github.io/cs491-2.github.io/

Project Specification Report

December 24, 2021

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491.

Table of Contents

1. Introduction

In today's world, it is a common practice that many traditional daily activities are supported with tools implemented in engineering domains. Thanks to a wide range of available software, users can perform daily activities more comfortably. Some examples include: *Spotify* [1], by which the users can basically carry around all the music that they would like to listen to throughout the day; this activity would otherwise be more complicated, having in mind that the users would had to carry the music records and the hardware to listen to the music with them, which would be extremely inefficient. *Google Maps* [2], by which the users can navigate easily in the streets of even the most complicated and congested cities; again in the absence of this software, the users would have to use some traditional methods like some physical maps, which are large to carry and do not have any interactive component like Google Maps like telling the traffic, offering alternative routes, suggesting different paths for different means of transportation, etc. There are more examples on a similar basis that we could grow our list on, but we believe that these examples suffice in terms of clarifying our justification. Among explained examples, one other important common point is that all of the examples given include virtualization, that the users can install these software as mobile applications in their mobile devices, and use them wherever they want as long as they have their mobile devices with them.

Having made the previous points, our project *Rhapso*, which is named after a goddess with the meaning "The Stitcher" [3], will be based on easing a particular daily activity, while using the virtualization aspect of mobile applications. As explained, our project will be a mobile app. With this app, we want to implement and provide the functionality of creating a virtual wardrobe using the photos of the clothes of the user. In comparison to a regular wardrobe, our wardrobe will be interactive which will give shopping suggestions in terms of the already existent items in itself. We want to pay attention to the sustainability of the environment, such that with this application we aim to reduce overall clothing consumption. Lastly, in our application, the users will have the opportunity to try on the clothes virtually, so that on occasions that the time is constrained, our application will provide practicality, which we believe will be essential.

Other than the virtualization aspect of the application, one other issue that we are trying to solve with *Rhapso* is the unnecessary consumption of clothing pieces. The statistics show that people bought 60% more clothing garments in 2014 than in 200, while another statistics to note here is that fashion production makes up 10% of humanity's carbon emissions [4]. Exceeding fashion consumption dries up water sources and causes pollution of rivers and streams [4]. These show that consumption of clothing is an ever growing environmental problem. A study on shopping behavior from researchers in University of Michigan state that the participants involved in the study described wanting deliberation or reflection before buying a clothing by asking the shopper what they would use this new clothing for and if they truly needed it [5]. Another way the participants wanted an application to encourage deliberation was through questioning their possessions, reporting if they have similar ones and how many they have of it [5].

Individual activities such as thrifting aim to offer a solution to the problem of pollution of clothing garments, but they do not have a wide impact as they are and they still do not affect customer behavior. One aim of *Rhapso* is to offer another solution to the present over-consumption of clothes. The application aims to do so by:

1. Comparing a new clothing garment to existent ones saved in the closet,
2. Finding similar clothes from online thrift stores to the new clothing such that it would encourage purchase of secondhand clothing instead of a new one,
3. Making the user "try on" the new clothing garment with the clothes they own so that they can see if the new clothes fit in with their closet,
4. Recommending outfits using the clothes inside their closet so that the user can see if they can make combinations of the new clothing with the existing ones.

By doing these tasks, our aim is to have reduced consumption of clothes for customers over time.

In the technical part of the argument, we want to make use of computer vision algorithms. The algorithms will mainly be used in the detection of the cloth from the photo, and while comparing clothes with each other in terms of their similarity. On the other hand, in the part of enabling users to virtually try on clothes, again we will be making use of computer vision algorithms, which for this part will be embraced in *GANs (Generative Adversarial Networks)*.

In this report, we intend to provide details about the high-level design of our system. Firstly, we discuss similar applications, which are providing similar functionalities and the domain of Rhapso. Followingly, we discuss our proposed system, in terms of elaborating the subsystems we will implement. Then, we discuss some multidisciplinary values that we consider cautiously for our design. And finally we include the details about our teamwork plan.

1.1 Purpose of the system

Rhapso is a tool that uses image processing and machine learning to create a virtual wardrobe, to match outfit as suggestions, and raise awareness to users for existing similar clothes or secondhand options of similar clothes in websites such as Dolap and Gardrop to support sustainability.
The purpose of the system is to provide the user the access to their closet in every place and in virtual manners, the outfit combination ideas, and environment friendly suggestions that involve similar or secondhand clothes. Rhapso aims to enhance the mobility of your closet and to help the environment through your closet.

1.2 Design goals

In this section, Rhapso's design goals will be explained and significant features to improve the user experience will be discussed.

1.2.1 Sustainability

One of the aims of the project is sustainability. We aim at reducing the purchase of new clothes by providing outfit recommendations. We are trying to prevent people from doing unnecessary shopping. Furthermore, if users want to buy a new piece of clothing, they can ask Rhapso to find similar clothes from second-hand online shopping websites. While reducing shopping, we prevent people from spending unnecessary money. Moreover, when the demand for new clothes decreases, the materials required for the production will not be consumed. We designed Rhapso's features to sustain consumption and economic condition of people so that we chose sustainability as a design goal.

1.2.2 Usability

We are going to focus on usability as another design goal. We are going to make the application as easy as possible. We will clearly indicate the aim of the GUI components by using appropriate headers, texts or icons. Furthermore, the app will guide the user about the actions that the user should perform. For example, while taking the photo of the clothes, there will be instructions on the screen such as 'Please show a clothe.', or 'The clothe could not be detected, please try to show the image clearly.'

1.2.3 Efficiency

Efficiency is another important design goal, especially in terms of time. Some of the Rhapso's features are delay sensitive. For example, the time required to get the similarity score for a cloth (a score generated based on the clothes the user has) must be minimized since we do not want the user to wait for the calculations too much during shopping. In this case, the user may not prefer using the feature. We are going to implement our models and optimize them in order to achieve efficiency in time.

1.2.4 Understandability

One general aim of Rhapso is to make people's lives easier by reducing the time spent to decide on an outfit. In order to achieve this, the application should be easy to use so that it will not require too much time. In order to achieve this, the app must be easy to use. A person should understand the meaning of the buttons, labels and the screens without requiring further explanation. Thus, it will be one of the design goals that we will be focusing on. We will make the user interfaces as simple as possible, with necessary indicators. We will also provide a tutorial when the user uses Rhapso for the first time. Furthermore, we will provide a help section which includes any necessary information regarding Rhapso.

1.2.5 Modifiability

We are going to implement Rhapso using MVC approach to ease making modifications without damaging the current system. The application will be modified to meet the needs of the users, thus it must have a modifiable structure. We might change the current functionalities or add new ones according to the feedback.

1.2.6 Readability

We aim at writing our code in a way that is as readable as possible since we are working in a group of five people. Thus, following the best coding practices is important to reduce the development time and increase the efficiency in terms of development time. We are going to give meaningful names to models, functions or variables and also provide comments if necessary. With the help of those, our source code will be easy to be traced and understood.

1.3 Definitions, acronyms, and abbreviations

Machine Learning: An algorithm that alters itself over time based on the data that it is exposed to.
Neural Network: Machine learning structures that aim to mimic the human brain that are comprised of connected nodes that have “weights” of their own which are altered as necessitated and pass on data to each other. [6]
MVC :
CNN:
Semantic Segmentation:
Deep Learning: A subset of machine learning that defines the dependation of the algorithm onto a neural network structure with three or more layers. [7]
Computer Vision: Subset of artificial intelligence that defines a specialization in deriving meaningful information from visual inputs. [8]
GAN: An abbreviation for Generative Adversarial Network. This is a network model that is designed with two components: generator that creates data, and discriminator that determines the validity of the generated data. Its aim is to generate data that is similar to the training data [9].
GPU: An abbreviation for Graphics Processing Unit. A GPU is an electronic circuit that is designed specifically to speed up the production of graphical units in a frame buffer to provide outputs to a display unit. [10]
Euclidean Distance: A calculation of the distance of vectors of real-values [11]. The square root of the sum of the squared differences between the two vectors is used to compute Euclidean distance [11].
Cosine Similarity: The similarity of two vectors in a n - dimensional space is measured by cosine similarity. It detects if two vectors are pointing in the same general direction by measuring the cosine of the angle between them [12].

2. Current software architecture

As far as present systems go, there are two related applications that may be explored. TryNDBuy and Your Closet [13] [14] are two of the applications in question. Users of TryNDBuy may try on garments from a catalog on a 3D model provided by the app. The app may also be used to purchase clothing. The primary goal of this program is to persuade users to purchase clothes through it. Your Closet is the second app that needs to be examined. The main goal of this program is to store a user's clothing items. Users of the program may add apparel to their profiles by uploading photos of the items. Dolap and Gardrops, two of Turkey's most popular web secondhand clothing buying applications, are two additional programs with the similar purpose of lowering new clothing consumption. These programs are used to catalog used items that users put out and have the intention of giving a platform for popularizing secondhand clothing purchases, consequently sharing the same target of reducing consumption as Rhapso but still not delivering any of the proposed system's advantages.

3. Proposed software architecture

3.1 Overview

In Rhapso, we offer many functionalities, such as offering combinations, combination recommendations, body visualization, and more. This follows an attribution of significance to the software architecture of our project. We believe that amplifying the architecture of our system design by allocating the subsystems and their components in their correct places allows us to come back and remind ourselves about our design during implementation. And other individuals interested in our system to better understand the architecture we implement. Hence, we pay significant importance to our architectural design of Rhapso. Our design can be viewed to a greater extent in the following parts, yet a brief discussion shall be influential to be made before delving into the visual information.

From our diagrams, one should capture the characteristics of our architecture; each figure tries to elaborate on the interactions of the components with each other. From these interactions, the flow of data can be interpreted more thoroughly. Subsystem decomposition, hardware/software mapping, persistent data management, access control and security, and global software control parts discussed below give detailed information about:

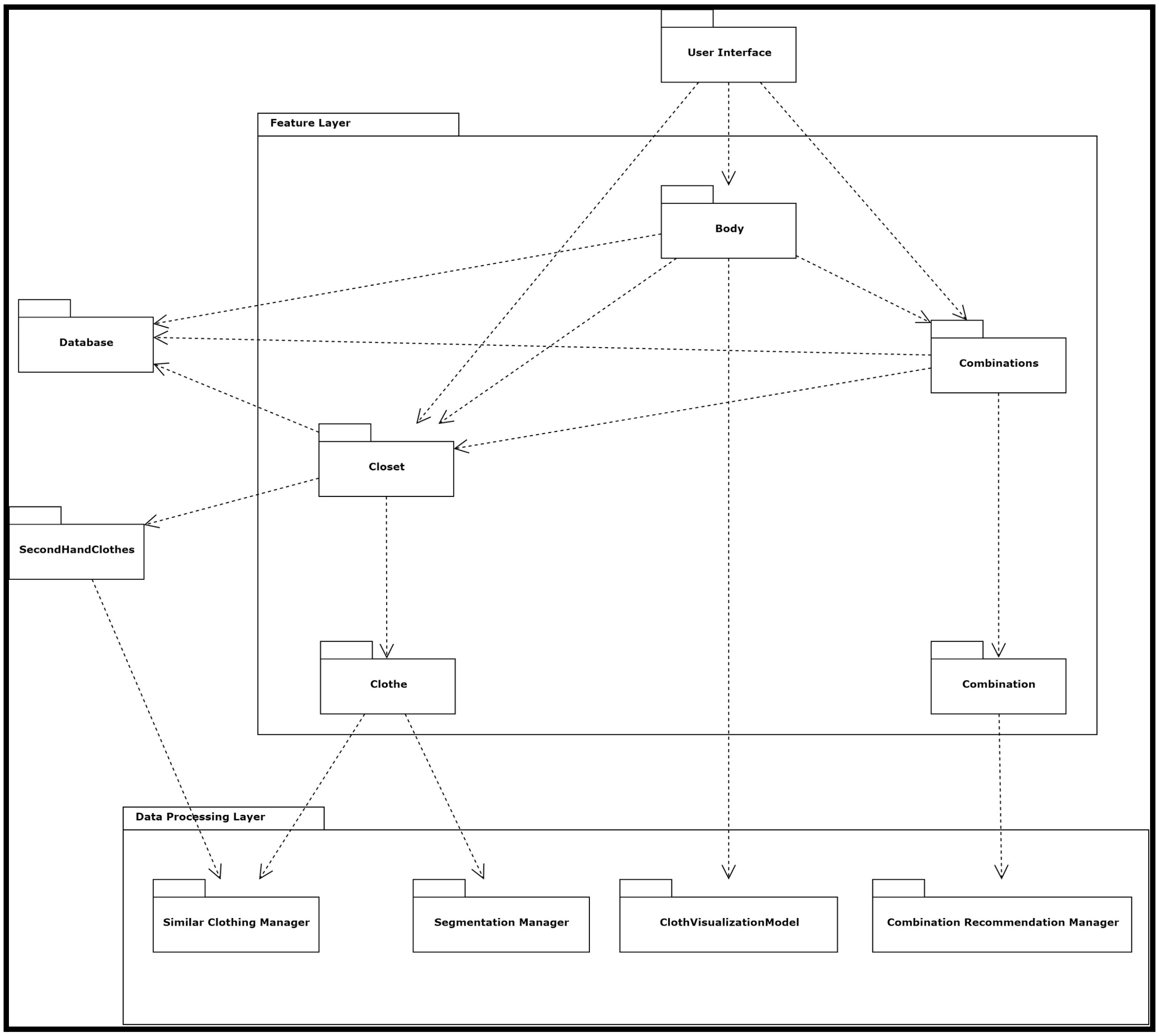
- The main subsystems, and their components, which compose our total structure
- The underlying connection between hardware and software we employ
- Management of data
- The roles and rights of different actors accessing our system
- The description of the real-time functioning of all subsystems we have

Below is a comparison of the proposed system and other similar previous systems discussed in the Current software architecture section.

Tools / Features	Rhapso	TryNDBuy	Your Closet	Dolap	Gardrops
Virtual Closet	✔	✘	✔	✘	✘
Virtual Try-On with 2D Model	✔	✔	✘	✘	✘
Similarity Comparison	✔	✘	✘	✘	✘
Aid in Shopping	✔	✔	✘	✔	✔
Thrifting Suggestions Based on Similarity	✔	✘	✘	✘	✘
Outfit Recommendations	✔	✘	✔	✘	✘
Goal of Reducing Consumption	✔	✘	✘	✔	✔

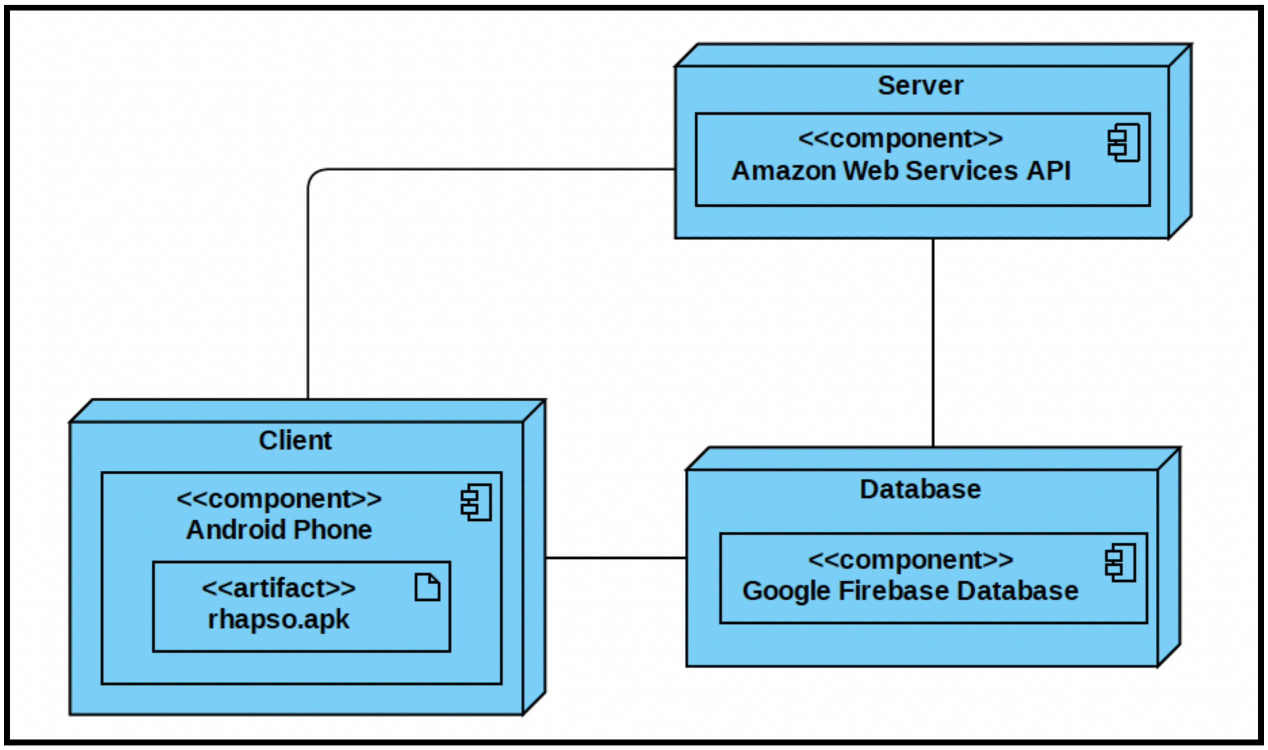
Tab. 1: Comparison of Rhapso and other applications according to features

3.2 Subsystem decomposition



Subsystems are divided into five layers, User Interface, Feature, Database, Second Hand and Data Processing Layer. Clients will access the whole application with User Interface Layer and this Layer is connected with the Feature Layer. The Feature layer consists of Closet, Body, Combinations, Clothe and Combination subsystems. Closet, Body and Combinations subsystems are connected to the Database Layer. Body, Clothe and Combination subsystems connected to the Data Processing Layer. Also, the Closet subsystem is connected to the Second Hand Cloth subsystem. Moreover, Data Processing Layer consists of Similar Clothing Manager, Segmentation Manager, Cloth Visualization Model and Combination Recommendation Manager.

3.3 Hardware/software mapping



Rhapso is a mobile application; in particular, we will design the application to run on mobile Android devices. Moreover, Rhapso will be released as an .apk (Android Application Package) file for our users to install Rhapso on their devices. We will make use of Android Studio IDE to construct the base interface of Rhapso, in which the users can perform operations like logging in, viewing their closets, adding clothing items, and getting suggestions. Java as the programming language will be used to implement the said interface. The visual design of Rhapso (i.e., user interface) will be determined by Extensible Markup Language (i.e., XML).

The functionalities we want to include into our design are more trivial to implement in Python programming language than Java, with the help of many ready-to-use libraries prepared for machine learning, and computer vision domains. On the other hand, the approach of locally running the complex image analysis and generation tasks is costly in terms of matching the efficiency goals we set for our design. Therefore, we are applying the help of Amazon Web Services API to host our costly operations initiated by our users.

Apart from the costly operations, and the base interface of our application, we must store and maintain our users' data globally. The said data includes registration information and application data for each user. We will be making use of Google's Firebase database to store and use user data.

3.4 Persistent data management

Rhapso has to maintain a database in order to provide the user functionalities such as maintaining a virtual closet, keeping outfit combinations within a reach, remembering 2D model options, preserving the Wishlist, and remembering the access camera option. Creating, manipulating, and maintaining the persistent data is required to save the information of the user. As the listed functionalities that require the persistent data shows, slow retrieval or the corruption of the data will affect the entire system and may cause system failure.

The persistent data will be maintained by the relational database that is accessed by queries and API services. In our mobile application, we plan to use SQL databases.

3.5 Access control and security

The system of Rhapso consists of different types of users according to their different authentication levels. Two different users of the system can be given as:

- Administrator/Developer (Admin): Developers and maintainers of the system.
- Customer Client: Customer’s smartphone app.

The access of a client to the database occurs in two different ways: read or update. Mainly, clients will access the database to get the data/monitor it, which is the reading functionality. Clients will use access points that are buttons to invoke actions in the system and consequently verified action will update the database accordingly. The updated values will be given via a text field, input stepper, selector, or button that are the user interface components. The update will only be performed if the action requested by the client is permissible to it which is controlled by the visibility/accessibility of the access points. For example, the client user will be given every flexibility to update the database for the 2D model attributes; however, the client will not be given the opportunity to add a cloth to the wishlist if it is already in his/her closet. The prevention of such an action will be done by not providing the access point (a button that adds the cloth to the wishlist) for the clothes in the closet.

Data privacy is one of the main concerns of the Rhapso. User information will be saved to the database and camera access should be given to the Rhapso in order to provide any functionality. Such actions may violate the data privacy of the client; therefore, the client user will be warned that the information s/he gives to the system will be saved and will not be distributed to 3rd parties. Moreover, the camera permission will be asked to the client user and if s/he permits, the choice for the camera access will be saved also and will not be asked every time the camera opens.

Admins are the users that are given the full permission and access to the system so that they can maintain the system and act on in the case of error and problems. Admins have full access to the database such as read, write, or update. Moreover, they have the privilege to change the authentication level of a client user such as changing access points.

3.6 Global software control

In this section, how the system will be controlled on a global scale is explained. We discuss the initialization of the requests and the synchronization of the subsystems. Lastly, we mention possible issues.

There will be two subsystems in Rhapso which are Client and Server. The general workflow is as follows: Requests will be sent to the servers from clients and the responses are sent back to the clients from the servers after being processed and taking the action. The whole procedure can be explained in three phases:

3.6.1 Request Phase

The phase starts with receiving a request from the client side. A request is sent once the user takes the photo and uploads the image to the virtual closet. This user action triggers a request which sends the photo to the server as an input to be processed by our models. One possible problem that might occur is that the server may be overloaded by the requests. The reason for this situation might be an unpredictable increase in the number of users or an attack to the servers.

3.6.2 Processing Phase

When the request is received at the respective server port, a Python script should run. It firstly applies the Segmentation model to the input image. Then, Pattern Recognition, Color and Type Detection models are applied. This phase delays the response the most. Thus, it is significant that this phase is parallelized in order to decrease the computation time. We are not able to use many servers because of the high cost. However, if the number of users increases dramatically, new servers must be accommodated which increases the cores. As a result, different inputs can be processed concurrently. In order to distribute jobs equally, Load Balancing Algorithm might be implemented which requires an extra hardware, Load Balancer, to the server side.

3.6.3 Response Phase

In this phase, the model outputs will be sent back to the clients from the ports when there is a request. In order to solve concurrency issues, an optimal port configuration will be found such that we are going to have minimized delay in the client-side.

3.7 Boundary conditions

Three boundary conditions will apply to Rhapso. Starting and terminating the application, and encountering an application failure are such circumstances. The next subsections go through these circumstances in further depth.

3.7.1. Initialization

To utilize the app, the user needs to have it installed on their phone - this process will be done through Google Play Store. The program necessitates the creation of an account and logging in with that account which will require a server connection. List of clothes in the "closet" of the user is accessed through this server connection. Such a feature requires internet connection to initialize the application.

3.7.2. Termination

When the user terminates the app, any running processes will be terminated. Any operations that were not properly saved before termination will not be able to be retrieved in the following successful runs. Such operations could include the liking/disliking option for outfits, adding a new clothing, making and adding a new combination of clothes. If these operations were not saved properly using the appointed buttons for saving before termination, then they will not be saved and retrieved.

Loss of internet connection will cause the application to be closed, however the process running will not be killed abnormally. This is also involved with the case of failures, and how such a failure will be handled is explained in more detail in the Failure section. In the case of termination during the process of attempts at reconnecting, the same processes described with the case of user termination of the app will be applied.

3.7.3. Failure

When a user enters an invalid username-password combination or attempts to create an account with a username which already appears in the database, then the authentication process fails. These failures of user authentication will be handled by prompting the user to enter the inputs for username and password again.

Internet connection failures can happen during the use of Rhapso. Since all of the features in Rhapso require internet connection to work as it extracts data of each user from a database, the application needs to handle such an issue properly. In the case of a loss of connection, the system will make the user wait for a while with a waiting screen. The application will try to reconnect once, then the waiting screen will have a prompt to try to reconnect or close the application. In the case of reconnection at a proper time, the user has a chance to retrieve their last process as the data will first be stored in cache memory. However, cache memory is very small and it is only temporary as new data from the phone may be stored over the data of the last process.

One case of failures happens with the use of pictures taken by the camera accessed from the application. The user needs to open the camera for them to take a new photo of clothing to add to their closet. The picture taken is sent to the data layer where the clothing in the image is identified, and if there are no clothes detected by the machine learning model then the application informs the user as so. The application will prompt the user to either go back to the main screen or take the photo again.

When a failure occurs because one of the modules ceases operating caused by a faulty within code or perhaps a server problem, then the system will notify the user that such a module is unavailable at that moment and as such the user would be informed once the module is available.

Lastly, because the server will always be operational, meaning the application will need data extracted from a database in a server as well as using different machine learning models from a server, it must be fault-tolerant. It has to not only have a system in place to deal with exceptions and boundary cases, but also deliver correct error messages to the client to alert the user as it manages these errors. Various failure and response models will be used to implement this system.

4. Subsystem services

4.1 User View Layer

The user view module is entirely constructed for the use of the client. This layer connects the user to the system. Every system functionality, data, and action is presented via the user view layer to the client. Moreover, the system is invoked by the user interface and actions performed in sublayers takes the parameters and execution signal from this layer. Then, the results come back to the user view layer to be presented to the client.

4.2 Feature Layer

4.2.1 Closet

This component is used to contain users' clothing items. After the pictured clothing item is processed by the related component (i.e., the Segmentation Module) in Rhapso, the item is added to the closet. This container module can be both used by the users to view their items in the closet, and the system to run other modules (Combination, Outfit Recommendation Manager) for further operations like combination, and recommendation generation.

4.2.2 Body

This subsystem is for getting body information of the users from the user with the user interface. Measures of the body components will be held within it and these values will be used to visualize the body, later. Therefore, it is connected to the Closet and Combinations subsystems to show the Clothes and the Combinations of the user on the specific body. It will use ClothVisualizationModel to visualize clothes on the bodies.

4.2.3 Combinations

With this subsystem, combination recommendations will be accessible by users and new combinations can be created by the users. Therefore, it is connected to the Closet subsystem to get the Clothes of the user.

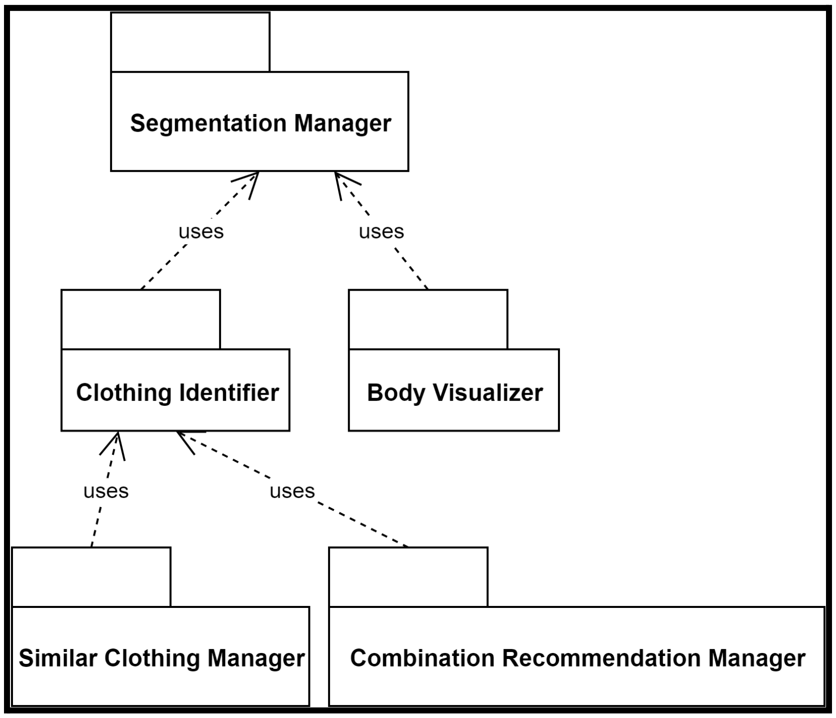
4.2.4 Combination

This subsystem will be used by Combinations subsystem to differentiate the combinations created by the user and recommended by our system. It will consist of the Clothes. Combination Recommendation Manager is used by this subsystem to create combination recommendations.

4.2.5 Clothe

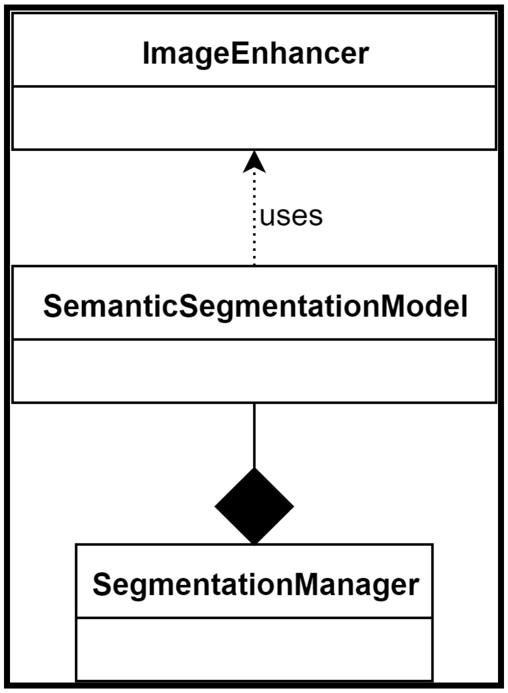
Most used subsystem is this Clothe subsystem. Image and the properties of clothes will be implemented in this subsystem. Also, it depends on the Second Hand Clothe Layer to recommend clothes. Moreover, Similar Clothing Manager is used by this subsystem to find similar clothes and Segmentation Manager is used to separate clothes in the images from the background.

4.3 Data Processing Layer



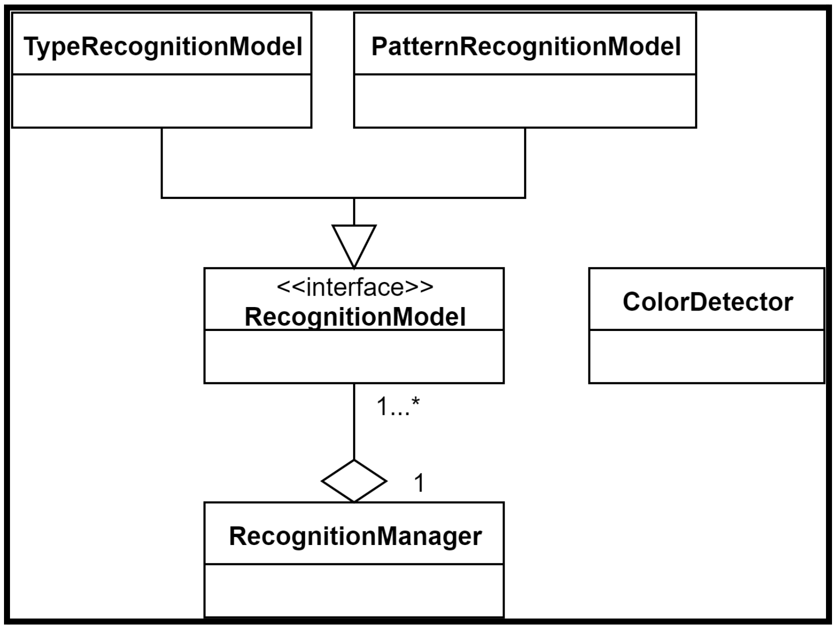
This layer of subsystems represents the layer at which data retrieved from the user, namely the clothes in the closet, are processed and passed to be used in different aspects of the application. Each of the subsystems above is described below in their own subsections in further depth.

4.3.1. Segmentation Manager



The meaning of segmentation in this case will be to remove the background of a clothing object. We aim to achieve this by combining two different methods. First, the image will be enhanced by applying contrast to better identify the difference of background and foreground (the clothing) pixels using image processing techniques. The second method aforementioned is called Semantic Segmentation for which we will train a model with a dataset of segmentation of clothes to achieve removal of background from foreground. The SegmentationManager class will be responsible for applying the output of SemanticSegmentationModel to the original, unaltered image for displaying to the user.

4.3.2. Clothing Identifier

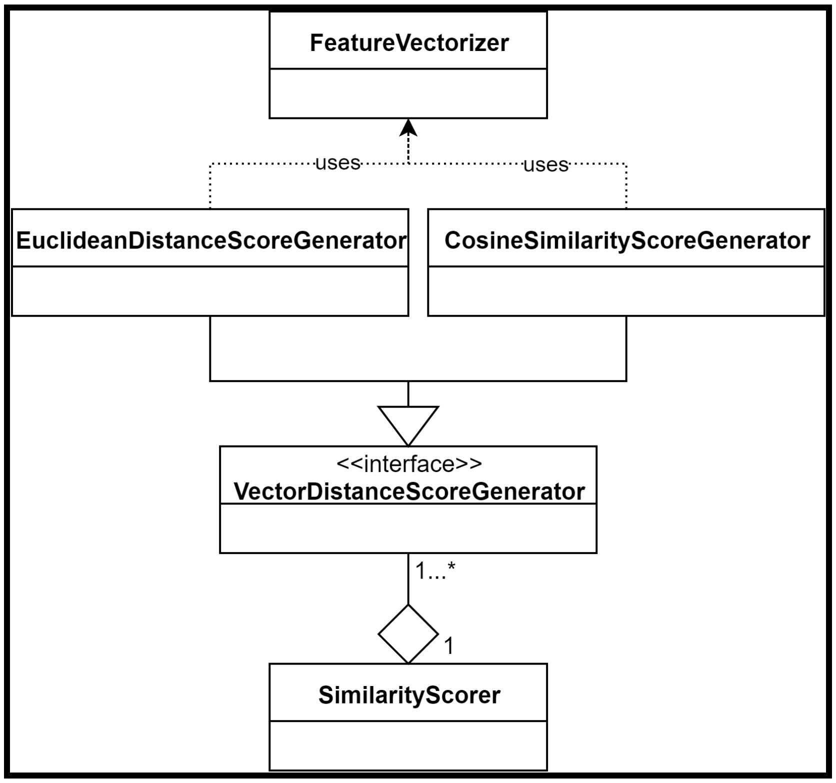


A specific model trained for clothing type recognition and pattern recognition will be used in this subsystem. The classes for clothing types, as of planning now, will include shirts, t-shirts, tops, long-sleeves, pants, shorts, jackets, blazers, and dresses; there will be more classes added as such a need arises. There will be four top classes that these smaller classes will be in, named Top (for t-shirts, shirts, long-sleeves and so on), Bottom (for pants, shorts, and so on), Outwear (for blazers, jackets, and so on), and Dress. A specific model is also trained for pattern recognition such as plain, floral, polka dot, squares, or stripes. Residual CNNs will be used for image recognition.

ColorDetector class will not be using this model, as it will only act to extract colors from an image of a clothing using image processing techniques. The colors defined in this system are from keywords of basic colors as included in the World Wide Web Consortium standard [15].

This subsystem primarily utilizes the segmentation subsystem. Outputs of these classes will be used for classifying and identifying each clothing.

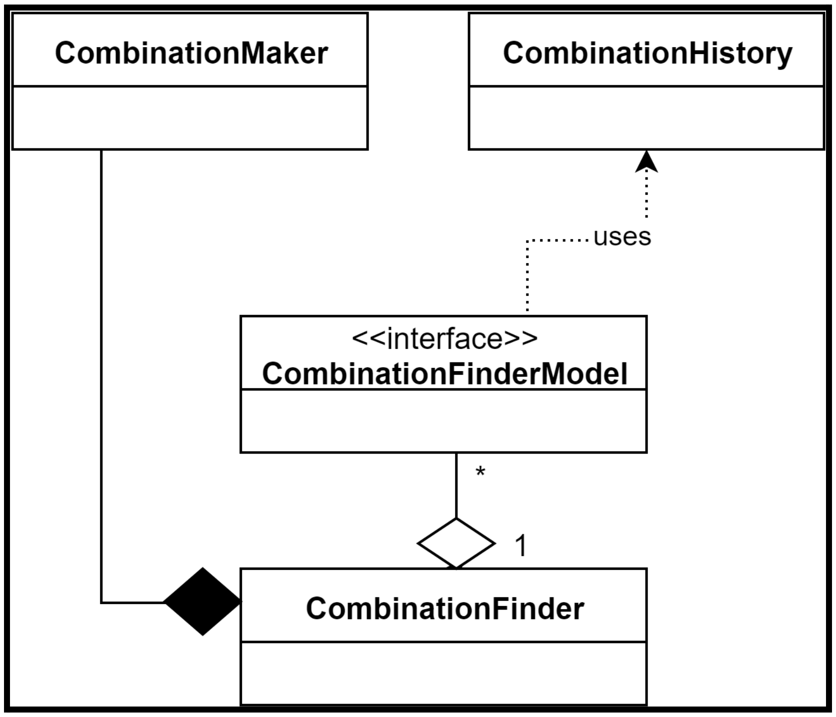
4.3.3. Similar Clothing Manager


FeatureVectorizer
uses
uses
EuclideanDistanceScoreGenerator
CosineSimilarityScoreGenerator
<<interface>>
VectorDistanceScoreGenerator
1...*
1
SimilarityScorer

FeatureVectorizer is responsible for creating vectors for the clothes being compared, and it will output two feature vectors of two clothes. Feature vectorization process will use the outputs of the Clothing Identifier system from which it will derive the color/s, clothing type, and pattern of the clothing. The output of this will be passed down to produce a similarity score for these vectors using two different vector similarity calculations: Euclidean distance, and cosine similarity. Euclidean distance calculates the distance of two vectors, and the similarity score for this distance value will be generated by normalizing the distance. The result of distance will be higher as the two vectors are more similar which will be dealt with in the score generation process. Cosine similarity produces a result that gets higher as the similarity increases, this will also be dealt with in the score generation process. Each of these methods will output a score percentage. Finally, SimilarityScorer will take the average of these results and output this result.

This class is responsible for finding the similar clothes using the outputs of Clothing Identifier, thus consequently it also utilizes the Segmentation subsystem.

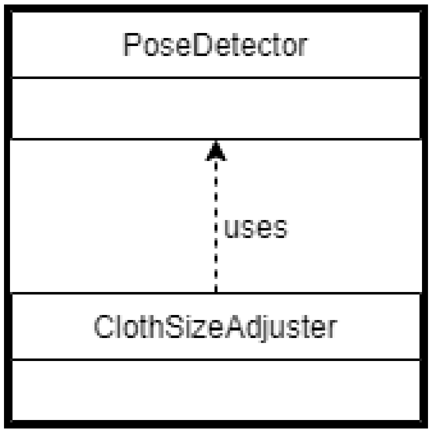
4.3.4. Combination Recommendation Manager



The CombinationMaker class will be responsible for generating combinations of clothes utilizing clothing data from the database layer. The recommendations will be given using a content-based recommendation system, meaning the properties of combinations that the user has liked or submitted previously will be taken into consideration when a recommendation is given. The CombinationFinder model will look at the euclidean distance of combinations in accordance with the model that has benefitted from the user's records, and will recommend combinations of closest distances to the vectorial representation that the model acquired.

The subsystem will utilize the outputs of Segmentation and Clothing Identifier subsystems.

4.3.5. Body Visualizer



The Body Visualizer subsystem will be responsible for showing outfits on a body. There will be a PoseDetector model which is responsible for detection of body parts. A human body photo will be given as an input to the PoseDetector model and this model will detect the body parts.The ClothSizeAdjuster model will be responsible for adjusting the size of the cloth with respect to the dimensions of the detected human body. It will be based on GAN (Generative Adversarial Network) technology. It will take the body image and the cloth, the output of the Clothing Identifier System, as input and produce a resized image as an output. Finally, the resized image will be demonstrated on the detected body.

4.4 Database Layer

We consider the Database Layer as a peerless layer, such that the one and only component we consider is the database layer itself. The functionality of the Database Layer is relatively straightforward in comparison with the previous layers and their contained components. While considering the Database Layer, we think of a component, which will be closely related to the "Feature" and "Data Processing" layers. One can interpret that for this layer, we are adopting such an approach which will communicate with other layers in an interactive manner rather than a hierarchical appeal. To further elaborate, some of the sample functionalities of the Database Layer include providing data for user registration directly to the base interface of the application, without consideration of the complex operations like pattern recognition, or providing the visual data for complex operations. Hence, whenever needed by other layers, Database Layer as a component provides this information as stored data to these layers.

4.5 SecondHandClothes

This layer is for accessing "Dolap" website, to show the similar second hand recommendations for clothing.

5. Consideration of Various Factors in Engineering Design

During the development of Rhapso, a number of things will be taken into account. At some point, such circumstances will restrict us.

5.1 Social Factors

Age, economic position, religion, and race have all been taken into account by our team. How individuals utilize Rhapso will alter.

A teenager, for example, will mostly use Rhapso for outfit ideas, but an adult may use it more for not purchasing identical clothing. As one gets older their preferences will change - the outfit recommendation system will be affected by one's preferences and their attitude towards recommended outfits and their own submitted outfits.
People belong to many religions, and their attire will alter depending on their religious attitude. Rhapso will recommend ensembles based on one's current clothing, and will recommend such that the outfits reflect their attitude towards clothing combinations. The algorithm for outfit recommendation will behave in an individual way. The model will first follow a general set of aesthetic standards and will recommend outfits as such. As it learns about the user's choice of outfits, liking or disliking the recommendations and/or adding their own outfit combinations, it will behave such that the combinations now will match the user's preferences.
Depending on their financial situation, those with lower income may prefer to only get the item they truly require. They will be especially interested in the similarity score feature of Rhapso from this perspective.

5.2 Economic Factors

There are economic benefits provided by Rhapso. The main aim of Rhapso is to reduce the shopping tendencies for new clothing garments by encouraging buying secondhand. Of many causes of over-shopping, Rhapso aims to delve into two main reasons of this behaviour: The lack of information on whether the shopper needs a new clothing garment, and the complaint of no combinations of clothes available.
When the problem of no combinations of current clothes is arrived at, the issue of having no outfits to wear is solved artificially by buying a new combination of clothes. Our team has recognized that this problem could be solved by an app with a feature of outfit recommendation from existing pieces of clothes. Rhapso will have a feature that recommends outfits by using the clothes in the virtual wardrobe. Therefore, even if the people cannot decide on what to wear, Rhapso will help them by generating suggestions and this will decrease the desire of an individual to buy new clothes instead.
As for the problem of the lack of information on needs for clothes: For instance, in the winter, summer garments are packed away in bags, and summer items are often discounted in the winter. Many individuals take advantage of this chance to acquire cheaper items; nevertheless, they may forget about their old garments and purchase items that are extremely similar. Rhapso also has a similarity score function, which calculates a similarity score for the provided garments based on the ones in the virtual closet. As a result, people may not favor the same outfits and save money.

5.3 Environmental Factors

There is a high environmental damage caused by excessive shopping behaviour as discussed in According to data, fashion production accounts for 10% of global carbon emissions [16]. Overconsumption of fashion dries up water supplies and pollutes rivers and streams [16]. These figures illustrate that clothes use is becoming an increasingly serious environmental issue. The team for Rhapso takes such environmental factors into account while developing the application. One of our main aims is to partake in reducing this behaviour of excessive shopping as much as possible. Rhapso delves into this issue in two different ways. One is reflection on the user's wants of consumption of clothes, by keeping the user in touch with how they would use each item through the combination recommendation mechanism. This mechanism will make it so that the user will have different ways of utilizing their already owned clothes, and as such our hope is that their belief on "need" of other clothes will cease. The other main way of delving into this issue of overconsumption will be through the similarity functionality of the application. Through this mechanism the user can feed the application a photo of a clothing garment that they want to purchase. The application will report on whether the user already has similar clothes in their closet, or if they could replace this new piece of clothing with a more environmentally aware choice - such choice will be a similar clothing choice from a selection of clothes in a secondhand clothing purchase website. So, the user will be incentivized to either not buy a new garment because of their information on whether they truly need the new garment, or replace their choice of a new clothing garment with a secondhand clothing piece through which they will not be contributing to the overconsumption of new clothing pieces.

5.4 Global Factors

In our discussion, what is meant by global factors is the rules, norms and some regulations that countries have an agreement on. Some of the standards are General Data Privacy Regulation (GDPR) [17], Google Play Terms of Service [18] and other global agreements between the developers, users and governments.It is significant that we limit ourselves to these factors since those are the opinions of the public who will be using our product. Thus, considering those factors during development is important.

5.5 Cultural Factors

The combinations of clothes created for each user will use a model that will change its behaviour based on the features of the outfits that the user has declared as "liking" to the application. Each new recommendation will be generated by using the previous data on pairs of clothes. A different suggestion algorithm that doesn't take each user's different preparations into account would have been not ideal because of cultural factors that differentiate each preference. These models would consequently be biased towards what the original dataset used for training mostly represents, i.e. a dataset consisting of outfits with western standards would not take outfit decisions with more eastern influences into account. As such, an individualistic model would prove more efficient in preventing such cultural bias.

6. Teamwork Details

6.1 Contributing and functioning effectively on the team

In our team, tasks are divided equally as much as possible while the responsible person for these tasks is chosen voluntarily. Therefore, everyone in the group is willing to do her/his task and finish them before the chosen deadlines. While choosing a task, we also care about the wisdom of each other and we increase the contribution of each while helping each other.

6.2 Helping creating a collaborative and inclusive environment

As mentioned earlier, everyone in the group helps when they know and/or learn new features on a topic. Also, we are paired on tasks to increase the collaboration to ease the learning process.

6.3 Taking lead role and sharing leadership on the team

When we are paired for a task, one of the members is leading the pair. Therefore, everyone becomes the leader of her/his task and updating others about the obstacles and improvements. Therefore, we divide the responsibility of the leader of the whole project as leaders of the tasks. Also, in each meeting, one of us leads the meeting and takes the important decisions in the meetings.

7. References

[1] "Listening is everything," *Spotify*. [Online]. Available: https://www.spotify.com/us/. [Accessed: 11-Oct-2021].

[2] *Google maps*. [Online]. Available: https://maps.google.com/. [Accessed: 11-Oct-2021].

[3] Wolfreys, J. (2009). In *Glossalalia: An alphabet of critical keywords* (pp. 358). essay, Edinburgh Univ. Press. [Online]. Available: https://books.google.com.tr/books?id=9b1UZF3pzWkC&printsec=frontcover&redir_esc=y#v=onepage&q=rhapso&f=false. [Accessed: 21-Oct-2021]

[4] M. McFall-Johnsen, "The fashion industry emits more carbon than international flights and maritime shipping combined. Here are the biggest ways it impacts the planet.," *Business Insider*, 21-Oct-2019. [Online]. Available: https://www.businessinsider.com/fast-fashion-environmental-impact-pollution-emissions-waste-water-2019-10#in-europe-fashion-companies-went-from-an-average-offering-of-two-collections-per-year-in-2000-to-five-in-2011-3. [Accessed: 11-Oct-2021].

[5] C. Moser, S. Y. Schoenebeck, and P. Resnick, "Impulse buying," *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019.

[6] By: IBM Cloud Education, "What are neural networks?," *IBM*. [Online]. Available: https://www.ibm.com/cloud/learn/neural-networks. [Accessed: 29-Oct-2021].

[7] By: IBM Cloud Education, "What is deep learning?," *IBM*. [Online]. Available: https://www.ibm.com/cloud/learn/deep-learning. [Accessed: 29-Oct-2021].

[8] By: IBM Cloud Education, “What is computer vision?,” *IBM*. [Online]. Available: https://www.ibm.com/cloud/learn/computer-vision. [Accessed: 29-Oct-2021].

[9] “Machine learning glossary | google developers,” *Google*. [Online]. Available: https://developers.google.com/machine-learning/glossary?hl=en#generative-adversarial-network-gan. [Accessed: 25-Oct-2021].

[10] “Graphics Processing Unit,” *Wikipedia*, 21-Oct-2021. [Online]. Available: https://en.wikipedia.org/wiki/Graphics_processing_unit. [Accessed: 25-Oct-2021].

[11] J. Brownlee, “4 distance measures for Machine Learning,” Machine Learning Mastery, 19-Aug-2020. [Online]. Available: https://machinelearningmastery.com/distance-measures-for-machine-learning/. [Accessed: 23-Dec-2021].

[12] J. Han, M. Kamber, and J. Pei, “2 - Getting to Know Your Data,” in Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, 2012, pp. 39–82.

[13] “Tryndbuy,” Brand. [Online]. Available: https://tryndbuy.com/. [Accessed: 15-Nov-2021].

[14] “YourCloset,” Closet Organizer & Smart Fashion App for Android. [Online]. Available: https://www.yourclosetapp.com/. [Accessed: 15-Nov-2021].

[15] “CSS/properties/color/keywords,” *CSS/Properties/color/keywords - W3C Wiki*. [Online]. Available: https://www.w3.org/wiki/CSS/Properties/color/keywords. [Accessed: 23-Dec-2021].

[16] M. McFall-Johnsen, “The fashion industry emits more carbon than international flights and maritime shipping combined. Here are the biggest ways it impacts the planet.,” Business Insider, 21-Oct-2019. [Online]. Available: https://www.businessinsider.com/fast-fashion-environmental-impact-pollution-emissions-waste-water-2019-10#in-europe-fashion-companies-went-from-an-average-offering-of-two-collections-per-year-in-2000-to-five-in-2011-3. [Accessed: 11-Oct-2021].

[17] “General data privacy regulation.” https://eugdpr.org/. [Accessed: 6- Nov- 2019].

[18] “Google play terms of service.” https://play.google.com/about/play-terms/ index.html. [Accessed: 6- Nov- 2019].