Clothing Virtual Try-On

Jorge Olivares
Robin Martinez
Alan Vela

FINAL REPORT

REVISION - 0.1
23 February 2022

Table of Contents

Execution Plan

Android Application Subsystem	**Cloud Database Subsystem**	**Machine Learning Subsystem**	**Date**
Learning Android Studio	Setting up AWS server host	Studying Python	2/14/22
Create Setup for GUI	Communicate with server Host	Verifying original Model runs - Dependencies	2/21/22
GUI calling for phone operational components	Study SQL	Verifying Original Machine Learning Model runs - Output	2/28/22
GUI receives information / Testing	Create SQL database on MySQL	Translating original model to Python 3 - Dependencies	3/7/22

Create log in services	Populate database	Translating original model to Python 3 - Source Code	3/14/22
Establish host for log in services	Test database information retrieval.	Translation original model to Python 3 - Source Code	3/21/22
Test log in services	Create clothing designs.	Translation original model to Python 3 - Source Code	3/28/22
Connect GUI with neural network	Communicate with a neural network to provide design.	Testing for 3D model output	4/4/22
Test transfer of information	Testing for communication with neural network	Setting up Python-for-Android	4/11/22
Connect GUI with database	Communicate with UI for User Login information	Testing communication between app and model	4/18/22
Test information storage and retrieval with databases.	Testing for communication with UI	Outputting 3D model to application	4/25/22

Validation Plan

Test Name	**Success Criteria**	**Status**	**Engineer Responsible**
Verify original model output	Dependencies must not interfere with one another. Original Python 2 code must output a 3D model.	Untested	Robin M.
GUI information request	Success when receiving video information from an android device.	Untested	Alan V.
Access to storage Host	Able to save information on cloud	Untested	Jorge O.
Log in services	Successful login for the user.	Untested	Alan V.
Database information retrieval	Success if extraction of information is possible.	Untested	Jorge O.
Verifying new	Dependencies must not interfere with one another.	Untested	Robin M.

model output	Python 3 code must output a 3D model.		
Transfer of database information to neural network	Success when the neural network retrieves clothing designs from a database.	Untested	Jorge O. Robin M.
Transfer of GUI information to database	Success when database gives access to user GUI login	Untested	Alan V. Jorge O.
Communication between app and neural network	Successful when GUI is able to show 3D model to user	Untested	Robin M. Alan V.

Change Record

Rev.	Date	Originator	Approvals	Description
0	[02/1/22]	[Jorge Olivares] [Alan Vela] [Robin Martinez]		Draft Release
1	[02/6/22]	[Jorge Olivares] [Alan Vela] [Robin Martinez]		Revision 1
2	[02/22/22]	[Jorge Olivares] [Alan Vela] [Robin Martinez]		Midterm Report
3	[04/29/22]	[Jorge Olivares] [Alan Vela] [Robin Martinez]		Final Report

Clothing Virtual Try-On

Jorge Olivares
Robin Martinez
Alan Vela

CONCEPT OF OPERATIONS

REVISION - Draft
06 February 22

Concept of Operations
for
Clothing Virtual Try-On

Team <38>

Approved by:

_____
Jorge Olivares Date

_____
Prof. Kalafatis Date

_____
Skyelar Head Date

List of Tables

List of Figures

1. Executive Summary

The Internet is a great tool that has provided a great number of innovations. Online shopping is a big part of it; however, it does create an issue. People don't get to have a personal look at the object they are purchasing leading to dissatisfaction with the product when it finally arrives and then returning it. This cycle affects the apparel market heavily because Americans tend to return 10% of their purchases, but in apparel, 35% of products get returned. For companies, this is money being lost and environmentally it creates waste.
This report details the implementation of an application that is accessible to all public and helps reduce the quantity of returns done by customers shopping for clothes. The application is going to take a recording of the user. Once the video is uploaded, our application will create a 3D model of the user. This model is able to change the clothing that it has on, therefore we are able to create an opinion of how an outfit would look on us. Now that we are able to create a criterion of how clothes look on the customer without purchasing, people are going to make shopping decisions more accurately. We believe that this will help reduce the amount of returns.

2. Introduction

In this document we are going to introduce the Virtual Clothing Try-on project. This project is going to be an application that has a wide range of utilities. Its versatility comes from the use of a simple RGB camera that is compatible with augmented reality. The camera is going to take a video recording of a user and turn it into a 3D model. Moreover, we are going to use the model to try-on different sets of clothing. Also, we are creating a database that will contain different clothing designs for all genders. In the very end, it will help with the reduction of returns to retail.

2.1.1. Background

Currently the process to verify that clothes fit is physically trying the clothes on. This process is not very efficient since it has a waiting time that can vary from a pair of days to even a month for clothing to arrive if bought overseas. This costs time for the customer and money to the company when the product is returned. Our main goal is that the application will allow for the customer to have a preview of the clothing on them and allow for a more informed decision when purchasing.

2.1.2. Overview

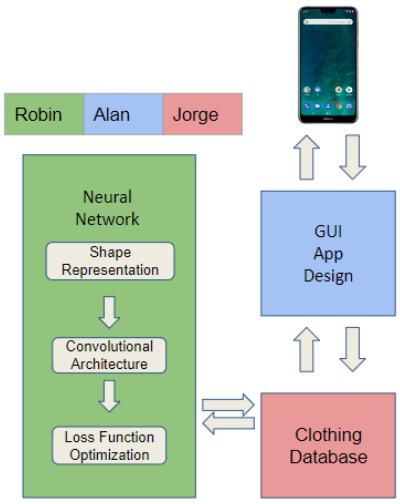


Figure 1 System Overview

In order for the application to work we decided to divide the system into two areas. The first area will be the neural network which is going to take care of the heavy lifting for the application. This neural network will process the video information. Once the information is processed by the system we will have a 3D model that will be stored on the android phone.

The second area of our system is the UI. This will be interacting with both the neural network system and the user. One area that will be interacting directly with the user is the front end, therefore we have to give a comfortable environment for the user. We want to ensure the user's enjoyment when using the application. Another area of the UI is the back end. This area is mainly in charge of the database containing different clothing designs. Nevertheless, it will aid with the proper transfer of information between the neural network and the front end of the application.

2.1.3. Referenced Documents and Standards

- Alldieck,Thiemo;Magnor,Marcus;Bharat Lal Bhatnagar;Theobalt,Christian;Pons-Moll,Gerard "Video Based Reconstruction of 3D People Models"
- https://www.mckinsey.com/industries/retail/our-insights/returning-to-order-improving-returns-management-for-apparel-companies

3. Operating Concept

3.1.1. Scope

The Clothing Virtual Try-on application proposed in this document was designed to allow customers to freely choose clothing on a 3D model of themselves and get an opinion of how the clothes will look on them before purchasing. The customer will be able to choose between a selection of clothing having different designs, sizing, and colors of which they can see how it looks on their 3D model. This will help customers reduce the amount of returns when purchasing clothing.

3.1.2. Operational Description and Constraints:

Clothing Virtual Try-on application is intended to be used for people who shop online and companies that want to advertise their clothing. Several selections of clothing will be displayed for the user to choose from and will be projected on their 3D model. The user will be able to select clothing they are satisfied with and purchase it on the application.

The application must meet all the following criteria:

- Must be an eCommerce Application
- The user must have an android AR compatible phone
- Avatar Display
- Must include shirts and trousers to select
- User must use backend camera for video
- User must use clothing that is fitting
- User privacy. The software side of the program must be able to handle 3D Model data without transmitting personal information
- The video needs to be at most a 24 fps, 30 second video that is colored

3.1.3. System Description

Our Clothing Virtual Try-on clothing application will consist of three main subsystems: the front end, backend, and neural network.

Front End: This is the main subsystem that will include the graphic user interface (GUI). This is the aesthetic area of the application of the user will see. This will allow the user to interact with the application on their phone. The GUI enables display customization such as viewing objects that the user can interact with. This will include buttons, navigation screens, alerts, etc.

Back end: The back end of the application will be interacting with both the main UI to receive the information from the video recording and the neural network to process the information. However, the most important part is going to be establishing the database with the catalog of different clothing the model can change and supplig the user's log in information and avatar.

Neural Network: Using a monocular RGB video of a person moving, a 3D model of them would be created. The neural network normalizes the videos data to 24 fps and a certain resolution. The model then does semantic segmentation in each frame to isolate the person into parts such as arms, legs, torso, etc. From there a skeleton is generated using preexisting work. The neural network model then generates textures onto the skeletal model. Finally, clothing textures taken from the clothing database are applied onto the 3D model.

3.1.4. Modes of Operations

The Clothing Virtual Try-on application will have a primary mode of operation when the user opens the application on the phone. In this mode, the user will use the backside camera of their android phone and upload a video to the application. The collected data from the user will be used to construct a 3D model of themselves via the Virtual Clothing Try-On neural network. With this model and the selection of clothing stored in our database, the user is presented with a model of themselves having a variety of different clothing to choose from.

The application will also have an error notification system where it would output a message to the user in the event of a problem that occurs within the database, internet speed, or simply the application itself. This will allow for the application to not all crash and let the user know when something is wrong to try again later.

3.1.5. Users

Our Clothing Virtual Try-on application will be marketed to people of all ages, but that mostly shop online. This will allow customers to be more comfortable shopping for clothing online as well as mitigate the need to return clothes after unsatisfactory purchases.

Our application can be also marketed to clothing companies that would want to implement their clothing using the application. This will create a great way of advertising their clothing and getting more purchases without the loss of money from returns.

3.1.6. Support

Support for the Clothing Virtual Try-on application will be provided on the application itself. This will include instructions on how to use the application, a video example of how to video record yourself for the 3D model using the back camera, and tech support to be able to contact.

4. Scenario(s)

4.1.1. Clothing Retail Industry

The Clothing Virtual Try-On app will find its main use in the clothing retail industry scene. The app would allow companies such as Amazon to advertise clothes from their catalog to customers. This would eliminate the loss produced by customers returning clothing to their retailers.

Manufacturers of clothes and established brands can also use the app to advertise upcoming clothing lines.

4.1.2. Virtual Reality

Using augmented reality from an application can help companies use virtual reality by using the 3D models created by the neural network. Companies such as Metaverse could use the data from the neural network to personalize models for individual users. This would allow more flexibility for the users to create an avatar that resembles themselves.

5. Analysis

5.1. Summary of Proposed Improvements

The Clothing Virtual Try-On app will provide improvements such as

- Easier access to clothing catalogs of retail stores
- Fewer returns on unsatisfactory sales
- Selling upcoming lines, allowing for pre-ordering on clothes

5.2. Disadvantages and Limitations

The Clothing Virtual Try-On app will have limitations that include:

- Only works on android phones with ARCore compatibility
- 3rd party GPU sources must be used to train the model
- The video needs to be at least a 24 fps, 30-second video that is colored

5.3. Alternatives

Some alternatives to the Virtual Clothes Try-On app are:

- Conventional shopping and trying on clothes in-store
- Stores with lenient return policies that reduce loss from returns

5.4. Impact

If the app were to be successful this would have a major impact on the clothing retail scene. There have not really been successful applications for the construction of personalized 3d models and applying textures onto said models. Clothing chains would be able to retain a bigger audience for their product as they would be able to easily advertise their merchandise.

There would be privacy and security concerns in regards to the manner in which these personalized models are stored. Questions would be raised on the rights to these models.

Virtual Clothes Try-on

Jorge Olivares
Robin Martinez
Alan Vela

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – 0.2
19 February 2022

Functional System Requirements
FOR
Virtual Clothes Try-on

Prepared by:

_____
Author Date

Approved by:

_____
Jorge Olivares Date

_____
Prof. Kalafatis Date

_____
Skyelar Head Date

List of Tables

List of Figures

6. Introduction

6.1. Purpose and Scope

The Virtual Clothing Try-on application is a more effective way of shopping online. It will provide a visual of how different clothes will fit. The document will help to identify the requirements for the Virtual Clothes Try-on project. An overview of the system and how it is divided is given in figure 1.

6.2. Responsibility and Change Authority

The team leader is going to be responsible for the approval of any changes done to the project and that the desired requirements are met in the provided deadlines. These changes need to be accepted by the sponsor before actual implementation. Subsystems responsibilities are shown in the following table.

Subsystem	**Responsability**
3D People Model application	Robin Martinez
GUI Design	Alan Vela
Cloud Database	Jorge Olivares

Table 1 - Assigned Responsibilities

7. Applicable and Reference Documents

7.1. Applicable Documents

The following documents, of the exact issue and revision shown, form a part of this specification to the extent specified herein:

Document Name	Revision/Release Date	Publisher
Learning to Reconstruct People in Clothing from a Single RGB Camera	2019	Thiemo Alldieck
Firebase Documentation	2022-11-15	Google Developers

Table 2 - Applicable Document

7.2. Reference Documents

The following documents are reference documents utilized in the development of this specification. These documents do not form a part of this specification and are not controlled by their reference herein.

Document Name	Revision/ Release Date	Publisher
SMPL	2015	Max Planck Institute for Intelligent Systems
Instance-level Human Parsing via Part Grouping Network	2018	EECV
OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields	2019	arXIV
Open GLView	2022/2017	Expo
Expo Three	2022/2018	Keith-Kurak

React-Navagation	2022	Satya164
ImagePicker	2022/2018	Expo
React-Redux	2022	Markerikson

7.3. Order of Precedence

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions.

All specifications, standards, exhibits, drawings or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

8. Requirements

This section is going to identify the minimum requirements needed for the application to work. These constraints were decided in an attempt to create a comfortable environment for the user and good reliability.

8.1. System Definition

The virtual clothes try-on system will be split into three different subsystems: Neural Network, UI application, and Cloud Database. The neural network is going to be the main focus of the application since it is going to create the 3D model of the user with a video recording. Additionally, it will allow the user to change the textures of the model allowing different designs on the avatar. The UI application will interact directly with the user of the application. This subsystem has to ensure user satisfaction and be easy to use. Lastly, we have the cloud database. The database is going to store different designs of clothing the user will be able to change the 3D model.

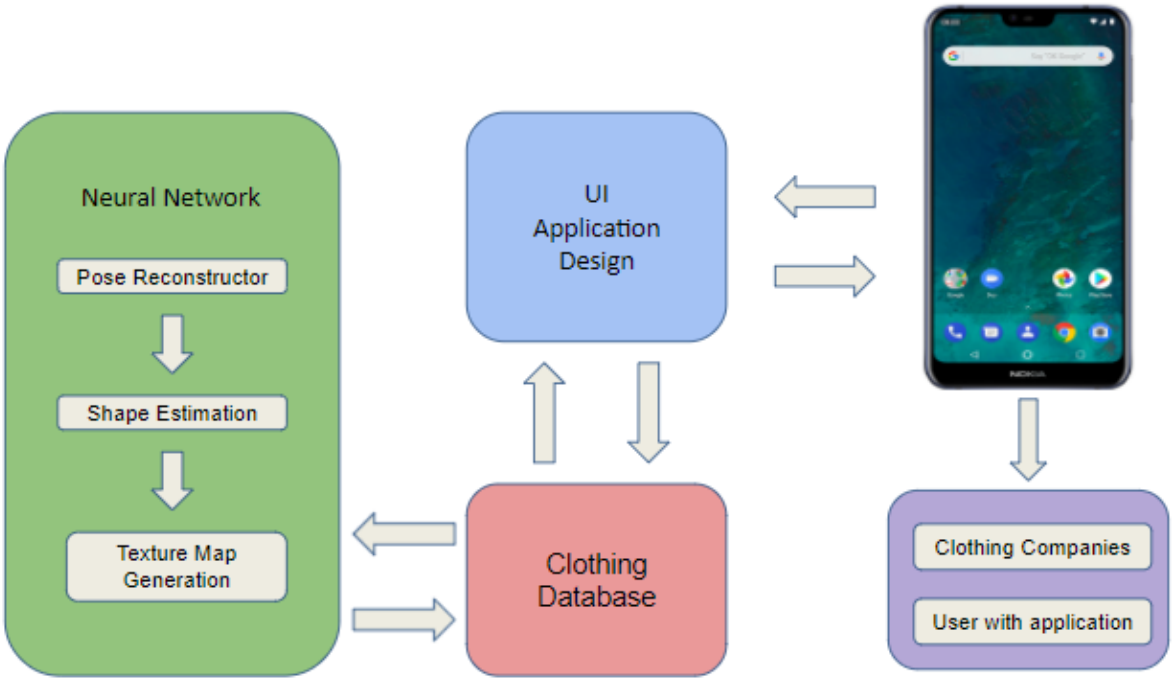


Figure 2 - System overview flowchart

8.2. Characteristics

8.2.1. Functional / Performance Requirements

8.2.1.1. Video

The video rendered by the neural network has to be at least 24 frames per second, at most 30-second video, and color.

Rationale: This is a core system performance requirement that the software requires to render the video.

8.2.1.2. Analysis Time

The total time needed to evaluate the video to render a 3D Model should not exceed 45 seconds.

Rationale: The time needed to render a 3D model should not take a long time for the user to see a result.

8.2.1.3. Database Size Requirements

The database should be able to hold 50 to 100 different textures of shirts and trousers

Rationale: Specified by sponsor

8.2.2. Physical Characteristics

8.2.2.1. Android Phone

The user has to have an android phone that can use OS 7.0 Nougat that is compatible with Augmented Reality applications. A back camera is required.

Rationale: This is a requirement specified by the sponsor.

8.2.3. Software Requirements

8.2.3.1. Use of SMPL

The detection and characterization algorithms will use the SMPL model to track the pose of the User

Rationale: A learned model of human body shape and pose-dependent shape variation that is compatible with the Python coding language

8.2.3.2. Programming Language - App Subsystem

The Android application will be coded using Javascript

Rationale: Due to ease of implementation when one is using React Native

8.2.3.3. Programming Language - Machine Learning Subsystem

The GitHub repository on the “Learning to Reconstruct People in Clothing from a Single RGB Camera” will be translated from Python 2.6 to python 3.6. The scope of the this subsystem later changed to just get it running on Python 2.6.

Rationale: Specified by the team sponsor due to the constraints of the report this project is building off of

8.2.3.4. Programming Language - Database Subsystem

The database for User accounts and clothe textures will be written in python and Node.Js.

Rationale: The Python language is a cross-platform language that is easy to pick up for novice programmers.

8.2.3.5. Database

The database will be constructed using Firebase

Rationale: Firebase is an open source relational database management system.

8.2.3.6. Android Application

The application will be designed using React Native

Rationale: An integrated development environment for Google's Android operating system that is very customizable for ease of use.

8.2.4. Communications Requirements

8.2.4.1. Database to Application

The database is going to use Firebase to have the proper organization of queries. Queries will be done in python and Node.Js to retrieve or store data from the Firebase database.

8.2.4.2. Failure Propagation

Any exceptions are caught within the script of code and will throw errors to be read by the user if one or all the subsystems is down. All data will remain safe in the database and a system restart should fix the problem. System failures caught in exceptions will be analyzed for future resolutions.

Appendix A: Acronyms and Abbreviations

GUI	Graphical User Interface
JSON	JavaScrip Object Notation
RGB	Red Green Blue
SMPL	Skinned Multi-Person Linear Model
3D	3-dimensional
IP	Internet Protocol

Appendix B: Definition of Terms

Neural Network - A series of algorithms that attempt to understand relationships between data sets that they receive

Database - An organized collection of data stored on a server

Monocular - With, for or in one eye

Augmented Reality - Interactive experience of a real world environment replicated by a computer

Internet Protocol - Method by which data is sent from one computer to another

Virtual Clothes Try-on

Jorge Olivares
Robin Martinez
Alan Vela

INTERFACE CONTROL DOCUMENT

REVISION – 0.2
19 February 2022

Interface Control Document
for
Virtual Clothes Try-on

Prepared by:

_____
Author Date

Approved by:

_____
Jorge Olivares Date

_____
Prof. Kalafatis Date

_____
Skyelar Head Date

List of Tables

List of Figures

9. Overview

This document describes the interfaces between all the different subsystems of Virtual Clothing Try-On. The system is designed to be centered on three different subsystems: the android application, database, and neural network. A system diagram of how the different subsystems interact is shown in the figure 3.

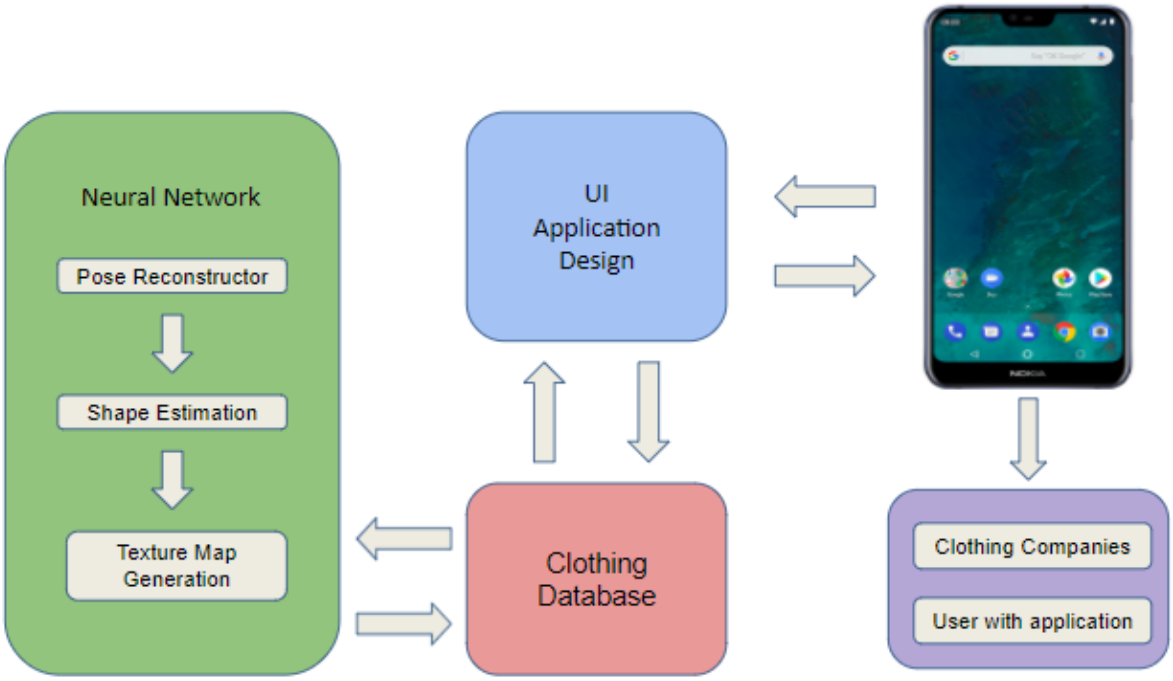


Figure 3 - System overview flowchart

10. References and Definitions

10.1. References

Document Name	Revision/ Release Date	Publisher
SMPL	2015	Max Planck Institute for Intelligent Systems
Instance-level Human Parsing via Part Grouping Network	2018	EECV
OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields	2019	arXIV
Open GLView	2022/2017	Expo
Expo Three	2022/2018	Keith-Kurak
React-Navagation	2022	Satya164
ImagePicker	2022/2018	Expo
React-Redux	2022	Markerikson

Table 4 - Reference Documents

10.2. Definitions

GUI	Graphical User Interface
JSON	JavaScrip Object Notation
RGB	Red Green Blue
SMPL	Skinned Multi-Person Linear Model
3D	3-dimensional
IP	Internet Protocol

11. Physical Interface

11.1. Model

An Android Phone that can use OS 7.0 Nougat and that is also compatible with Augmented Reality applications is needed to provide a video able to create the 3D model.

11.2. Camera

The phone has to use a back camera that can capture video in RGB. The camera needs to process videos that are at least 24 frames per second and colored.

12. Database Interface

The subsystems will interact with Firebase database through an IP established in the host services. Data requests will be managed with python and Node.Js to ensure flexibility between the subsystems. Each of the systems will be able to communicate with the database independently. Database will be able to hold data string information for users to log in and data files upload that will be provided by the users. Moreover, storage of database can be scalable for distribution.

12.1. Django RESTframework

Django is an API that will help with the communication between the Machine Learning system, the GUI and the host database in use. This is a python language software and it contains many open source libraries that help with the production of applications storing information in database.

13. Graphic User Interface

The user will interact with the GUI on their android phone that was created on React Native using JavaScript. Users will be able to login and have a profile with information they will be able to edit on. There will also be a shopping cart that the user will be able to add or remove shopping items they want to purchase. A section to upload a video of themselves for the avatar to be displayed. This allows the user to freely and conveniently use the application.

14. Machine Learning Model to Server Communication

Communication between the Octopus machine learning model will be done through the Database Interface and its Firebase server. Firebase hosting and Cloud run will be used to host a flask server. Firebase hosting is a framework that support python scripts. Cloud run is a way to serverlessly run containers.

Virtual Clothes Try-on

Jorge Olivares
Robin Martinez
Alan Vela

SUBSYSTEM REPORT

REVISION – 0.3
29 April 2022

Subsystem Report
for
Virtual Clothes Try-on

Prepared by:

_____
Author Date

Approved by:

_____
Jorge Olivares Date

_____
Prof. Kalafatis Date

_____
Skyelar Head Date

Virtual Clothes Try-on

Jorge Olivares
Robin Martinez
Alan Vela

GUI Subsystem Report

REVISION – 0.3
29 April 2022

Subsystem Report
FOR
Virtual Clothes Try-on

Prepared by:

_____
Author Date

Approved by:

_____
Alan Vela Date

_____
Prof. Kalafatis Date

_____
Skyelar Head Date

List of Figures

List of Tables

15. GUI Subsystem Introduction

The GUI is the main part of the subsystem where the user interacts with. This is what the user will be seeing and is considered the front end subsystem.

15.1. React Native

The reason for choosing react native is because are faster to build, easier to use, offer more resources to use for open source. It also allows the user to use tools for development, debugging, testing, and performance that make it faster and easier to develop apps. At first android studio was used, but switch to react native for a better application.

16. General GUI Description

The GUI Phone Application is separated by different screens (navagation which the app draws its UI) where the user is able to see and interact with.

16.1. Login Screen

The login activity is a display to the user where the user is allowed to login & register. This will give access to the user with the correct credentials to be able to enter the main display of the application. If the user is not able to correctly input information it will prompt the user to input that information or it won't register. This can be said if the user inputs the same email it will prompt the user to input another email. Figure 1 will show the login screen. Figure 2 will display the register screen.

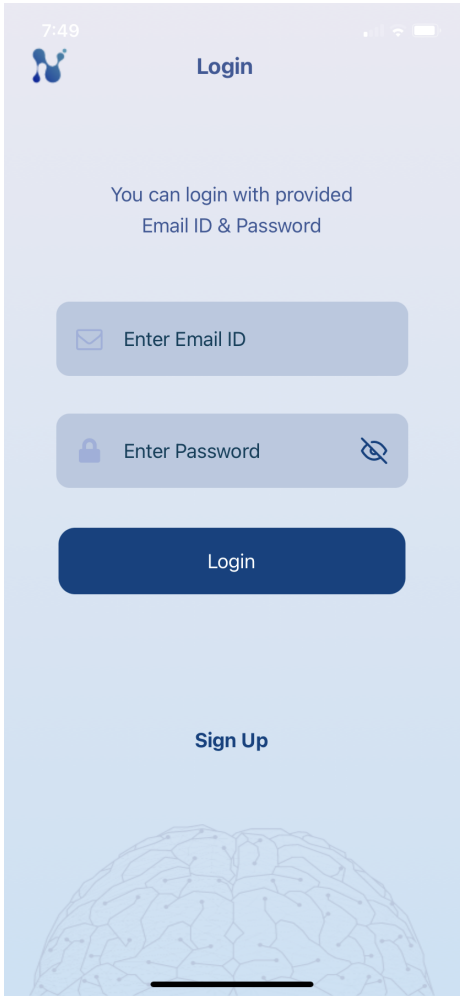


Figure 1 - Login Screen

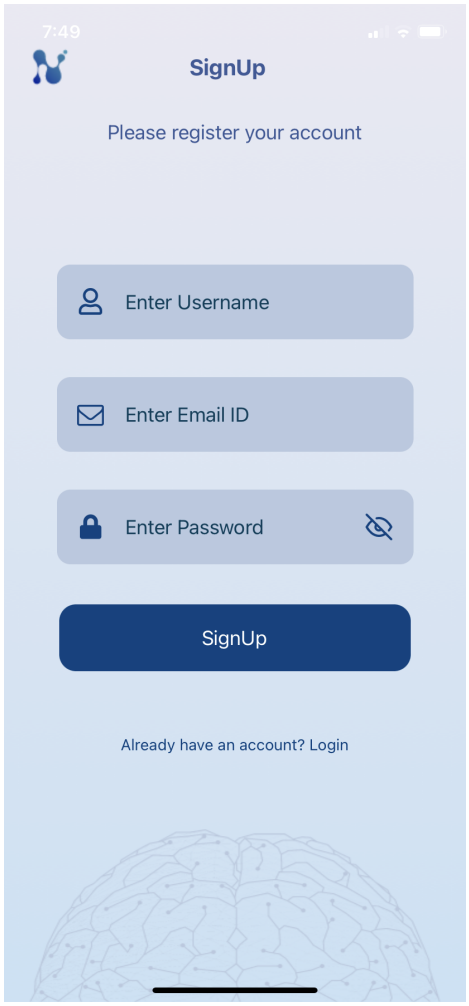


Figure 2 - Register Screen

16.2. Home Page Screen

When logging in, the user is prompted to the home page screen. On the home page screen, the user is able to select the options to choose from. The options are as follows: Store Products, Upload Video, and Shopping Cart. These options allow the user to freely navigate through the application. Figure 3 shows the home page screen.

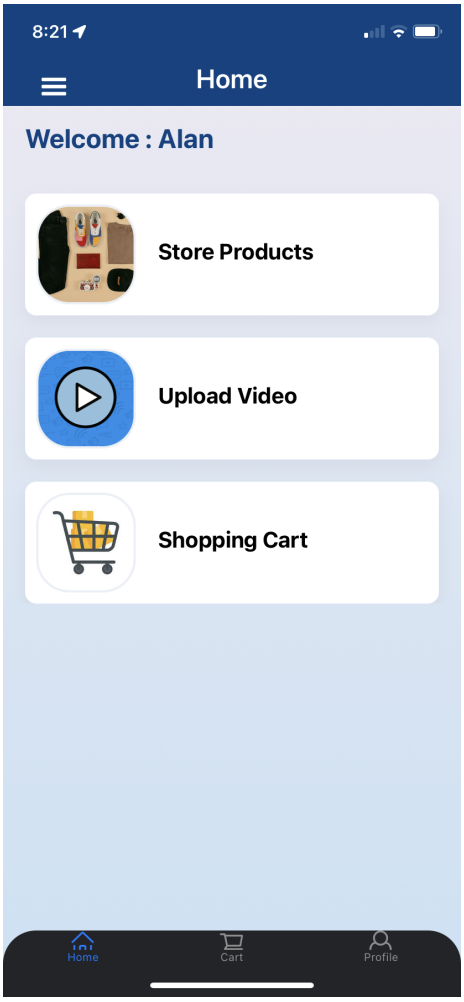


Figure 3 - Home Page Screen

16.3. Profile Information Screen

When the user selects the profile tab at the bottom of the application, it will prompt them to the profile information screen. Here the user can update their profile information that includes: username, email, and profile picture. Figure 4 displays the profile information screen.

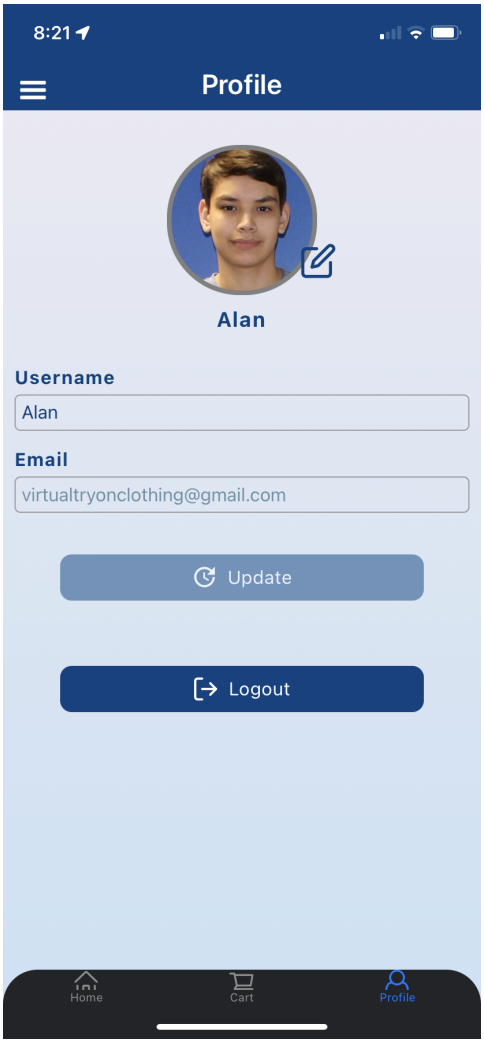


Figure 4 - Profile Information Screen

16.4. Menu Tab Screen

The menu tab screen allows the user to navigate through the application quickly. This includes: home, cart, profile, contact, and logout. When selecting any of the listed, it will navigate them to those designated areas. It will also display the user their profile information on the top to let the user know it's their account logged in. The contact button will alert the user of an email to ask any inquiries about the application or help. The logout button will prompt the user back to the login screen. Figure 5 displays the menu tab screen.

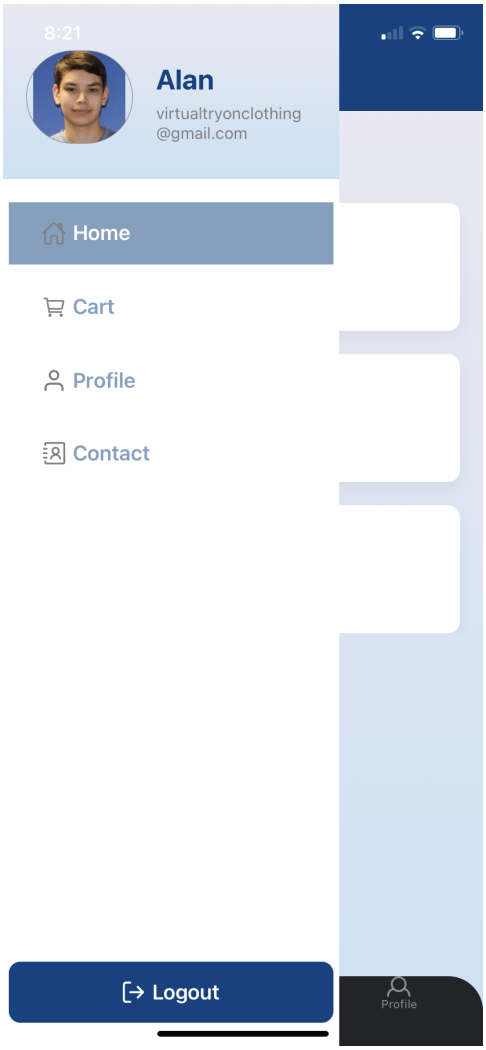


Figure 5 - Menu Tab Screen

16.5. Store Products Screen

The user will be able to scroll, browse, and select different types of shirts and trousers for both men and women in the store products screen. When selected on an item(s), the user will be able to read information about the product and select different sizes ranging from: small, medium, large, and x-large. Once the user selects the appropriate size, they will be able to add & remove the item to the shopping cart. An alert will be displayed for the user. Figure 6 displays the store products and figure 7 displays the details screen of the item(s) selected.

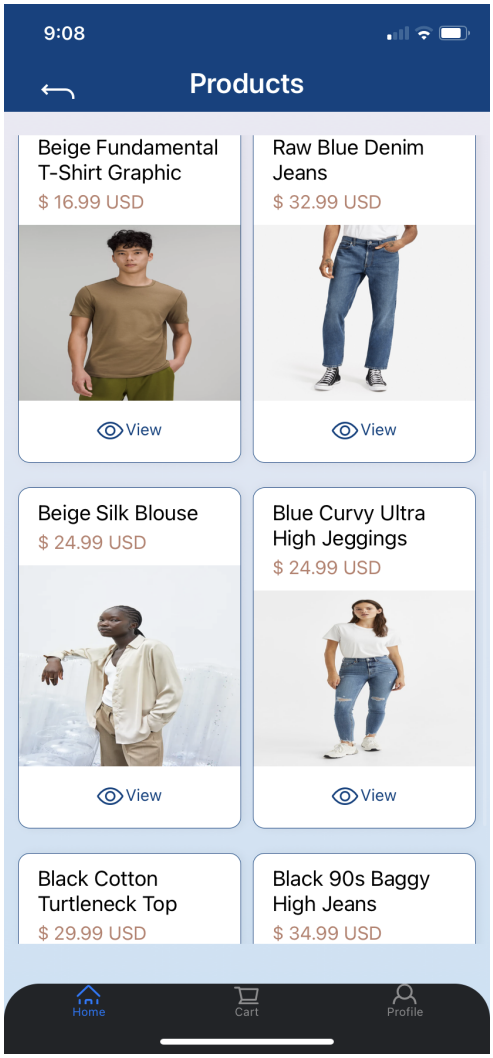
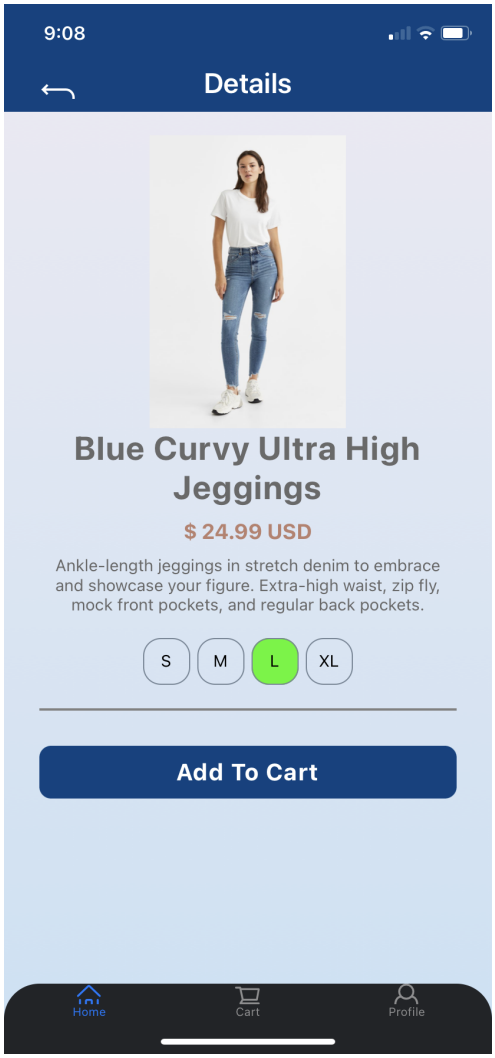


Figure 6 - Store Products



Screen Figure 7 - Details Screen

16.6. Upload Video Screen

The user will be able to upload a video of themselves. An alert will be displayed to the user with instructions of what type of video to upload. Once the user has read the instructions and acknowledged the instructions, it will automatically open the video gallery on their phone. After selecting a video, a load screen will appear to show the video is rendering. This video information will be used to create a 3D avatar of the user. Figure 8 shows the upload video screen.

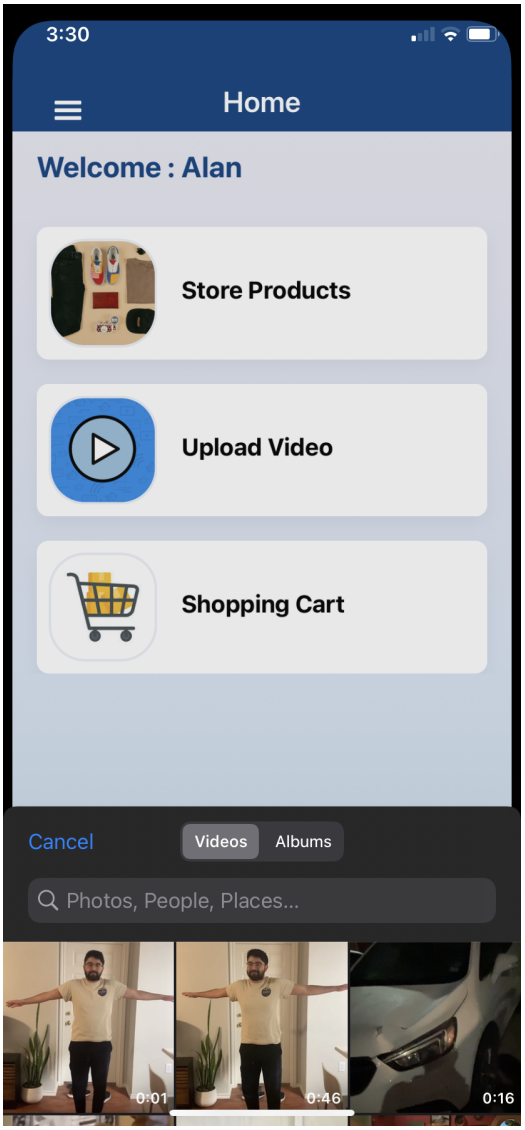


Figure 8 - Upload Video Screen

16.7. Shopping Cart Screen

The shopping cart activity will show the user if any products they selected will be saved to be able to purchase. They will be able to increase or decrease the number of the same product selected. The user will also be able to remove the item(s) from the shopping cart. A checkout button will be used by the user and prompted an alert of the order successfully placed. Figure 9 shows the shopping cart screen.

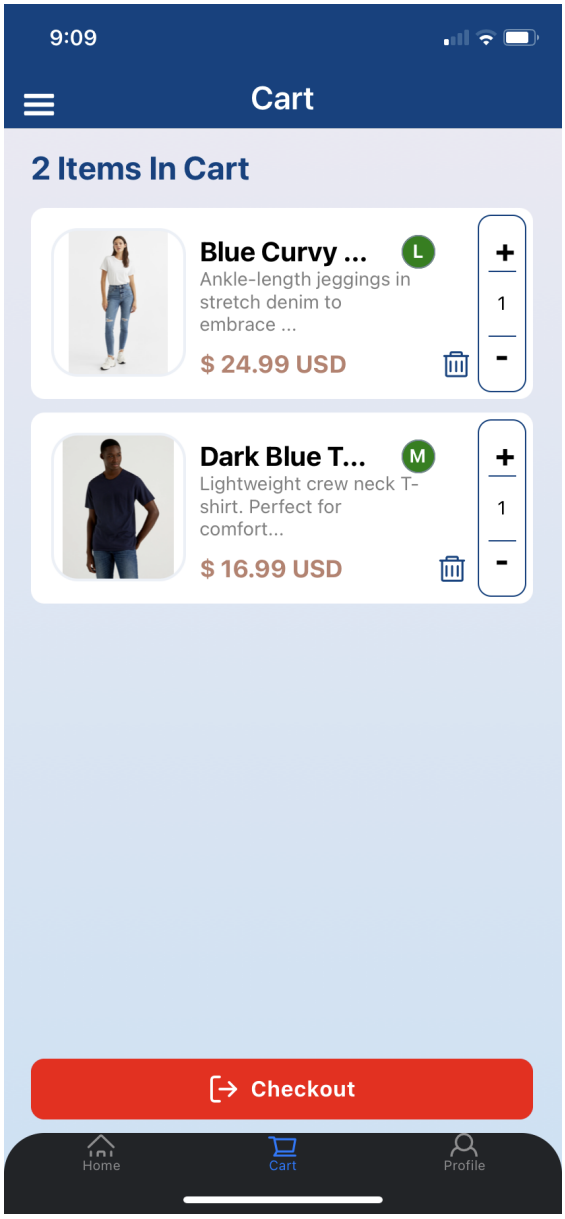


Figure 9 - Shopping Cart Screen

17. DataBase Interaction

The GUI will be interacting with firebase to store and receive information.

17.1. Firebase

Firebase is a realtime database that allows secure access to the database directly. Data is stored locally so even when offline the data will still be on fire giving the user a

responsive experience. Firebase was the easiest and most known to use with react native for mobile applications.

18. Validation

18.1. Log in services

Validation was performed for the login services by requesting data from the user, storing that data into firebase and accepting that data to login on the application. This was successful and any user is able to register and login.

18.2. Video Upload

Validation was performed for the video upload by requesting a video from the user from the gallery of the phone. Firebase once again was used to store the video in storage and able to display the user with an alert to show that it was uploaded successfully.

19. Subsystem Conclusion

In conclusion, the GUI subsystem was successful and completed in time as expected.

19.1. Further Development

19.1.1. Integrating with Other SubSystems

For most of the GUI, to be able to test the application firebase database was used to able to test and validate if anything worked. In ECEN 404, the GUI subsystem will have to change to Jango and use machine learning to create the 3D avatar.

19.1.2. Creating 3D Avatar

The GUI subsystem will have to be able to read an obj file that the octopus outputs and display a 3D avatar to the user. This will be done in 404 working with the machine learning subsystem.

19.1.3. Creating Clothing Product

The main part of the subsystem will be displaying clothing product to the user in the application to shop. A lot of the clothing product is already in process and will be soon completed in 404.

Virtual Clothes Try-on

Jorge Olivares
Robin Martinez
Alan Vela

Database Subsystem Report

REVISION – 0.3
29 April 2022

SUBSYSTEM REPORT
FOR
Virtual Clothes Try-on

PREPARED BY:

_____
Author Date

APPROVED BY:

_____
Jorge Olivares Date

_____
Prof. Kalafatis Date

_____
Fardeen Mozumder Date

List of Figures

List of Tables

20. Firebase

Firebase is a cloud hosted, NoSQL database that doesn't need a relation when storing data on it. It can assist with creative and non-conventional data storage. Furthermore, even though it is not a strict non-relational database, it is still very good at managing high volumes of data at very good speeds. This is the database selected for the project since it can manage information storage files of any kind and it has tools that can allow for files of up to 256Mb per upload. The scalability for the deployment of a project is very easy because the cloud offers different storage options at their respective rates.

20.1. Realtime Database

Realtime Database is one of the tools offered by Firebase. It helps with the synchronization of data between every connected client. Data is stored in JSON for quicker and easier access. It can also provide feedback regarding the speed of the processes of the queries and alert that a more efficient way is possible. This allows for a better interaction between the application and users.

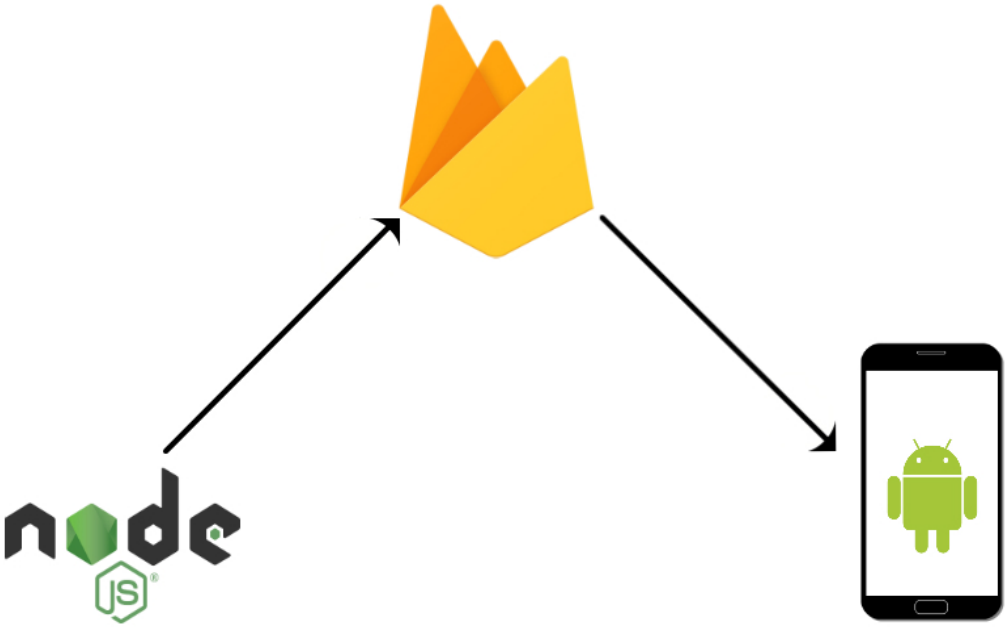


Figure 1: Firebase and Node.Js process of communication

21. Schema for Database

In Firebase the database is set up by a set of collections. Inside each collection, you can decide what kind of data is stored for later use. These collections have a schema layout that simplifies the organization process of information. The use of Firebase helps to store data files within the database.

21.1. Users

The user collection contains the information that is going to be provided by the user. The information is going to be the username and password. Only one 3D model avatar is going to link with the user itself.

Tag	Description
User ID	Unique ID for user information storage
Username	Username provided by user
Email	Email provided by user
Image Url	User's profile picture

Table 1: User information stored in the database

21.2. Storage

The collections for clothing are going to contain the information needed for the texture to be retrieved from the database and the name provided whenever the file is uploaded to storage. Also, the video provided for the ML algorithm is going to be stored individually for each user.

Tag	Description
Clothing ID	Unique ID for clothing storage
Name	Name provided for a piece of cloth
Image	Path for the image in the database
Price	Price listed for the item

Table 2: Clothing information stored in the database

Tag	Description
Video ID	Unique ID for video storage
Name	Name provided for the video

Table 3: Video information stored in the database

22. Validation

To validate a series of queries that will upload and retrieve the information to and from the database was use. Data needs to be displayed in the database; nevertheless, reading the data from the database is an essential part of the subsystem. Queries displaying the proper data and files requested from the database. Once the file was downloaded and opened on a different computer to verify that it worked and wasn't corrupted.

23. Subsystem Conclusion

The subsystem was completed on plan and met the requirements set in the validation process.

23.1. Further Development

23.1.1. Integration with other subsystems

The settings for fully automated queries match the inputs provided by the GUI. The GUI is also going to be able to request the information and scale it as needed for display. This will take care of the inputs needed for the GUI. Moreover, the machine learning subsystem will need to communicate with the database to store the model avatar created, for the user to have access to whenever this information is needed.

23.1.2. Horizontal Scaling

The number of documents provided for the development of the application are subject to change whenever the application is ready for deployment. However, Firebase provides tools to facilitate the upgrading of storage processes. The use of JSON in order to store documents and the use of nodes for data storing facilitates the upscale of storage.

Virtual Clothes Try-on

Jorge Olivares
Robin Martinez
Alan Vela

MACHINE LEARNING SUBSYSTEM REPORT

REVISION – 0.3
29 April 2022

Subsystem Report
for
Virtual Clothes Try-on

Prepared by:

_____
Author Date

Approved by:

_____
Jorge Olivares Date

_____
Prof. Kalafatis Date

_____
Skyelar Head Date

List of Figures

List of Tables

N/A

24. Octopus

Octopus is convolutional neural network model that predicts human shape. With just a few frames taken from a video in which the subject slowly turns, it is able to create a model within 4 to 5mm accuracy. Using semantically segmented images and a skeletal model, Octopus produces a clothed model based off SMPL, a learned model of human body shape and pose-dependent shape variation.

Octopus is written around depreciated libraries and APIs from 2019. Therefore, dependencies were either updated or replaced to eventually achieve a working model.

24.1. SMPL

Skinned Multi-Person Linear Model is a skinned vertex-based model Octopus is built off of. It can produce various body shapes in natural human poses as it's parameters can be modified. The model's parameters are learned from rest post templates, blend weights, pose-dependent blend shapes, identity-dependent bled shapes and a regressor from vertices to joint locations. The model itself can be download from website along with its tools required to run and added into the container with the machine learning model.

24.2. Openpose

Is a program that can detect human body, hand, facial and foot keypoints to construct a 3D skeletal model. It receives a .mp4 video and maps out 25-keypoints on the body for every frame in the video. These output frames are one of the inputs that are going into the Octopus model.

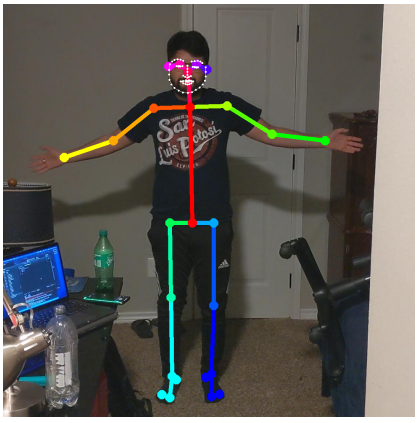


Figure 1: OpenPose Skeletal frame

24.3. PGN Semantic Segmentation

PGN is a deep learning method for semantic segmentation, instance-aware edge detection and instance-leve human parsing built using tensorflow. These output frames are one of the inputs that are going into the Octopus model.

25. Machine Learning Project Flow

The 3D model creation process starts at preparing the video to be fed into OpenPose and PGN Semantic Segmentation. This is done so in a Data Proceessing Pipeline where a given .mp4 video is split and cropped into frames. These frames are then fed into OpenPose and PGN semantic segmentation while prove the inputs into the Octopus Model. These inputs consist of datapoints outlining a skeletal frame overlaid on the human bodyp as well as a semantically segmented photo. This data will then be fed into the Octopus machine learning model that will use the SMPL model to predict pose and shape of a person to create a human model. Finally, Octopus will overlay clothing on this human model.

25.1. Data Processing Pipeline

Before an image is fed to OpenPose and PGN, the data must be formatted correctly to receive them. The dataprocessing pipeline is a python code that recieves an .mp4 video, split the video into images, crop those images to 1080x1080.

```
def CropImage(InputPath):
    # Opens a image in RGB mode
    im = Image.open(InputPath)

    # Size of the image in pixels (size of original image)
    # (This is not mandatory)
    width, height = im.size

    # Setting the points for cropped image
    left = width / 2 - 540
    top = height / 2 - 540
    right = width / 2 + 540
    bottom = height / 2 + 540

    # Cropped image of above dimension
    # (It will not change original image)
    im1 = im.crop((left, top, right, bottom))

    #saves the image
    return im1
```

Figure 2: Frame Size Manipulation for Dataprocessing Pipeline

Figure 2. Shows how the video's frame size is manipulated by changing the parameters of left,top,right and bottom. These parameters restrict the the pixels taken from the width and height.

25.2. Interacting with the Olympus Server

Configuring the container in which octopus would be maintained ending up being a larger undertaking than previously considered. Within the container, I built a python environment that held the octopus code as well as any dependencies it needed to run.

The largest issue that was run into was when I needed to configure Cmake. Cmake controls the software compilation process and is needed for Dirt, a renderer used. This issue was surpassed by requesting root access to the servers via a sandboxed root that would allowed the download of any dependency required.

It had to be investigated where the linux singularity container nvidia driver files are hidden on olympus as dirt was not pointing at the right location. csrc text files were edited to further assist Dirt in defining the correct gpu-architecture location. Another large problem, was finding the OpenGL and EGL files needed to make cmake work. This again required editing csrc text files and find the correct location. Finally, issues were ran into as cuda architecture had to be changed between 10.2, and 11.0 to allow compatibility with Dirt.

26. Validation

Was able to output a skeletal model using Openpose. Was not able to properly update the dependencies required for Octopus to run on the Olympus server.

26.1. Dataprocessing Pipeline

A 1080x1080 frame was able to be achieved. By setting the parameters, the frames were cropped so that the center of the image was captured.

26.2. OpenPose Output

An output was able to be received from Openpose, shown through Figure 1. Shown there is the skeletal frame outlining the datapoints that will be going into the Octopus model as inputs.

26.3. Semantic Segmentation Output

An output was not able to be achieved through the use of PGN semantic segmentation.

27. Subsystem Conclusion

The subsystem was not completed on plan and did not meet the requirements set in the validation process.

27.1. Further Development

27.1.1. Integration with other subsystems

Once the machine learning model has been trained off of hundreds of data inputs, it would be located on the Firebase database server. This would be done so via a Docker container Firebase hosting and Cloud run will be used to host a flask server. From there, the server would communicate with the app to receive video files that it would then send through Openpose, PGN Semantic Segmentation and Octopus. Once done, Octopus would spit out an obj file that would be collected by the server and delivered back to the application.

27.1.2. Future Improvements

The subsystem could be improved by replacing OpenPose and PGN with Facebook's Detectron 2. Detectron 2 is an object detection platform that uses AI to create both a skeletal model and a semantically segmented image. It improves upon OpenPose and PGN by taking many more keypoints on the body, therefore, producing a smoother model. It also would significantly reduce the runtime needed to create the inputs for the Octopus model.

Virtual Clothes Try-on

Jorge Olivares
Robin Martinez
Alan Vela

FINAL REPORT

REVISION – 0.3
03 December 2022

Subsystem Report
for
Virtual Clothes Try-on

Prepared by:

_____
Author Date

Approved by:

_____
Alan Vela Date

_____
Prof. Jang Date

_____
Fardeen Mozumder Date

List of Figures

List of Tables

28. Overview

The frontend components were integrated with the backend into a single cohesive system. Integration with machine learning was not possible due to the subsystem being incomplete. Integration consisted of creating a system that could communicate with a GUI and a database storage system. The lack of a machine learning subsystem hindered the actual purpose of the system.

29. Execution

A gantt chart will be representing our execution plan shown in table 1. Integration for the frontend and backend were complete with machine learning still in progress. Throughout the semester issues were encountered when integrating and required for us to make changes to the system. These problems delayed progress and made the project more challenging; nevertheless, the GUI and database communication was established. This advancement provided a working application. The machine learning subsystem encountered multiple problems with the Olympus server and it wasn't possible to run the subsystem.

	09/05	09/12	09/19	09/26	10/3	10/10	10/17	10/24	10/31	11/7
Setting access and connection to Olympus										
Prepare budget and approve for parts										
Get the GUI writing on Firebase										
Set up algorithm on Olympus										
Create a catalog for the avatar										
Write .obj and .png files into database										
Train Model										
GUI reading information from Firebase										
Display clothing catalog and dummy model in GUI										
Test app on phone and bug fix										
Create app package with ML model										
Test app on phone with ML model and bug fix										

Table 1: Gantt chart explaining the execution process.

30. Design Plan

The project experienced several changes after each subsystem was ready for integration. The GUI subsystem changed drastically, as android studio was giving many errors and not enough resources online to continue on the application. These errors caused ineffectual video upload and displaying avatar. Figure 1 and 2 shows the first application created. Also, connectivity between the systems was an issue. The in used was Mongodb; this was a database easy to scale for a higher volume of users; however, the difficulty of establishing a connection between the GUI and the database subsystem led to huge changes in the end system result.

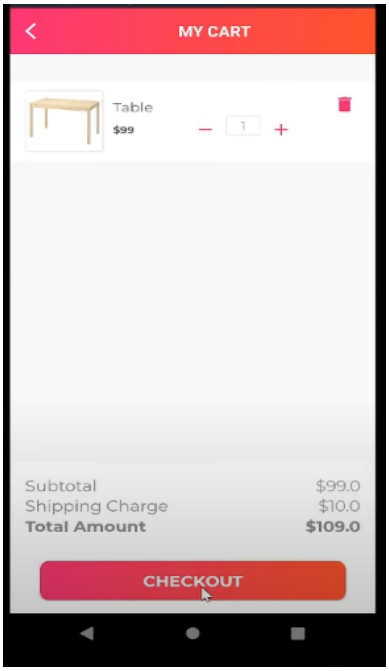


Figure 1: Previous Shopping Cart Activity

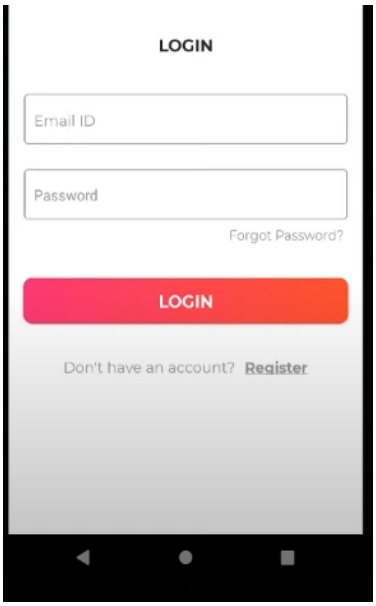


Figure 2: Previous Login Activity

For the frontend, react native was used instead of android studio. These changes took time to understand how to use javascript, git, and nodejs. Using react native, the application was able to fix the errors of uploading a video and displaying an avatar. Moreover, Firebase was the database selected to work with, allowing for greater progress with the GUI. The machine learning subsystem was incompleted and based on what the sponsor requested it was discontinued from the project.

31. Validation

The application was tested with different parameters. The parameters for the GIU are established on Table 2. The GUI passes all of the purposed tests except for the upload of image & video alert during the offline state. All the buttons of the system worked with great response time and it was able to communicate with the database. Also, when encountered with possible errors a pop up window was displayed with the error message.

Next, in table 3 are the results of the test done to validate the connectivity of the database. The latency is tested by pinging the database. Different sizes of files were introduced to verify the capabilities of the database. The database was tested by reading and writing data. The different types of files that the database accepted and the emails provided were in the proper format.

Parameters	Amount/Type	Validated
IOS Latency	74ms (Average)	Pass
Android Latency	68ms (Average)	Pass
Buttons	47	Pass
Alerts	19	Pass
Loading Screens	2	Pass
Navagation Screens	25	Pass
Multiple Devices Used	3	Pass
No Network Upload Alert		Fail
Offline Mode		Pass
Information Saved (cart, username, etc)	3	Pass
Start/Stop Application	2	Pass
Avatar Displaying		Pass
Profile Image Displaying		Pass
External SMS Operate with Application Open		Pass
Page Scrolling	4	Pass

Password Minimal Character Limitation	6	Pass
Registration with existing email		Pass
Insufficient Session Expiration		Pass
Video Length	<30secs	Pass
Keyboard Input Minimized		Pass
Going back & Undoing Method		Pass

Table 2: GUI Validation Results

Parameters	Amount/Type	Validated
Database Latency	146ms(average)	Pass
Video Files	All Android/IOS File types (.Mp4, Quicktime, .Mp3, .3gp, .mov)	Pass
Image Files	All Android/IOS File types (.jpeg, .png, .jpg, .bmp)	Pass
Sign-In Method	Any type of email (gmail, tamu, hotmail, yahoo)	Pass
One Video File per profile storage	3	Pass
Profiles Tested	5	Pass
Clothes Stored	10	Pass
File Size Cap	333 Mb	256Mb
Storage		10Gb

Table 3: Database Validationi Results

31.1. Alerts

To be able to validate the application, many alerts were used for the user to be able to use the application without fail. A total of 19 alerts were used in the application and below shows the console alerts given to the user and what they mean. Figures 1, 2 ,3, and 4 show some of the alerts on the application.

Signup/Login:

1. “Missing email” - User doesn’t type an email
2. “Internal error” - User type’s email but not password
3. “Email already in use” - Already have same email
4. “Password should be at least 6 characters/weak password” -password shorter than 6 characters
5. “User not found” - when trying to login but is not sign up
6. “Wrong password” -when user types in wrong password when login
7. “Invalid Email” - When user tries to login without an email
8. “Success sign up, please login” - when user signs up

Home Page/Menu:

9. “Contact Us” - When wanting to contact email will pop up
10. “Are you sure you want to logout?” -asks user if they want to logout

Store Products:

11. “Item added to cart” - let’s user know the item has been added to the shopping cart
12. “Item removed from cart” -let’s user know item has been removed from the shopping cart

Shopping Cart:

13. “Do you want to remove item from cart” - let’s user know if they are certain to remove
14. “Do you want to checkout” - let’s user know if they really want that item checked out
15. “Congrats order successful” - let’s user know the item’s have been checked out

Profile:

16. “User name updated successfully” -let’s the user know they updated their name successfully
17. “Are you sure you want to logout?” -asks user if they want to logout

Upload Video:

18. “Instructions” -Tell’s user what and how to upload video for avatar display
19. “Video too long” - Tell’s user video is too long

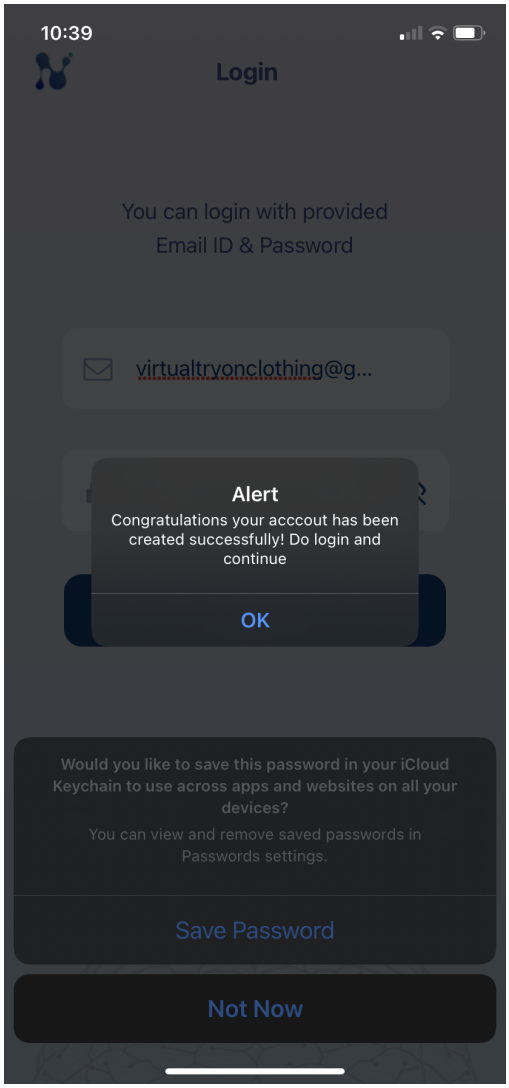


Figure 3: Successful Sign Up Alert

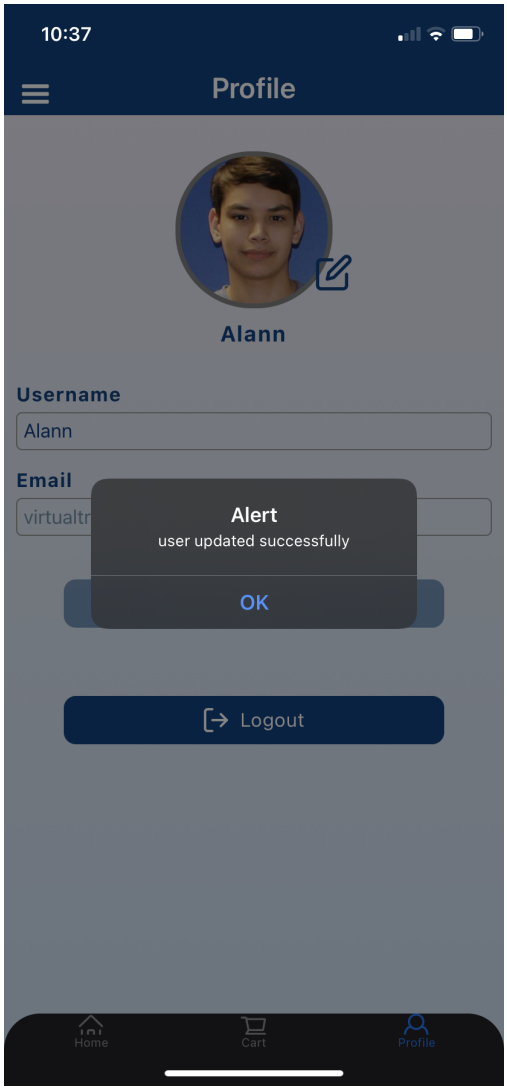


Figure 4: User Information Updated Alert

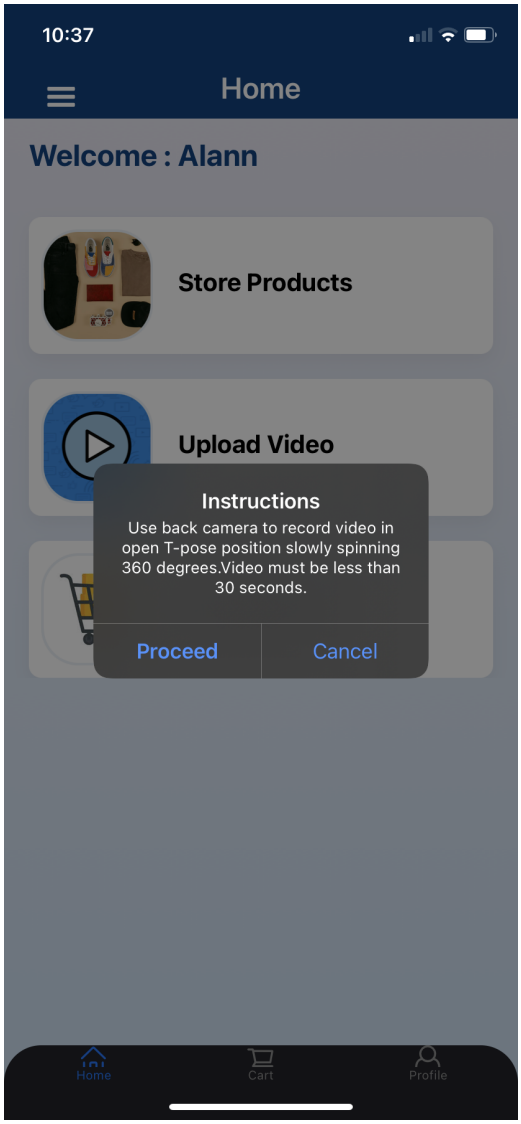


Figure 5: Successful Sign Up Alert

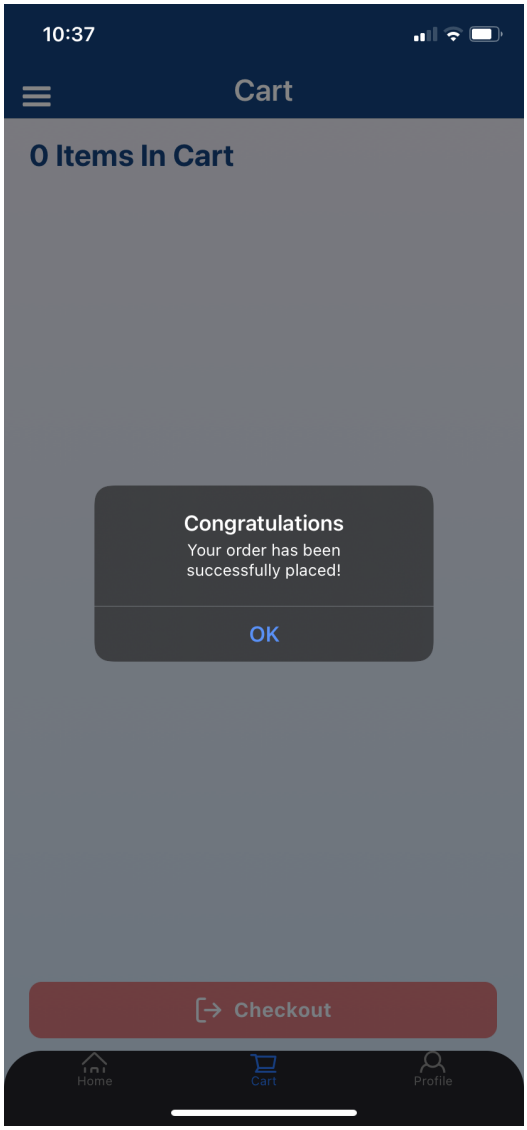


Figure 6: User Information Updated Alert

31.2. Machine Learning Model

31.2.1. Design Changes

One major design change implemented was what language the Octopus model would be running on. Previously it was requested to translate the Octopus model from python 2 to python 3. This design parameter was altered as it proved too challenging to modify Dirt, the renderer Octopus uses for the model, within the time constraints.

31.2.2. Interaction with the Database

The problems described below are gone more into detail in the Server1 and Server 2 text files on the Github repository.

31.2.2.1. Ubuntu Server

One of the solutions endeavored when trying to create an environment for the Octopus model was using a private Ubuntu Server. Doing so would allow the bypass of permission issues required to create containers, environments and the upload of dependencies.

Issues were ran into when trying to configure a container that could handle OpenPose. In particular, the installation of CUDA and cudNN, a program interface allows the handling of gpu's for general purpose processing, stopped progress into creating said container.

31.2.2.2. Outdated PC

One major issue encountered was the breakdown of a Mac computer being used to contact the Olympus server. A replacement Mac computer from 2014 was found but this introduced new issues with compatibility in regards to connecting to the server.

Since an older version of Mac was used, the Wireguard VPN used was no longer supported and so had to be built from source. Yet, the package manager that was suggested by Wiregaurd Technical Documentation, Homebrew, was also outdated. Alternatives, such as Cakebrew, were attempted to be used but those still threw errors when downloaded via the terminal.

31.2.2.3. Olympus Server

Permissions issues when interacting with the Olympus server continued to be a recurring issue. In particular the download of dependencies such as vim, cmake, python-virtualenv and libegl1-mesa-dev wasn't allowed in student home directories due to security issues. This was attempted to be remedied by going to a higher level within the student directories on a drive call /mnt/shared-scratch, suggested by Linux IT located at Wisenbaker. Yet, these permissions restrictions still would throw errors when software was being installed. A root container was also attempted but this was not successful in remedying the installation of these dependencies.

32. Overall Performance

The application ran smoothly with low latency and was able to run on multiple devices simultaneously. The application both supports Android 5+ (API level 21) and iOS 11+. The integrated system did not crash and information of the user was always stored in the database. The machine learning model was not able to be successfully displayed as it was not uploaded successfully to olympus due to permission issues.

33. Conclusions

Overall, the app and server system functioned well and according to the requirements of the sponsor. However, there was some issues with the application itself that needed to be improved on.

33.1. Application Improvements

The application had some minor errors that included misspelled words, negative value for removing items, and not having alert for email taken (gives alert “email invalid” instead). These minor errors can be changed easily. However, the major change of improvement of the application needed is being able to select different sizes of clothing on the same item. If the user selected a size small, and added the item to cart, then the user was not able to select the same item for a different size. The user will have to remove the item and select the correct size. This is a limitation of the application that needed to be improved. This can be done when implementing a function that allows the user to select the same item with different sizes.

33.2. Machine Learning Model

To get past problems when downloading the Machine Learning model it would be more advantageous to pursue the use of Google Cloud. This would allow you to download any container images and dependencies without running into permission issues that often occur on university servers.