

IBM – NALAIYA THIRAN PROJECT

SMART FASHION RECOMMENDER APPLICATION

INDUSTRY MENTOR : KRISHNA CHAITANYA
FACULTY MENTOR : DAYANA

TEAM ID :PNT2022TMID25168

TEAM LEAD : Mos Richard E

TEAM MEMBER : Kathiravan S

TEAM MEMBER : Saran N

TEAM MEMBER : Karan V

ABSTRACT

Fashion is perceived as a meaningful way of self-expressing that people use for different purposes. It seems to be an integral part of every person in modern societies, from everyday life to exceptional events and occasions. Fashionable products are highly demanded, and consequently, fashion is perceived as a desirable and profitable industry. Although this massive demand for fashion products provides an excellent opportunity for companies to invest in fashion-related sectors, it also faces different challenges in answering their customer needs.

In recent years, the textile and fashion industries have witnessed an enormous amount of growth in fast fashion. On e-commerce platforms, where numerous choices are available, an efficient recommendation system is required to sort, order, and efficiently convey relevant product content or information to users. Smart Fashion Recommender Application has attracted a huge amount of attention from fast fashion retailers as they provide a personalized shopping experience to consumers. Smart Fashion Recommender Application has been introduced to address these needs.

TABLE OF CONTENTS

SL NO.	CONTENTS	PAGE NO.
	INTRODUCTION	
1	1.1 PROJECT OVERVIEW 1.2 PURPOSE	05
	LITERATURE SURVEY	
2	2.1 EXISTING PROBLEM 2.2 REFERENCES 2.3 PROBLEM STATEMENT DEFINITION	06
	IDEATION & PROPOSED SOLUTION	
3	3.1 EMPATHY MAP CANVAS 3.2 IDEATION & BRAINSTROMING 3.3 PROPOSED SOLUTION 3.4 PROBLEM SOLUTION FIT	08
	REQUIREMENT ANALYSIS	
4	4.1 FUNCTIONAL REQUIREMENT 4.2 NON-FUNCTIONAL REQUIREMENTS	13
	PROJECT DESIGN	
5	5.1 DATA FLOW DIAGRAMS SOLUTION & TECHNICALARCHITECTURE 5.2 USER STORIES	15
	PROJECT PLANNING & SCHEDULING	
6	6.1 SPRINT PLANNING & ESTIMATION 6.2 SPRINT DELIVERY SCHEDULE 6.3 REPORTS FROM JIRA	18

	CODING & SOLUTIONING	
7	7.1 FEATURE 1	
	7.2 FEATURE 2	23
	7.3 DATABASE SCHEMA	
	TESTING	
	8.1 TEST CASES	33
8	8.2 USER ACCEPTANCE TESTING	
	RESULTS	
9	9.1 PERFORMANCE METRICS	35
10	ADVANTAGES & DISADVANTAGES	36
11	CONCLUSION	37
12	FUTURE SCOPE	37
13	APPENDIX	38
	13.1 SOURCE CODE	
	13.2 GITHUB & PROJECT DEMO LINK	

1. INTRODUCTION

PROJECT OVERVIEW

The Fashion industry is one of the larger industries around the world. One of the things that have remained constant throughout human civilization is humans covering their bodies with a piece of cloth. Initially, this cloth was worn as protection from the harsh climates of those ages. Later on, as we humans learned to fend for ourselves from the unforgiving climates, the cloth started to serve a different purpose. Fashion these days showcases the individuality of the person. There are many things that can be said about a person based on their fashion sense.

PURPOSE

There is currently no existing system that is capable of recommending clothes based on the occasion. Different occasions call for different clothing. Moreover, a lot of fashion is based on the color combinations of outfits. A person with no or little fashion sense will have a hard time to decide on clothes that leave a lasting impression. The proposed Fashion Recommendation System is intended to be used by individual users in order to store images of the clothes that they own in what is called a digital wardrobe and also to get recommendations by the system on what clothes to wear for a given occasion. The main aim of the project is to recommend the most appropriate clothes for a given occasion based on the clothes existing in the user's wardrobe to relieve the user of the burden of making decisions about what clothing to wear. Such a system should be capable of helping someone who has no fashion sense to wear clothes that leave a good impression on others. The system should be such that it is easily accessible and easy to take advantage of the various features that it provides. One of the features should be the ability to store images that the user uploads into a wardrobe. A wardrobe is a very useful entity that the user can use to view and manage the images of clothes that they have uploaded. This feature can also be used by the recommendation algorithm to recommend the clothes. Another feature is the classification of the type and color of the clothing that is uploaded by the user. The system should be capable of handling the 4 basic clothing types: Shirt, T-Shirt, Pants and Shoes.

2. LITERATURE SURVEY

Chatbot design, consumer trust and privacy: Chung, Ko, Joung, & Kim stated that since consumers might see chatbots with negative eyes if there are privacy concerns, consumer trust is another factor that can be explored further. Aspects such as transparent advice and problem-solving could be investigated in future research, addressing the role that design plays in this context and if other factors (e.g. social elements, cultural values, self-identity) influence consumer's trust on chatbots.

Multi-user chatting: Merrilees and Miller observed that traditional shopping with a companion influences the consumer experience. Alone consumers tend to be more price sensitive. Future studies may explore the way fashion consumers seek for advice from chatbots that could be experimented with by adjusting social factors (e.g. including a friend in the conversation), evaluating the impact of these factors on user acceptance levels.

Design bots: Colombi, Kim, & Wyatt suggested that fashion chatbots may be made to behave as a fashion designer, providing a platform to support co-creation of value. Conversational platforms can provide insights for brands to recognize consumer value, which means that future research in this area can also enhance the consumer experience.

Consumer autonomy and identity in chatbot consumer experience: Ameen, Hosany, & Tarhini suggested that consumer autonomy is related to the perceived sense of control that consumers have over the interaction with chatbots and it can be attached to motivational factors. Future studies might address the role that consumer autonomy and identity play in consumer trust and acceptance, for example, by measuring chatbots design approaches that can trigger these states.

2.1 EXISTING PROBLEM:

In existing system only simple web application and their rating has been implemented in existing system, An ecommerce product recommendation engine is a piece of technology that displays recommended products to shoppers throughout your store. It uses machine learning to get smarter and show increasingly relevant products to shoppers based on their interests and previous browsing behavior.

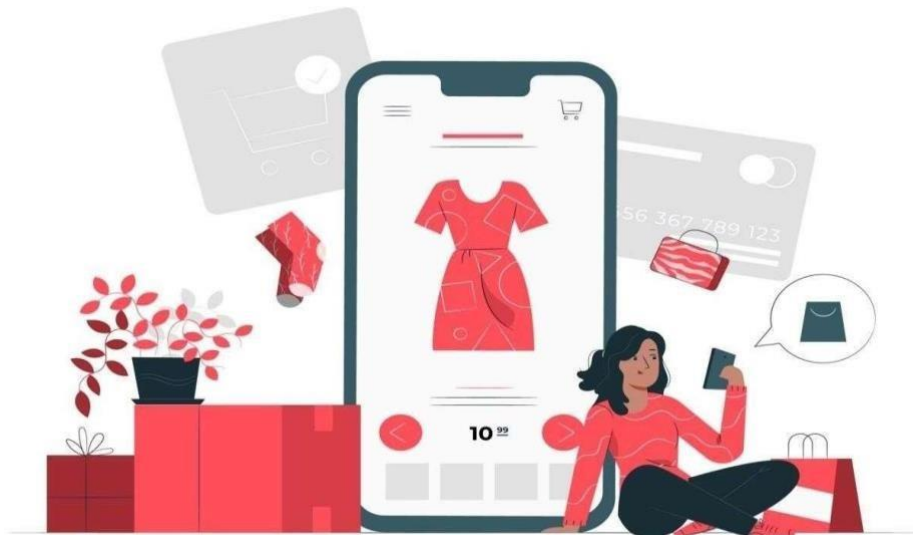
REFERENCES:

1. GloablInfoResearch: Global Fast Fashion Apparel Market 2021 by Key Countries, Companies, Type and Application. GloablInfoResearch, HongKong, 2021.
2. Hou, M., Wu, L., Chen, E., Li, Z., Zheng, V. W., & Liu, Q.: Explainable fashion recommendation: A semantic attribute region guided approach. In Proceedings of the 28th Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019; pp. 4681-4688.
3. Hidayati, S. C., Hsu, C. C., Chang, Y. T., Hua, K. L., Fu, J., & Cheng, W. H.: What Dress Fits Me Best? Fashion Recommendation on the Clothing Style for Personal Body Shape. In Proceedings of the 26th ACM International Conference on Multimedia (MM '18). Association for Computing Machinery, New York, NY, USA, 2018; pp. 438-446.
4. Wang, H., Wang, N., & Yeung, D. Y.: Collaborative Deep Learning for Recommender Systems. In Proceedings of the 21th CM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, 2015; pp. 1235- 1244.
5. McAuley, J., Targett, C., Shi, Q., & Van Den Hengel, A.: Image-based Recommendations on Styles and Substitutes. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015; pp. 43-52. 2015.

2.2 PROBLEM STATEMENT DEFINITION

The personal information collected by recommenders raises the risk of unwanted exposure of that information. Also, malicious users can bias or sabotage the recommendations that are provided to other users. In recent years, the textile and fashion industries have witnessed an enormous amount of growth in fast fashion. On e-commerce platforms, where numerous choices are available, an efficient recommendation system is required to sort, order, and efficiently convey relevant product content or information to users.

- The problem of the work is to design static web applications deployments with customer deployment
- Lack of interaction between application and user
- User need to navigate across multiple pages to choose right product
- Confusion in choosing product
- Lack of sales
- Complex User Interface.
- Lack of proper guidance.

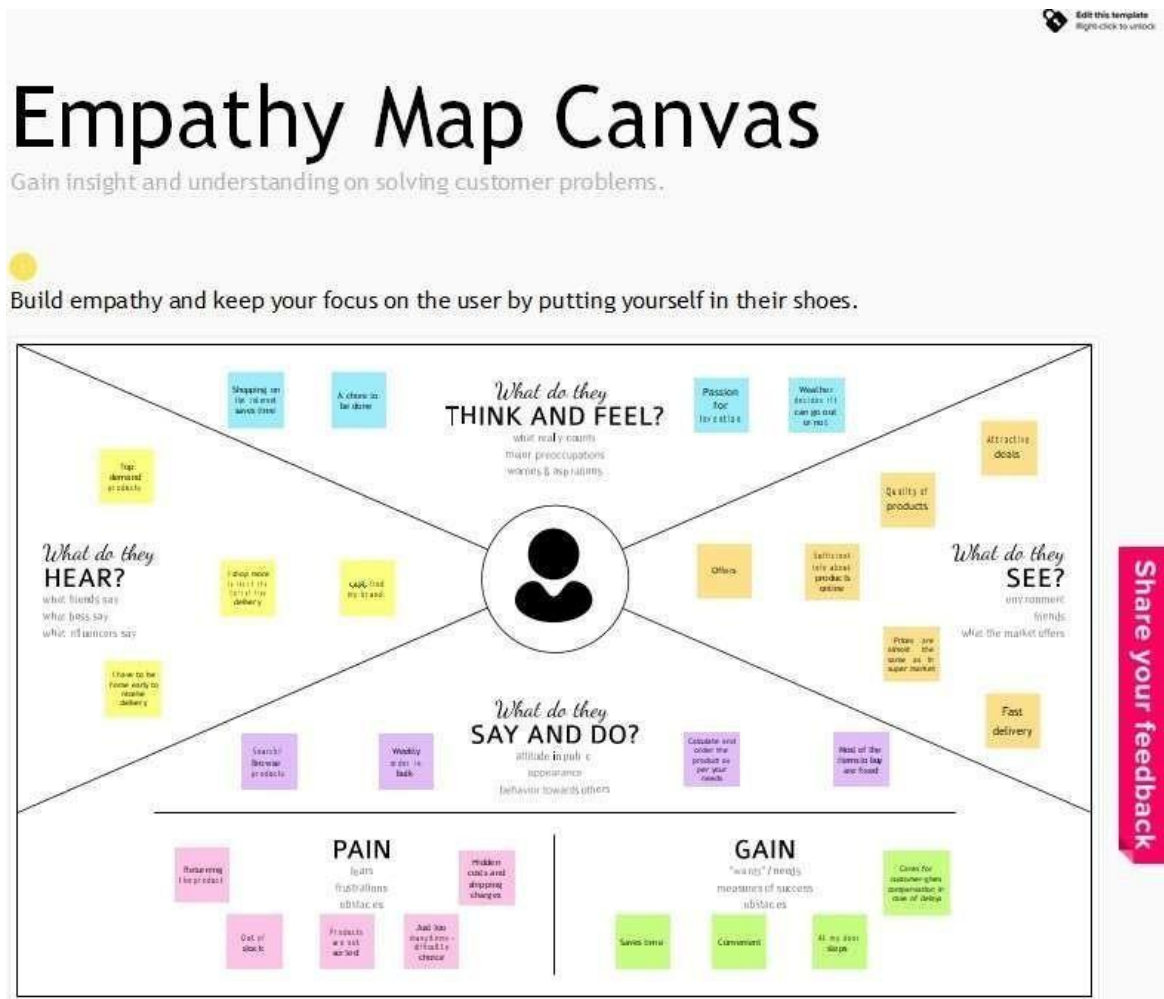


3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges. An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers.

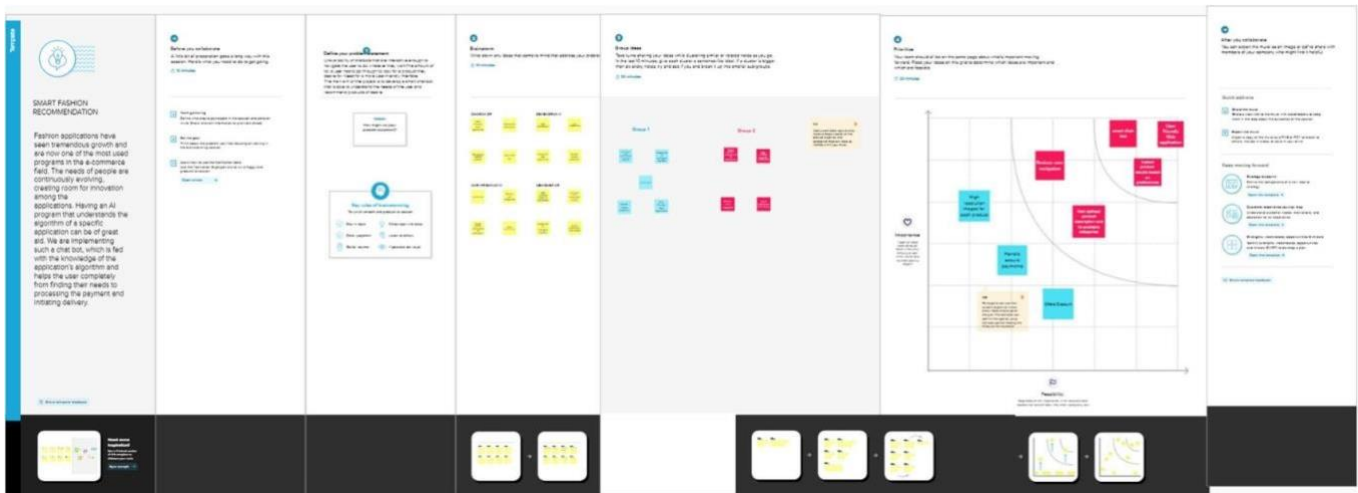
SMART FASHION RECOMMENDER APPLICATION



3.2 IDEATION & BRAINSTROMING:

A group problem-solving technique that involves the spontaneous contribution of ideas from all members of the group.

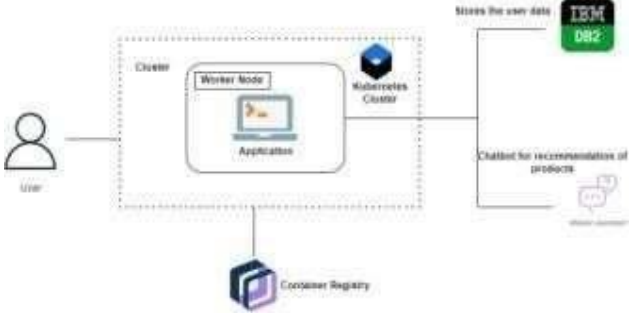
The mulling over of ideas by one or more individuals in an attempt to devise or find a solution to a problem.



3.3 PROPOSED SOLUTION:

SMART FASHION RECOMMENDER APPLICATION

S.NO.	PARAMETER	DESCRIPTION
1.	Problem statement(problem to be solved)	<ul style="list-style-type: none">• In E-commerce websites, users need to search for products and navigate across screens to view the product and order product.• A new innovative solution came up through which can directly make online shopping based on the choice of the user without any search.• It can be done by using the chatbot which can be achieved by a smart fashion recommender application.
2.	Idea/ solution description	<ul style="list-style-type: none">• The smart fashion recommender application leverages the use of a chatbot to interact with the users, gather information about their preferences, and recommend suitable products to the users.• User can be able to mention their preferences by interacting with chatbot.• The user must receive a notification on order confirmation/failure.• The chatbot must gather feedback from the user at the end of order confirmation
3.	Novelty/ Uniqueness	<ul style="list-style-type: none">• Chatbot asks and learns from user preference which recommends appropriate products to the user without making them search through various filters which reduces time and thus increases sales.• Instead of searching manually a chatbot will help to find the right product effectively, with this feature user can save time and it is an easy process, chat keep sending a notification about new collections
4.	Social impact/Customer satisfaction	<ul style="list-style-type: none">• Feedback from the user at the end of the session or after placing an order is one of the most important factors in deriving customer satisfaction and providing better services.• The model can recommend products that are more suitable to the customer.• Directly do online shopping based on customer

		<p>choice without any search.</p> <ul style="list-style-type: none"> • It can also save a lot of time.
5.	<p>Business model (Revenue model)</p>	<ul style="list-style-type: none"> • Due to market dynamics and customer preferences, there is a large vocabulary of distinct fashion products, as well as high turnover. • This leads to sparse purchase data, which challenges the usage of traditional recommender systems. • Better experience and Feasibility. 
6.	<p>Scalability of the solution</p>	<ul style="list-style-type: none"> • The solution can be made scalable by using micro service architecture provided that each server is responsible for certain functionality of the application. • Storing user preferences along with the product in the browser cookie will enable it to provide a response instantly and allows for fetching related products. • The scalability can be increased by increasing the number of products and also the accuracy of the product suggestions

3.4 PROBLEM SOLUTION FIT

In this project, we propose a model that uses Convolutional Neural Network and the Nearest neighbour backed recommender. As shown in the figure Initially, the neural networks are trained and then an inventory is selected for generating recommendations and a database is created for the items in inventory. The nearest neighbour's algorithm is used to find the most relevant products based on the input image and recommendations are generated.

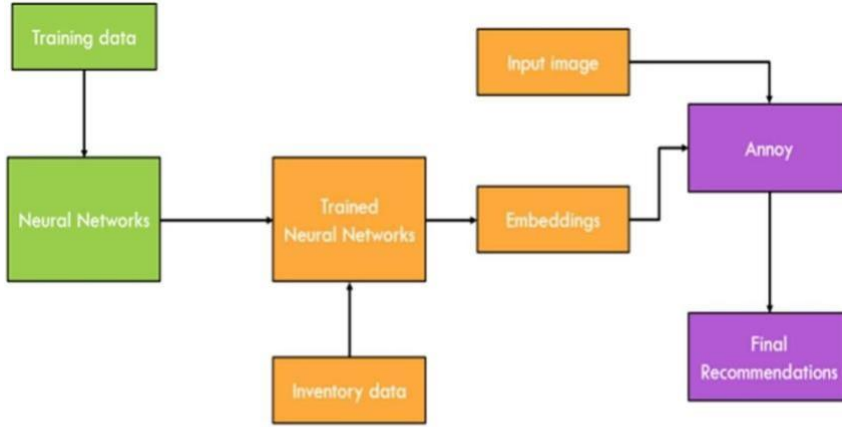


Figure 1. Block diagram of proposed system

Training the neural networks:

Once the data is pre-processed, the neural networks are trained, utilizing transfer learning from ResNet50. More additional layers are added in the last layers that replace the architecture and weights from ResNet50 in order to fine-tune the network model to serve the current issue. The figure shows the ResNet50 architecture.

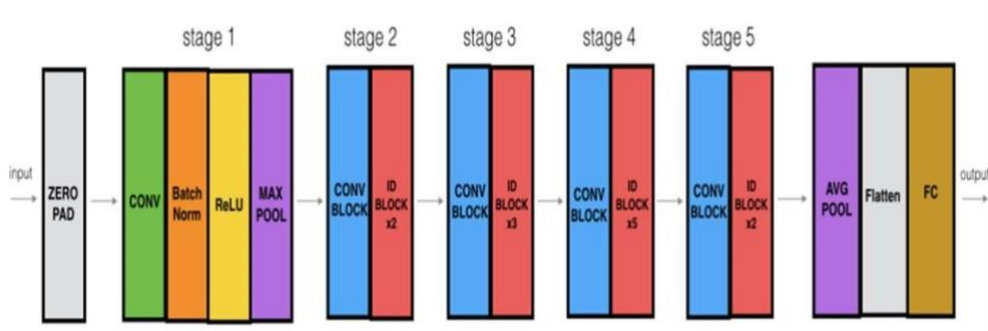


Figure 2. ResNet50 architecture

Getting the inventory:

The images from Web Scrapping from myntra Fashion Product Images and creating a Dataset. The inventory is then run through the neural networks to classify and generate embeddings and the output is then used to generate recommendations. The Figure shows a sample set of inventory data

Recommendation generation:

To generate recommendations, our proposed approach uses Sklearn Nearest neighbours Oh Yeah. This allows us to find the nearest neighbours for the given input image. The similarity measure used in this Project is the Cosine Similarity measure. The top 5 recommendations are extracted from the database and their images are displayed.

Experiment and results:

The concept of Transfer learning is used to overcome the issues of the small size Fashion dataset. Therefore we pre-train the classification models on the DeepFashion dataset that consists of 44,441 garment images. The networks are trained and validated on the dataset taken. The training results show a great accuracy of the model with low error, loss and good f-score.

The Jaccard Index:

Let's return to the question of what cloths to recommend to Bob given Sally and Judy's rating history. In our heads, we might have determined that Bob may be more of a western fan based on his sole rating of The BTS. In other words, Bob follows a pattern that other users share. We need a way to see how similar Bob is to other users. This is the purpose of the Jaccard Index.

$$S(U_1, U_2) = \frac{|L_1 \cap L_2|}{|L_1 \cup L_2|}$$

Project Title: Smart Fashion Recommender Application

Project Design Phase-I – Problem Solution Fit Template

Team ID: PNT2022TMD49171

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) <small>Who are your customers? Is anything different about them?</small> <div>Our customers are children and adults.</div>	6. CUSTOMER CONSTRAINTS <small>What constraints could hinder customers from adopting or fully utilizing your solution? (e.g., lack of resources, competing solutions, lack of knowledge, etc.)</small> <div>As much as service providers make the needs of their customers, it is just as important for them to satisfy their customers.</div>	5. AVAILABLE SOLUTIONS <small>What solutions are available to the customer when they face the problem? Are there any pros and cons? What are the pros and cons of these solutions? What are the pros and cons of these solutions?</small> <div>Online shopping gives new collections. The pros are easy to use. The cons are customer confused when have lost of collections.</div>	Explore AS, differentiate
	2. JOBS TO BE DONE / PROBLEMS <small>What jobs do your customers have? What problems do they face? What do they want to achieve? What do they want to avoid?</small> <div>From the customer can easily to choose a best out fitting product. And to even manage time in effective way.</div>	9. PROBLEM ROOT CAUSE <small>What is the root cause of the problem? Why is it a problem? What are the root causes? What are the root causes?</small> <div>Customers need to be with new fashions for current trends. Lot of time is wasted and an best product of his/her outfits not selected.</div>	7. BEHAVIOUR <small>What does your customer do to address the problem when they face the job? What are the pros and cons? What are the pros and cons? What are the pros and cons?</small> <div>Customer experience, content performance, and perfection in the product review, spend time to find new clothes.</div>	
Focus on J&P, fit into BE, understand RC	3. TRIGGERS <small>What triggers customer to get into the problem? What are the triggers? What are the triggers? What are the triggers?</small> <div>This software like as a merchant. It can access the customer location and give the related identification.</div>	10. YOUR SOLUTION <small>How are you solving the problem? What are the solutions? What are the solutions? What are the solutions?</small> <div>Make the Chat bot Assistant for shopping with customers and send notifications when new collections arrived.</div>	8. CHANNELS OF BEHAVIOUR <small>What channels are available to the customer? What are the channels? What are the channels? What are the channels?</small> <div>ONLINE: This application depends upon the internet connectivity, because we use the API and data connection through internet. OFFLINE: This is not applicable in online mode.</div>	Identify strong TR & EM
	4. EMOTIONS: BEFORE / AFTER <small>How do customers feel when they face a problem? What are the emotions? What are the emotions? What are the emotions?</small> <div>Feeling sad and frustration > Self-confident</div>	11. YOUR SOLUTION <small>How are you solving the problem? What are the solutions? What are the solutions? What are the solutions?</small> <div>Make the Chat bot Assistant for shopping with customers and send notifications when new collections arrived.</div>	12. OFFLINE <small>What channels are available to the customer? What are the channels? What are the channels? What are the channels?</small> <div>ONLINE: This application depends upon the internet connectivity, because we use the API and data connection through internet. OFFLINE: This is not applicable in online mode.</div>	
Identify strong TR & EM				Identify strong TR & EM

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS:

SMART FASHION RECOMMENDER APPLICATION

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Sign up	Register by using mobile number/ Register by using email id.
FR-2	User Verification	Verify via Email Verify via OTP
FR-3	Login	Login by using username / password
FR-4	Profile Updation	Update the profile details like Name,Gender, Age ,Address & mobile number ,etc,.
FR-5	Chatbot	Chatbot is useful to search products , view offers,discounts and stock availability. It is also used to solve queries and issues.
FR-6	Ordering the product	After confirming the product , buy the product via Cash on Delivery or online transactions.
FR-7	Tracking the ordered Product	After ordering the product , track the delivery via link received to your registered mobile number through SMS or registered email id.
FR-8	Logout	After receiving the product ,user can logout the account when he/she needs

4.2 NON-FUNCTIONAL REQUIREMENTS:

SMART FASHION RECOMMENDER APPLICATION

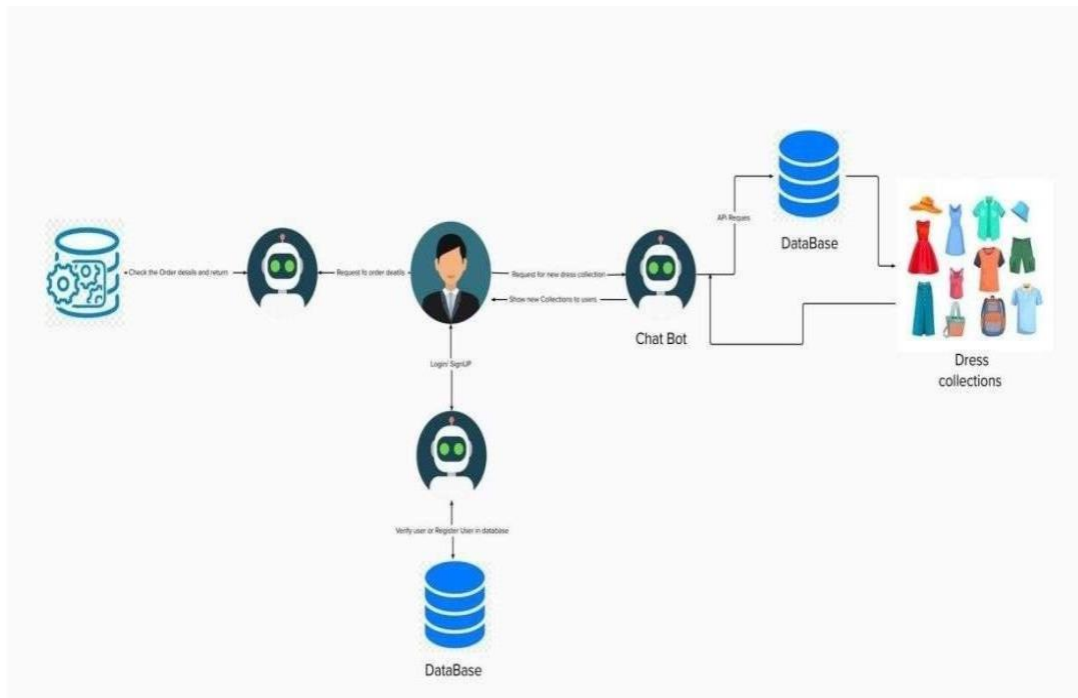
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The application will be designed in such a way that any user can easily navigate through it and user can easily view , order and track the product until delivery.(Easy and Compact design.)
NFR-2	Security	Using of SSL (Secure Socket Layer) certificate (Python Flask to Cloud connect) will provide security to the project. The user details will be kept as more secure.
NFR-3	Reliability	To make sure the application doesn't go down due to network traffic and the details entered in this application is kept as highly confidential, so it is highly reliable.
NFR-4	Performance	It focus on loading the application as quickly as possible irrespective of the number of users/integrator traffic.
NFR-5	Availability	This application will be available to all users (network connectivity is necessary) at any given point of time. Users can access the chatbot for raising any queries/ questions.
NFR-6	Scalability	Chatbot can be very useful during festival season to know about offers and discounts. It will be helpful whenever we make online shopping.

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 SOLUTION & TECHNICAL ARCHITECTURE:

We have developed a new innovative solution through which you can directly do your online shopping based on your choice without any search. It can be done by using the chatbot. In this project you will be working on two modules:

- Admin
- User

Instead of searching for products in the search bar and navigating to individual products to find required preferences, this project leverages the use of chatbots to gather all required preferences and recommend products to the user. The solution is implemented in such a way as to improve the interactivity between customers and applications. The chatbot sends messages periodically to notify offers and preferences. For security concerns, this application uses a token to authenticate and authorize users securely. The token has encoded user id and role. Based on the encoded information, access to the resources is restricted to specific users.

The proposed solution is a Content-Based recommendation system that involves various modules that carry specific tasks that help in achieving the overall goal. For the system proposed in the published article by Myntra the first module consists of the Human Key Point Detection, looks for specific human key points like the ankle and the eyes in order to classify the image as a full-shot image i.e. an image that consists of the full body of a model, however the solution proposed in this blog skips this step, instead we'll recommend items using all the fashion products detected in an image. For better product recommendation we'll first try to determine the gender of the person in the query image. The pose can either be front, back, left, right or detailed. After determining the gender, the input is fed is then fed to the next module, it's an Object Detection module trained to identify objects from these broad

categories - top-wear, bottom-wear, and foot-wear. The goal of this module is not just the identification of these objects in the image but also to localize the objects with bounding-boxes. Once the bounding boxes are obtained for a certain object, that part of the image is cropped and the semantic embeddings for the cropped image are obtained, which will help in identifying similar products from the catalog/database of products whose semantic embeddings are already known. We'll train a Siamese Network for generating the embeddings for the catalog of images. This system helps in automatically and efficiently recommending new items to a user who has selected or shown interest in a certain product display page.

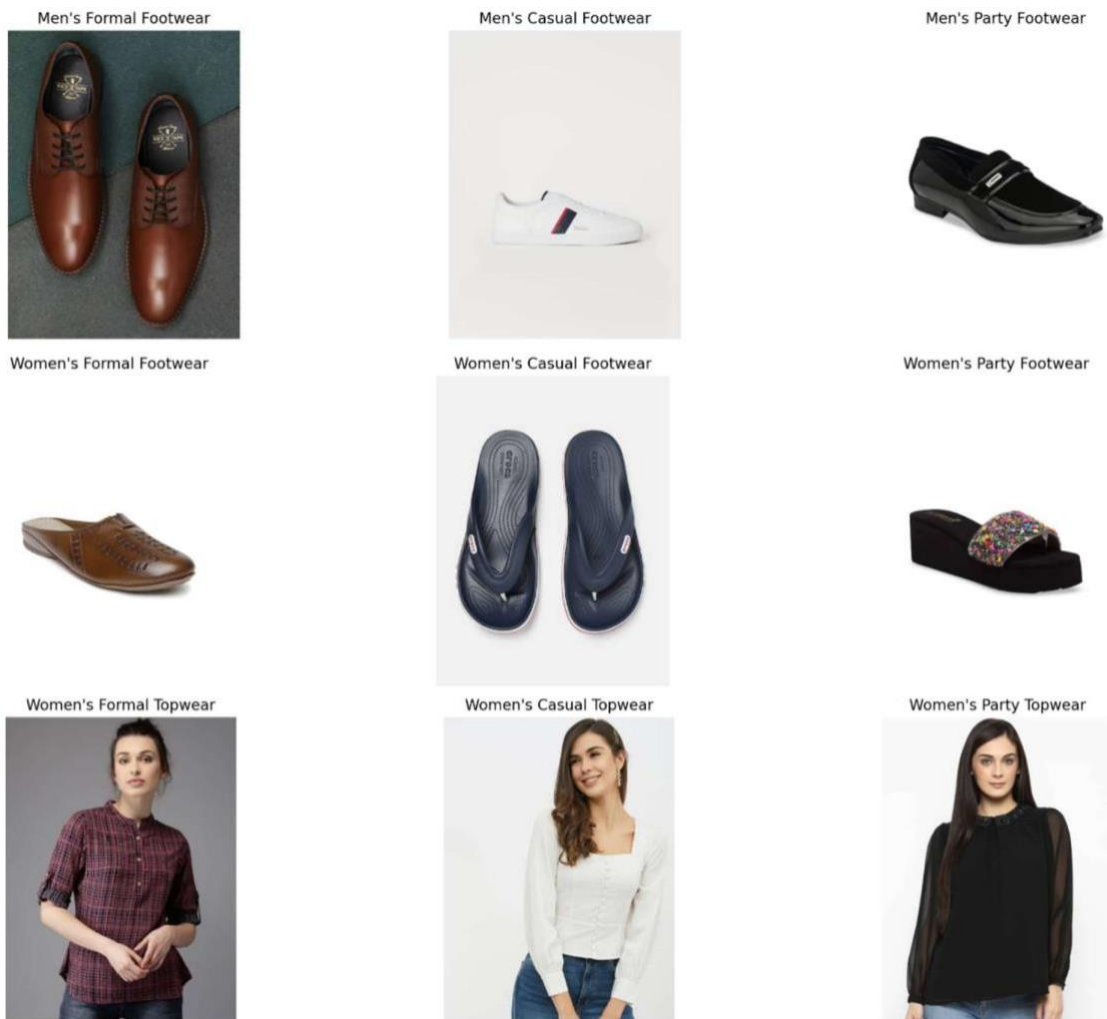
Data Acquisition:

To get started with the Fashion Recommendation Engine, we'll need a catalog/database consisting of fashion products to recommend. Unless you have a database of these items lying around we'll need to acquire this data using simple web scraping tools in python. Since we'll focus on recommending items across various categories we'll need a diverse set of fashion products to populate the database. We can use Myntra's e-commerce website to query images using specific keywords and later save the product details for the results obtained. This way we can categorize products into various sections which would make it easier while recommending items. We'll use Selenium package from python, its the most popular browser automation tool mainly used to carry out web-product testing. For that we'll require webdriver, for chrome users, you can get it from [here](#), make sure the version that you download matches with the version of chrome on your computer. For more details on webscraping e-commerce websites using python follow this article "[Webscraping e-commerce websites using python](#)". After scraping the image and product urls, you either use

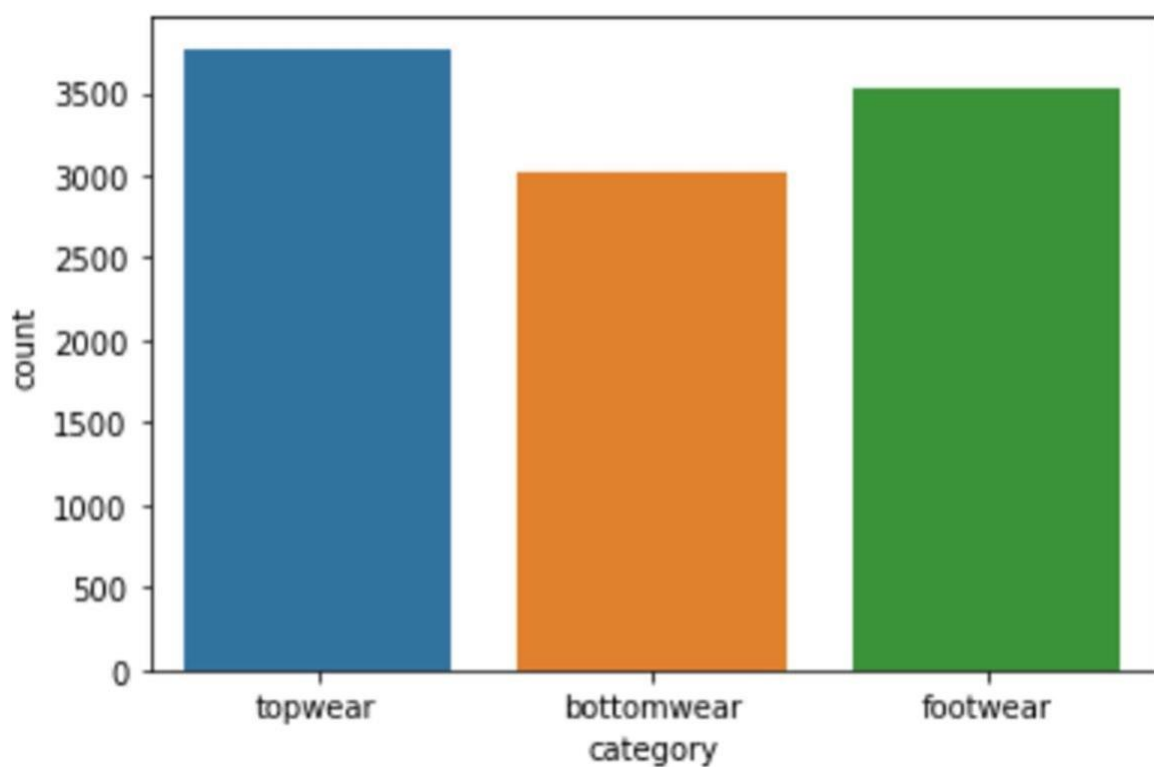
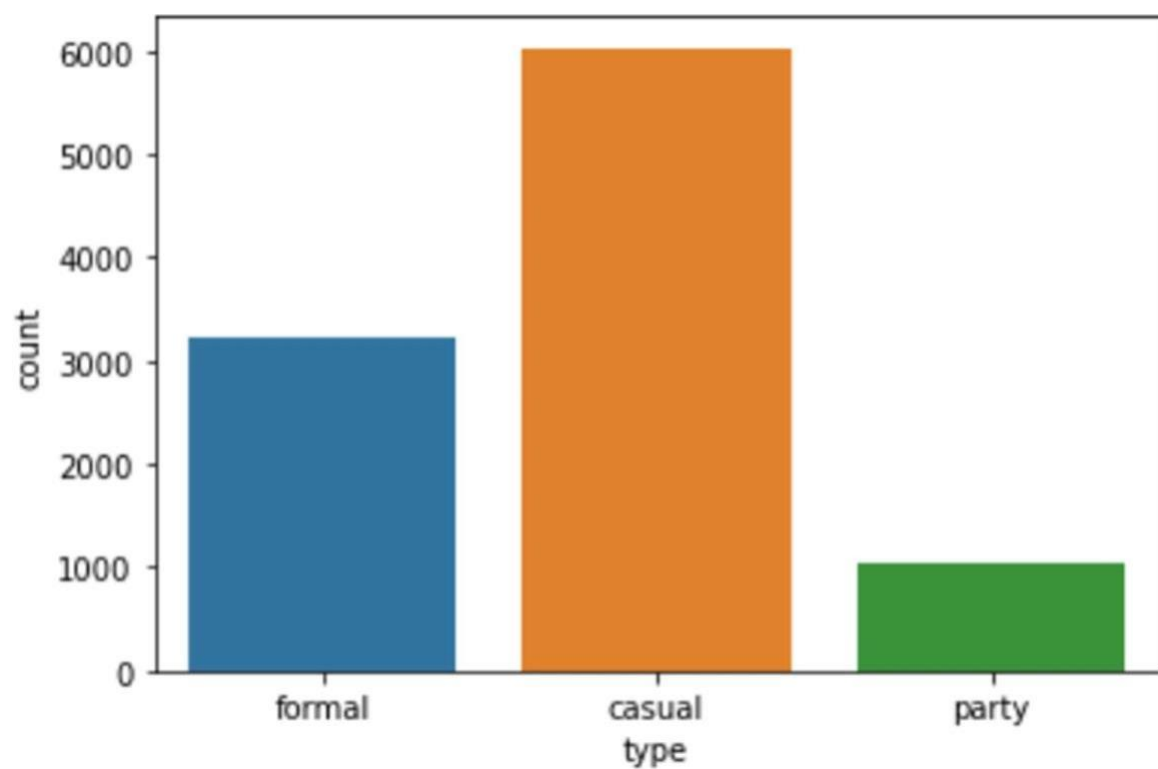
it to download the images into your local system or save the image and product urlpair to a flat file and use it later.

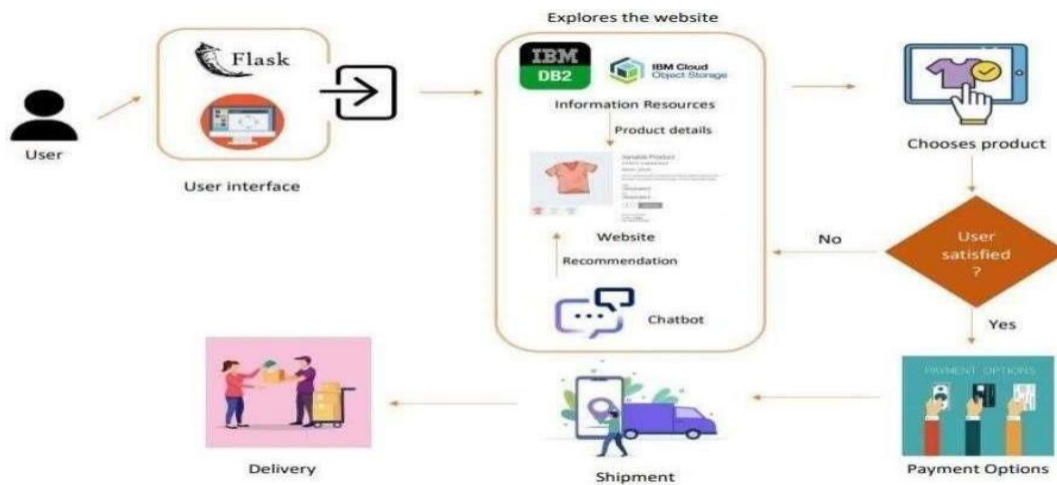
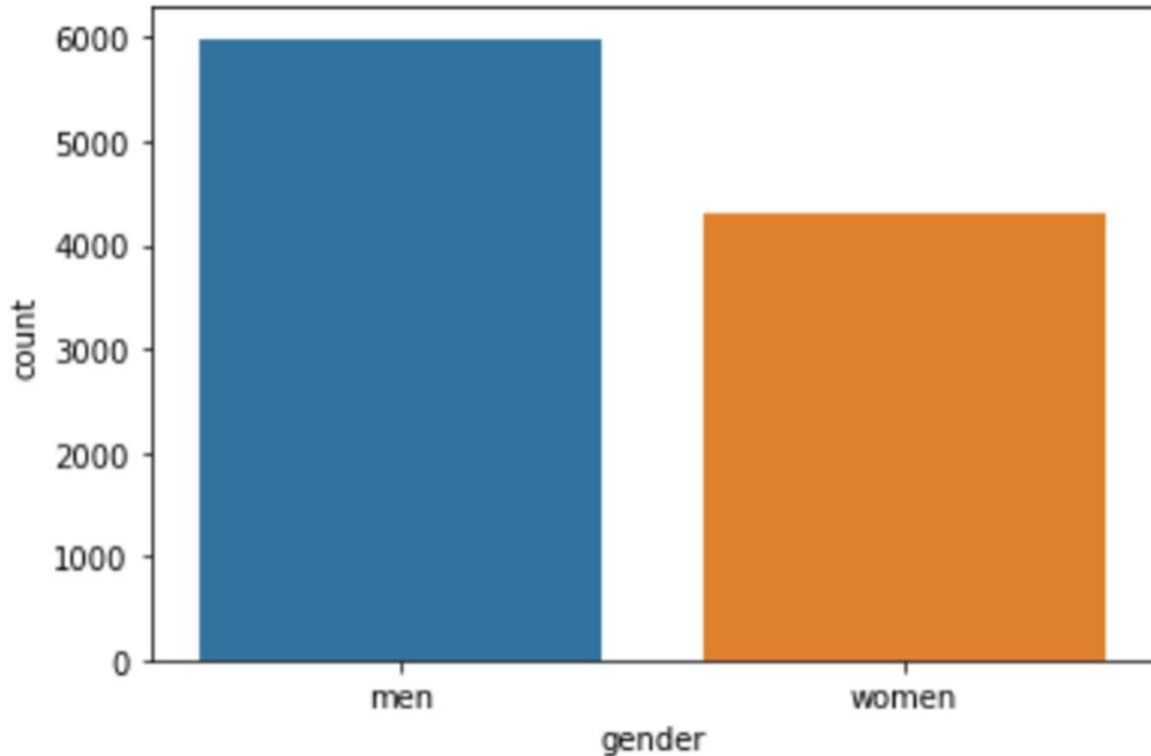
Exploratory Data Analysis:

Now that we've gathered the catalog, lets do some exploratory analysis on it. The dataset consists of 10310 fashion products in various categories. The categories being:



The fashion products are not equally split among the two genders, there are 5992 Mens Products whereas as just 4318 Womens products. The plot below is a visual representation of the products split by genders.



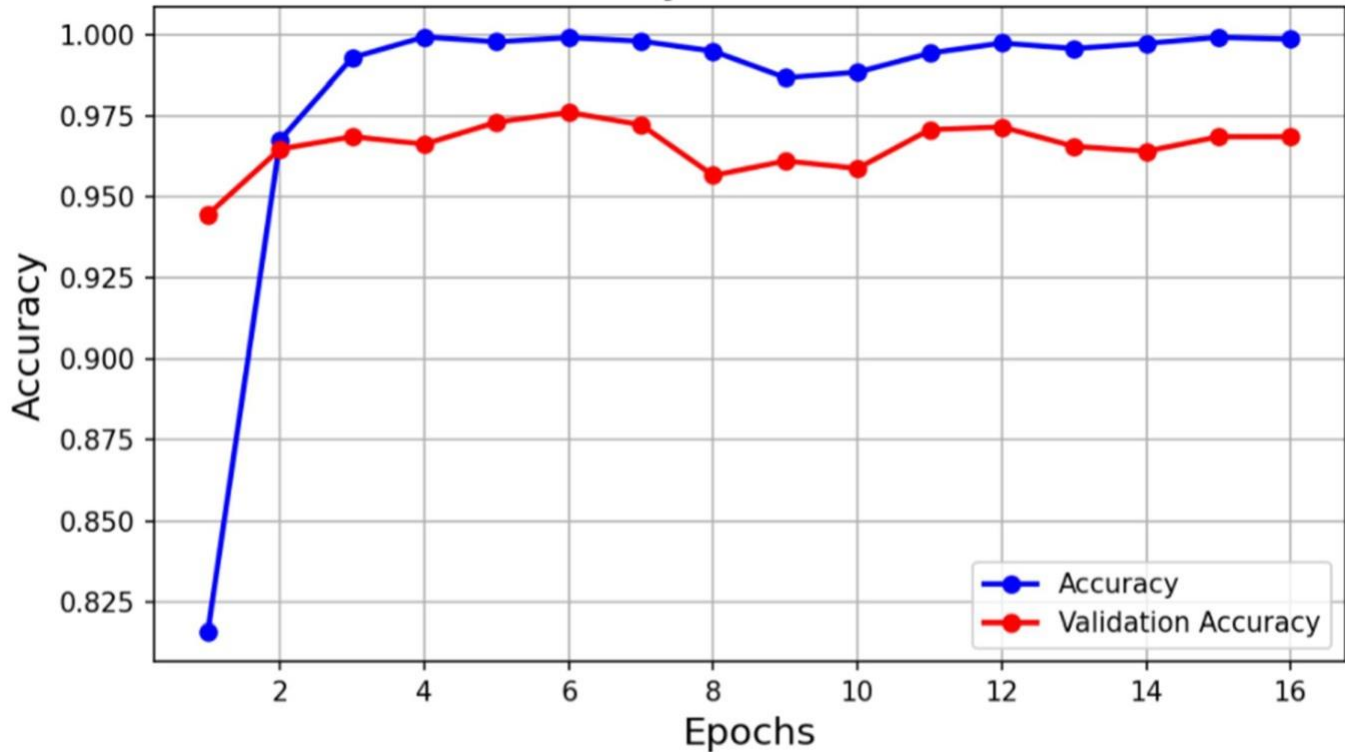


Gender Classification:

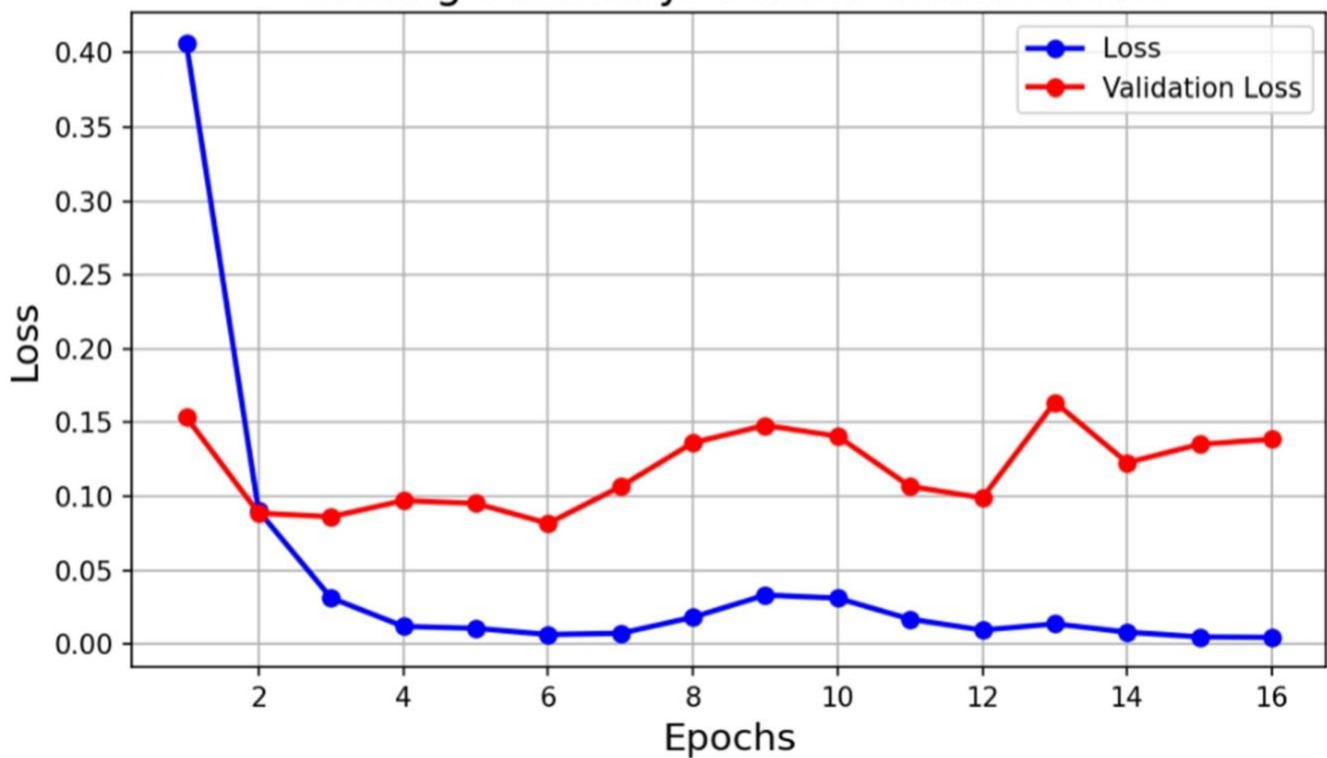
In order to recommend relevant items to the user we'll have to first detect the gender of the person present in the query image. Doing so will allow us to generate and query embeddings only on certain subsets of the catalog instead of the whole database. This will also help us in reducing irrelevant recommendations to the user based on their gender. For this task we'll make use of the Man/Woman classification dataset from Kaggle. This is a manually collected and cleaned dataset containing 3354 pictures (jpg) of men (1414 files) and women (1940 files) that includes full body and upper body images of men and women. This is ideal for our use case since we'll mostly have full body images of models wearing various fashion items. For classification we can use a

ResNET50 backbone initialized with 'Imagenet' weights attached to a fully connected network that outputs the probability scores for the two classes. We'll freeze all the layers from the backbone since the task is comparatively easy to achieve and in order to avoid overfitting to this dataset. Below is a code snippet for the model used for classification. We are able to achieve accuracy of 95% on the test set which is decent for our use case.

Metric Summary: Gender Classification



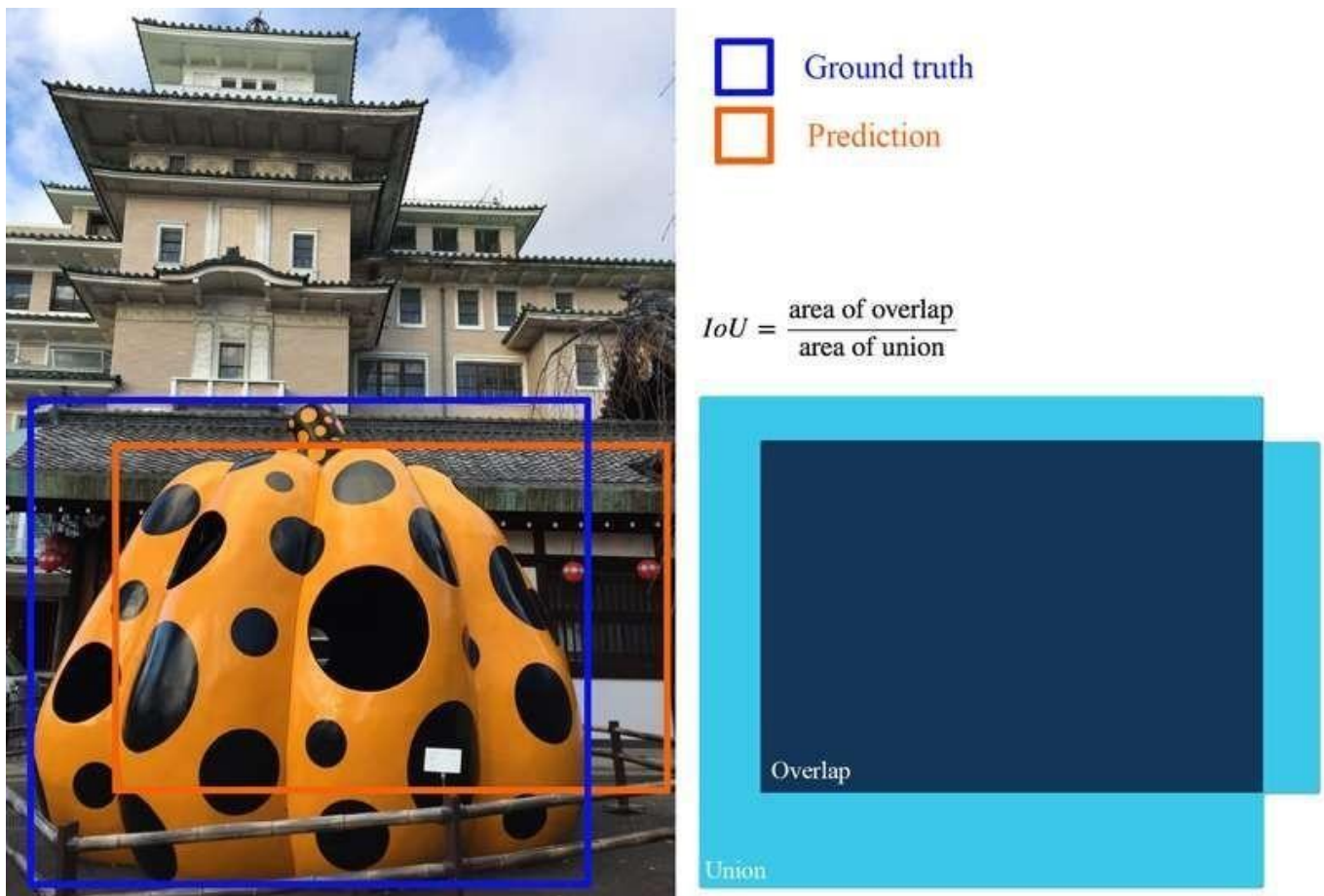
Training Summary: Gender Classification



Object Detection & Localization:

In order to recommend similar products we'll first need to detect all the products in a given image, and for generating the embeddings we'll need to crop out the product from an image. Object detection is one of the most interesting aspect of Computer Vision. Most object detection systems have some kind of trade off between inference speed and detection accuracy. In order to better understand object we'll need to understand a metric called 'mAP' or Mean Average Precision, for object detection whether a detection is considered correct or not depends on the IoU (Intersection over Union) threshold. The IoU score for a predicted bounding box and actual bounding box is defined as

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

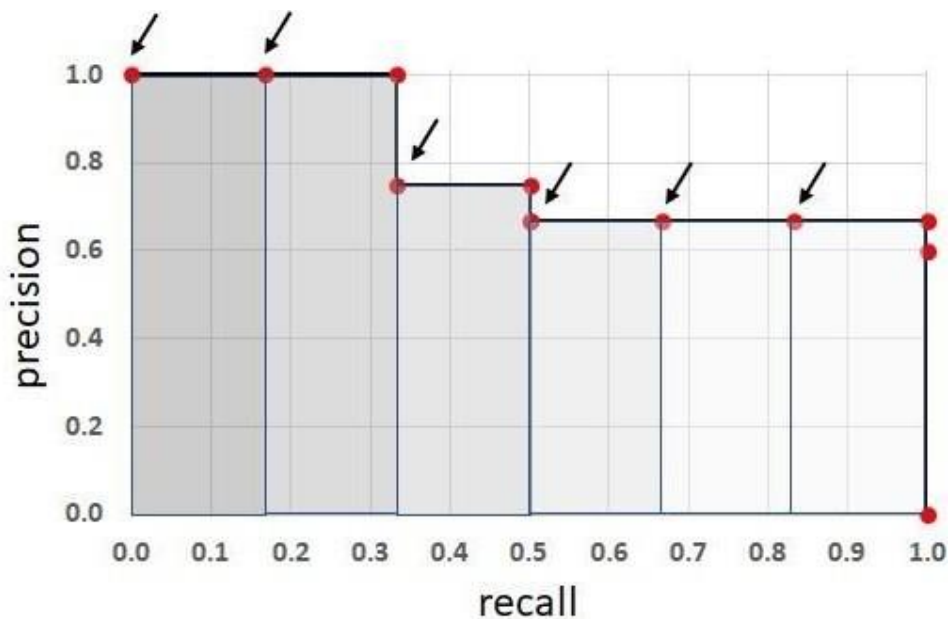


For a pre-defined IoU threshold we can define if a detection was accurate or not if the IoU is greater than the threshold. Based on the detections Precision and Recall is defined as.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \& \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Where TP = True Positive, FP = False Positive & FN = False Negative

The average precision is calculated by using the Area Under the Precision and Recall Curve.

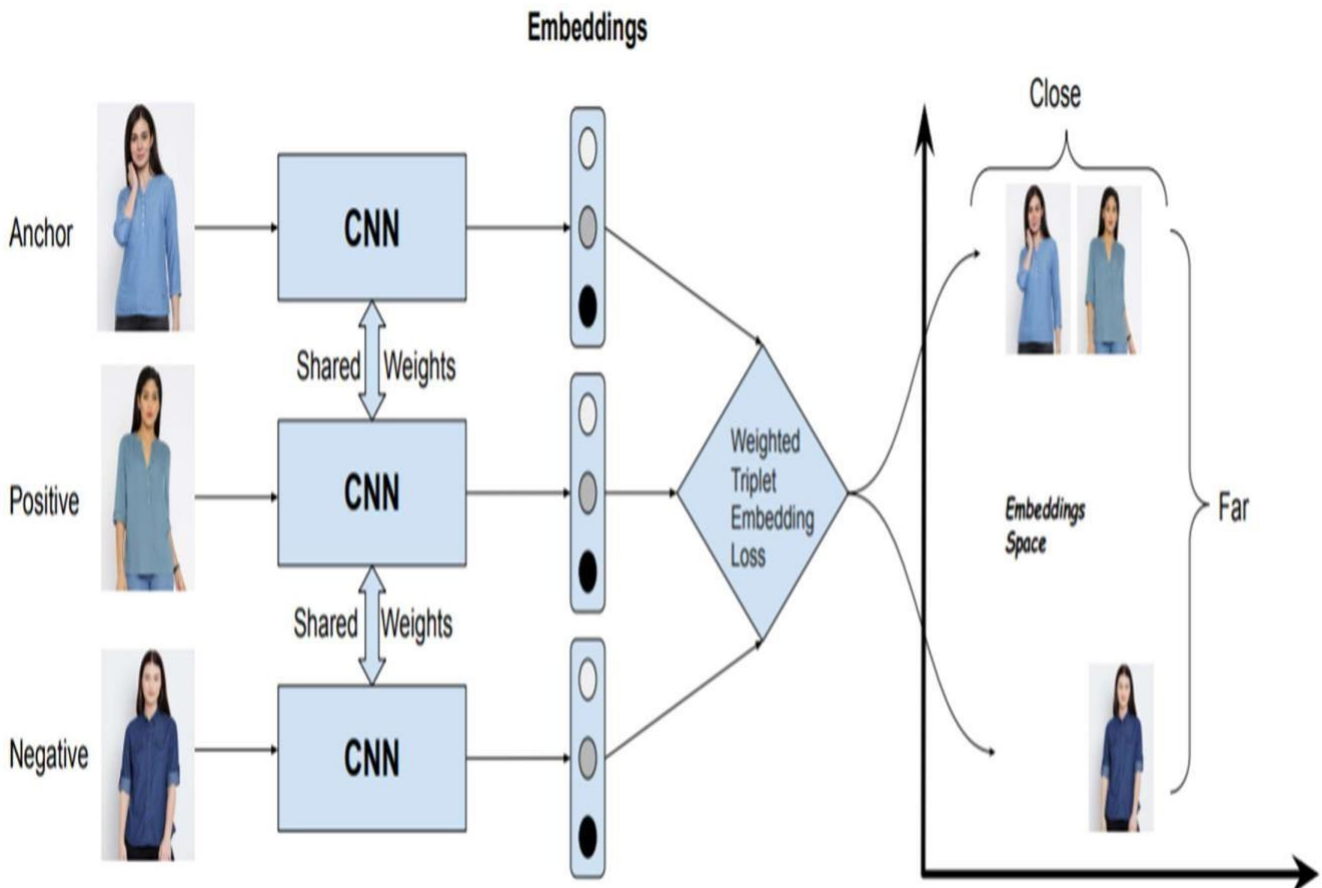
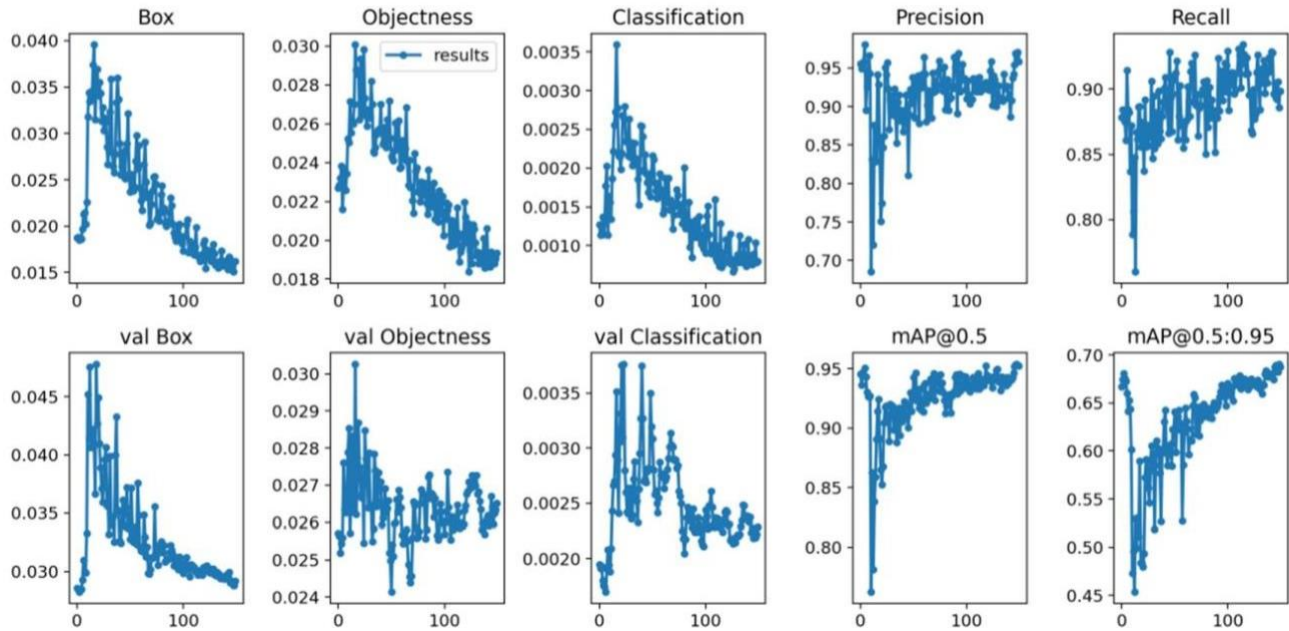


The Mean Average Precision is calculated by taking mean of the Average Precision values over different values of threshold, for example in the COCO Primary Challenge the mAP was calculated by averaging the AP scores over a range of IoU thresholds from 0.5 to 0.95 with a step size of 0.05 and finally taking the mean of the AP scores over all the classes. For more detailed explanation of mAP follow this blog - "Breaking Down Mean Average Precision". Now that we know how Object Detection Systems are evaluated let's checkout various methods for object detection. We'll compare various object detection model's performance against the MS COCO Dataset that contains 80 classes. We'll look at the Inference speed and mAP scores. Below is a plot showing the highest and lowest Frames Per Second (FPS) values reported in their respective papers.

```

from utils.plots import plot_results
plot_results(save_dir='runs/train/exp5') # plot all results*.txt as results.png
Image(filename='runs/train/exp5/results.png', width=800)

```



5.3 USER STORIES:

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can access my data by login	High	Sprint-1
	Dashboard	USN-6	As a user, I can view the dashboard and by products		High	Sprint -2
Customer (Web user)	Registration / Login	USN-7	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard		Sprint -1
Customer Care Executive	Contact with Customers	USN-8	As a Customer care executive, I solve the customer Requirements and feedback	I can receive calls from customers	High	Sprint-1
Administrator	Check stock and Price , orders	USN_9	As a Administrator , I can Check the database And stock details and buying and selling prices	I am the administrator of the company	High	Sprint -2

6. PROJECT PLANNING & SCHEDULE

6.1 SPRINT PLANNING & ESTIMATION:

Milestones	Activities	Description
Project Development Phase	Delivery of Sprint – 1,2,3,4	To develop the code and submit the developed code by testing it
Setting up App environment	Create IBM Cloud account	Signup for an IBM Cloud account
	Create flask project	Getting started with Flask to create project
	Install IBM Cloud CLI	Install IBM Command LineInterface
	Docker CLI Installation	Installing Docker CLI on laptop
	Create an account in send grid	Create an account in sendgrid. Use the service as email integration to our application for sending emails
Implementing web Application	Create UI to interact with Application	Create UI <ul style="list-style-type: none"> • Registration page • Login page • View products page • Add products page
	Create IBM DB2 & connect with python	Create IBM DB2 service in IBM Cloud and connect with python code with DB
Integrating sendgrid service	Sendgrid integration with python	To send emails form the application we need to integrate the Sendgrid service
Developing a chatbot	Building a chatbot and Integrate to application	Build the chatbot and Integrate it to the flask application
Deployment of App in IBMCloud	Containerize the App	Create a docker image of your application and push it to the IBM container registry
	Upload image to IBM container registry	Upload the image to IBM container registry
	Deploy in kubernetes cluster	Once the image is uploaded to IBM Container registry deploy the image to IBM Kubernetes cluster
Ideation Phase	Literature Survey	Literature survey on the selected project & information gathering
	Empathy Map	Prepare Empathy map to capture the user Panis & Gains, prepare list of problem statement
	Ideation	Organizing the brainstorming session and priorities the top 3 ideas based on feasibility & Importance
Project Design Phase I	Proposed Solution	Prepare proposed solution document which includes novelty, feasibility of ideas, business model, social impact, Scalability of solution
	Problem Solution Fit	Prepare problem solution fit document
	Solution Architecture	Prepare solution architecture document
Project Design Phase II	Customer Journey	Prepare customer journey map to understand the user interactions & experience with the application
	Functional requirement	Prepare functional & non functional requirement document
	Data Flow Diagram	Prepare Data Flow Diagramand user stories
	Technology architecture	Draw the technology architecture diagram
Project Planning Phase	Milestones & Activity list	Prepare milestones and activity list of the project
	Sprint Delivery Plan	Prepare sprint delivery plan

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story points	Priority	Team Members
Sprint-1	Setting up App environment	USN-1	As a user, I can register in ICTA Academy and create IBM cloud account.	2	High	Mos Richard E Saran N
Sprint-1		USN-2	As a user, I will create a flask project	1	Low	Kathiravan S Karan V
Sprint-1		USN-3	As a user, I will install IBM Cloud CLI	2	Medium	Karan V Saran N
Sprint-2	Setting up App environment	USN-4	As a user, I can install Docker CLI	1	Low	Saran N Mos Richard E
Sprint-2		USN-5	As a user, I will Create an account in sendgrid	2	Medium	Mos Richard E Karan V

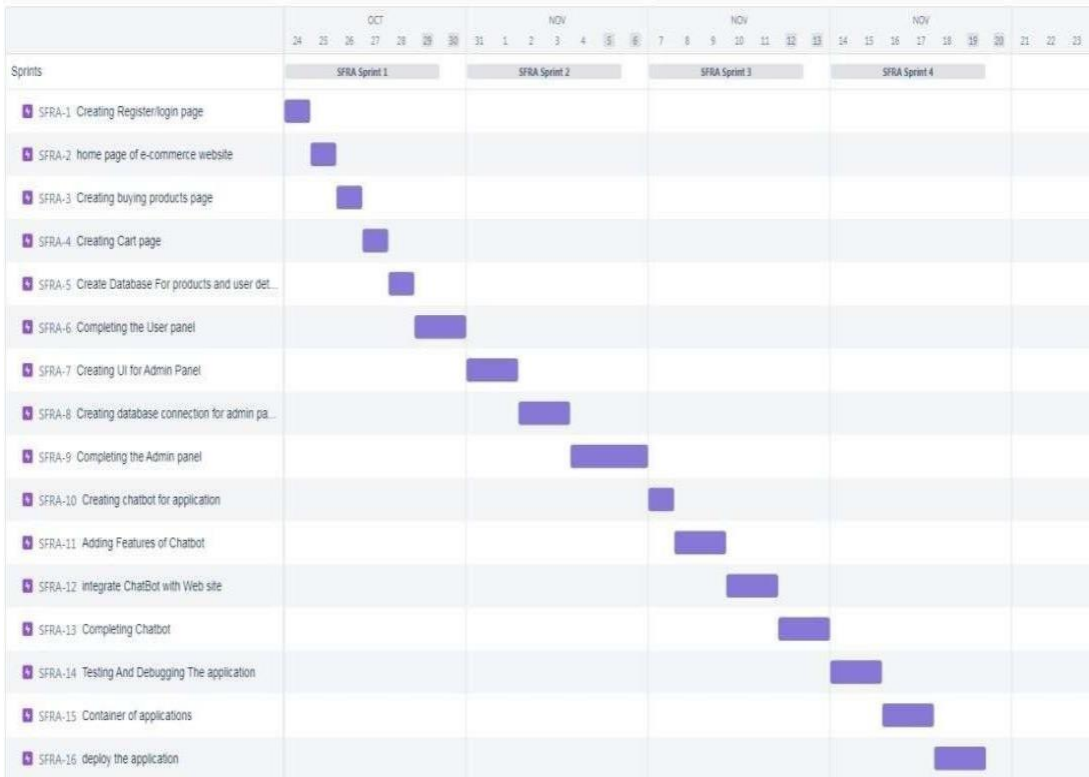
Sprint-3	Implementing web application	USN-6	As a user, I Create UI to interact with the application	1	High	Kathiravan S Saran N
Sprint-3		USN-7	As a user, I Create IBM DB2 and connect with Python	3	High	Karan V
Sprint-3	Integrating Sendgrid Service	USN-8	As a user, I will integrating sendgrid with python code	2	High	Kathiravan S
Sprint-3	Developing a chatbot	USN-9	As a user, I have to build a chatbot and Integrate to application	1	Medium	Mos Richard E
Sprint-4	Development of App in IBM Cloud	USN-10	As a user, I will Containerize the App	1	Low	Karan V
Sprint-4		USN-11	As a user, I will upload image to IBM Container registry	2	Medium	Saran N
Sprint-4		USN-12	As a user, I will deploy App in Kubernetes cluster	3	High	Mos Richard E
Sprint-4	User panel		As a user <ul style="list-style-type: none"> • Register, Login, Email, Verification • Manual Search • Order placement, Order Details 	3	High	Kathiravan S Karan V Saran N Mos Richard E

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	5 Days	19 Nov 2022	24 Nov 2022	20	24 Nov 2022
Sprint-2	20	5 Days	19 Nov 2022	24 Nov 2022	20	24 Nov 2022
Sprint-3	20	5 Days	19 Nov 2022	24 Nov 2022	20	24 Nov 2022
Sprint-4	20	5 Days	19 Nov 2022	24 Nov 2022	20	24 Nov 2022

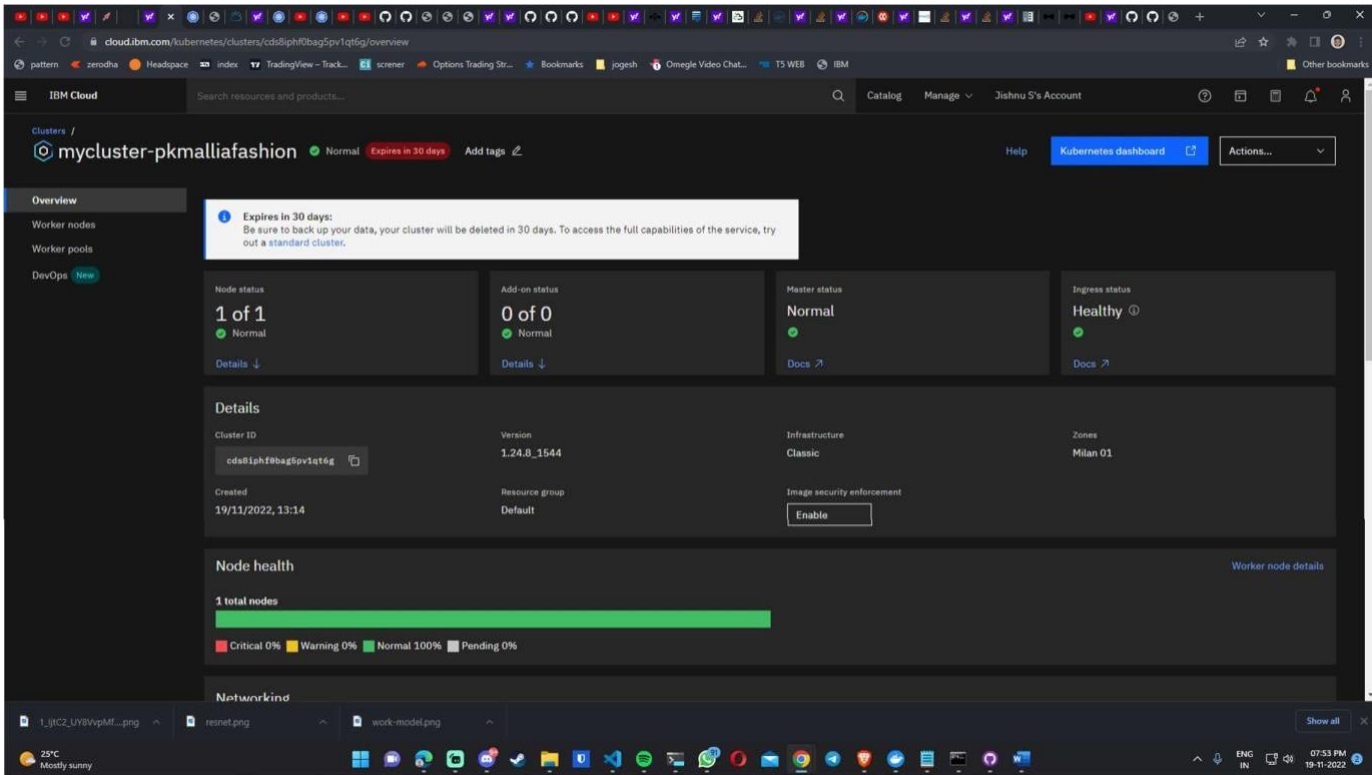
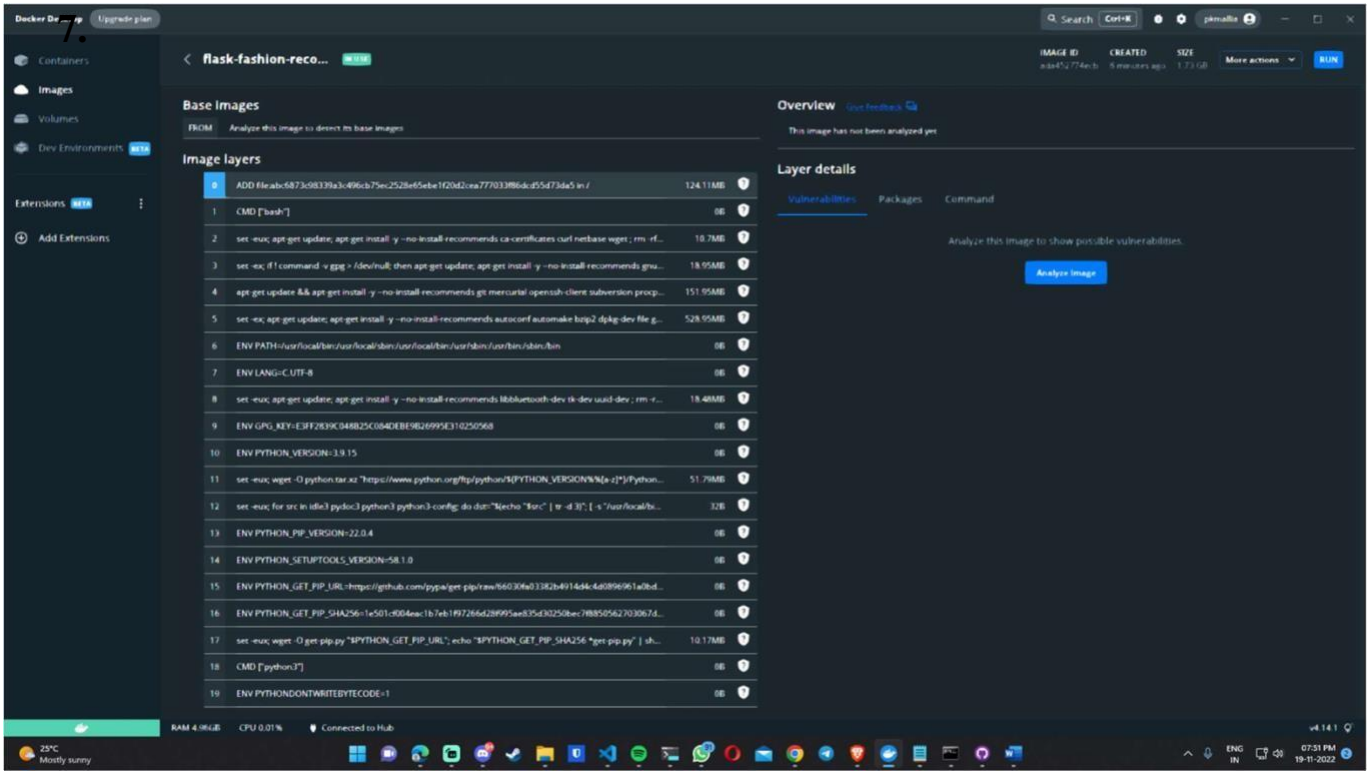
6.3 REPORTS FROM JIRA:

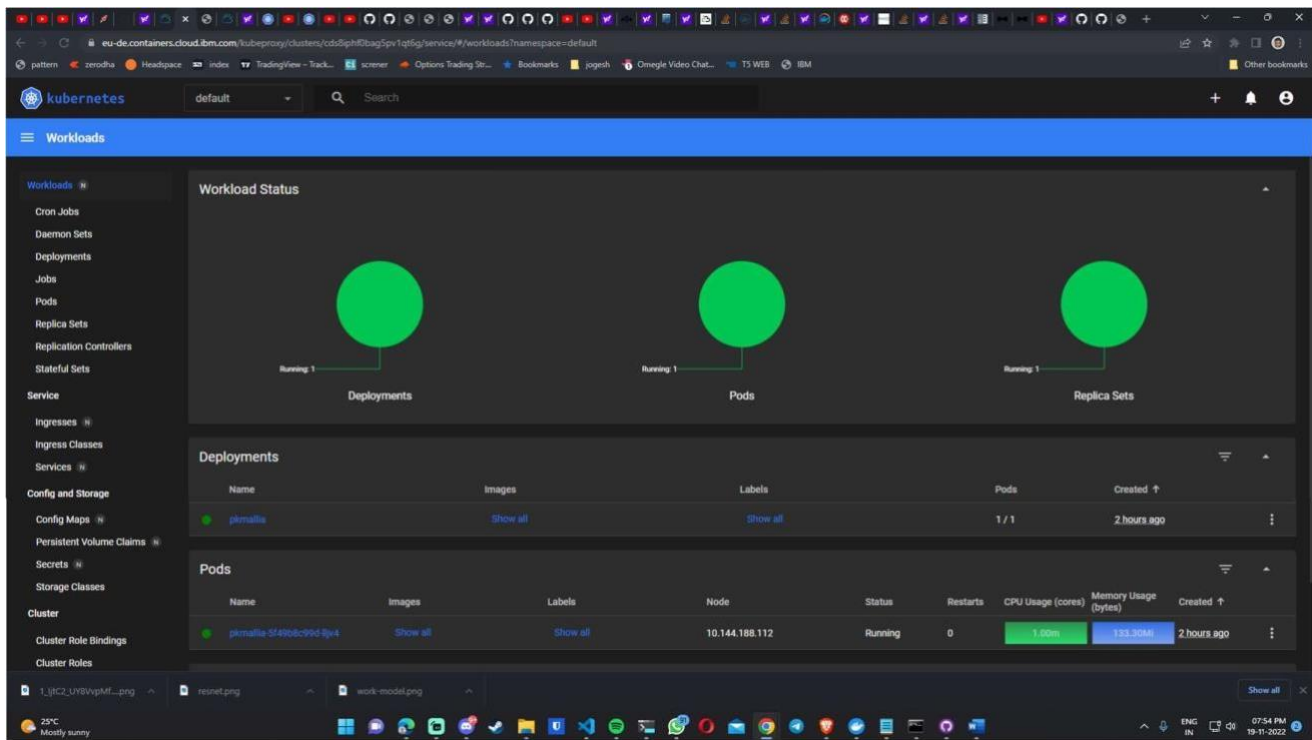
Burndown Chart:



CODING & SOLUTIONING

FEATURE 1:





7.1 homepage.html

```
{% extends 'layouts/base-fullscreen.html' %}

{% block title %} Sign IN {% endblock title %}

<!-- Specific CSS goes HERE -->
{% block stylesheets %}{% endblock stylesheets %}

{% block body_class %} sign-in-illustration {% endblock body_class %}

{% block content %}

{% include "includes/navigation-auth.html" %}

<section>
  <div class="page-header section-height-100">
    <div class="container">
      <div class="row">
        <div class="col-xl-4 col-lg-5 col-md-7 d-flex flex-column mx-lg-0 mx-auto">
          <div class="card card-plain">
            <div class="card-header pb-0 text-left">
              <h4 class="font-weight-bolder">Sign In</h4>
              <p class="mb-0">
                {% if msg %}
                  {{ msg | safe }}
                {% else %}
                  Add your credentials
                {% endif %}
              </p>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>
```



```

<div class="card-body">

    <form method="post" action="" role="form">

        {{ form.hidden_tag() }}

        <div class="mb-3">
            {{ form.username(class="form-control form-control-lg",
placeholder="Username") }}
        </div>
        <div class="mb-3">
            {{ form.password(class="form-control form-control-lg",
placeholder="Password", type="password") }}
        </div>
        <div class="form-check form-switch">
            <input class="form-check-input" type="checkbox" id="rememberMe">
            <label class="form-check-label" for="rememberMe">Remember me</label>
        </div>
        <div class="text-center">
            <button type="submit" name="login"
                class="btn btn-lg bg-gradient-primary btn-lg w-100 mt-4 mb-
0">Sign in</button>
        </div>
    </form>
</div>
<div class="card-footer text-center pt-0 px-lg-2 px-1">
    <p class="mb-4 text-sm mx-auto">
        Don't have an account?
        <a href="{{ url_for('authentication_blueprint.register') }}" class="text-
primary text-gradient font-weight-bold">Sign UP</a>
    </p>
</div>
</div>
<div class="col-6 d-lg-flex d-none h-100 my-auto pe-0 position-absolute top-0 end-0
text-center justify-content-center flex-column">
    <div class="position-relative bg-gradient-primary h-100 m-3 px-7 border-radius-lg
d-flex flex-column justify-content-center">
        
        <div class="position-relative">
            
        </div>
        <h4 class="mt-5 text-white font-weight-bolder">
            SMART FASHION RECOMMENDER
        </h4>
        <p class="text-white">
            &copy; <a target="_blank" class="text-white" href="https://bit.ly/3fKQZaL"
target="_blank">Vel Tech High Tech</a>

```

```

        - Coded by <a class="text-white" target="_blank" href="https://appseed.us"
target="_blank">Haritha, Niranjana, Krishika, Juneha</a>.
    </p>
  </div>
</div>
</div>
</div>
</div>
</div>
</section>

{% endblock content %}

<!-- Specific JS goes HERE -->
{% block javascripts %}

  <script src="/static/assets/js/soft-design-system.min.js?v=1.0.1"
type="text/javascript"></script>

{% endblock javascripts %}

{% extends 'layouts/base-fullscreen.html' %}

{% block title %} Sign UP {% endblock title %}

<!-- Specific CSS goes HERE -->
{% block stylesheets %}{% endblock stylesheets %}

{% block body_class %} sign-in-illustration {% endblock body_class %}

{% block content %}

  {% include "includes/navigation-auth.html" %}

  <section>
    <div class="page-header section-height-100">
      <div class="container">
        <div class="row">
          <div class="col-xl-4 col-lg-5 col-md-7 d-flex flex-column mx-lg-0 mx-auto">
            <div class="card card-plain">
              <div class="card-header pb-0 text-left">
                <h4 class="font-weight-bolder">Sign UP</h4>
                <p class="mb-0">
                  {% if msg %}
                    {{ msg | safe }}
                  {% else %}
                    Add your credentials
                  {% endif %}
                </p>
              </div>
            <div class="card-body">

```

```

        <form method="post" action="" role="form">

            {{ form.hidden_tag() }}

            <div class="mb-3">
                {{ form.username(class="form-control form-control-lg",
placeholder="Username") }}
            </div>
            <div class="mb-3">
                {{ form.email(class="form-control form-control-lg", placeholder="Email",
type="email") }}
            </div>
            <div class="mb-3">
                {{ form.password(class="form-control form-control-lg",
placeholder="Password", type="password") }}
            </div>
            <div class="form-check form-switch">
                <input class="form-check-input" type="checkbox" id="rememberMe">
                <label class="form-check-label" for="rememberMe">
                    Agree with <a href="#">terms</a>
                </label>
            </div>
            <div class="text-center">
                <button type="submit" name="register"
                    class="btn btn-lg bg-gradient-primary btn-lg w-100 mt-4 mb-
0">Register</button>
            </div>
        </form>
    </div>
    <div class="card-footer text-center pt-0 px-lg-2 px-1">
        <p class="mb-4 text-sm mx-auto">
            Have an account?
            <a href="{{ url_for('authentication_blueprint.login') }}" class="text-
primary text-gradient font-weight-bold">Sign IN</a>
        </p>
    </div>
</div>
</div>
<div class="col-6 d-lg-flex d-none h-100 my-auto pe-0 position-absolute top-0 end-0
text-center justify-content-center flex-column">
    <div class="position-relative bg-gradient-primary h-100 m-3 px-7 border-radius-lg
d-flex flex-column justify-content-center">
        
        <div class="position-relative">
            
        </div>
        <h4 class="mt-5 text-white font-weight-bolder">
            SMART FASHION RECOMMENDER

```

```

        </h4>
        <p class="text-white">
            &copy; <a target="_blank" class="text-white" href="https://bit.ly/3fKQZaL"
target="_blank">QBIT STUDIOS</a>
            - Coded by <a class="text-white" target="_blank" href="https://appseed.us"
target="_blank">PKMALLIA</a>.
        </p>
    </div>
</div>
</div>
</div>
</div>
</div>
</section>

{% endblock content %}

<!-- Specific JS goes HERE -->
{% block javascripts %}

    <script src="/static/assets/js/soft-design-system.min.js?v=1.0.1"
type="text/javascript"></script>

{% endblock javascripts %}

```

FEATURE 2:

7.2 finalhome.html:

```
{% extends 'layouts/base-presentation.html' %}

{% block title %} Presentation {% endblock title %}

<!-- Specific CSS goes HERE -->
{% block stylesheets %}{% endblock stylesheets %}

{% block body_class %} index-page {% endblock body_class %}

{% block content %}

    <header class="header-2">
        <div class="page-header section-height-75 relative" style="background-image:
url('/static/assets/img/curved-images/curved.jpg')">
            <div class="container">
                <div class="row">
                    <div class="col-lg-7 text-center mx-auto">
                        <h1 class="text-white pt-3 mt-n5">
                            <a class="text-white" target="_blank"
                                href="">Job Recommender</a>
                        </h1>
                        <p class="lead text-white mt-3">
                            recommends using closest near neighbours
                        <br />
                        <a class="text-white" target="_blank" href="" target="_blank"></a>.
                    </p>
                </div>
            </div>
        </div>
        <div class="position-absolute w-100 z-index-1 bottom-0">
            <svg class="waves" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 24 150 40" preserveAspectRatio="none"
shape-rendering="auto">
                <defs>
                    <path id="gentle-wave" d="M-160 44c30 0 58-18 88-18s 58 18 88 18 58-18 88-18 58
18 88 18 v44h-352z" />
                </defs>
                <g class="moving-waves">
                    <use xlink:href="#gentle-wave" x="48" y="-1" fill="rgba(255,255,255,0.40)" />
                    <use xlink:href="#gentle-wave" x="48" y="3" fill="rgba(255,255,255,0.35)" />
                    <use xlink:href="#gentle-wave" x="48" y="5" fill="rgba(255,255,255,0.25)" />
                    <use xlink:href="#gentle-wave" x="48" y="8" fill="rgba(255,255,255,0.20)" />
                    <use xlink:href="#gentle-wave" x="48" y="13" fill="rgba(255,255,255,0.15)" />
                    <use xlink:href="#gentle-wave" x="48" y="16" fill="rgba(255,255,255,0.95)" />
                </g>
            </svg>
        </div>
    </div>
```

```

    </div>
  </div>
</header>

<script>
  window.watsonAssistantChatOptions = {
    integrationID: "0bb96b92-4e98-44c7-9dab-3a5fe2ff8562", // The ID of this integration.
    region: "au-syd", // The region your integration is hosted in.
    serviceInstanceID: "e5babddc-2ad5-4eac-a0c5-6dad126622cb", // The ID of your service
instance.
    onLoad: function(instance) { instance.render(); }
  };
  setTimeout(function(){
    const t=document.createElement('script');
    t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>

{% for items in product %}

<section class="my-5 py-5">
  <div class="container">
    <div class="row align-items-center">
      <div class="col-lg-4 ms-auto me-auto p-lg-4 mt-lg-0 mt-4">
        <div class="card card-background " data-tilt>
          <div class="full-background" style="background-image:
url('https://images.unsplash.com/photo-1567095761054-7a02e69e5c43?ixlib=rb-
4.0.3&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=format&fit=crop&w=687&q=80')"></
div>
          <div class="card-body pt-7 text-center">
            <h2 class="text-white up mb-0">{{product[items].Position}} <br />
{{product[items].Company}} <br> {{product[items].location}}</h2>
            <a href="{{product[items].url}}" class="btn btn-outline-white mt-5 up btn-
round">Apply</a>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>

{% endfor%}

{% endblock content %}

<!-- Specific JS goes HERE -->
{% block javascripts %}

```

```

<script src="/static/assets/js/plugins/countup.min.js"></script>
<script src="/static/assets/js/plugins/choices.min.js"></script>
<script src="/static/assets/js/plugins/rellax.min.js"></script>
<script src="/static/assets/js/plugins/tilt.min.js"></script>
<script src="/static/assets/js/plugins/choices.min.js"></script>

<script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDTTfWur0PDbZWPr7Pmq8K3jiDp0_xUziI"></s
cript>
<script src="/static/assets/js/soft-design-system.min.js?v=1.0.1"
type="text/javascript"></script>
<script type="text/javascript">
  if (document.getElementById('state1')) {
    const countUp = new CountUp('state1',
document.getElementById("state1").getAttribute("countTo"));
    if (!countUp.error) {
      countUp.start();
    } else {
      console.error(countUp.error);
    }
  }
  if (document.getElementById('state2')) {
    const countUp1 = new CountUp('state2',
document.getElementById("state2").getAttribute("countTo"));
    if (!countUp1.error) {
      countUp1.start();
    } else {
      console.error(countUp1.error);
    }
  }
  if (document.getElementById('state3')) {
    const countUp2 = new CountUp('state3',
document.getElementById("state3").getAttribute("countTo"));
    if (!countUp2.error) {
      countUp2.start();
    } else {
      console.error(countUp2.error);
    }
  }
</script>

{% endblock javascripts %}

```

7.3 DATABASE SCHEMA:

```
from flask import Flask
from flask_login import LoginManager
from flask_sqlalchemy import SQLAlchemy
from importlib import import_module

db = SQLAlchemy()
login_manager = LoginManager()

def register_extensions(app):
    db.init_app(app)
    login_manager.init_app(app)

def register_blueprints(app):
    for module_name in ('authentication', 'home'):
        module = import_module('apps.{0}.routes'.format(module_name))
        app.register_blueprint(module.blueprint)

def configure_database(app):

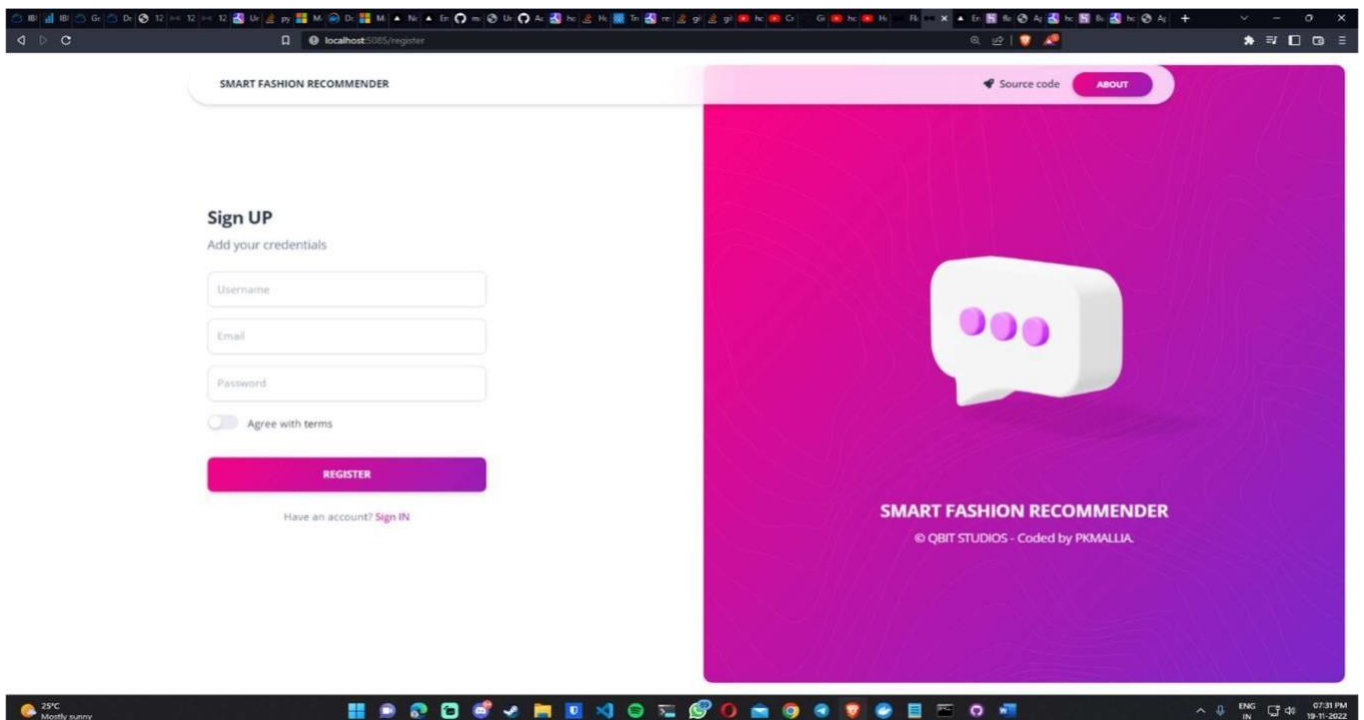
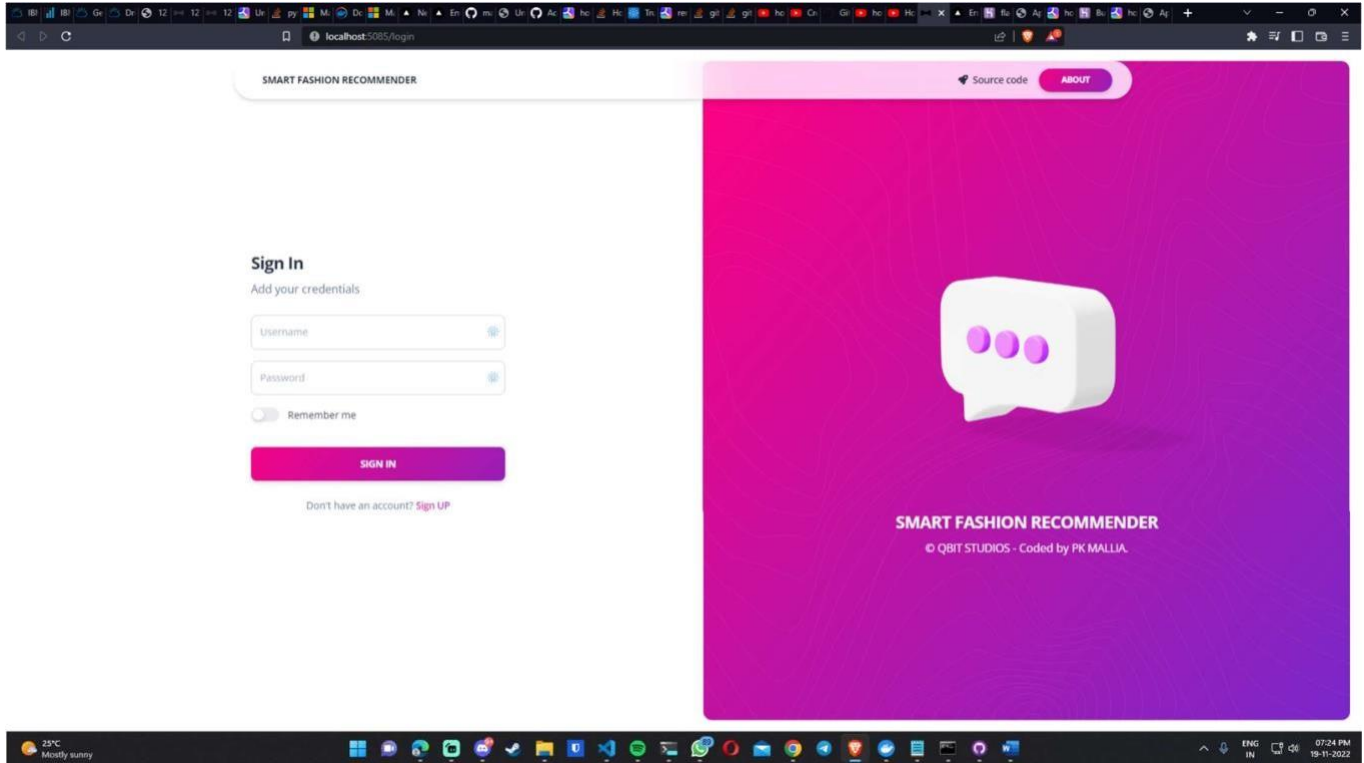
    @app.before_first_request
    def initialize_database():
        db.create_all()

    @app.teardown_request
    def shutdown_session(exception=None):
        db.session.remove()

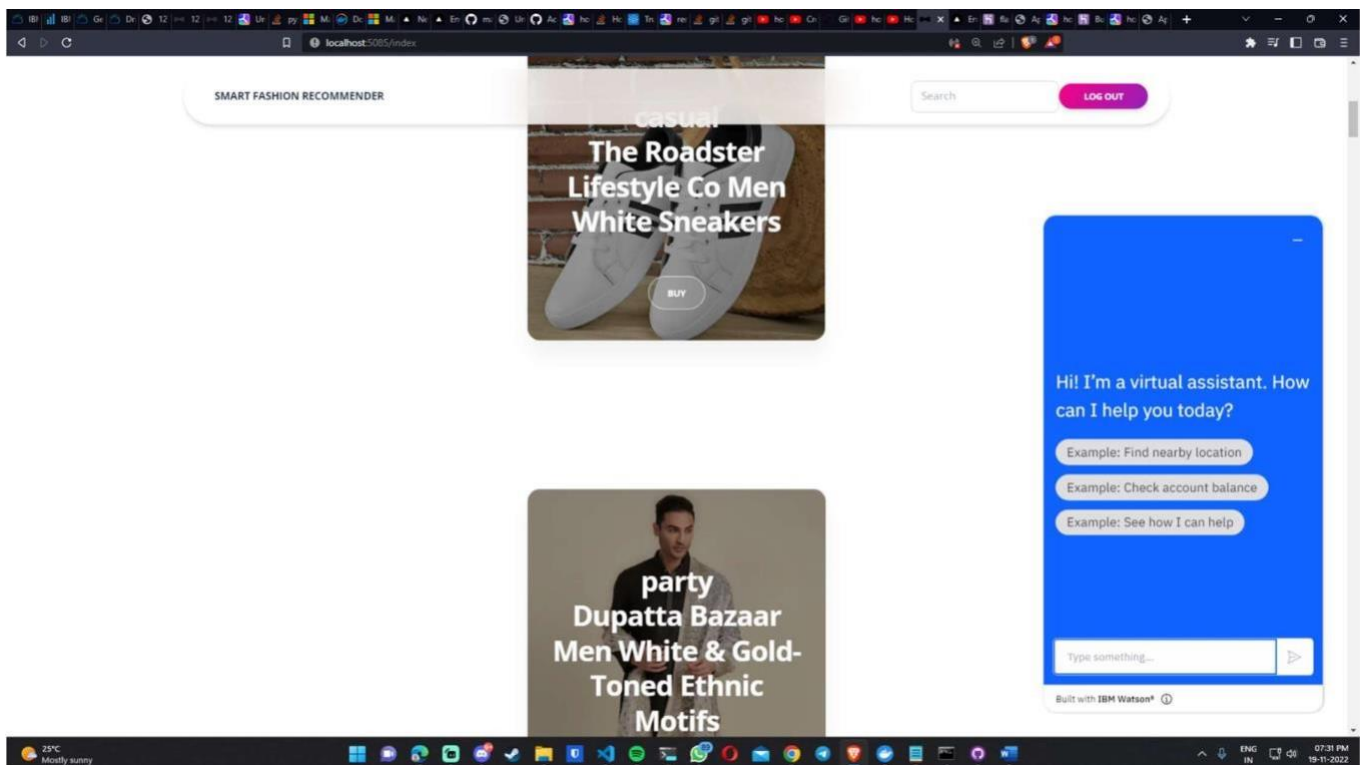
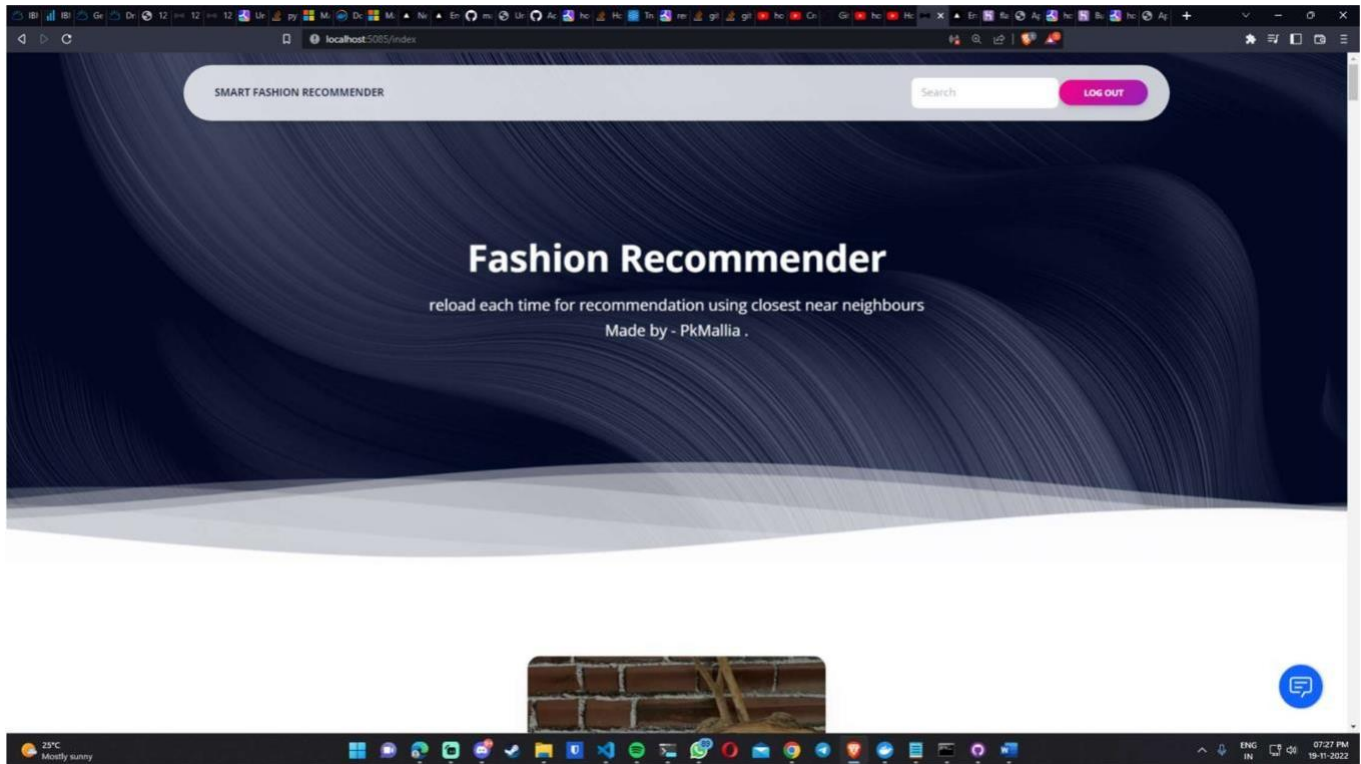
def create_app(config):
    app = Flask(__name__)
    app.config.from_object(config)
    register_extensions(app)
    register_blueprints(app)
    configure_database(app)
    return app
```


8. TESTING

8.1 TEST CASES



8.2 USER ACCEPTANCE TESTING



9. RESULTS

PERFORMANCE METRICS:

The performance of a recommendation algorithm is evaluated by using some specific metrics that indicate the accuracy of the system. The type of metric used depends on the type of filtering technique. Root Mean Square Error (RMSE), Receiver Operating Characteristics (ROC), Area Under Cover (AUC), Precision, Recall and F1 score is generally used to evaluate the performance or accuracy of the recommendation algorithms.

Root-mean square error (RMSE). RMSE is widely used in evaluating and comparing the performance of a recommendation system model compared to other models. A lower RMSE value indicates higher performance by the recommendation model. RMSE, as mentioned by, can be as represented as follows:

$$RMSE = \sqrt{\frac{1}{N_p} \sum_{u,i} (p_{ui} - r_{ui})^2} \quad (1)$$

where, N_p is the total number of predictions, p_{ui} is the predicted rating that a user u will select an item i and r_{ui} is the real rating.

Precision. Precision can be defined as the fraction of correct recommendations or predictions (known as True Positive) to the total number of recommendations provided, which can be as represented as follows:

$$Precision = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive\ (FP)} \quad (2)$$

It is also defined as the ratio of the number of relevant recommended items to the number of recommended items expressed as percentages.

Recall. Recall can be defined as the fraction of correct recommendations or predictions (known as True Positive) to the total number of correct relevant recommendations provided, which can be as represented as follows:

$$Recall = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)} \quad (3)$$

It is also defined as the ratio of the number of relevant recommended items to the total number of relevant items expressed as percentages.

F1 Score. F1 score is an indicator of the accuracy of the model and ranges from 0 to 1, where a value close to 1 represents higher recommendation or prediction accuracy. It represents precision and recall as a single metric and can be as represented as follows:

$$F1\ score = 2 \times \frac{Precision * Recall}{Precision + Recall} \quad (4)$$

Coverage. Coverage is used to measure the percentage of items which are recommended by the algorithm among all of the items.

Accuracy. Accuracy can be defined as the ratio of the number of total correct recommendations to the total recommendations provided.

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- Runs on IBM Kubernetes cluster
- Recommends top of the line products
- Smart fashion recommender application is the user friendly.
- With the help of chatbot user can find the products very easily.
- This application used to discover the product based on the user's choice, very easily and quickly.
- It has the ability to reduce transaction costs for consumers, and increase revenue for retailers.

DISADVANTAGES:

- It needs active internet connection.
- Privacy concerns.
- Too many choices.
- Cold-start problem.



11. CONCLUSION

The Fashion Recommendation System is mainly used to recommend the best possible outfit combinations to a user who has no fashion sense based on their wardrobe . It may not always provide the best possible outfit to wear for an occasion as the system is dependent completely on the clothes present in the user's wardrobe. Also another reason is that fashion is highly dependent on the time period. However the system does a great job in inculcating a fashion sense among the users and can provide the best recommendations based on the user's wardrobe. Since the system is implemented as a website, it is very easy for the end users to access as well as use. The scope of this system can be expanded by including the ability to detect the various design and patterns on clothing, and to increase the number of occasions.

12. FUTURE SCOPE

In the future, to implement this recommendation system to be extended to include male and non-binary fashion items including apparel, footwear, accessories etc. This work can further be enhanced to predict fashion items based on the skin colour and weather conditions.

Future research should concentrate on including time series analysis and accurate categorization of product images based on the variation in colour, trend and clothing style in order to develop an effective recommendation system. The proposed model will follow brand-specific personalization campaigns and hence it will ensure highly curate and tailored offerings for users. Hence, this research will be highly beneficial for researchers interested in using augmented and virtual reality features to develop recommendation systems.

13. APPENDIX

SOURCE CODE:

Flask app.py:

```
from apps.home import blueprint
from flask import render_template, request
from flask_login import login_required
from jinja2 import TemplateNotFound
import pandas as pd
import numpy as np
from ftfy import fix_text
from nltk.corpus import stopwords
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neighbors import NearestNeighbors
import re

@blueprint.route('/index')
@login_required
def index():
    with open('/job_final.csv', encoding="utf-8") as f:
        df1 = pd.read_csv(f)
        df1.to_string()
        df2 = df1.sample(25)
        df3 = df2.to_dict('index')
        print(df3)
        print(type(df3))

    return render_template('home/index.html', product = df3, segment='index')

# @blueprint.route('/search', methods=['GET', 'POST'])
# def recommender():
#     with open('G:\\Flask-Fashion-Recommender\\flask-pixel\\myntra-database.csv') as f1:
#         stopw = set(stopwords.words('english'))
#         df11 = pd.read_csv(f1)
#         df11['test'] = df11['gender', 'category', 'type', 'brand', 'description'].apply(lambda x:
#             ' '.join([word for word in str(x).split() if len(word) > 2 and word not in (stopw)]))

#     def ngrams(string, n=3):
#         string = fix_text(string) # fix text
#         string = string.encode("ascii", errors="ignore").decode() # remove non ascii
#         string = string.lower()
#         chars_to_remove = [")", "(", ".", "|", "[", "]", "{", "}", "'", '"']
#         rx = '[' + re.escape(''.join(chars_to_remove)) + ']'
```

```

#         string = re.sub(rx, '', string)
#         string = string.replace('&', 'and')
#         string = string.replace(',', ' ')
#         string = string.replace('-', ' ')
#         string = string.title() # normalise case - capital at start of each word
#         string = re.sub(' +', ' ', string).strip() # get rid of multiple spaces and
replace with a single
#         string = ' ' + string + ' ' # pad names for ngrams...
#         string = re.sub(r'[,.-/]|\\sBD', r'', string)
#         ngrams = zip(*[string[i:] for i in range(n)])
#         return [''.join(gram) for gram in ngrams]

#         vectorizer = TfidfVectorizer(min_df=1, analyzer=ngrams, lowercase=False)
#         tfidf = vectorizer.fit_transform(org_name_clean)

#         def getNearestN(query):
#             queryTFIDF_ = vectorizer.transform(query)
#             distances, indices = nbrs.kneighbors(queryTFIDF_)
#             return distances, indices

#         nbrs = NearestNeighbors(n_neighbors=1, n_jobs=-1).fit(tfidf)
#         unique_org = (df11['test'].values)
#         distances, indices = getNearestN(unique_org)
#         unique_org = list(unique_org)
#         matches = []
#         for i,j in enumerate(indices):
#             dist=round(distances[i][0],2)

#             temp = [dist]
#             matches.append(temp)
#         matches = pd.DataFrame(matches, columns=['Match confidence'])
#         df11['match']=matches['Match confidence']
#         df11=df11.sort_values('match')
#         df22=df11[['Product_url',
'image_url','type','category','description','brand']].head(30).reset_index()
#         return render_template('includes/card-each.html', sproduct = df22, segment='index')

@blueprint.route('/<template>')
@login_required
def route_template(template):

    try:

        if not template.endswith('.html'):
            template += '.html'

        # Detect the current page

```

```

        segment = get_segment(request)

        # Serve the file (if exists) from app/templates/home/FILE.html
        return render_template("home/" + template, segment=segment)

    except TemplateNotFound:
        return render_template('home/page-404.html'), 404

    except:
        return render_template('home/page-500.html'), 500

# Helper - Extract current page name from request
def get_segment(request):

    try:

        segment = request.path.split('/')[-1]

        if segment == '':
            segment = 'index'

        return segment

    except:
        return None

```


INTEGRATING APPLICATION WITH CHATBOT USING WATSON ASSISTANT

CODE :

```
<script>
  window.watsonAssistantChatOptions = {
    integrationID: "614a4315-ff80-4187-8fe4-2fd9b506b723", // The ID of this integration.
    region: "au-syd", // The region your integration is hosted in.
    serviceInstanceID: "9670dcf8-789f-4609-8d7a-6e25c412a9ec", // The ID of your
    service instance.
    onLoad: function(instance) { instance.render(); }
  };
  setTimeout(function(){
    const t=document.createElement('script');
    t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
    (window.watsonAssistantChatOptions.clientVersion || 'latest') +
    "/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>
```

GITHUB & PROJECT DEMO LINK

GITHUB LINK:

<https://github.com/IBM-EPBL/IBM->

[Project-54564-1662271812](https://github.com/IBM-EPBL/IBM-Project-54564-1662271812)

VIDEO LINK:

<https://drive.google.com/file/d/1vJZnfGPiI2b5hjlynW7RSd6I1g3bb>

[OhG/view?usp=share link](https://drive.google.com/file/d/1vJZnfGPiI2b5hjlynW7RSd6I1g3bb/view?usp=share_link)

