Bilkent University

Department of Computer Engineering

CS 492: Senior Design Project

Rhapso

Final Report

Group Members:

Işık Özsoy	21703160	isik.ozsoy@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
Burak Yetiştiren	21802608	burak.yetistiren@ug.bilkent.edu.tr

Supervisor : *Prof. Dr.* Halil Altay Güvenir

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

Innovation Expert: Aras Bilgen

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

Final Report

May 6, 2022

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 CS491/2.

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 have 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 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 this 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 the 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.

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.

2. Requirements Details

2.1. Functional Requirements

In this section, we will discuss the functional requirements of Rhapso. Those requirements are helpful to understand the intended behavior of the system.

2.1.1. System Functionality

- System should obtain authorization from the user to use the resources of the phone, i.e. camera, etc.
- System should make it possible to provide camera input with an option.
- System should provide an option for registration with a new account, making the user provide a username, an email and a password.
- System should provide an option to change username, email, and password.
- System should provide an option for loginning to the app with a pre-existing account.
- When a new photo of a clothing is provided, the system should distinguish the clothing itself from the background.
- The system should provide an option for the user to add their own outfits to the application by combining the existing clothes in the closet.
- When a new photo of a clothing is provided and when the system has successfully distinguished it from the background, the system should provide options to add the clothing to the wish list, add the clothing to the user’s virtual closet, or compare the clothing with the rest of the clothes in the closet.
- When the user has clicked on a clothing item inside their “closet” or their wish list, the system should provide an option to show creations of combinations of the selected clothing with existing clothes inside the closet. The “outfit creator” that creates custom combinations of clothes in the closet should do these combinations to ascribe to certain aesthetical rules, such as how the colors of the clothes the outfit generator combined should match in terms of color theory (considering hue and value of the color of the clothing) and in terms of patterns.
- The system should provide an option to keep or discard the outfit recommendations the outfit creator recommended. If they keep the outfit recommendation, it will be added to the list of their own outfits. If they choose to discard the recommendation, the system will show another recommendation of combinations.
- The system should learn from the user’s choice of outfits, from the clothes they combined as submitted outfits, and from the recommendations of outfits the user decided to keep and discard.

2.1.2. User Functionality

- Users should be able to accept or reject the permission for the use of the camera. However, as the acceptance of such permissions is essential for the application to work, primary features of the app will be unavailable in the situation of rejection. The rejection will not be taken permanently and the application will continue to ask for permission when the camera feature is tried to be used.
- Users should be able to give photo input to the application through the phone camera.
- Users should be able to register to the application with a username, an email and a password.
- Users should be able to add clothes to and remove clothes from their “closet” or their “wish list” stored in the database.
- Users should be able to get recommendations of new clothes they want to buy from online alternatives of secondhand clothings.
- Users should be able to retrieve their previous information stored in the application later and with different applications if they choose to store their data with a registered account.
- When the user takes the photo of a new clothing piece using the camera feature of the application, the user should be able to see possible combinations of this new clothing piece with the clothes already present in their “closet”.
- When the user takes the photo of a new clothing piece using the camera feature of the application, the user should be able to see a comparison score of the new clothes with the ones already present in their “closet”.
- Users should be able to see possible outfit combinations of clothings in their “Closet” or “Wish List” with the clothes already present in their “closet”.
- Users should be able to virtually try the clothes present in their “closet” as well as a new dress they want to buy by trying them on a body model provided by the application, provided they took photos of each piece of clothing beforehand with the application camera.
- Users should be able to add new clothing to their “wish list” in the application.
- Users should be able to keep or discard the outfit recommendations created by the “outfit creator”

2.2. Nonfunctional Requirements

2.2.1. Security

The application must be secure enough with passwords and usernames so that no other outsider could access another account’s information directly. The application must ensure that the camera inputs given to the application will not be accessed by a 3rd party application. The application should ask the user permission for enabling access to camera and photo gallery before the first use of such features. Such features should only be accessible by the application only when the application is running and these features are called by the application itself.

2.2.2. Usability

The application must be supported by Android and supported by different versions of Android phones. The system should have an interface that is easily understandable by users. Its interface design should be usable with different sizes of mobile phones.

2.2.3. Performance

The system needs to access the database for a very large number of users, for this reason we want the application to perform very fast. The application must be fast enough with its processing of clothing images such that its segmentation and the feature of comparing the newly acquired clothing piece with the ones already saved in the database must take no longer than 1 second. When the user wants to see recommendations for other clothes, it should take no longer than 1 second for the application to search through secondhand clothing websites, otherwise known as online thrift stores. As for the virtual clothing trying-on aspect of the application, the application must take no longer than 1 second to add each piece of clothing to the model the clothing piece is added onto.

2.2.4. Reliability

The application must detect clothes, particularly their structure and their textures if they have anything printed on them, correctly and render new images of the clothes, adjusted to the body model that we will put the clothes on to, as close to real-life as possible. The application should also give correct approximations of the closeness of the new clothing piece that is being scanned when it is compared to the clothes already present in the user's "closet", i.e. it should not give too high of a closeness score to two clothes that are similar in no way. These are factors that will determine the user's satisfaction with the application, hence the necessity for the reliability of them.

Additionally, the system should notify the user as so in the case of a bad or unintelligible clothing photo, e.g. the photo has no clothing in it, the photos of clothes are not taken from the front view which makes the photo a bad candidate for the virtual try-on aspect of it, etc. The system should have reliable failure management. It should not have any errors or exceptions upon release. Stack overruns, memory management problems, and such non-deterministic application failures should be handled by causing the app to rollback, and thus force a restart. The application should have reliable input/output management so that no failures in direct response to data input or a user input occur.

Hosting the app' APK on Google Play offers some reliability benefits [6] [7], including

- Hosting the app on high-uptime Google servers; through this, dealing with server failure problems is cheaper (as we will be leaving the issue mostly to Google services and will not need to buy better hardware for the application)
- Compatibility testing for devices will be automated and done for each mobile device for the application
- Download times are faster than apps hosted in external servers
- Data consumption is reduced when the app is hosted on Google servers

2.2.5. Scalability

The application should have a maximum of 1 second response time, with the general aimed response time being 45 milliseconds. The application should be able to handle 200 requests per second (12,000 per minute) as the count of requests around this value or more would result in a heavy load for smaller servers, and we aim to handle heavy load situations as well.

2.2.6. Accessibility

The application should be available and freely downloadable from the Play Store hosted by Google for easy access for Android phone users.

2.2.7. Extensibility

The application should be easy to manage for future extensions. Although the application will initially be tailored for Android phones, its interface design should be extensible for different mobile OS systems such as iOS. The development environment should allow for new features to be added with ease if the need occurs.

3. Final Architecture and Design Details

3.1. Overview

In Rhapso, we offer many functionalities, such as combination recommendations, and secondhand recommendations etc. 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.

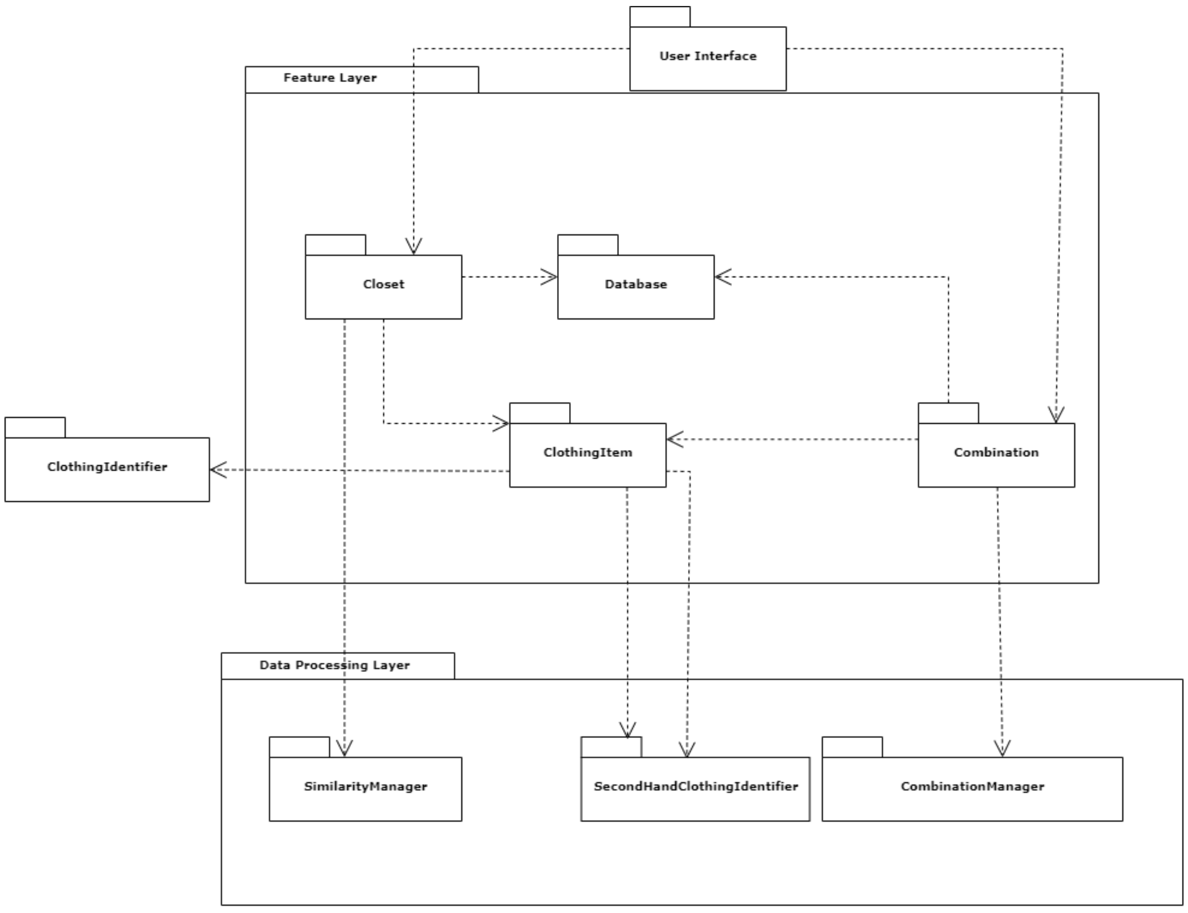


Figure 1: Subsystem decomposition diagram of Rhapso

Subsystems are divided into four layers, User Interface, Feature, Database, and Data Processing Layers. 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, ClothingItem, and Combination subsystems. Closet and Combination subsystems are connected to the Database Layer. ClothingItem and Combination subsystems are connected to the Data Processing Layer. Also, the Closet subsystem is connected to the Second-Hand Cloth subsystem. Moreover, Data Processing Layer consists of a Similar Clothing Manager, Segmentation Manager, Cloth Visualization Model, and Combination Recommendation Manager.

3.2. Server Architecture & Design Choices

Reliability and short response time are the main goals in our design since those servers are responsible for significant tasks. We tried to make our servers as reliable as possible and distribute different tasks independently in order to reduce the risk of a crash causing another crash. Thus, we have separated the server work into the following:

3.2.1. Clarifai API

Clarifai Inc. is artificial intelligence (AI) company that focuses on computer vision using machine learning and deep neural networks to identify images [8]. Clarifai provides a complete platform to deploy, maintain, and manage the AI models. Clarifai provides 27 different models and we have used 3 of them which are the apparel model, color model, textures, and patterns model.

3.2.2. Remove.bg API

Remove.bg is an online service of Kaleido AI used to remove the background (i.e., segment) of the given image. The service provides API access that the developers can integrate into their code [9]. This API was used to fetch the significant part of the given picture by removing the background of the clothing item. The resulting image was used to store the clothes of the users in their closets.

3.2.3. Dolap API

To serve options for secondhand clothes we used the Dolap. To get good choices we created a filtered URL and got secondhand clothes according to this filter. To create this filter, Dolap API gets the properties of the cloth from the Clarifai API.

3.3. UI Classes

With SignUp and SignIn classes, users can create an account and access their closet. They can see their information in the UserProfile class and update them in the Settings class.

When the user enters the application they can see their closet with the ClosetFragment class.

When they want to take a photo of their clothes the CameraFragment class will be used and it will direct the user to ItemActions class which provides the features of the application and calls the Clarifai API, image segmentation and DolapAPI. From the ItemActions class, users can be directed to SecondHand class that lists the images and buttons that direct to Dolap webpage or to the Similarity class which will provide similar clothes to the image that user took or CombinationAfterItemActivity class that will show the possible clothes that will be combined with the current clothe. Also, the user can add the clothes to the closet in ItemActions or they can add it to the wishlist by using the Wishlist class.

4. Development/Implementation Details

4.1. Mobile Application

Our implementation consists of one main part, namely the ‘Mobile Application’. We developed an Android application, hence we used Android Studio as our IDE to develop the application. We believe that Android development is more straightforward than other mobile platforms; moreover, the abundance of Android users, as well as the ease of development offered by Google by providing the IDE, was one of the main points behind our decision. Additionally, Android development can be conducted by Java programming language, with which all of the group members were familiar. Naturally, this was crucial as we wanted everyone to be actively involved in the implementation phase of the application. As we implemented Rhapso, the integrated emulator to Android Studio, and the personal phones of the members, who had Android phones were used. The main purpose we used these was to test our application by running it on said artifacts. We believe that testing was one of the main challenges we faced during the implementation of our project. Android Studio introduced some obstacles while we tested our implementation. These were faced mostly while we were diagnosing the errors we received. The inability to reach the main memory of the computer and the not straightforward image printing for diagnosis was challenging.

4.2. Database

We used Google's Firebase Database to store the information that would be provided by the users of Rhapso. Firebase is a NoSQL database, which is an enhanced version of the conventional relational databases. We know that such databases can house larger data more efficiently; having in mind that Rhapso has the potential to grow with many users, the data that will be stored in our database can grow exponentially. The other reason we employed Firebase was, that the service can be easily utilized in the Android Studio IDE. This way, we neglected any extra possible baggage that could be introduced in the forms of efficiency-decreasing or ease-eliminating aspects.

4.3. Version Control Systems

For a more efficient implementation procedure, we applied Git & GitHub to our development cycles. By introducing such version control systems (VCS) each of the team members could restfully implement their part of the code, without worrying about malfunctioning the code, as we also implemented 'Continous Integration' into our development cycles. Further details about this part of our development/implementation procedures are given in Section 5.

5. Testing Details

5.1. Continuous Integration

Our project is hosted on GitHub, where each member of the team had access. Our 'master' branch was the central repository, where each member of the team merged their new code after they tested it on their personal branch. Our approach was to create a new branch for the implementation of the different aspects of the project. For each aspect, the members assigned to that part created branches with their names and the name of the aspect and saved their work there. After the implementation was complete, the functionality was merged to the 'master' branch. This way, the members could be assigned to the implementation of many functionalities, and they were able to switch freely between their tasks.

5.2. Automated Code Review

Communication was one of the most important aspects of our project. While we ensured healthy communication between the team members, we followed a similar convention in the code we have written. As each team member needs to understand what any given member has written, or has done, we adopted the approach, where aspects like commits, code variables, and code comments should be as straightforward as possible. Moreover, the code, in general, should be readable with consistent indentation and proper spacing along with the above-stated conventions. We believe that such conventions increased our development pace, as we did not lose additional time to understand each other.

5.3. Testing the UI

One of the primary concerns was testing the user interface. In order to test it, we have talked with a group of people and asked them to use Rhapso and to give feedback on their experience. Those people were not software developers and did not have any technical knowledge. As we added new functionalities, they tested the UI. Then, they reported the bugs and issues and also provided their feedback. With the help of this group, we spent less time finding bugs and easily resolved the existing bugs. Furthermore, we have improved the design based on the reported feedback.

6. Maintenance Plan and Details

As similar to many software, Rhapso also needs to be regularly maintained to sustain its functionalities. In the following, some of the aspects of Rhapso that need further upkeep are listed.

6.1. Server Maintenance

We use Google's Firebase Database Service to store user information in our application. Currently, the free version of the database is sufficient to accommodate the small volume of data we have at the moment. But as the number of users, followingly the size of total data increases, our current approach will be not sufficient.

6.2. API Maintenance

As explained in Section 3.2., we utilize some third-party APIs to satisfy some of the functionalities we present in our application. While in the current phase of our application the low-volume usage of these APIs is sufficient, in the long run, some enhancements can be needed (i.e., increased subscription plans). Furthermore, the functionalities that the APIs are offering could be implemented as a future work of the implementation of Rhapso. We believe that we can reduce the dependence of Rhapso on third-party applications by following such approaches.

6.3. Google Play

One of our aims is to host Rhapso on the Google Play Store, making it available to be downloaded free of charge. The consultations with the experts should be completed prior to this public release of Rhapso.

6.4. Actions on Feedback

Any possible feedback on any aspect of Rhapso is crucial for us. Hence, the required updates regarding the feedback we receive are one of the main points of consideration about the maintenance part of Rhapso.

7. Other Project Elements

7.1. 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.

7.1.1. 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 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 recommendations 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.

7.1.2. 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 for this behavior: 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 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.

7.1.3. Environmental Factors

There is high environmental damage caused by excessive shopping behavior. According to data, fashion production accounts for 10% of global carbon emissions [10]. Overconsumption of fashion dries up water supplies and pollutes rivers and streams [10]. 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 behavior of excessive shopping as much as possible. Rhapso delves into this issue in two different ways. One is a 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 in the “need” for 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.

7.1.4. 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 the General Data Privacy Regulation (GDPR) [11], Google Play Terms of Service [12], 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.

7.1.5. Cultural Factors

The combinations of clothes created for each user will use a model that will change its behavior 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.

7.2. Ethics and Professional Responsibilities

We used Google Firebase to avoid data leaks and get user approval to send their data to Clarifai API. Other than that, their liked and disliked outfits are used to personalize the recommendations of outfits.

7.3. Judgements and Impacts to Various Contexts

Any given impact in the table below are out of 10.

Impact in:	**Impact Level**	**Impact**
Global Context	6	Rhapso is a virtual wardrobe app that can be used globally.
Economic Context	2	Costs of APIs and cloud services restrict the effectiveness.
Environmental Context	8	The 'sustainable alternative suggestion' module of Rhapso has the potential to reduce the orientation towards high textile consumption.
Societal Context	7	Rhapso collects personal information, as well as the images that the users produce. careful conservation of such data is highly crucial.

Table 1: Impacts of judgement

7.4. Teamwork Details

7.4.1. Contributing and functioning effectively on the team

Tasks are divided so that each of them can be done until the next meeting. Therefore, the progress of each member could be seen and obstacles could be detected and resolved.

7.4.2. Helping creating a collaborative and inclusive environment

Each task is directed by one member; however, when it affects others' parts, pair works are done. Also, seen errors are investigated by two members that collaboration decreased the time that we worked on the problems.

7.4.3. Taking lead role and sharing leadership on the team

As we mentioned before, each task has been led by a member and s/he assigned necessary subtasks to others. Also, during the meetings, the leader of the meeting has changed and necessary notes were taken by the leader of that meeting.

7.4.4. Meeting objectives

During the meetings each member talked about which task s/he was assigned to in the previous meeting and the process s/he got until. If there are any tasks left, we talk about the reasons and solutions and decide on what to do until the next meeting.

7.5. New Knowledge Acquired and Applied

Google Firebase was learned and applied to the database. From the website, values are get and processed to use in the application. Clarifai API is learned and POST/GET requests are used to get the necessary information. For clothes segmentation, machine learning and computer vision algorithms are used.

8. Conclusion and Future Work

8.1. Conclusion

In the last year, we converted our unique idea of transferring one's physical closet into virtuality by creating a mobile application. In our application, the users can find many helpful extensions to the traditional closet like gathering outfit recommendations and the suggestion mechanism we implemented to remind the user that there can be alternatives in the second-hand market, which can be a more sustainable alternative when the negative effects of the garment production can cause to the environment when the user wants to buy clothes.

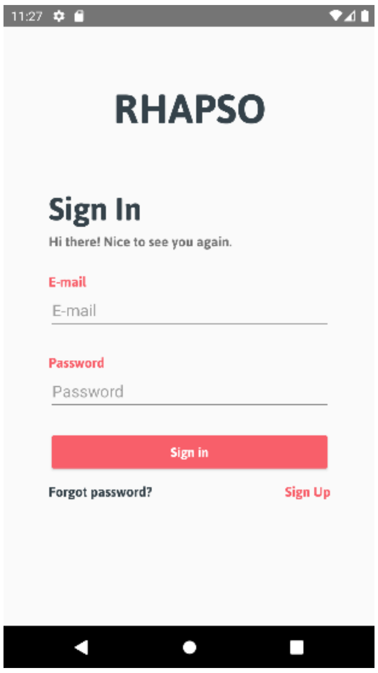
We are pleased to be able to deliver a functioning real-time model of what was once an idea in our minds. The process was educational for each member of the team; we reinforced many of our skills, including coding, communicating, and documenting.

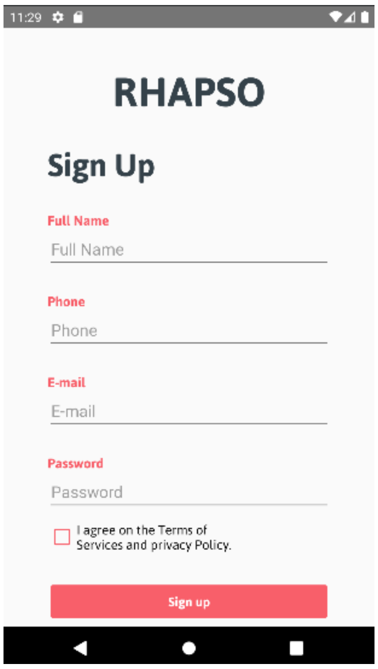
8.2. Future Work

We are eager to improve the possible aspects of our application in accordance with any feedback we may get from our peers, professors, or others. Needless to say, we welcome any positive or negative feedback, which we believe can carry our work further. While this is applied, we would like to reach to a wider audience than the scope of the senior design project course, and hopefully make people, especially the ones for which our application might be useful, aware that such an application exists. We believe that the wider our audience is, the more we can learn, and develop ourselves and our application.

9. User Manual

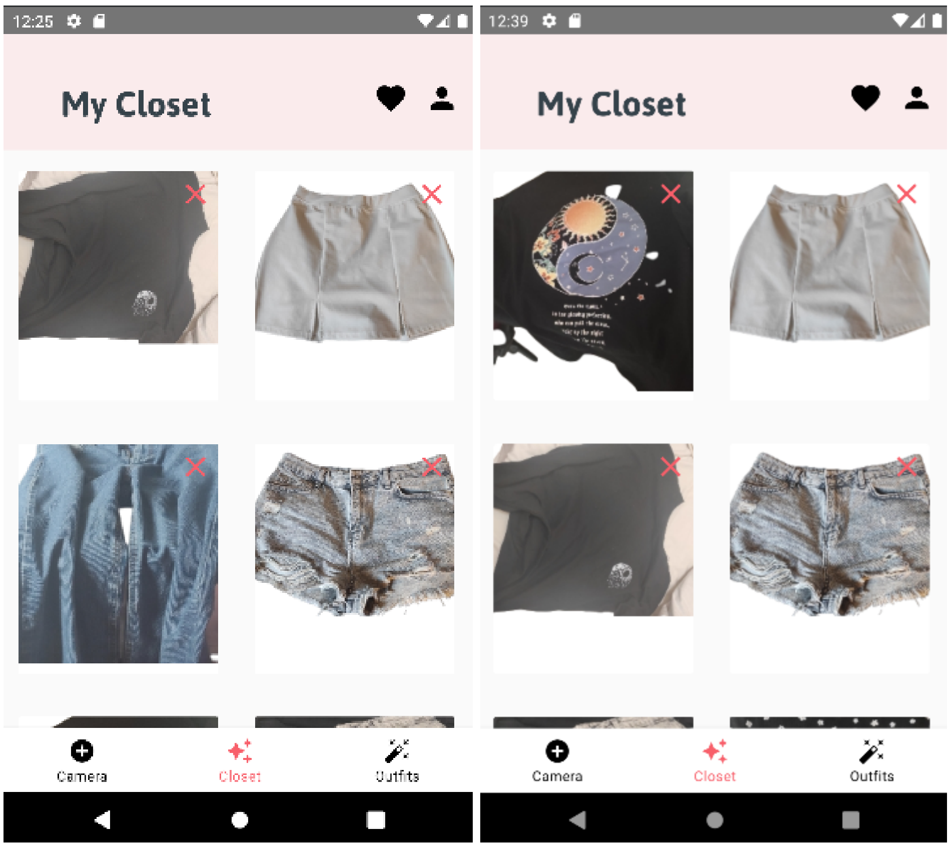
9.1. Sign-up / Sign-in Page





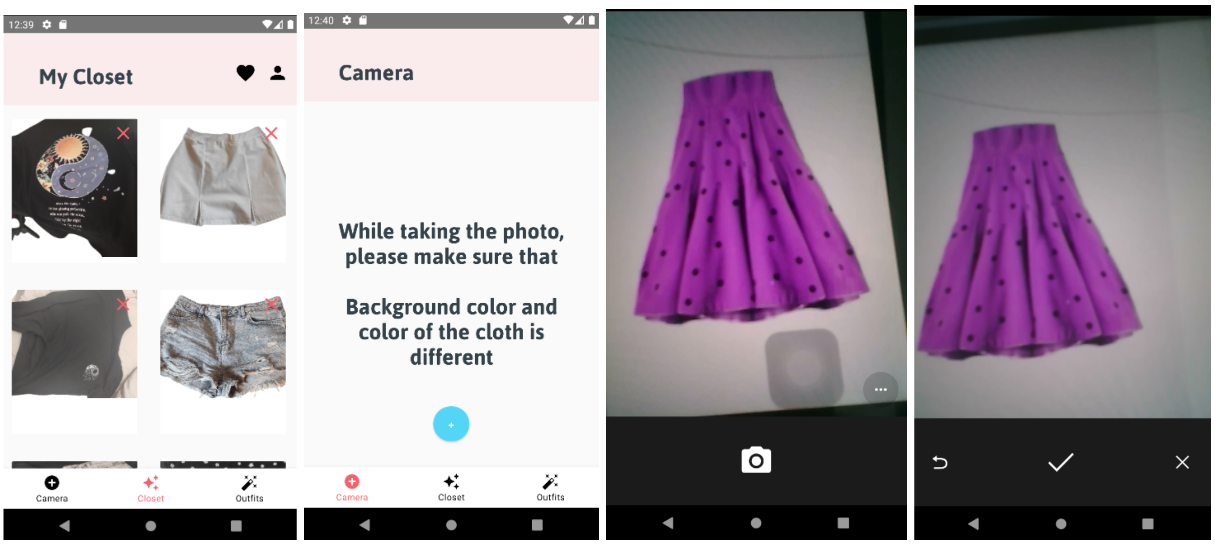
The first page that users see is the sign-in page. The user can enter his/her email and password and click on the Sign In button to start using Rhapso. If the given credentials are correct the user will be directed to the Closet Page (Main Page) which will be explained in Section 9.2. If the user does not have an account, s/he can click on the Sign-Up button to go to the Sign Up page. In the Sign-Up page the user is expected to enter full name, phone, email and password, and also to check the privacy policy. After filling the fields the user can click on the Sign-Up button. After Signing up, the user is directed to the Closet Page.

9.2. Closet Page



In the Closet Page, the user sees the clothing items that s/he has added. The user is also able to delete an item from the closet by clicking on the red button on the top right of each cloth image. In the screenshots above, we have deleted the bottom left cloth. Then, the deleted cloth is removed from the screen as can be seen in the second screenshot.

9.3. Camera Page

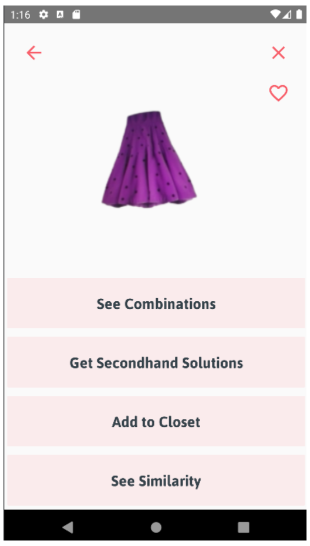


While taking the photo, please make sure that

Background color and color of the cloth is different

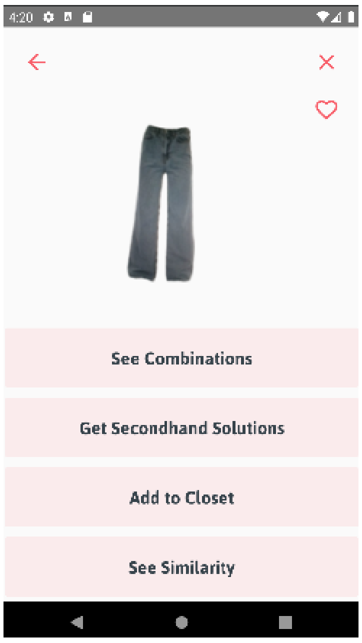
From the Closet Page, the user can go to the Camera Page by clicking on the camera button at the bottom left. After taking the photo of the cloth, the user can approve the image or can retake it. After approving the cloth image, the user will be directed to the Cloth Actions Page which will be explained in Section 9.4.

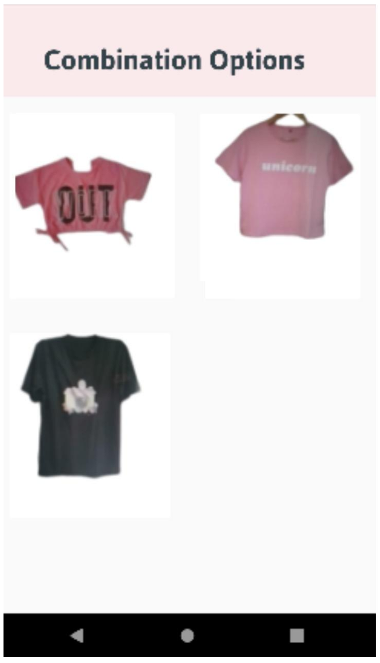
9.4. Cloth Actions Page



In the Cloth Action Page, the segmented version of the image of the cloth is shown. There are four options for this cloth. The first option is "See Combinations", which gives the possible outfits with the given cloth. The second option is "Get Secondhand Solutions", which gives similar clothes from the secondhand cloth website Dolap. The third option is "Add to Closet", which saves the given image to the Closet. Finally, the last option is "See Similarity", which shows the existing similar clothes to the given image with the similarity score. The details of those options will be explained in the subsections below.

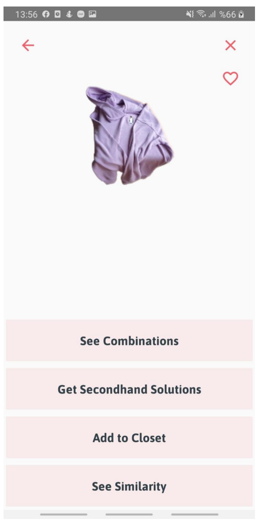
9.4.1. See Combinations

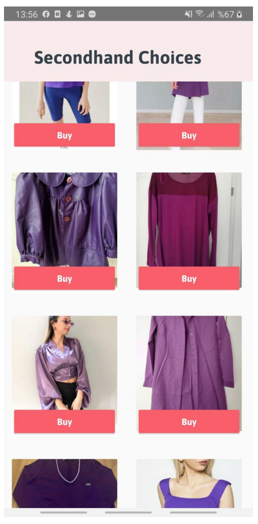


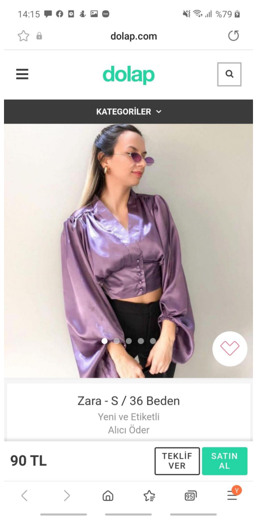


After clicking on the "See Combinations" button, Rhapso will open a new page and list clothes that can be combined with the given cloth.

9.4.2. Get Secondhand Solution

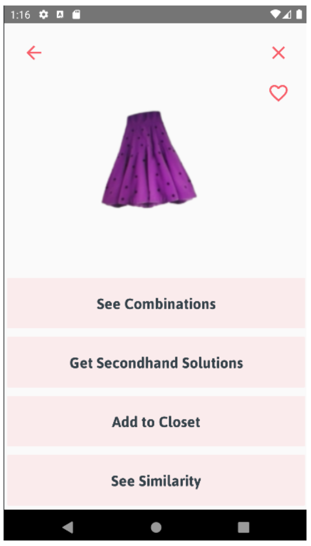


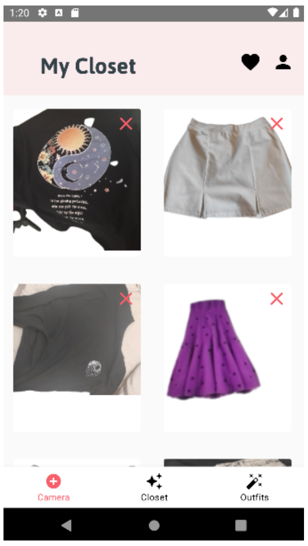




If the user selects “Get Secondhand Solutions”, Rhapso will show clothes that are similar to the given cloth. Secondhand clothes are obtained from a secondhand cloth website called Dolap. The user is able to visit the original website of the cloth to buy it by clicking on the buy button as can be seen in th4e third screenshot.

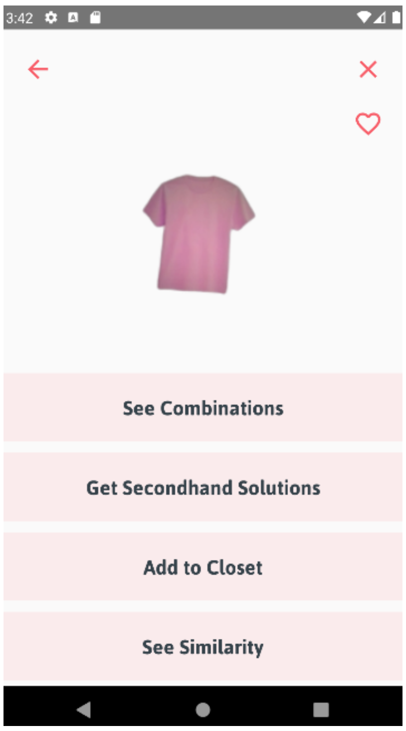
9.4.3. Add To Closet

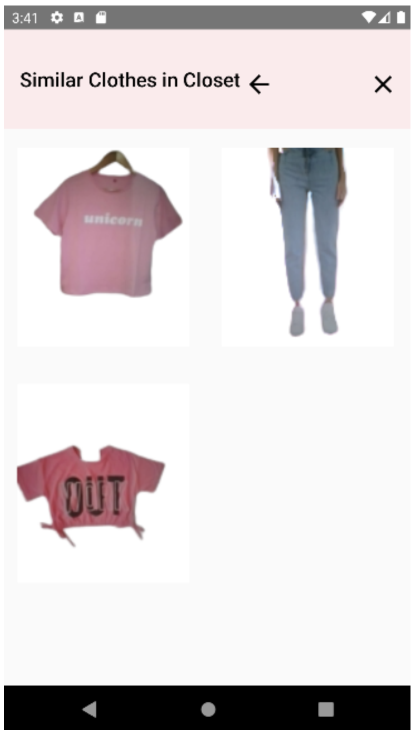




If the user selects “Add to Closet”, the cloth will be added to the Closet and the image of the cloth will be shown on the Closet page as can be seen in the second screenshot on the right.

9.4.4. See Similarity

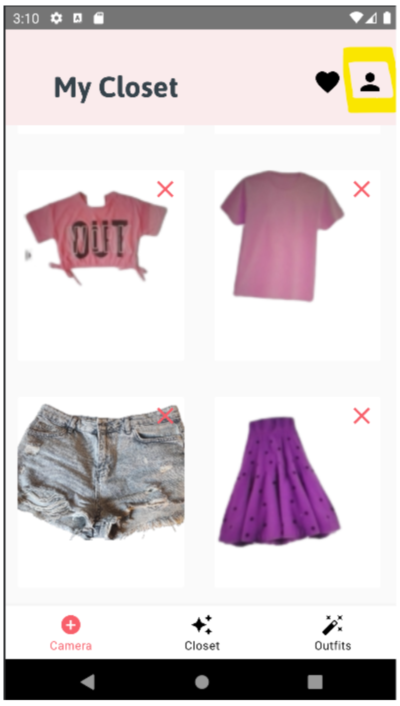


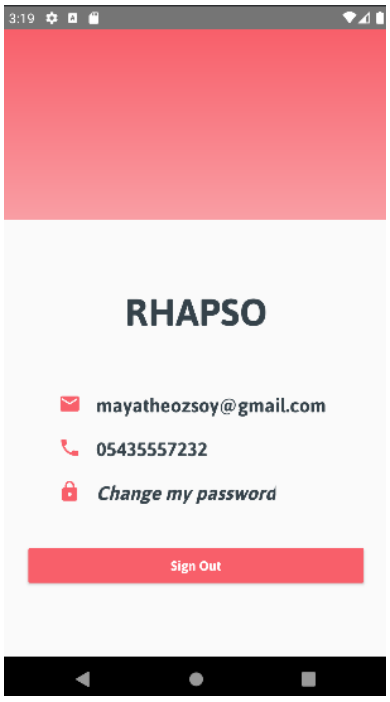


After choosing the “See Similarity” option, Rhapso shows the clothes similar to the given cloth from the closet. As shown in the screenshots, the given cloth is a pink top and the two similar clothes are also pink tops.

9.5. Outfits Page

9.6. Profile Page





mayatheozsoy@gmail.com

05435557232

The Profile Page can be opened by clicking on the “User” icon, which is on the top left in the first screenshot. On the Profile Page, the user can see the personal information and also is able to sign out.

10. Glossary

Google Firebase: The Firebase Realtime Database is a database stored in the cloud. When you use our Apple, Android, and JavaScript SDKs to create cross-platform apps, all of your clients share a single Realtime Database instance and are immediately updated with the most recent data. [13]

API: API stands for Application Programming Interface and is a set of definitions and protocols for creating and integrating application software. [14]

Machine Learning: An algorithm that alters itself over time based on the data that it is exposed to.

Deep Neural Networks: Deep Neural Networks (DNNs) are a class of machine learning algorithms similar to artificial neural networks that aim to mimic the way the brain processes information. [15]

Computer Vision: Subset of artificial intelligence that defines a specialization in deriving meaningful information from visual inputs. [16]

Git: Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

GitHub: GitHub is a provider of Internet hosting for software development and version control using Git.

Dolap: It is a website and application where the users can sell second-hand clothes. Also, it is used within our application to offer options for new clothes.

VCS: Version control, also known as source control, is a method of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time. As the development environment accelerates, version control systems help software teams work faster and smarter. These are especially useful for DevOps teams because they help reduce development time and help deploy successfully. [17]

Development Cycle: The software development life cycle (SDLC) is the process used by the software industry to design, develop, and test high-quality software. SDLC aims to create high-quality software that meets or exceeds customer expectations and is completed within time and budget estimates. [18]

IDE: An integrated development environment (IDE) is software for creating applications that combine common development tools into a single graphical user interface (GUI). [19]

Mobile Application: A mobile application, commonly known as an application, is a type of application software designed to run on a mobile device, such as a smartphone or tablet. Mobile applications are often used to provide users with services similar to those accessible on a PC. [20]

Continous Integration: Continuous Integration (CI) is the practice of automating the integration of code changes from multiple contributors into a single software project. This is core DevOps best practice, allowing developers to regularly merge code changes into a central repository, where builds and tests will then run. Automated tools are used to confirm the correctness of new code prior to integration. [21]

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] “Hosted private apps - managed google play help,” *Google*. [Online]. Available: https://support.google.com/googleplay/work/answer/6145197?hl=en#zippy=%2Creliability-and-service. [Accessed: 25-Oct-2021].

[7] “Deploy apps to enterprises using Google Play : android developers,” *Android Developers*. [Online]. Available: https://developer.android.com/distribute/google-play/work. [Accessed: 25-Oct-2021].

[8] Clarifai, “About Clarifai: Computer Vision & AI platform for Unstructured Data,” *Clarifai*. [Online]. Available: https://www.clarifai.com/company/about. [Accessed: 06-May-2022].

[9] Kaleido, “Remove background from image,” *remove.bg*. [Online]. Available: https://www.remove.bg/. [Accessed: 06-May-2022].

[10] M. McFall-Johnsen, “Is fashion bad for the environment?,” *World Economic Forum*, 31-Jan-2020 [Online]. Available: https://www.weforum.org/agenda/2020/01/fashion-industry-carbon-unsustainable-environment-pollution/. [Accessed: 15-Nov-2021]

[11] “What is GDPR, the EU's new Data Protection Law?,” *GDPR.eu*, 13-Feb-2019. [Online]. Available: https://gdpr.eu/what-is-gdpr/. [Accessed: 06-May-2022].

[12] “Google Play Terms of Service,” *Google play terms of service*. [Online]. Available: https://play.google.com/about/play-terms/index.html. [Accessed: 06-May-2022].

[13] “Firebase realtime database | firebase documentation,” *Google*. [Online]. Available: https://firebase.google.com/docs/database. [Accessed: 06-May-2022].

[14] “What is an API?,” *Red Hat - We make open source technologies for the enterprise*. [Online]. Available: https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces. [Accessed: 06-May-2022].

[15] “Deep Neural Network,” *Deep Neural Network - an overview | ScienceDirect Topics*. [Online]. Available: https://www.sciencedirect.com/topics/engineering/deep-neural-network. [Accessed: 06-May-2022].

[16] By: IBM Cloud Education, “What is computer vision?,” *IBM*. [Online]. Available: https://www.ibm.com/cloud/learn/computer-vision. [Accessed: 29-Oct-2021].
[17] Atlassian, “What is version control: Atlassian Git Tutorial,” *Atlassian*. [Online]. Available: https://www.atlassian.com/git/tutorials/what-is-version-control. [Accessed: 06-May-2022].

[18] *SDLC - Overview*. [Online]. Available: https://www.tutorialspoint.com/sdlc/sdlc_overview.htm. [Accessed: 06-May-2022].

[19] “What is an IDE?,” *Red Hat - We make open source technologies for the enterprise*. [Online]. Available: https://www.redhat.com/en/topics/middleware/what-is-ide. [Accessed: 06-May-2022].

[20] Techopedia, “What is a mobile application? - definition from Techopedia,” *Techopedia.com*, 07-Aug-2020. [Online]. Available: https://www.techopedia.com/definition/2953/mobile-application-mobile-app. [Accessed: 06-May-2022].

[21] Atlassian, “What is continuous integration,” *Atlassian*. [Online]. Available: https://www.atlassian.com/continuous-delivery/continuous-integration. [Accessed: 06-May-2022].