VIETNAM NATIONAL UNIVERSITY OF

HO CHI MINH CITY

THE INTERNATIONAL UNIVERSITY

School of Computer Science & Engineering

THESIS REPORT

FASHION PRODUCT RECOMMENDATION PLATFORM WITH FREQUENT ITEMSET MINING

by

Nguyễn Giang Ngọc Trâm - ITITIU18158

Advisor: Assoc. Prof. Nguyễn Thị Thúy Loan

FASHION PRODUCT RECOMMENDATION PLATFORM WITH FREQUENT ITEMSET MINING

APPROVED BY:

__________________________________ ,

Assoc. Prof. Nguyen Thi Thuy Loan

THESIS COMMITTEE

(Whichever applies)

A thesis submitted to the School of Computer Science and Engineering

in partial fulfillment of the requirements for the degree of

Bachelor of Information Technology/Computer Science/Computer Engineering

Ho Chi Minh City, Vietnam

2020

ACKNOWLEDGEMENT

First and foremost, I sincerely express our deep gratitude to my advisor Assoc. Prof. Nguyen Thi Thuy Loan for accepting to be the pre-thesis and thesis's instructor, accompanying to find out the suitable thesis topic, lead me to the right path for my research in next semester and giving me unwavering guidance on planning, managing the work, preparing for necessary knowledge for graduation as well as guiding me mentally and technically. Thanks for making massive support, always encouraging me to explore my fullest potential to complete the work in the most efficient way in the process of doing my thesis. Thank the Bay-Loan seminar team for supporting me a lot.

I would love to express my sincerest thanks to the International University teachers, especially the Computer Science and Engineering Department teachers, who gave me a great background of technical knowledge and practical skills to apply in the project. In this thesis, I have involved the knowledge that we gain from the Principle of Database management of Assoc. Prof. Thuy Loan, Information System Management of MSc. Hoang Chau and solid foundation knowledge of the Web Application Development course which is taught by Dr. Nguyen Van Sinh.

Finally, I would like to show appreciation to my family, all of my friends, who have facilitated me to make efforts as much as possible to complete the thesis report successfully and on time.

TABLE OF CONTENT

LIST OF ABBREVIATIONS

ERD	Entity Relationship Diagram
API	Application Programming Interface
DAM	Dynamic Asset Management
REST	Representational State Transfer
JSON	JavaScript Object Notation
APR	Automatic Product Recommendation
CPS	Conditional Pattern Base
DOM	Document Object Model
ERD	Entity Relationship Diagram
I/O	Input / Output
SPA	Single Page Application
UI	User Interface

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The thesis proposes a fashion product recommendation system presented simulated in the form of on a web application platform with many other features such as post clothing item to sell, to share style between users like a social media, to visualize how a product looks like when mixed with other items, to save that collection for a later occasion in Virtual Look section of the website.

In this work, I focus on building up a social media website with a recommendation model run below the web application system by using Frequent itemset mining, focusing on the information of the product to combine and generate a clothing collection for creating an association rule. I describe an implementation of this framework on a JSON server, and user interface as a Web Application to demonstrate this clothing recommendation system. Lastly, I discuss the approaches to applying Frequent itemset in a dataset by trying different algorithms to find out which is the most suitable for the model, determine the proposed quality of the system with a Confidence value.

CHAPTER 1. INTRODUCTION

Fashion has a massive influence on our culture. Clothing is the most crucial consideration that individuals should make on a daily basis. People spend a lot of time and effort thinking about their outfits of the day because they desire to appear beautiful, women especially, as seen by the numerous online social media platforms such as Instagram and Facebook, where people can share their fashion photos with the rest of the world.

Furthermore, shopaholic loves to buy a bunch of clothes to try on different styles to make themselves look more appealing. This is evidenced by the growing online clothing sales, which reached 370 billion dollars in the United States by 2017 and 191 billion euros in Europe and over 103 billion in a Chinese e-commerce website named Taobao.com, on a single festival day.

Despite facing the coronavirus pandemic: cutbacks by consumers on inessential items and an increased focus on household essentials, the heat of the fashion industry has not abated and US fashion industry sales are expected to rebound from 2021 onwards. Total sales for 2022 are forecast at $473.42 billion, an 8.3% annual increase, before rising 4.3% more to $494.89 billion in 2023.

Fashionistas love to update fashion trends, catch on trending items every day by consulting the combinations of top stars, influencers in the fashion industry. By understanding the desire of customers, they not only want to experience how clothes when combined, but they also want to be suggested mix n match style, suggest products that many people like, and used in their wardrobe, how other people mix that product to create their outfits, the system has applied frequent itemset mining to suggest items that are often combined together, making it easier for users to prepare clothes every day in a quick way, saving effort and time to create an outfit that captures the trend, which is loved by many people in the fashion industry.

The thesis idea – Fashion Product Recommendation platform with Frequent Itemset Mining was implemented form on ReactJS, Three, Drei, JSON Server, some other JavaScript libraries and especially hinge on the knowledge we gain in the Web Application Development course on applying basic web design, MVC architecture, essential skills in design and develop web-based applications and other knowledge about Machine Learning that I have obtained by myself to create a Clothing fashion Recommend system Web application, to utilize critical and analytic thinking to the planning and apply to both execution and evaluation of the software development process.

1.1 Background

Until now, the world's fashion demand has always had an unpredictable development, from styles to diverse clothing designs. Have you ever wondered how can we try on all the clothes in the closet or the store? How to find the style that suits you best. Can you take advantage of the available fashion accessories?

With the thought of helping to change your fashion style more quickly, make the most of the products and clothes you currently have or can use in the future. I have developed a website that can help you solve these problems.

Initially, I came up with the idea to make it online like Shopee- is one of the e-commerce to see and buy the product from one user to another user. After many discussions and opinions with the advisor, I finalized to build a social network for everyone to share their style, learn from other users on a mix and match tips with the support of a recommendation system to increase the profits of the chain stores on the web, speeding up the sales of a product, users can self-sell their own second-hand clothing products and exchange clothes with other users to earn some source of income to try other fashion products. In order to prolong the life cycle of clothing which give a hand in protecting the environment. I named it is FinnTA

1.2 Problem Statement

With the rapid development of the fashion industry, fast fashion has become an alarming global phenomenon when fashion waste is one of the causes of environmental pollution. The cost of this increased consumption of uncontrolled, discarded clothing after only a few uses or even never being used is waste, pollution, and the greenhouse effect.

Furthermore, users spend too much time both in online shopping and direct store deciding what products or accessories to buy that match their wardrobe, wondering which color is best suited with, how other people dress with this item; they can't visualize the products they intend to buy and products they own when mixing will look like. They even cannot imagine how a product on a website will be in real life.

Therefore, to solve this problem we bring a solution - a website platform to search for products, styles and collections, and connect fashion shopaholics to create a community to help each other in sharing and exchanging their style, their tips on how to mix and match, to dress up well every day. My website application is not aimed to be an e-commerce platform, I have not supported the shipping or made an order. I focus on building up a social network to share ideas to mix and match the outfit of the day, to view clothes in the set, to determine which item is best suited in different occasions, to discuss upcoming trending styles, as well as hot-sale product for each season.

1.3 Scope and Objectives

1.3.1. Scope:

The main objective of this application is to facilitate the review of products and services between the online shops and consumers on the basis of the items or products that the business/users supplies

1.3.2. Objective:

The main goal of FinnTA is to reach the maximum customers at the right time to develop more fashions and styles, recommend products for users with fast speed and the high precision degree to receive satisfaction from users. The functions of FinnTA include viewing and trying to mix and match products, transferring fashion style or data over the internet.

1.4 Assumption and Solution

1.4.1. Assumption

From users’ view: Assuming that new applications are often difficult to reach by users, or that the rate of change in the number of users grows very slowly from the early days, that is also one of the difficulties in spreading out our application.

From platform developers’ view: Creating innovation and differentiating from other apps with similar features. FinnTA's functionality should also be as easy to understand and use as possible

1.4.2. Solution of this problem

I have designed the interface to maximize possible functionality and look catchy, in keeping with current trends. Provide a friendly and easy-to-use experience to users. Thereby the application can increase its accessibility to users and stores to suit its purposes.

1.5 Structure of Thesis

The introduction will be chapter 1.

Chapter 2 take a review of the report.

From then, I am going to introduce the methodologies behind some features, some algorithms to handle logical problems in detail is presented in the Chapter 3.

After that, discuss the fundamental design that I have been working on for the web application and demonstrate the flow of the website; the report will state all implementation and some brief discussion, comparison, and evaluation of our project in Chapter 4.

Ultimately, the final chapter – Chapter 5 will sum up our achievement in building fashion virtualization, fashion product recommendation platform and share our future plan development.

CHAPTER 2. LITERATURE REVIEW

2.1 Similar Applications/Systems

Magic Closet recommends clothes for different occasion by using Latent SVM model and suggest clothes matching with color skin by applying Fuzzy Logic and Questionnaire. [1]

To be specific, Magic Closet's recommendations are based on an occasion-based latent SVM model that is taught through the use of labelled clothing. The model produces suggestions based on physical characteristics, clothing characteristics, and events. Feature occasion potential is taken into consideration when wearing properly-occasion-attribute potential. Features are taken and combined into a vector to represent a human part. Some qualities are manually defined in order to address the semantic gap between clothing features and event ideas. There are also features like pattern and category attributes like “jeans” and “skirts”. From all of the fully annotated clothing photographs, the algorithm deduces the underlying rules.

Latent SVM: assumes that (x, y) pairs are insufficient to describe the input-output relationship, and that this relationship is also dependent on unobserved latent variables z. The model combines four possibilities: a visual feature with an attribute, a visual feature with an occasion, an occasion with an attribute, and an attribute with an attribute. Incorporating these matching rules into the Latent SVM to ensure that selected clothes meet the condition of being worn correctly and attractively.

Fuzzy Logic and Questionnaire: The YCbCr color model has been fixed, and 30 samples have been divided into three races. To create fuzzy rules, the race classification is used to determine the fuzzy membership function. The suitability of clothing with skin color must next be determined. A questionnaire with photographs of human races and a wide range of clothing was provided to 30 participants. The three highest scoring overall responders (ratings of 1 to 10 denote highly mismatch to very harmonious) were assigned to each skin tone as the dominating hue. A color model suitable for clothes is chosen, and a fuzzy membership function is designed to find the association between garments color and skin color. The fuzzy rule association between the fuzzy membership function of clothes and skin color was created in order to obtain the optimum appropriateness index. The index was needed to determine the acceptability of the input skin and clothing colors. Men and women have distinct appropriateness indices.

Table 2.1 - Comparison with Similar Application

Application	**Advantages**	**Disadvantages**
MixDress – Combine Clothes App	▪ Help to choose the outfit of the day by asking some question ▪ Distinguish between favorite outfit and dislike outfit ▪ Nice experience with welcome survey	▪ Not a good layout webpage which makes user confused. ▪ Old website style, unfriendly user interface. ▪ Do not provide various categories, patterns, colours and materials. Only divide into 4 types: Bottom, top, solo and extra. ▪ Cannot mix a layered outfit: not allow outfit with 2 tops: T-shirt and jacket. ▪ Recommended outfit based on the hashtag of product from upload, not an ideal suggestion.
Virtuality Fashion	▪ Professional 3D website with detailed simulation. ▪ Convert automatically by asking the user to provide relevant data such as pictures, sketches, 2D files and choosing stitches. ▪ Support multiple languages.	▪ Doesn't popular in Vietnam. ▪ Doesn't allow to discover other items, collection of other's users, or mix and match clothes to create a collection ▪ Doesn't have a newsfeed for finding inspirable collections, not a sharing platform.

2.2 Platform and Tools Review

- Main framework: ReactJS
- Template Engine: MaterialUI, Boostrap
- Database management system: JSON Server
- Integrated development environment (IDE): Visual Studio Code (VSCode)

- Supported library for Image processing and 3D Object: Three.js, Drei, Fiber, Valtio, Html2Canvas, React - Colorful.
  - Three.js: used to create and display animated 3D computer graphics in a web browser using WebGL
  - Drei: a growing collection of useful helpers and abstractions which support project by implement Camera: Perspective and Orthographic camera, Controls: orbit controls, Loaders: use GLTF
  - Html2Canvas: manage, rearrange position of each product's image and capture the final image of a collection.
  - React Colorful: use to Changing Color of Product by HexColorPicker

- Programming language: JavaScript, CSS, SCSS
  - JavaScript: give web page interactive elements that engage users, add behavior and effects to web page, build Front-end based on React, call API, handle data from Back-end, solve logical problems
  - CSS (Cascading Style Sheet) and advanced version SCSS (Syntactically Awesome Style Sheet): give style of appearance to web page
- Managing and sharing code community: Github
- Task management tools: Trello

2.3 Background and Applied Technologies

2.3.1. Overview of web 2.0 technologies

The most typical web application structure nowadays is a dynamic website, often known as web 2.0 with REST API. The main benefit of a dynamic website is that it allows you to easily update any data in an organized and structured manner in order to build multiple product pages or categories based on the needs of the users.

Dynamic Websites are considerably more functional websites that are much easier to maintain. They allow for the addition of new information, which draws visitors back to the site and aids in search engine optimization.

2.3.2 NodeJS - Running Environment

Node.js is a framework for simply constructing fast and scalable network applications based on Google Chrome's JavaScript Engine (V8 Engine) or Chrome's JavaScript runtime. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js is a cross-platform runtime environment for building server-side and networking applications that is free source. Node.js applications are written in JavaScript and run on OS X, Microsoft Windows, and Linux using the Node.js runtime.

Node.js also comes with a large library of JavaScript modules, which greatly facilitates the building of Node.js web applications.

2.3.3 ReactJS - Front-End / Client Side

React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript toolkit for creating UI components-based user interfaces. With frameworks, React may be used as a foundation for developing single-page, mobile, or server-rendered apps. React, on the other hand, is solely concerned with state management and displaying that information to the DOM, therefore building React apps frequently necessitates the usage of extra frameworks for routing and client-side functionality.

2.3.4 ExpressJS with Nodemon - Back-End / Server Side

The most prominent Node web framework is Express, which also serves as the foundation for a number of additional Node web frameworks. It has provisions for:

- ✓ Create handlers for requests with various HTTP verbs at different URL paths or routes.
- ✓ Additional request processing "middleware" could be added at any point within the request handling pipeline.
- ✓ A typical web application setting like the connection port is set up and the location of templates that will be used to display the answer.
- ✓ Integrate with "view" rendering engines to provide replies by filling templates with data.

Despite the fact that Express is a relatively simple framework, developers have written compatible middleware packages to solve practically any web development issue. Cookies, sessions, user logins, URL parameters, POST data, security headers, and many other topics are covered by libraries. At Express Middleware, you'll find a collection of middleware packages maintained by the Express team (along with a list of some popular 3rd party packages).

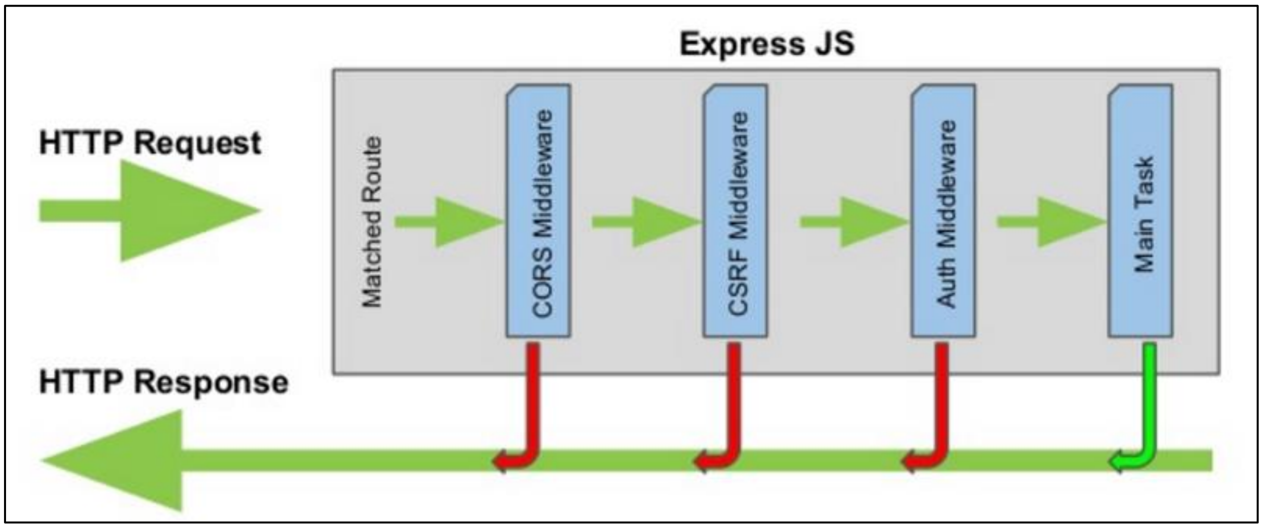


Figure 2 1 – ExpressJS Workflow

2.3.5 Cloudinary - Dynamic Asset Management (DAM)

Cloudinary is a cloud-based service, a dynamic asset management solution for websites and mobile apps. It provides an optimal, powerful visual experiences solution to manage media formats, covering everything from uploads, storage, transformations, manipulations, and optimizations, to delivery of all media assets and digital experiences across any browser, device, and bandwidth with easy-to-use SDKs, APIs, widget and user interface.

Figure 2 2 – Cloudinary Logo

With this end-to-end media asset management, the website or mobile application launches, content delivery in a faster and hassle-free way. Cloudinary is considered as a Rich media capabilities to address all Web and Mobile App needs with some highlight provided features:

Media APIs and SDKs for High Performance:

Control the media lifecycle in simple steps: upload rich media to the cloud for storage; transform and optimize with dynamic URLs; and deliver at high speeds over single CDN, multi-CDN, and geo-specific CDNs; integrates seamlessly with existing application code with APIs for delivery capabilities.

Manage rich media format:

Work with a wide variety, an exhaustive list of media types delivered – images, videos, GIF, 3D images, PDF, PSD, AVIF, …

Real-time Image Enhancements:

Create multiple banners on-the-fly from the same image, overlaid with logos, text, badges, and watermarks.

Website Load Time Optimizations – Automate responsive:

Automatically serve images and videos in browser or viewport- optimized formats and quality to ensure pretty look on any device with automatic width selection, responsive breakpoint support, HTTP client hints and adaptive bitrate streaming for videos.

Leverage AI, Machine Learning Algorithms:

Automate media management activities like file tagging, background removal, content-aware cropping, AI-based cropping, video previews, and more.

Image & Video Hosting:

Storage via AWS and Google Cloud with enterprise-class backup, revision history, and disaster recovery.

2.3.6 JSON Server

One of the most significant aspects of a typical corporate application is collaborating with several teams and third-party APIs. Consider using a third-party RESTful web service to obtain JSON data to work on. Due to a tight schedule, the website cannot afford to wait for them to finish their job before starting yours. JSON Server is the NPM package, the prototype Rest Web service we need to retrieve the demo data in a faster way. JSON Server makes it easy to set up a REST API with CRUD operations (Create, Read, Update, Delete) by using HTTP requests (Post, Get, Put, Delete) simply and quickly within one minute.

Not inferior to RESTful web service, JSON Server also support to handle sort, filter, slice, paginate, query by full-text search, operators and relationship to obtain a specific range of data.

Besides the Express Server applied in the thesis, JSON Server is also a suitable approach for quickly establishing a backend for an application because JSON is an easy-to-parse and lightweight data-interchange format. For this application, which integrates website features, functions, and centralized to present the outcome of the training model of

Frequent Item-set Mining, the combination of Node, Express, and MongoDB for the backend-side is not a viable solution owing to time demand.

CHAPTER 3. METHODOLOGY

3.1 User requirement analysis

Capturing the expectations about the features integrated into a product that users have is an endless task to be done from the very first stages of product development to the improvement phase of the published product. Bearing in mind that the user target of this website is the youngsters who are fascinated by fashion, user requirement analysis was carried out.

Fashion enthusiasts demand to save outfits worn by trendsetters and other users that catch their eyes. Additionally, what differentiates this website from other social media sites available market lies in its capability of enabling users to mix and match a set of outfits by combining separate items. Its function allows users to view their or others' products in a more close-up view and suggest which item they should integrate with items they already have in the closet to give a hand in deciding whether they should buy that item or not. As the user requirement analysis was carried out, this website has been developed to become a promising social media site that should be put into consideration of publishing online to satisfy the passion in fashion.

3.2 System design

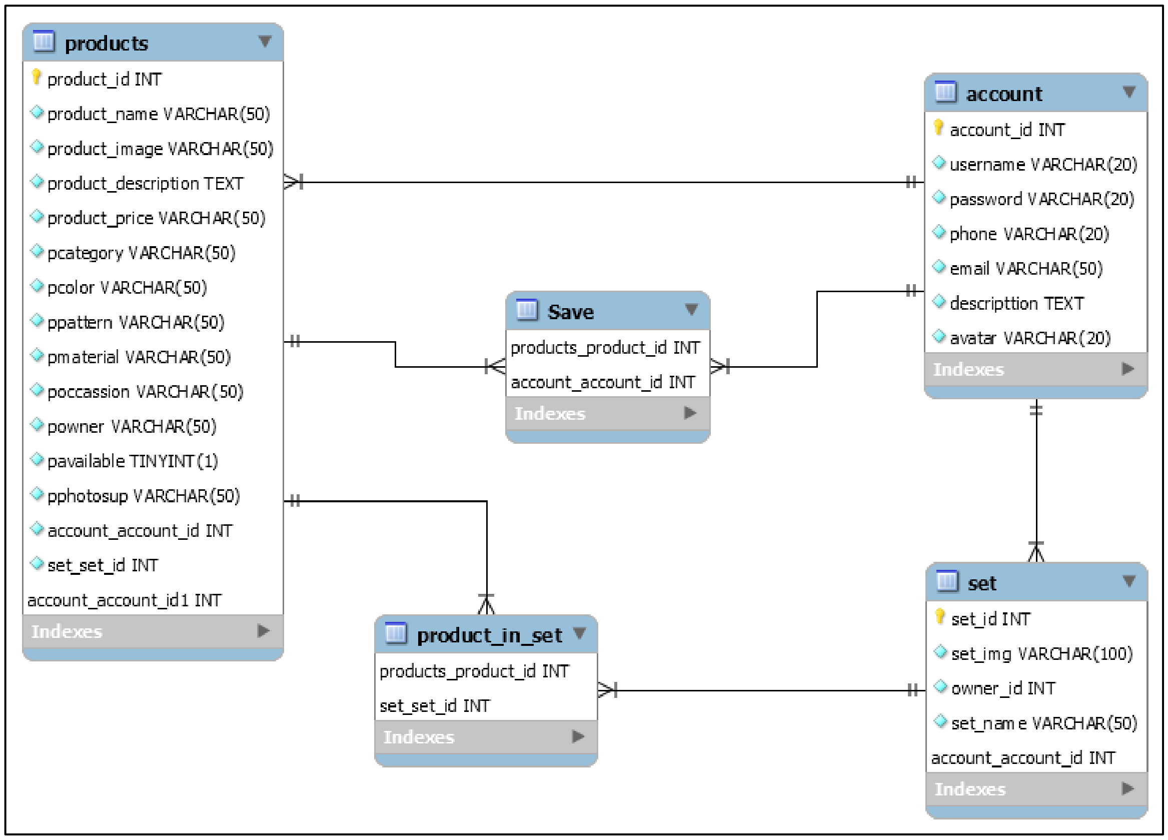


Figure 3 1 – Database Design of FinnTA

In database design, there are 3 main tables in the database system: products, sets, accounts. The product table stores information about the product includes name, image, description, price, etc. The Set table stores id of a set, the owner – who create the set and the relationship between set and product is M-N means 1 set contains many products and 1 product belongs to many sets.

3.3 Frequent Item-set Mining

3.3.1 Overview

We are living in the era of industrial revolution 4.0 - the most brilliant development up to date in the digital age. In this era, data plays a role as one of the important factors in business - which is considered as an abundant resource that is strictly secured by each enterprise and data analytics has a significant impact on the decision-making process. Thanks to data, we understand the problems facing the organization and the effective use of data helps businesses improve vastly.

Data analytics has a great impact on diverse industries such as retail, consumer goods, (FSI) financial services, manufacturing (Manufacturing), insurance, etc. The insights gained from such pattern analysis can bring significant benefits such as increased sales and reduced costs, increased competitive advantage, improve business results, come up with better market strategies, give help to businesses in making faster and more accurate decisions.

With the great benefits that data brings up, I propose to build up a recommendation system by using Frequent Item-set mining into the FinTA web application. In the purpose of exploiting these valuable statistic data, insights in suggesting suitable products to combine with in order to help users create satisfactory outfits quickly, easily, and stylishly.

3.3.2 Frequenst Item-set Mining (FIM)

With the help of recommendations, analyzing and finding customer behavior becomes more easily and discovering more informative the relationship between unrelated data. It plays a vital role in Market basket Analysis. Recommendations using Frequent Item-set mining Algorithm help to identify more frequently occurring patterns and recommend the relationship or correlations between the items in transactional and relational databases are found. Applying the idea of Frequent Item-set mining to my website, this recommended system will suggest matching items based on their frequency with other items, based on the relationship between entities that have been combined regularly by many users. The system will propose products and sort the priority in decreasing order (top-down) based on the percentage of Confidence - the high probability that item A will mix and combine with item B compared to other itemsets.

Frequent Item-set Mining is also known as Association Rule Mining. Association rule mining [2] is an important part of data mining, a rule-based machine learning algorithm. The algorithm can find mutual relations in big data. Its purpose is to use some attribute criteria to identify strong rules that exist in the database. Frequent Mining shows which items appear together in a transaction or relation.

Therefore, it is an unsupervised machine learning method within four methods, types of Machine Learning problems: Supervised learning, Unsupervised learning, Semi-supervised learning and Reinforcement learning.

3.3.3 Prerequisites definitions of Frequent Item-set Mining

a) Important definition - Baseline

Let's denote the combination of item (collections) database is (T) which contains a set of completed outfits, and each outfit has a set of mixing items. **|T|** is a number of total collections. There are multiple collections in our database and denoted each transaction as (t). Each unique product in overall collections is called "item", denoted as (i), for all sets of unique items in our transaction database, denoted as **(I)** – a number of totally unique items. Therefore, we will have the collections of (I) and (T) as follows:

$$I = \{i_1, i_2, \dots , i_{|I|}\}$$

$$T = \{t_1, t_2, \dots, t_{|T|}\}$$

The term "***itemset***" refers to a collection of specific product sets; an itemset might contain one or more products/items. We'll use the following statistic to calculate the frequency of a certain itemset, indicated as X in the whole transaction database. The following equation is used to compute ***support(X)***:

$$\text{support(X)} = \frac{n(X)}{|T|} = \frac{|t \subset T,\ \ X \subseteq t}{|T|}\ [3]$$

The support value of (X) is the frequency level of (X) itemset in overall database collections, where n(X) is the number of transactions that contain (X) and (|T|) is the total number of collections. Support is one of the measures of interestingness that demonstrates the usefulness and certainty of a rule. To be more exact, 10% support means that 10% of the database's collections follow the association rule.

$$\text{support}(A \ \rightarrow B) = \text{support\_count } (A \cup B)\ [4]$$

where support_count(X) denotes the number of collections in which X appears, A ∪ B is represent for the set of collections that contains both A and B. If X is A union with B then it is a number of collections in which A and B are both present.

Another important definition is ***Confidence*** which is used to identify how often a “T-shirt”, “skirt” and “jacket” are mixed together in comparison with other itemsets, for example, a confidence metric is used – denoted as confidence (A → B) is calculated as bellow equation:

$$\text{confidence } (A \ \rightarrow B) = \frac{support(\ A \cup B)}{support\ (A)}\ [4]$$

Let’s take an example to illustrate this meaning. Let’s consider the association rule is [‘tee’] ^ [‘skirt’] → [‘jacket’] with the confidence of 70% means that 70% of customers/visitors who mixes a tee with a skirt also looking for a jacket to combine, to create an outfit.

In addition, we have some other terms such as: ***Maximal itemset, Closed itemset, K-itemset***. If none of an itemset's supersets are frequent, it's called a maximum itemset. An itemset is a closed itemset if none of its immediate supersets has the same support count same as itemset. A K-itemset is an itemset that contains K items.

3.3.4 Problem Analysis

The association rule mining process mainly includes two stages:

- The first stage is to find all the high-frequency item sets from the collection.
- The second stage is to generate the association rules from these high-frequency item sets. The high-frequency item set found in the first stage refers to the fact that the frequency of occurrence of a certain project group must reach a certain level called minimum support count with respect to all records, the frequency of occurrence in a project group is called Support.

The problem is how can we determine whether the itemsets are frequent or infrequent. We will initiate a threshold to determine how frequent that we need, called: minimum support value, denoted as ***minSup***, the range value of minSup is from 0 to 1 ($0 < minSup < 1$). We would take the itemset if its support value is higher than minSup, otherwise will be ignored.

The confidence level of association rules is similar to the itemset frequency threshold, we also have a threshold for the confidence of rules, called as minimum confidence value, denoted as ***minConf*** ($0 < minConf < 1$) which means that product association rules could be taken if its confidence values larger or equal the predefined minConf threshold.

When the Association rules that satisfy the minimum supportability threshold – minSup and also meet the minimum confidence threshold – minConf are called strong association rules. The set of items that meet the minimum support threshold limit is called a ***Frequent itemset***. The most important part of determining the performance of association rules mining is mining frequent itemsets [5].

3.4 Dataset

I was seeking for a shared dataset on Kaggle and decided to collect a dataset from a clothing-oriented online website named Polyvore.com, which is the most popular fashion-oriented website in the United States. This dataset consists of 68,306 polyvore.com outfits organized into 224 categories, and their meta-data crawled from the Polyvore website.

Dataset splits: There are two versions of the data: Disjoint and non-disjoint Polyvore outfits.

- ***Non-disjoint outfits:*** Outfits are divided at random, so certain items (but not whole outfits) may appear in both training and test splits.
- ***Disjoint outfits:*** There are no items in the test/validation set that are shared with the training set (although some items in the test set may be present in outfits in the validation set)

Within each version folder, data is divided into 3 JSON files – a list of outfits with their item id and ordering as known as index: train, valid, test.json for training, validation, and testing.

Images and Meta-data: contains 261K images stored by their item id, which is organized in lists of outfits for each version of the dataset.

- ***polyvore_item_metadata.json:*** contains a dictionary where each key is an item_id, and the values are its associated meta-data labels.
- ***polyvore_outfit_titles.json*** - contains a dictionary where each key is a set_id and the values are its associated meta-data labels.

categories.csv: each row contains three items: category id, fine-grained category, and semantic category.

Polyvore provides tools, templates and friendly interfaces for users to create fashion outfits from a pool of items by themselves and post the outfit to share with others, and tens of thousands of fashion outfits are created every day. Each shared outfit includes details such as the outfit name, goods included in the outfit, similar styles, comments about the outfit, the number of people who enjoy it, and so on. However, the website's name has been changed to SSENSE.

The fashion goods are meticulously and tastefully placed to represent distinct fashion styles, as demonstrated in the illustration.

Visitors glance at, like, and recreate these fashion outfits, and some of those get a vast amount of attention. We crawl fashion outfits from Polyvore.com, which are related to the number of likes and the number of fashion goods. The image, title and category for each fashion item have been crawled.

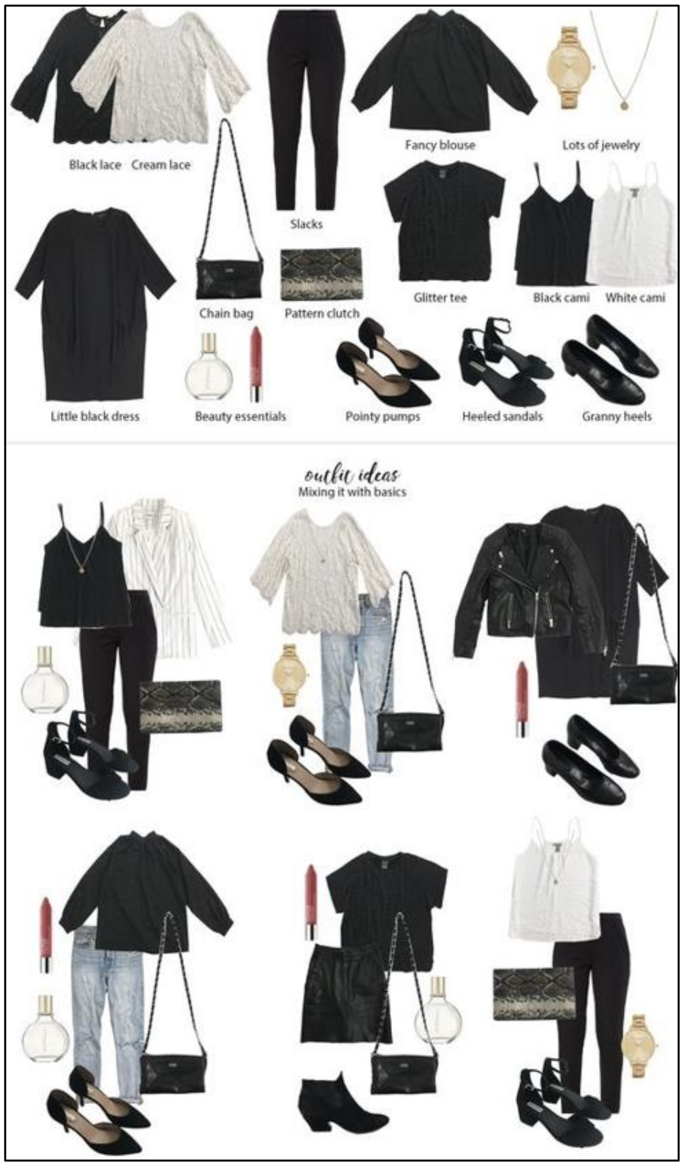


Figure 3 2 – Shows an example of fashion clothes created by users.

3.5 Pre-processing Data

Data preprocessing is a process of inspecting, cleaning, transforming, and modeling data with the goal of discovering useful information, drawing conclusions, and supporting decision-making. In a nutshell, it's a method for transforming raw data into a usable and efficient format.

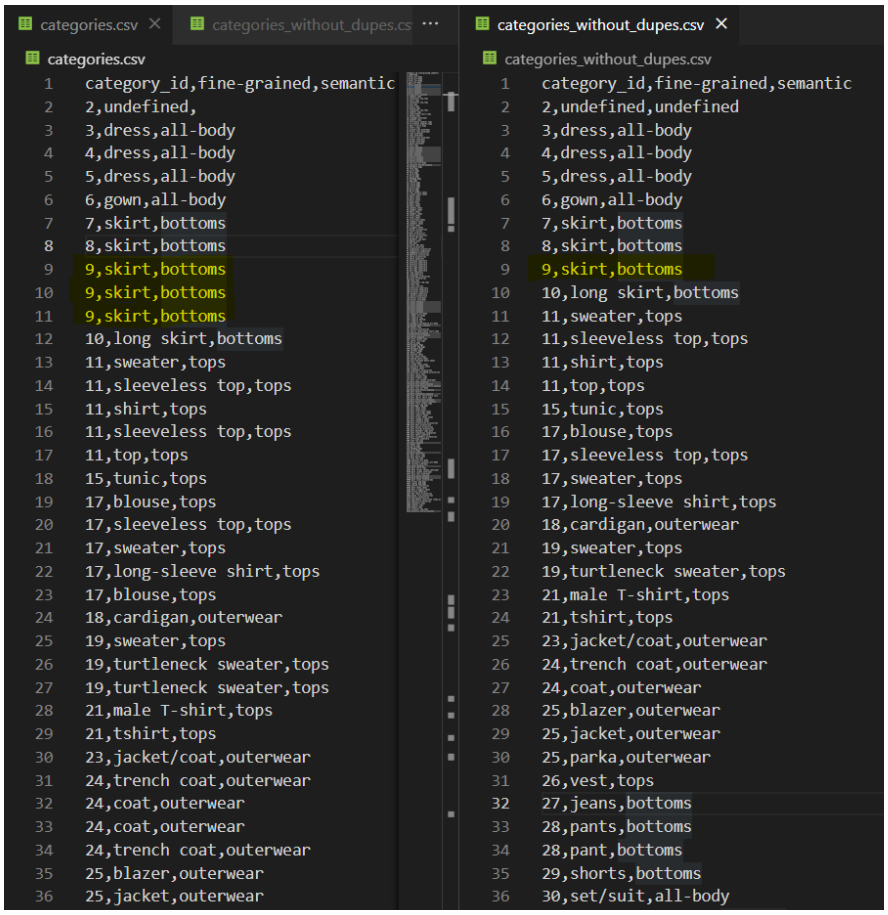
3.5.1 Import crucial libraries

Python is the most widely used and recommended data science library worldwide. Predefined Python libraries, which can execute certain data preparation activities, are one of the first critical steps for data preprocessing in Machine Learning. For data preprocessing in Machine Learning, the following three fundamental Python libraries are used:

- ***NumPy:*** the core package for scientific calculation in Python. It is used to add big multidimensional arrays and matrices to code, as well as entering any form of mathematical calculation.
- ***Pandas:*** a fantastic open-source Python for data manipulation and analysis toolkit. It is extensively used for importing and managing datasets, as well as offering efficient, user-friendly data structure and data analysis tools. It packs in high-performance, and can handle a wide range of data types, including tabular data, ordered and unordered time series, and arbitrary matrix data with row and column labels.
- ***Matplotlib***: a Python 2D plotting library that is used to plot any type of graphs or charts in Python. It can produce publication-quality figures in numerous hard copy formats and interactive data visualization, environments on a variety of platforms (IPython shells, Jupyter notebook, web application servers, etc.).

3.5.2 Identifying and handling the missing values - Transactional database

- **Remove duplicate categories:** remove duplicate rows (have the same value in 3 columns: category_id, fine-grained, semantic) of data, and keep only one row that represents those duplicates. The number of existing categories is reduced from 224 to 195 as a result of this data filtering.



categories.csv

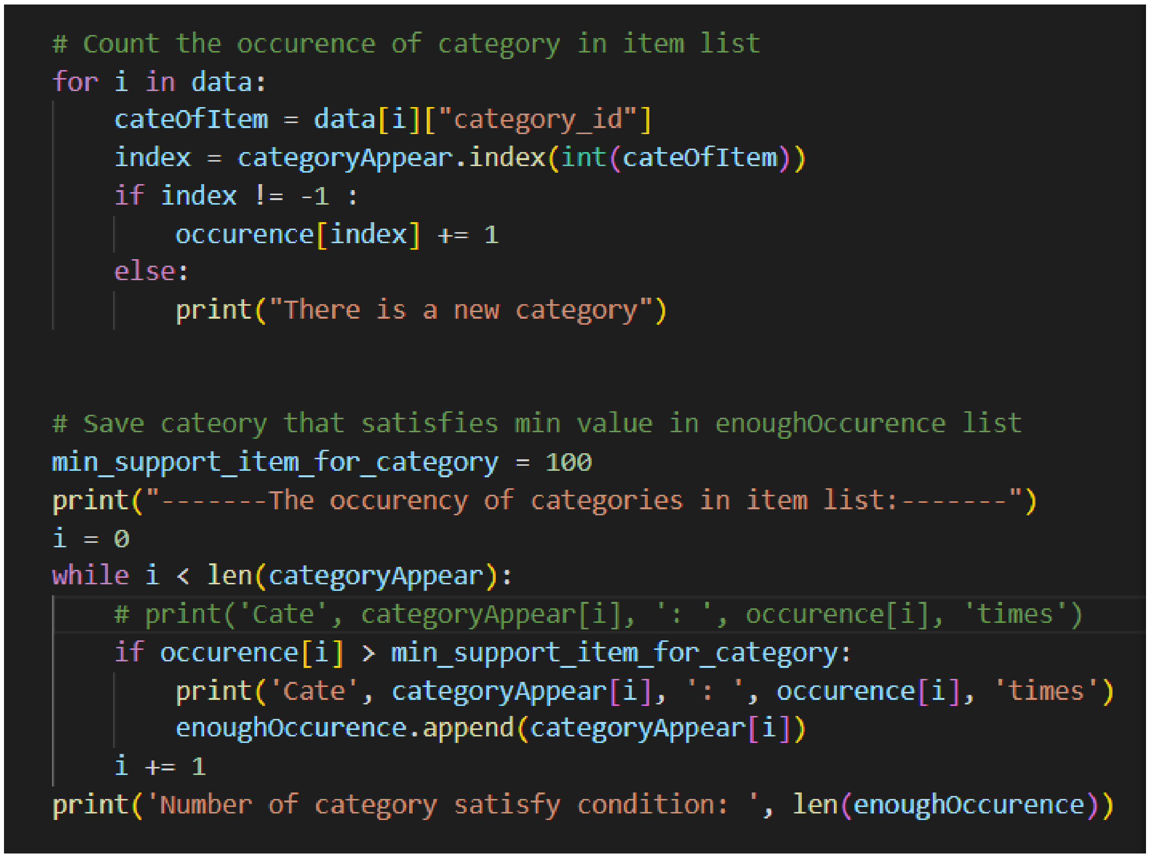
```
category_id,fine-grained,semantic
2,undefined,
3,dress,all-body
4,dress,all-body
5,dress,all-body
6,gown,all-body
7,skirt,bottoms
8,skirt,bottoms
9,skirt,bottoms
9,skirt,bottoms
9,skirt,bottoms
10,long skirt,bottoms
11,sweater,tops
11,sleeveless top,tops
11,shirt,tops
11,sleeveless top,tops
11,top,tops
15,tunic,tops
17,blouse,tops
17,sleeveless top,tops
17,sweater,tops
17,long-sleeve shirt,tops
17,blouse,tops
18,cardigan,outerwear
19,sweater,tops
19,turtleneck sweater,tops
19,turtleneck sweater,tops
21,male T-shirt,tops
21,tshirt,tops
23,jacket/coat,outerwear
24,trench coat,outerwear
24,coat,outerwear
24,coat,outerwear
24,trench coat,outerwear
25,blazer,outerwear
25,jacket,outerwear
```

categories_without_dupes.csv

```
category_id,fine-grained,semantic
2,undefined,undefined
3,dress,all-body
4,dress,all-body
5,dress,all-body
6,gown,all-body
7,skirt,bottoms
8,skirt,bottoms
9,skirt,bottoms
10,long skirt,bottoms
11,sweater,tops
11,sleeveless top,tops
11,shirt,tops
11,top,tops
15,tunic,tops
17,blouse,tops
17,sleeveless top,tops
17,sweater,tops
17,long-sleeve shirt,tops
18,cardigan,outerwear
19,sweater,tops
19,turtleneck sweater,tops
21,male T-shirt,tops
21,tshirt,tops
23,jacket/coat,outerwear
24,trench coat,outerwear
24,coat,outerwear
25,blazer,outerwear
25,jacket,outerwear
25,parka,outerwear
26,vest,tops
27,jeans,bottoms
28,pants,bottoms
28,pant,bottoms
29,shorts,bottoms
30,set/suit,all-body
```

Figure 3 3 – Shows an example of clear duplicate category

- **Clear out categories not related to Fashion:** those unrelated categories appear too few time which have less than 100 products in that category of the dataset. This data filtering reduces the number of previous categories from 195 categories to 133 categories.



```python
# Count the occurence of category in item list
for i in data:
    cateOfItem = data[i]["category_id"]
    index = categoryAppear.index(int(cateOfItem))
    if index != -1 :
        occurence[index] += 1
    else:
        print("There is a new category")


# Save cateory that satisfies min value in enoughOccurence list
min_support_item_for_category = 100
print("-------The occurency of categories in item list:-------")
i = 0
while i < len(categoryAppear):
    # print('Cate', categoryAppear[i], ': ', occurence[i], 'times')
    if occurence[i] > min_support_item_for_category:
        print('Cate', categoryAppear[i], ': ', occurence[i], 'times')
        enoughOccurence.append(categoryAppear[i])
    i += 1
print('Number of category satisfy condition: ', len(enoughOccurence))
```

Figure 3 4 – Shows an example of clear duplicate category

- **Defined category based on gender:** divided into two separate categories, hence besides the category in general, having female and male category. The male category has the specific word indicator is “male”, hence every row of data has a male in their semantic columns will belong to the male with 8 categories, and the remainder is for the female with 125 categories.

- **Filtered out the item in removed categories:**

  Filter products from the list of products in Polyvore item that belongs to the category that has been removed for not meeting the conditions. Updating the outfit - sets of clothing containing that product, i.e., remove only the product from the collection while keeping valid items and not deleting the entire collection. The total number of items that meet the condition that are not in the category are removed from 68,306 to 66,921 products.

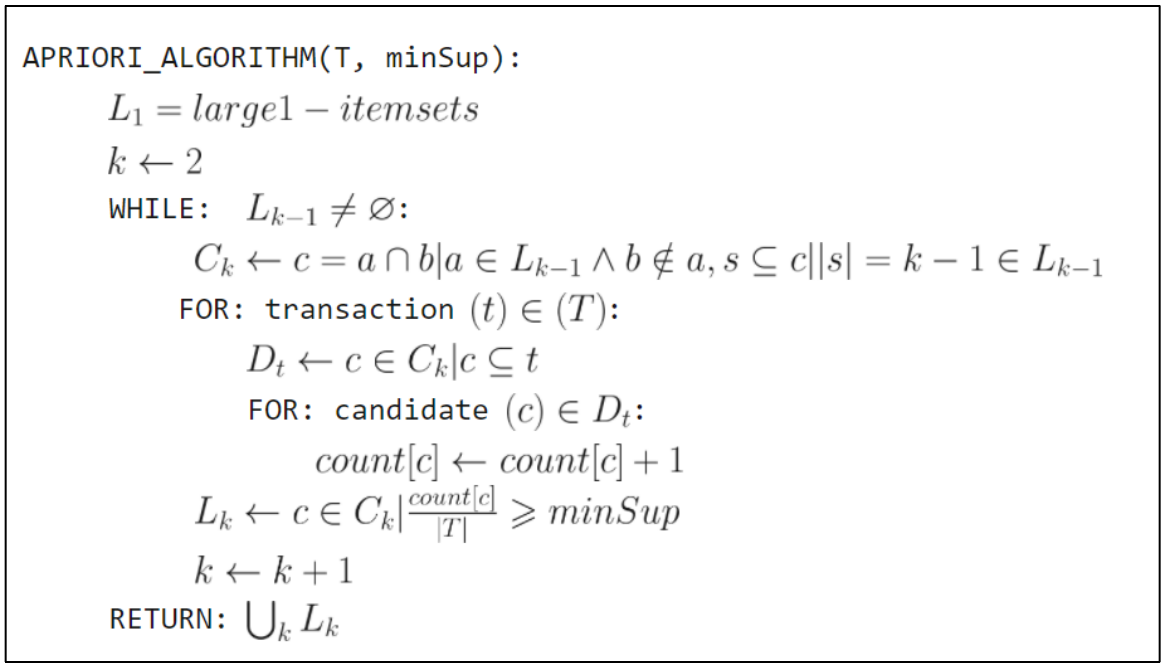
3.6 Apriori Alogirthms

3.6.1 Introduction

The first algorithm for mining frequent patterns was Apriori. R agarwal and R srikant presented it in 1994. It uses a database with a horizontal layout. It uses a generate and test approach and is based on Boolean association rules. It makes use of BFS (breadth first search). Apriori finds a larger itemset of k+1 items by using frequent k itemsets. When an item's Apriori support count is specified, the algorithm initially scans the database for all frequently occurring items based on support. The frequency of an item is calculated by counting how many times it appears in all transactions [7]. All infrequently used goods are discarded. Apriori property: All non-empty subsets of a frequent itemset are also frequent.

3.6.2 Frequent itemsets mining by Apriori algorithm

The initial stage in learning association rules from the collections database is to retrieve frequently used itemsets. Following that, association rules are generated from the obtained itemset. As a result, the Apriori method is used in the first phase to obtaining all frequent itemsets from the database (T) that satisfy the minimum support value threshold (minSup).



```
APRIORI_ALGORITHM(T, minSup):
    $L_1 = large1 - itemsets$
    $k \leftarrow 2$
    WHILE: $L_{k-1} \neq \varnothing$:
        $C_k \leftarrow c = a \cap b | a \in L_{k-1} \wedge b \notin a, s \subseteq c || s | = k - 1 \in L_{k-1}$
        FOR: transaction $(t) \in (T)$:
            $D_t \leftarrow c \in C_k | c \subseteq t$
            FOR: candidate $(c) \in D_t$:
                $count[c] \leftarrow count[c] + 1$
        $L_k \leftarrow c \in C_k | \frac{count[c]}{|T|} \geqslant minSup$
        $k \leftarrow k + 1$
    RETURN: $\bigcup_k L_k$
```

Figure 3 5 – Pseudo-code for Apriori algorithm which is introduced by R. Agrawal et al. [6]

There are 5 steps to give a clearer explanation for the Apriori algorithm's pseudo-code.

Step 1: Initializing 1-length itemsets: A set of 1-length itemset denoted as L_1 contains all itemset that only has 1 item. After that, initialize a variable k = 2, this variable is used to indicate the length of the next itemsets.

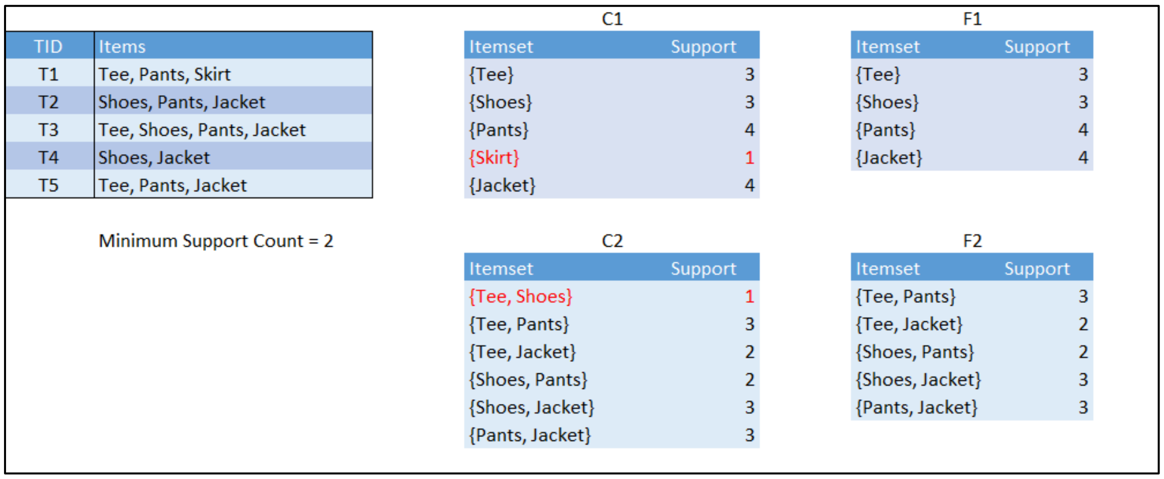
Step 2: Candidate generation: Entering the while loop for each iteration, a set of a candidate is generated denoted as C_k from the current itemset with (k-1) length items denoted as L_{k-1}. The retrieved C_k is also considered as a list of itemsets that have k-length.

Step 3: Database scanning: Iterate through all collections in the database to check if each candidate c – which is generated from the previous step in C_k is a subset of any transaction t, increase the frequency count by 1:

count(c) $\leftarrow$ count(c) + 1

Step 4: Frequent itemset pruning: To exclude all unsatisfied candidates with support values less than the set minSup threshold, calculate support value for all examined candidates in C_k as: $\frac{count[c]}{|T|}$. New satisfied candidates will be utilized to create a new collection of k-length itemsets, represented as L_k, and the value of k will be increased by one. The while iteration is then continued in Step 2 until no itemset exists in L_k (or $L_k = \varnothing$).

Step 5: Frequent itemsets collecting: From previous generated satisfied items with different lengths, we aggregate all of them to form our final frequent itemsets collection which all itemsets, denoted as: $\bigcup_k L_k$ have support values greater or equal to the given predefined minSup value.



TID	Items
T1	Tee, Pants, Skirt
T2	Shoes, Pants, Jacket
T3	Tee, Shoes, Pants, Jacket
T4	Shoes, Jacket
T5	Tee, Pants, Jacket

Minimum Support Count = 2

C1

Itemset	Support
{Tee}	3
{Shoes}	3
{Pants}	4
{Skirt}	1
{Jacket}	4

F1

Itemset	Support
{Tee}	3
{Shoes}	3
{Pants}	4
{Jacket}	4

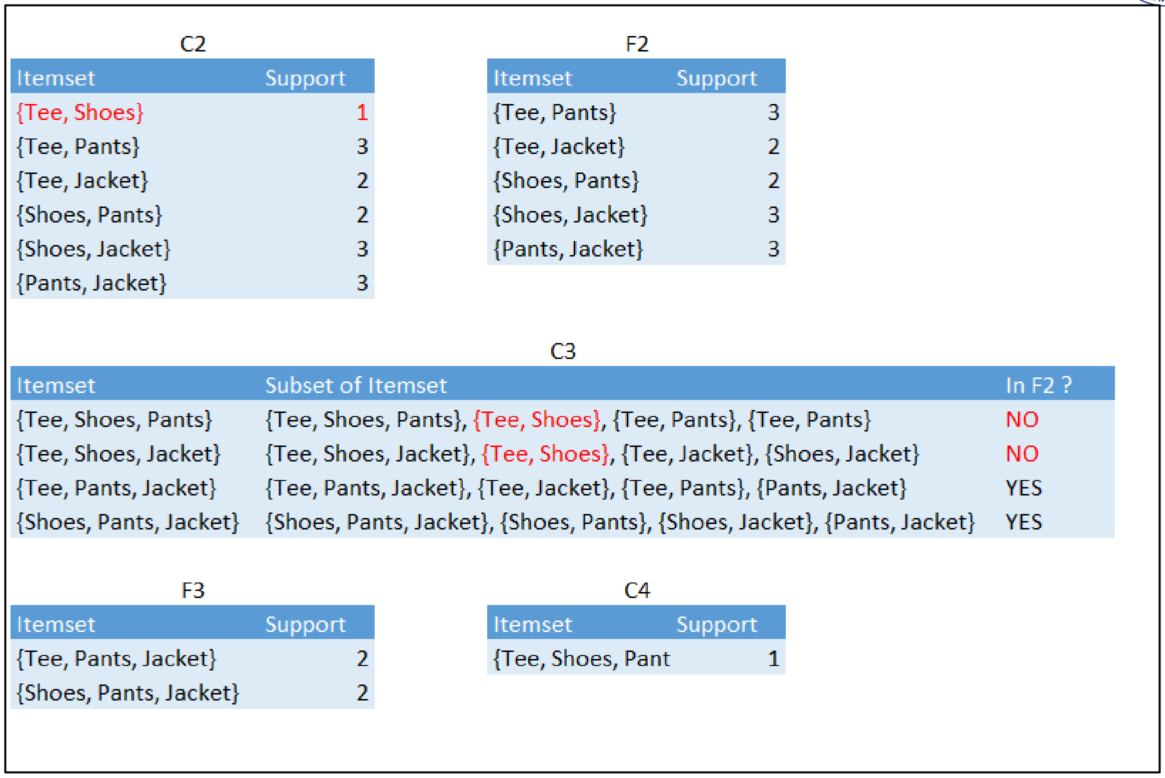
C2

Itemset	Support
{Tee, Shoes}	1
{Tee, Pants}	3
{Tee, Jacket}	2
{Shoes, Pants}	2
{Shoes, Jacket}	3
{Pants, Jacket}	3

F2

Itemset	Support
{Tee, Pants}	3
{Tee, Jacket}	2
{Shoes, Pants}	2
{Shoes, Jacket}	3
{Pants, Jacket}	3

Figure 3 6 – Given transaction data and apply Apriori Algorithm with 1-length, 2-length itemsets



C2

Itemset	Support
{Tee, Shoes}	1
{Tee, Pants}	3
{Tee, Jacket}	2
{Shoes, Pants}	2
{Shoes, Jacket}	3
{Pants, Jacket}	3

F2

Itemset	Support
{Tee, Pants}	3
{Tee, Jacket}	2
{Shoes, Pants}	2
{Shoes, Jacket}	3
{Pants, Jacket}	3

C3

Itemset	Subset of Itemset	In F2 ?
{Tee, Shoes, Pants}	{Tee, Shoes, Pants}, {Tee, Shoes}, {Tee, Pants}, {Tee, Pants}	NO
{Tee, Shoes, Jacket}	{Tee, Shoes, Jacket}, {Tee, Shoes}, {Tee, Jacket}, {Shoes, Jacket}	NO
{Tee, Pants, Jacket}	{Tee, Pants, Jacket}, {Tee, Jacket}, {Tee, Pants}, {Pants, Jacket}	YES
{Shoes, Pants, Jacket}	{Shoes, Pants, Jacket}, {Shoes, Pants}, {Shoes, Jacket}, {Pants, Jacket}	YES

F3

Itemset	Support
{Tee, Pants, Jacket}	2
{Shoes, Pants, Jacket}	2

C4

Itemset	Support
{Tee, Shoes, Pant	1

Figure 3 7 – Given transaction data and apply Apriori Algorithm with 1-length, 2-length itemsets

Generating association rules from mined frequent itemsets

After obtaining all of the frequent itemsets with support values greater than minSup in the previous part (section 3.4.2), denoted as $\bigcup_k L_k$ L = , a frequent item is used to generate product association rule, indicated as ***A*** $\rightarrow$ ***B***.

minConf – the minimum confidenece for association rules with each set contains of k-length itemsets, denoted as: ***L***$\boldsymbol{L_k}$, need to find all ***f*** stands for non-empty subsets where f belongs to L_k ($f \subset L_k$), to generate association rules such as: $\boldsymbol{f \rightarrow (L_k - f)}$ which confidence value: ***confidence*** $\boldsymbol{f \rightarrow (L_k - f)}$ satisfy the condition confidence value must be larger than or equal to the minConf threshold.

```
APRIORI_ASSOCIATION_RULES_GENERATION(L, minConf):
    $R = \varnothing$
    FOR: itemset $l \in L$
        FOR: subset $f \subset l (f \neq \varnothing \; and \; f \neq l)$
            IF: $confidence(\{f \rightarrow (l - f)\}) = \frac{support(l)}{support(f)} \geqslant minConf$
                $R = R \cup \{f \rightarrow (l - f)\}$
    RETURN: $R$
```

Figure 3 8 – Pseudocode for Apriori Association Rule Generation Algorithm

In general, the number of association rules that can be generated with k-length itemset is $2^k - 2$. The below pseudo-code for generating association rule from the given itemset L = $\bigcup_k L_k$

Let's take an example with a 3-length itemset, such as {'Tee', 'Skirt', 'Jacket'}, there are $2^3 - 2 = 6$ (with k =3) association rules can be generated are:

{'Tee'} → {'Skirt', 'Jacket'}, {'Skirt'} → {'Tee', 'Jacket'}, {'Jacket'} → {'Skirt', 'Tee'}

{'Skirt', 'Jacket'} → {'Tee'}, {'Tee', 'Jacket'} → {'Skirt'}, {'Skirt', 'Tee'} → {'Jacket'}

3.6.3 Applying Automatic product recommendation via mix and match analysis

From the knowledge, the concept of Frequent Itemset Mining and Apriori algorithm obtained, the Automatic Product Recommendation (APR) feature development is applied into my Fashion Recommendation Website platform. To be specific, the system will receive input from users as clothes that they would like to find an item to match with. The system will respond by suggesting potential products for customers depending on analyzing their current having/added clothes in their closet - mix and match sections.

To recommend to a website's visitor, the system will display the top items, top goods in target itemsets of matched association rule with greatest confidence values, ranking these products in descending order with highest confidence values on the top.

3.6.4 Limitation

It is possible that the Apriori Algorithm is slow. The biggest drawback is the amount of time it takes to hold a large number of candidate sets with frequent itemsets, low minimum support, or huge itemsets, making it inefficient for large datasets.

Furthermore, to recognize a regular pattern of size 100, i.e. v1, v2,... v100, it must generate 2100 candidate itemsets, which is costly and time-consuming. As a result, it will check for several sets from candidate itemsets, as well as scan the database multiple times to locate candidate itemsets. When memory capacity is restricted and there are a significant number of transactions, Apriori will be very low and inefficient.

3.7 FP-Growth Algorithms

3.7.1 Introduction

This algorithm is an improvement to the Apriori method. The difference between Apriori and FP-Growth [8] is that a frequent pattern is generated without the need for candidate generation in FP-Growth.

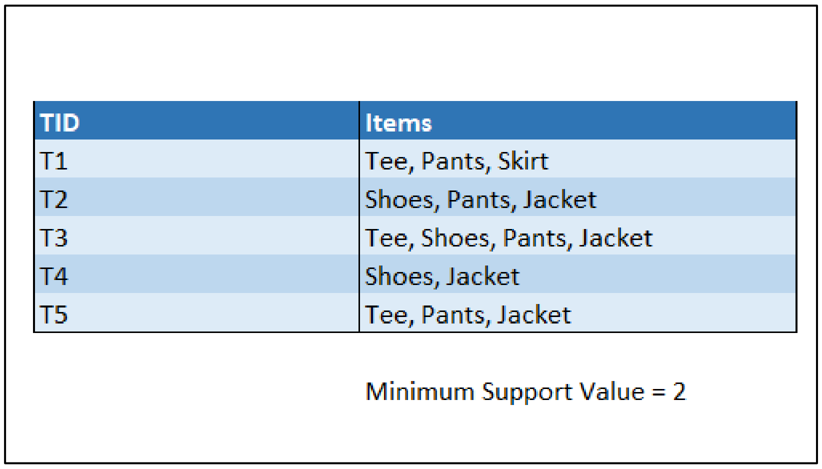
Frequent Pattern Algorithm (FP) is a short term for Frequent Pattern Algorithm, which depicts the database as a tree known as a frequent pattern tree or FP tree. FP-Growth identifies frequent item sets, encodes datasets using the FP Tree compact data structure, extracts frequent itemsets directly from FP Tree, and utilizes frequent itemsets to discover Association rules. Transaction table of dataset → FP Tree → Frequent itemset → Association Rule

3.7.2 FP Tree

The Frequent Pattern Tree is a tree-like structure that is created from the initial itemsets of the database. The FP tree's objective is to find the most common pattern.

Each item in the itemset is represented by a node in the FP tree. Null is represented by the root node, whereas itemsets are represented by the lower nodes. The association of the nodes with the lower nodes that is the itemsets with the other itemsets are maintained while forming the tree.

3.7.3 Frequent Pattern Growth Algorithm



TID	Items
T1	Tee, Pants, Skirt
T2	Shoes, Pants, Jacket
T3	Tee, Shoes, Pants, Jacket
T4	Shoes, Jacket
T5	Tee, Pants, Jacket

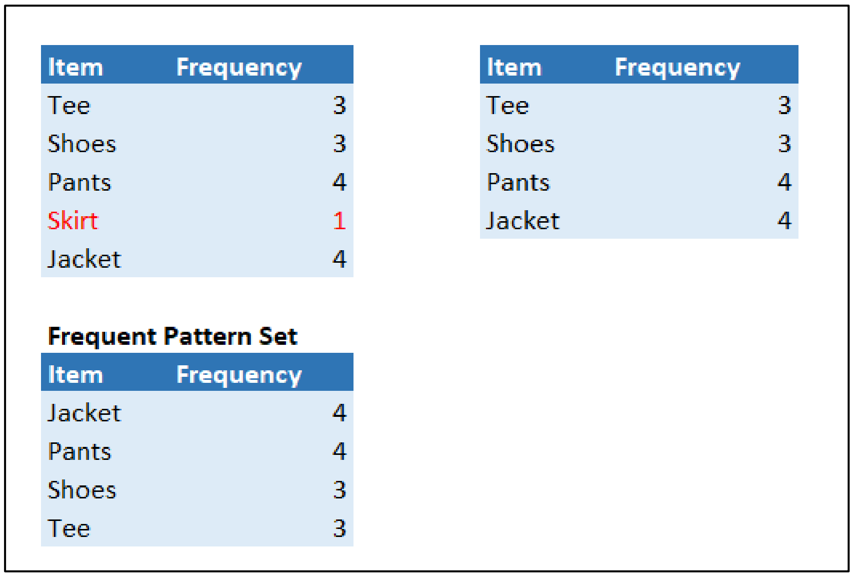
Minimum Support Value = 2

Figure 3 9 - Illustration of Transaction data to apply FP-Growth

Step 1: Find the occurrence of individual item: Examine the database for the frequency of the itemset. This phase is identical to Apriori's first step. The count of 1-itemsets in the database is called support count or frequency of 1-itemset (single item pattern).

Step 2: Find Frequent Pattern Set: Determine the minimum support value and establish the Frequent Pattern set which comprises all elements with a frequency is greater than or equal to the minimum support.

Step 3: Sort in frequency descending order: The Frequent Pattern set's elements are organized in descending order of their frequencies. The highest-counting itemset is at the top, followed by the next-lowest-counting itemset, and so on –means that the branch of the tree is constructed with transaction itemsets in descending order of count.



Item	Frequency
Tee	3
Shoes	3
Pants	4
Skirt	1
Jacket	4

Item	Frequency
Tee	3
Shoes	3
Pants	4
Jacket	4

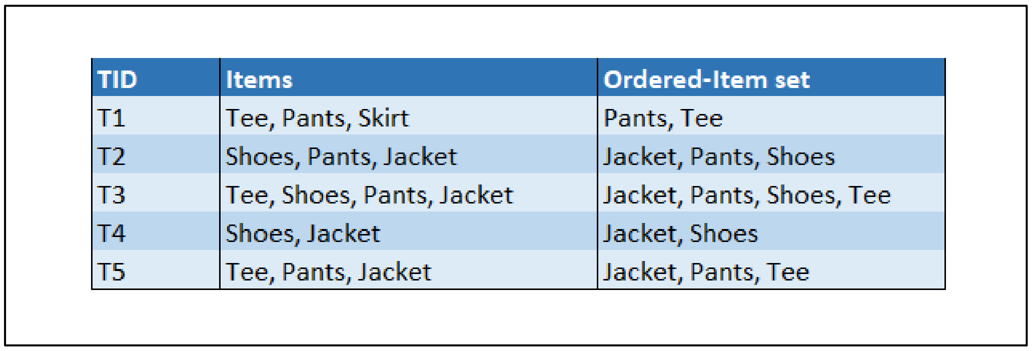
Frequent Pattern Set

Item	Frequency
Jacket	4
Pants	4
Shoes	3
Tee	3

Figure 3 10 - Illustration of applying FP-Growth Algorithm step 1, 2, 3

Step 4; Identify the Ordered-item set:

For each transaction, an Ordered-item set is created based on the previously discovered Frequent Pattern Set.

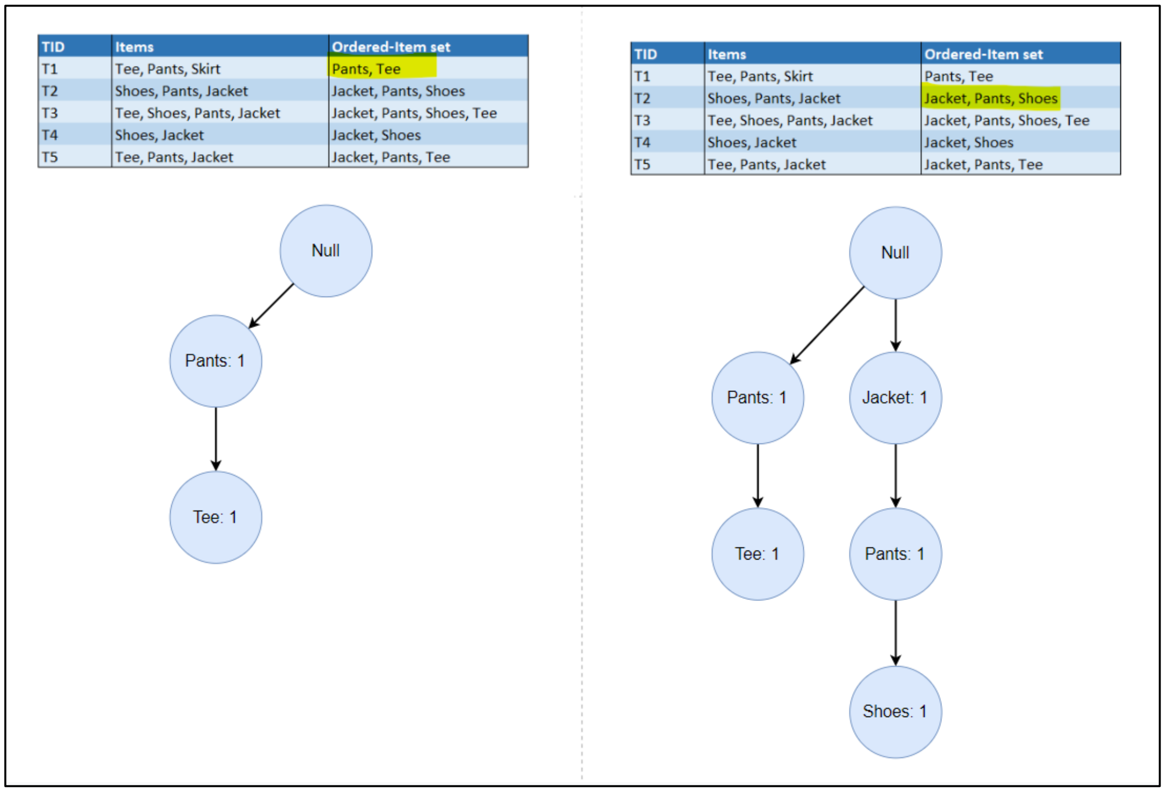


TID	Items	Ordered-Item set
T1	Tee, Pants, Skirt	Pants, Tee
T2	Shoes, Pants, Jacket	Jacket, Pants, Shoes
T3	Tee, Shoes, Pants, Jacket	Jacket, Pants, Shoes, Tee
T4	Shoes, Jacket	Jacket, Shoes
T5	Tee, Pants, Jacket	Jacket, Pants, Tee

Figure 3 11 – Illustration of applying FP-Growth Algorithm step 4

Step 5: Build to FP Tree - Trie Data Structure

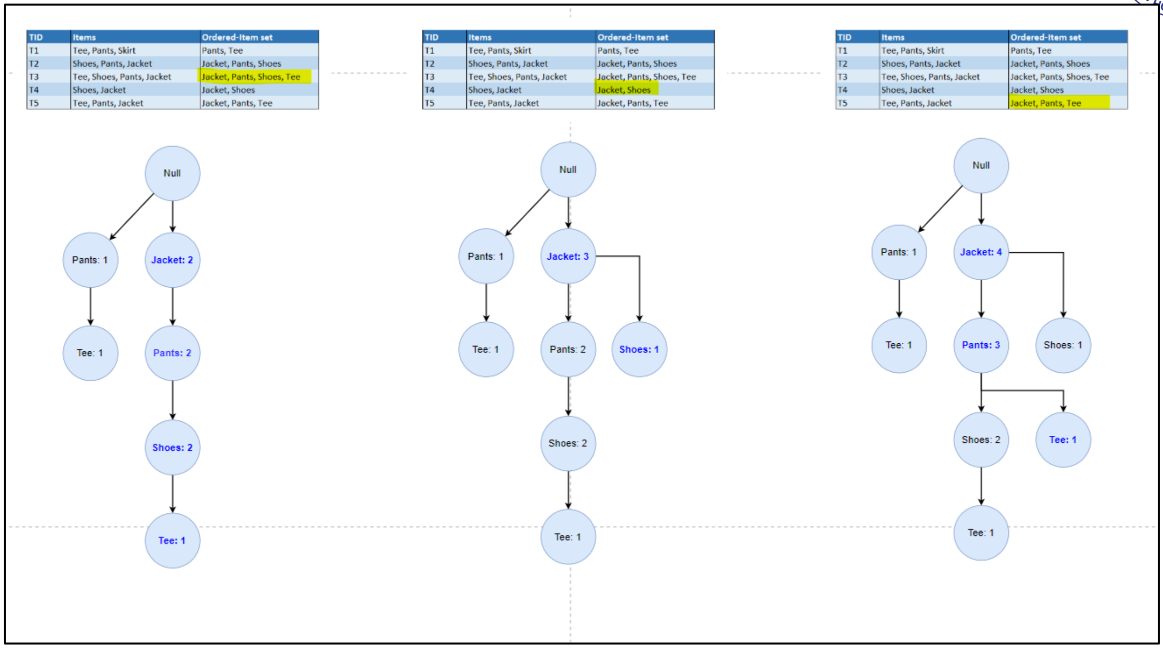
Scan the ordered itemset to construct the trie data structure. If any of the transaction's itemsets are already present in another branch, the transaction branch will have a shared root prefix. In addition, as transactions occur, the count of the itemset is increased. As nodes are formed and linked according to transactions, the count of both the common node and new node increases by one.



TID	Items	Ordered-Item set
T1	Tee, Pants, Skirt	Pants, Tee
T2	Shoes, Pants, Jacket	Jacket, Pants, Shoes
T3	Tee, Shoes, Pants, Jacket	Jacket, Pants, Shoes, Tee
T4	Shoes, Jacket	Jacket, Shoes
T5	Tee, Pants, Jacket	Jacket, Pants, Tee

TID	Items	Ordered-Item set
T1	Tee, Pants, Skirt	Pants, Tee
T2	Shoes, Pants, Jacket	Jacket, Pants, Shoes
T3	Tee, Shoes, Pants, Jacket	Jacket, Pants, Shoes, Tee
T4	Shoes, Jacket	Jacket, Shoes
T5	Tee, Pants, Jacket	Jacket, Pants, Tee

Figure 3 12 – Illustration of FP-Tree at step 5



TID	Items	Ordered-Item set
T1	Tee, Pants, Skirt	Pants, Tee
T2	Shoes, Pants, Jacket	Jacket, Pants, Shoes
T3	Tee, Shoes, Pants, Jacket	Jacket, Pants, Shoes, Tee
T4	Shoes, Jacket	Jacket, Shoes
T5	Tee, Pants, Jacket	Jacket, Pants, Tee

TID	Items	Ordered-Item set
T1	Tee, Pants, Skirt	Pants, Tee
T2	Shoes, Pants, Jacket	Jacket, Pants, Shoes
T3	Tee, Shoes, Pants, Jacket	Jacket, Pants, Shoes, Tee
T4	Shoes, Jacket	Jacket, Shoes
T5	Tee, Pants, Jacket	Jacket, Pants, Tee

TID	Items	Ordered-Item set
T1	Tee, Pants, Skirt	Pants, Tee
T2	Shoes, Pants, Jacket	Jacket, Pants, Shoes
T3	Tee, Shoes, Pants, Jacket	Jacket, Pants, Shoes, Tee
T4	Shoes, Jacket	Jacket, Shoes
T5	Tee, Pants, Jacket	Jacket, Pants, Tee

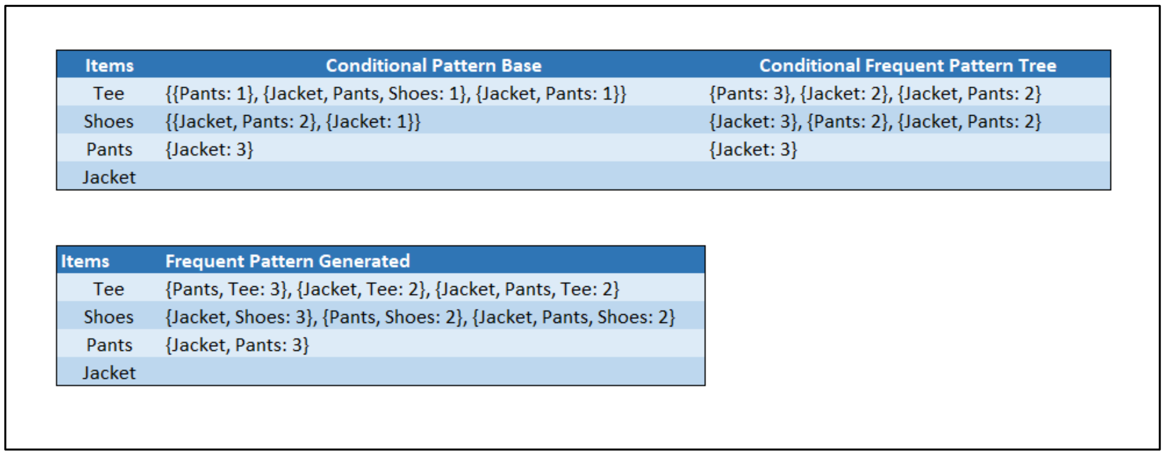
Figure 3 13 – Illustration of FP Tree with count of node have increased

Step 6: Generate Conditional Pattern Base (CPB): Conditional Pattern Base is computed which is path labels of all the paths which lead to any mode of the given item in frequent-pattern tree.

The lowest node, as well as the linkages between the lowest nodes, are evaluated first in order to mine the constructed FP Tree. The frequency pattern length 1 is represented by the lowest node. From this, traverse the path in the FP Tree.

Step 7: Form Conditional Frequent Pattern Tree: By taking the set of elements which is common in all the paths in CPB of that item and calculating its support count by summing the support counts in CPB. The Conditional FP Tree considers the itemsets that meet the threshold support.

Step 8: Generate Frequent Pattern Rules: Rules are generated by pairing the items of Conditional Frequent Pattern tree to the corresponding to the item.



Items	Conditional Pattern Base	Conditional Frequent Pattern Tree
Tee	{{Pants: 1}, {Jacket, Pants, Shoes: 1}, {Jacket, Pants: 1}}	{Pants: 3}, {Jacket: 2}, {Jacket, Pants: 2}
Shoes	{{Jacket, Pants: 2}, {Jacket: 1}}	{Jacket: 3}, {Pants: 2}, {Jacket, Pants: 2}
Pants	{Jacket: 3}	{Jacket: 3}
Jacket		

Items	Frequent Pattern Generated
Tee	{Pants, Tee: 3}, {Jacket, Tee: 2}, {Jacket, Pants, Tee: 2}
Shoes	{Jacket, Shoes: 3}, {Pants, Shoes: 2}, {Jacket, Pants, Shoes: 2}
Pants	{Jacket, Pants: 3}
Jacket	

Figure 3 14 – Result when applying FP-Growth Algorithm step 6, 7, 8

CHAPTER 4. IMPLEMENTATIONS AND RESULTS

4.1 Upload images to Cloudinary

4.1.1 Classify products by category

With this dataset, the number of item that satisfy the pre-processing condition and reach the min support value is 15,209 items distributed into 12 main categories as shown in Figure .

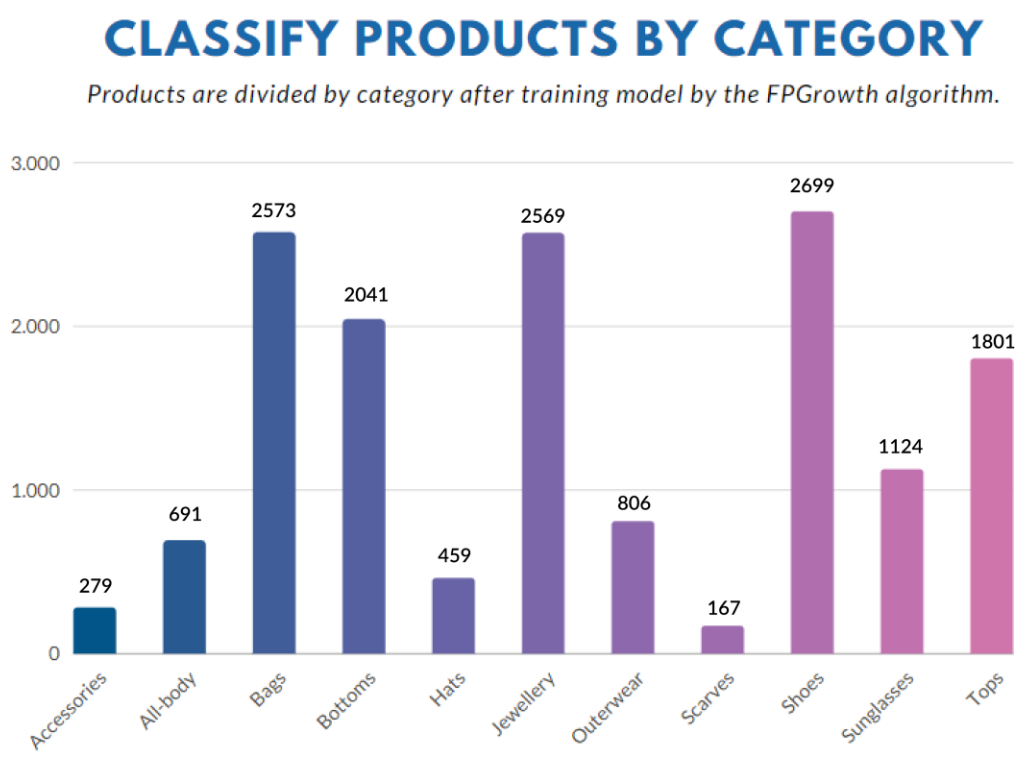
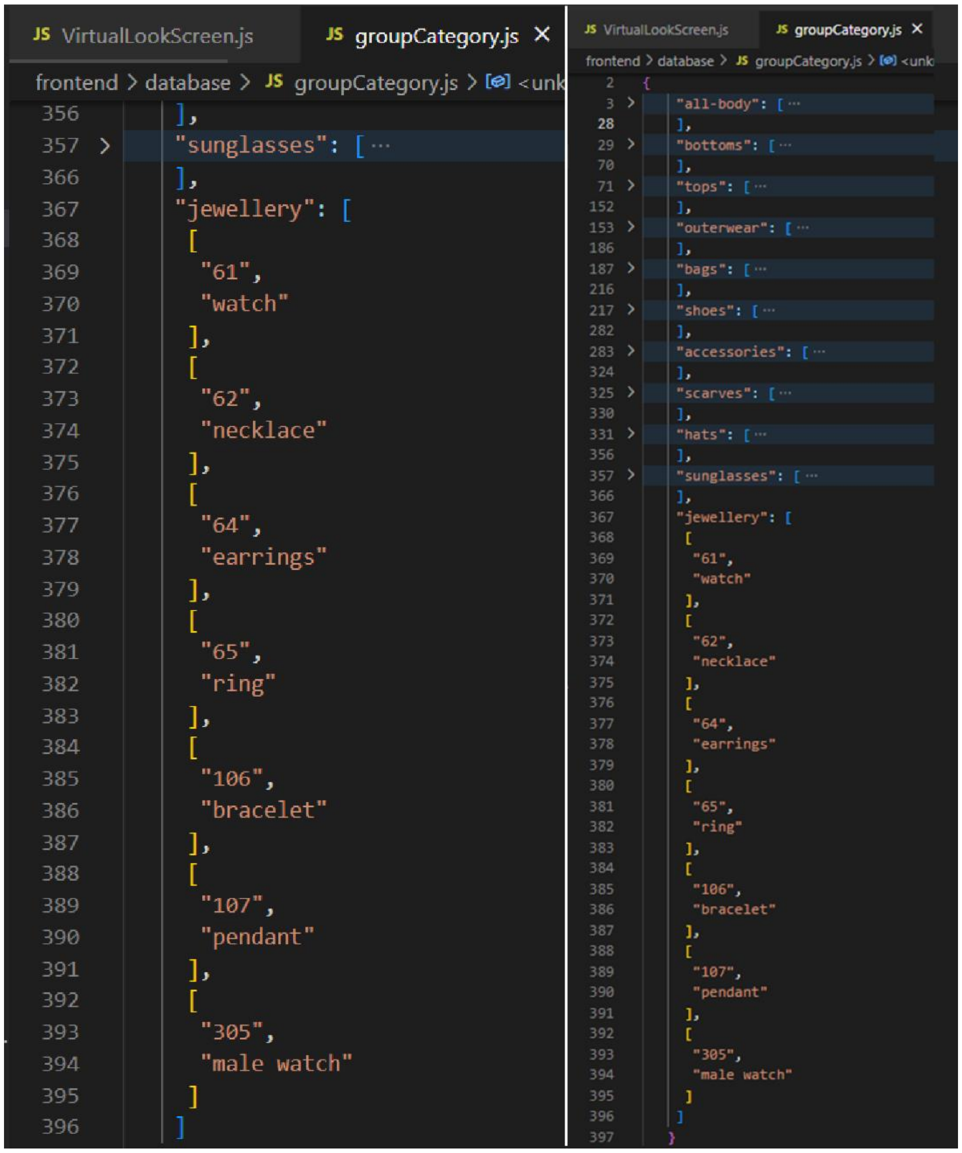


Figure 4 1 – Number of product within each categories



```
VirtualLookScreen.js    groupCategory.js ×
frontend > database > groupCategory.js > <unk
        ],
        "sunglasses": [⋯
        ],
        "jewellery": [
          [
            "61",
            "watch"
          ],
          [
            "62",
            "necklace"
          ],
          [
            "64",
            "earrings"
          ],
          [
            "65",
            "ring"
          ],
          [
            "106",
            "bracelet"
          ],
          [
            "107",
            "pendant"
          ],
          [
            "305",
            "male watch"
          ]
        ]
```

```
VirtualLookScreen.js    groupCategory.js ×
frontend > database > groupCategory.js > <unk
    {
        "all-body": [⋯
        ],
        "bottoms": [⋯
        ],
        "tops": [⋯
        ],
        "outerwear": [⋯
        ],
        "bags": [⋯
        ],
        "shoes": [⋯
        ],
        "accessories": [⋯
        ],
        "scarves": [⋯
        ],
        "hats": [⋯
        ],
        "sunglasses": [⋯
        ],
        "jewellery": [
          [
            "61",
            "watch"
          ],
          [
            "62",
            "necklace"
          ],
          [
            "64",
            "earrings"
          ],
          [
            "65",
            "ring"
          ],
          [
            "106",
            "bracelet"
          ],
          [
            "107",
            "pendant"
          ],
          [
            "305",
            "male watch"
          ]
        ]
    }
```

Figure 4 2 - Overview of category and Sub-category of jewelry

Inside each main categories contains sub-categories called fine-grained. For example, there are 7 sub-categories in jewelry: watch, necklace, earrings, ring, bracelet, pendant, male watch.

With a large and diverse number of products, the storage space for images is quite large, occupying up to 69.73 MB of memory. Classifying products by categories will make it easier to manage images on Cloudinary, and access images faster when photos are uploaded in separate folders for each of the 12 major categories.

Uploading product images to Cloudinary by drag and drop becomes extremely difficult because of the number of product images that need to be uploaded up to 15K. The warning message always pops up with standard drag and drop: "Browser not responding: Close program or Wait for the program to respond".

As a result, a Javascript program to automatically upload photographs is required, so that the images are uploaded in order and can make a verification of the upload status whether it was successful or not.

4.1.2 Add tags for the product’s image

After uploading photographs into 12 categories, the following step is to tag them in order to make searching for images using the tag's keyword more straightforward when utilizing Cloudinary's APIs when users do not know the product name. For example, the tag named "male" is added to the product's image in the male category for later searching and filtering on the website.

4.2 Training model

Following the preparation stage, the data is ready to be fed into the Mlxtend library, which will be trained to discover frequent sets and association rules. A vast number of data analysts and scientists enjoy and utilize the Mlxtend library.

Mlxtend (machine learning extensions) is a Python library of effective tools and extensions that provide many exciting functions for day-to-day data analysis, data science, and machine learning operational processes. Although numerous machine learning libraries for Python are available, including scikit-learn, TensorFlow, Keras, PyTorch, and others, however, MLxtend supplies additional functionalities and can be a valuable addition to the data science toolkit.

frequent_itemsets = fpgrowth(df, min_support=0.00005, use_colnames=True)

The chosen minimum support count is 0.00005 = 0.005% which is the percentage of the all transaction. The model has 0.005% support count and 53306 is the total transaction then in number the min support will be

$$53306 \text{ x } \frac{0.005}{100} = 2.66$$

With the property of polyvore item found on Kaggle, the enormous train data file includes 53306 different outfits (also known as transactions), and the ways in which the individual products are combined to generate a new set are very different. It is unlikely to have the same mix and match from two different users when each user has their own unique way of dressing. Therefore, it is only necessary to produce a product that appears over 2 times to participate in the model training process.

rule = association_rules(frequent_itemsets, metric="confidence", min_threshold = 0.01)

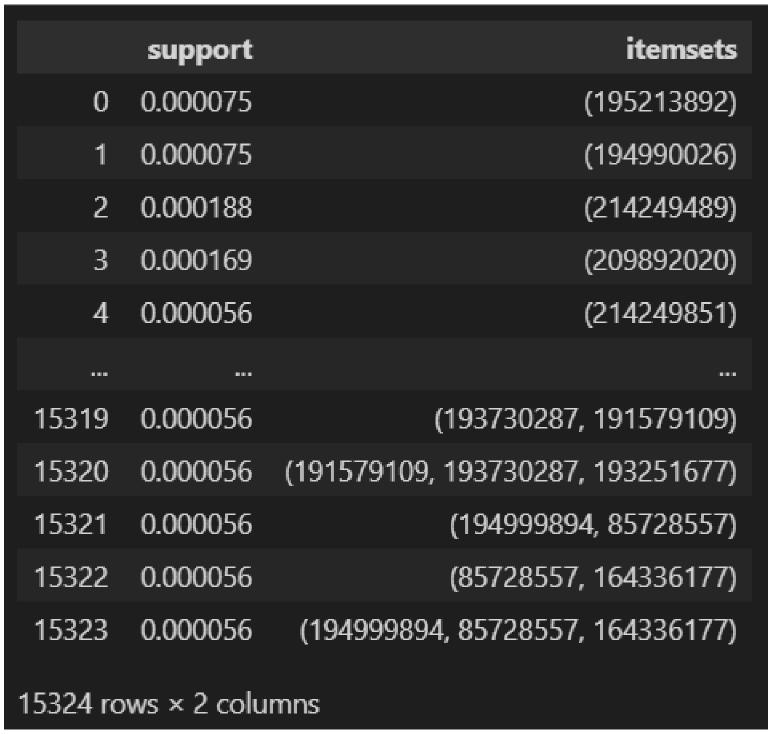
rule.sort_values('confidence', ascending=False)

The chosen confidence min threshold is 0.01 and sorted in descending order based on confidence value. Deliver results displayed on the FinnTA web application with higher priority for the combination used by most users.

4.3 Convert train result to JSON Server

The train_outfit.js file contains the number of transaction (number of outfit – the combination of clothes) is 53,306 sets. After the process of training model, giving result has been recorded as:

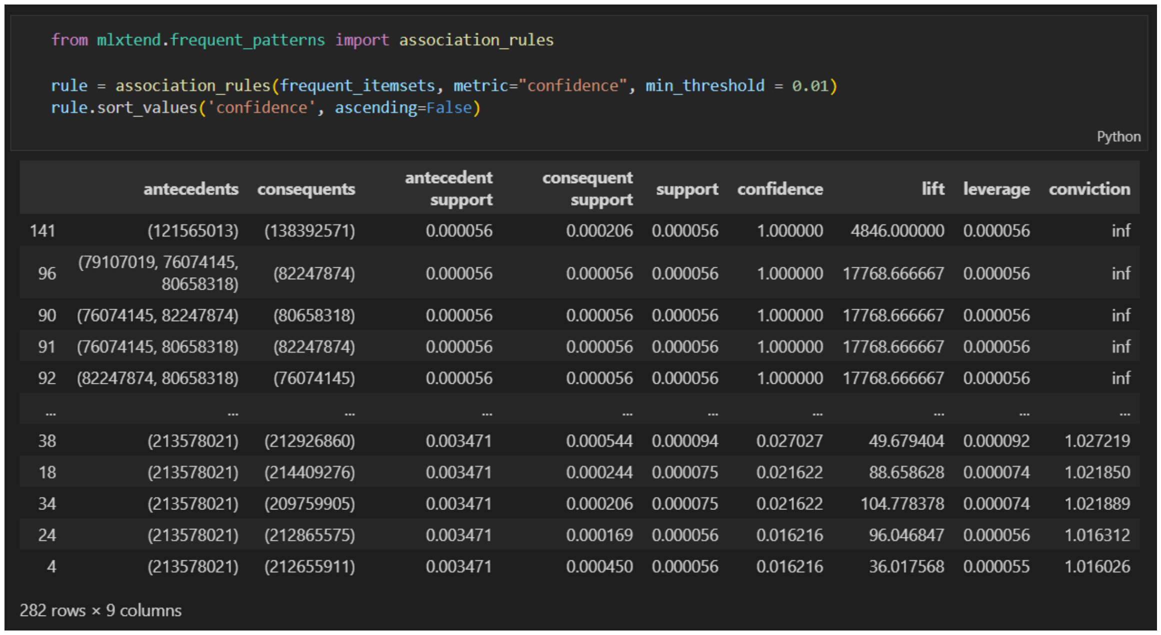
- ***15,324 itemsets*** which contains minimum and maximum amount of items in itemset is 1, and 3 respectively.
- ***15,209 single items*** that satisfy the minimum support value
- ***282 rules*** is generated in a set of association rule.



	support	itemsets
0	0.000075	(195213892)
1	0.000075	(194990026)
2	0.000188	(214249489)
3	0.000169	(209892020)
4	0.000056	(214249851)
...	...	...
15319	0.000056	(193730287, 191579109)
15320	0.000056	(191579109, 193730287, 193251677)
15321	0.000056	(194999894, 85728557)
15322	0.000056	(85728557, 164336177)
15323	0.000056	(194999894, 85728557, 164336177)

15324 rows × 2 columns

Figure 4 3 – Result of itemset after training model by FP-Growth



```
from mlxtend.frequent_patterns import association_rules

rule = association_rules(frequent_itemsets, metric="confidence", min_threshold = 0.01)
rule.sort_values('confidence', ascending=False)
```

Python

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
141	(121565013)	(138392571)	0.000056	0.000206	0.000056	1.000000	4846.000000	0.000056	inf
96	(79107019, 76074145, 80658318)	(82247874)	0.000056	0.000056	0.000056	1.000000	17768.666667	0.000056	inf
90	(76074145, 82247874)	(80658318)	0.000056	0.000056	0.000056	1.000000	17768.666667	0.000056	inf
91	(76074145, 80658318)	(82247874)	0.000056	0.000056	0.000056	1.000000	17768.666667	0.000056	inf
92	(82247874, 80658318)	(76074145)	0.000056	0.000056	0.000056	1.000000	17768.666667	0.000056	inf
...	...	...	...	...	...	...	...	...	...
38	(213578021)	(212926860)	0.003471	0.000544	0.000094	0.027027	49.679404	0.000092	1.027219
18	(213578021)	(214409276)	0.003471	0.000244	0.000075	0.021622	88.658628	0.000074	1.021850
34	(213578021)	(209759905)	0.003471	0.000206	0.000075	0.021622	104.778378	0.000074	1.021889
24	(213578021)	(212865575)	0.003471	0.000169	0.000056	0.016216	96.046847	0.000056	1.016312
4	(213578021)	(212655911)	0.003471	0.000450	0.000056	0.016216	36.017568	0.000055	1.016026

282 rows × 9 columns

Figure 4 4 Result of Association Rule after training model by FP-Growth algorithms

After obtaining this data and deciding to show training data on the web application's user interface, the following step is to transform the data format to meet the user interface displayable format. The mlxtend library's training data is stored in CSV format, however the data format is difficult to read and cannot be searched since its information format does not support searching.

Therefore, at the data format conversion stage, it is necessary to:

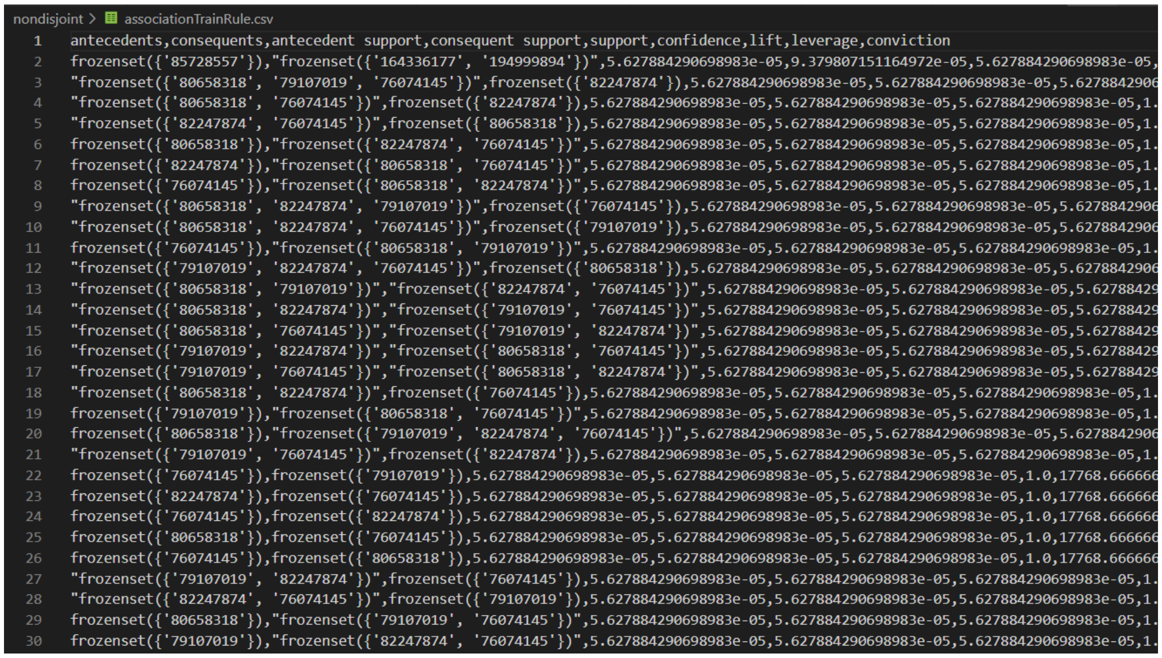
- Remove the unused information
- Convert the file format from CSV to JSON and from JSON to JS

in order to insert into the JSON Server. We can implement the API Query parameter to look for specific data to display by converting it to a JSON file.

The optional key-value pairs that occur after the question mark in the URL can be configured as API Query parameters. In a nutshell, they are URL extensions that help decide certain content or actions based on the supplied data.

The final result we will get after this stage includes:

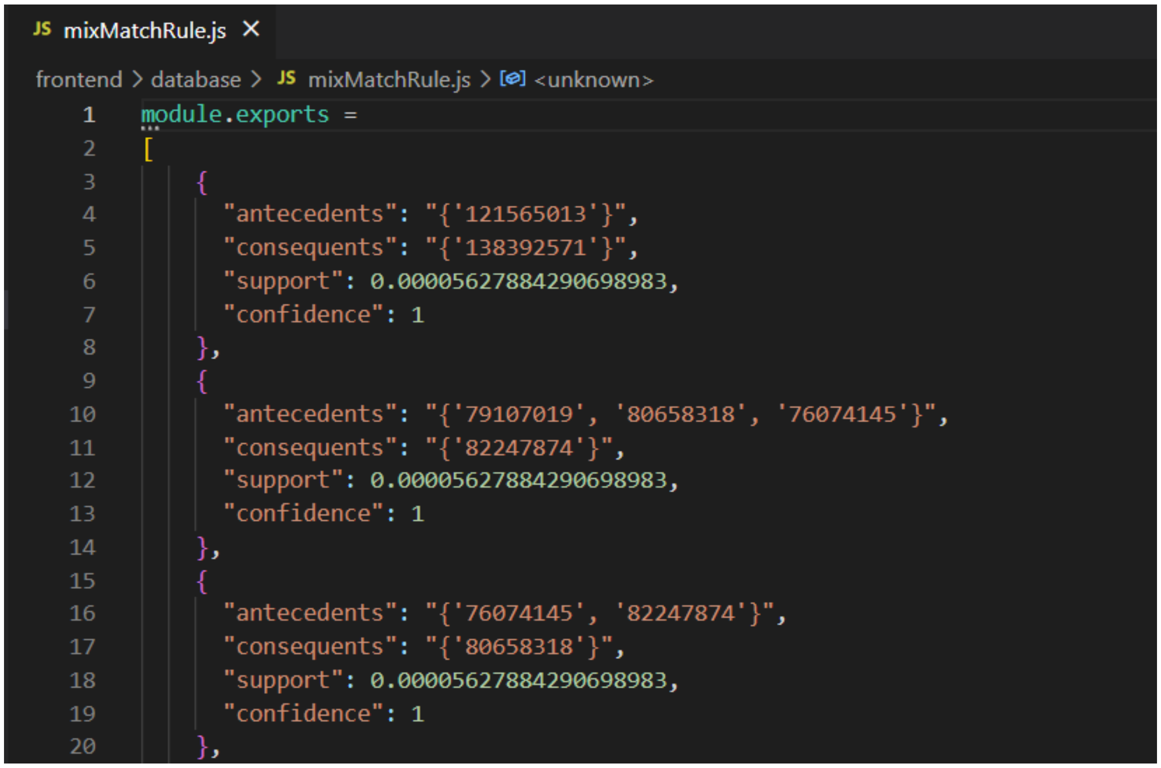
- A list of products that meet the model's minimum support value
- A list of association rule including useful parameters like antecedents, consequences, support and confidence.



nondisjoint > associationTrainRule.csv

```
antecedents,consequents,antecedent support,consequent support,support,confidence,lift,leverage,conviction
frozenset({'85728557'}),"frozenset({'164336177', '194999894'})",5.627884290698983e-05,9.379807151164972e-05,5.627884290698983e-05,
"frozenset({'80658318', '79107019', '76074145'})",frozenset({'82247874'}),5.627884290698983e-05,5.627884290698983e-05,5.6278842906
"frozenset({'80658318', '76074145'})",frozenset({'82247874'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
"frozenset({'82247874', '76074145'})",frozenset({'80658318'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'80658318'}),"frozenset({'82247874', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'82247874'}),"frozenset({'80658318', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'76074145'}),"frozenset({'80658318', '82247874'})",5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
"frozenset({'80658318', '82247874', '79107019'})",frozenset({'76074145'}),5.627884290698983e-05,5.627884290698983e-05,5.6278842906
"frozenset({'80658318', '82247874', '76074145'})",frozenset({'79107019'}),5.627884290698983e-05,5.627884290698983e-05,5.6278842906
frozenset({'76074145'}),"frozenset({'80658318', '79107019'})",5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
"frozenset({'79107019', '82247874', '76074145'})",frozenset({'80658318'}),5.627884290698983e-05,5.627884290698983e-05,5.6278842906
"frozenset({'80658318', '79107019'})","frozenset({'82247874', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.62788429
"frozenset({'80658318', '82247874'})","frozenset({'79107019', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.62788429
"frozenset({'80658318', '76074145'})","frozenset({'79107019', '82247874'})",5.627884290698983e-05,5.627884290698983e-05,5.62788429
"frozenset({'79107019', '82247874'})","frozenset({'80658318', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.62788429
"frozenset({'79107019', '76074145'})","frozenset({'80658318', '82247874'})",5.627884290698983e-05,5.627884290698983e-05,5.62788429
"frozenset({'80658318', '82247874'})",frozenset({'76074145'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'79107019'}),"frozenset({'80658318', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'80658318'}),"frozenset({'79107019', '82247874', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.6278842906
"frozenset({'79107019', '76074145'})",frozenset({'82247874'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'76074145'}),frozenset({'79107019'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.0,17768.666666
frozenset({'82247874'}),frozenset({'76074145'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.0,17768.666666
frozenset({'76074145'}),frozenset({'82247874'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.0,17768.666666
frozenset({'80658318'}),frozenset({'76074145'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.0,17768.666666
frozenset({'76074145'}),frozenset({'80658318'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.0,17768.666666
"frozenset({'79107019', '82247874'})",frozenset({'76074145'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
"frozenset({'82247874', '76074145'})",frozenset({'79107019'}),5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'80658318'}),"frozenset({'79107019', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
frozenset({'79107019'}),"frozenset({'82247874', '76074145'})",5.627884290698983e-05,5.627884290698983e-05,5.627884290698983e-05,1.
```

Figure 4 6 – Data format of Association Rule before converting process



mixMatchRule.js

frontend > database > mixMatchRule.js > <unknown>

```
module.exports =
[
    {
      "antecedents": "{'121565013'}",
      "consequents": "{'138392571'}",
      "support": 0.00005627884290698983,
      "confidence": 1
    },
    {
      "antecedents": "{'79107019', '80658318', '76074145'}",
      "consequents": "{'82247874'}",
      "support": 0.00005627884290698983,
      "confidence": 1
    },
    {
      "antecedents": "{'76074145', '82247874'}",
      "consequents": "{'80658318'}",
      "support": 0.00005627884290698983,
      "confidence": 1
    },
```

Figure 4 5 – Data format of Association Rule after converting process

4.4 Apply Lazy Loading

With a large amount of data and product images, it is not possible for the website to load all the images of each category in the Cloudinary image folder at once. Loading 15K images at once will lead to a slow website or no response, forcing to end the action.

Therefore, this web application have applied Lazy loading, which Cloudinary supports. Only when the users click to choose which sub-category they would like to view the product, then the system will call the APIs to get the image in an exact folder of the chosen category in Cloudinary.

Lazy loading essentially means that the pictures are not loaded until the user accesses them. Only the photos that are required are ever loaded, potentially saving gigabytes upon terabytes of bandwidth. By doing this, the website may minimize page weight, resulting in a faster page load time, preserve bandwidth by only providing material to visitors only if it is requested, and save both server and client resources.

4.5 User Interface

As a user of our website:

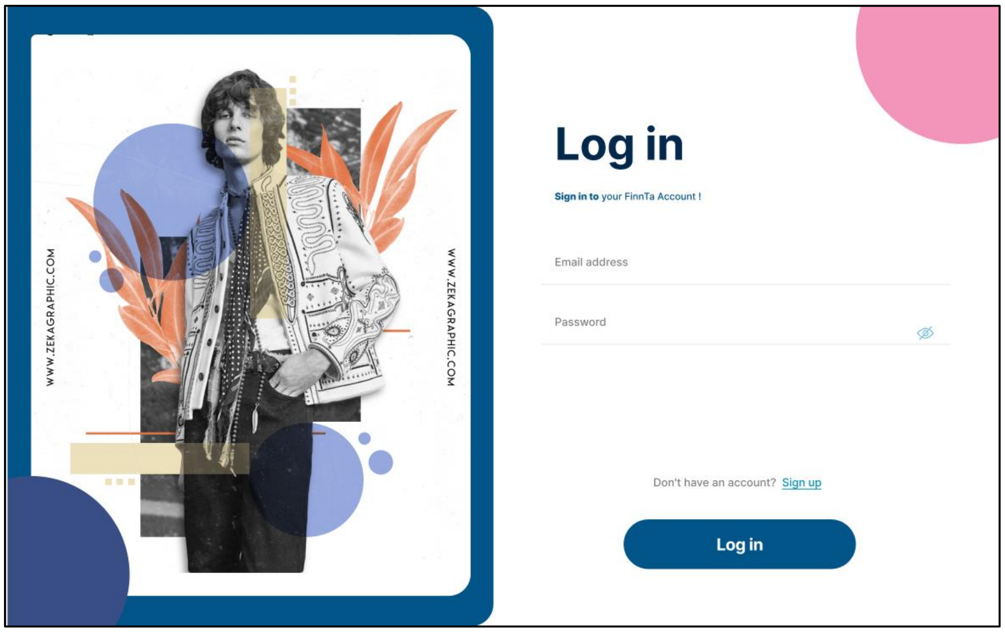
- Registering a new account, sign in to the website
- Uploading new product
- Creating a collection
- Editing own products
- Collecting other's user products
- Editing a profile and viewing other user's profile
- View, select to mix a fashion product
- Combine many separate items (both their own clothes and system clothes) to generate an outfit of the day, adjusting order, the position of the item in the whole set's image
- Save outfit to later occasions
- Receive the recommendation clothes to support the mix and match

4.6 Result

The website have been implemented into many screen with different functionality.

4.6.1 Login/ Sign up screen

There are 2 roles in our website: Business such as: shopping store, amatuer seller and User. To ensure the security of website, we have implemented the Hide/Show password by clicking the eye icon.



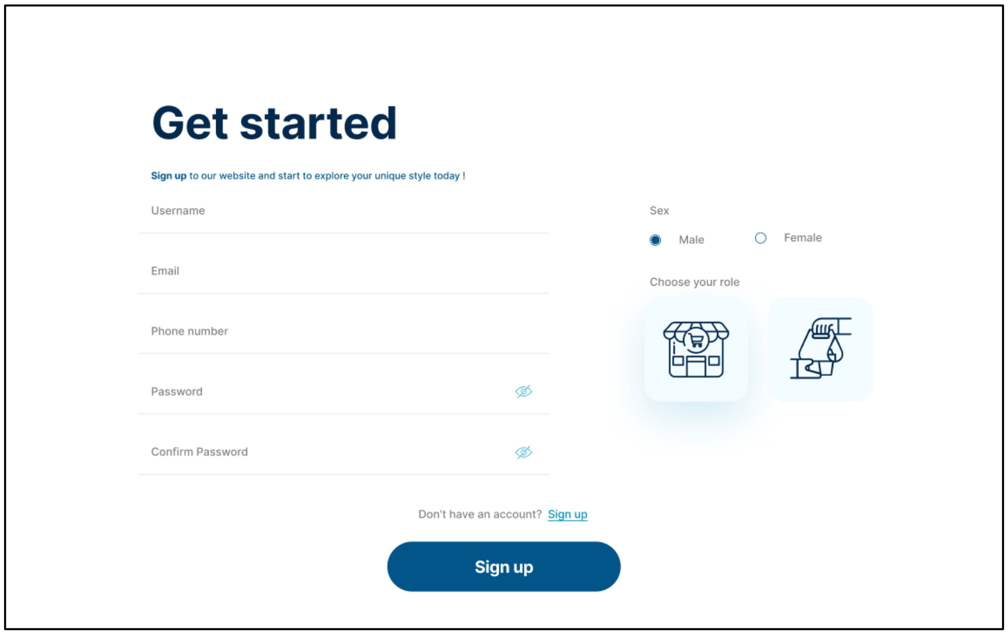
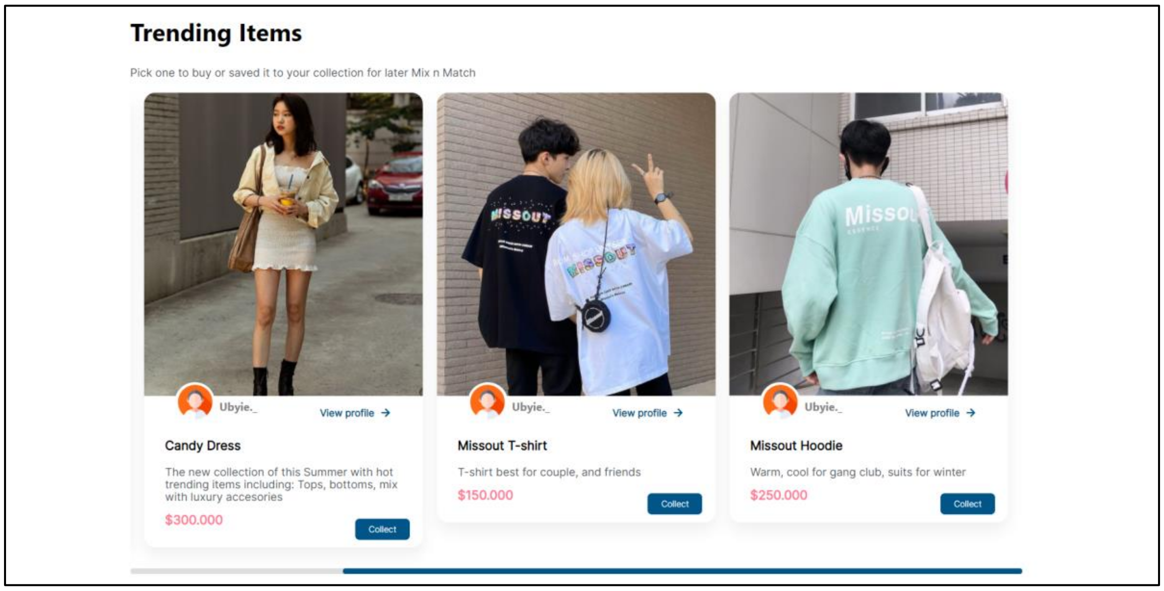

Get started
Sign up to our website and start to explore your unique style today !
Username
Email
Phone number
Password
Confirm Password
Sex
Male
Female
Choose your role
Don't have an account? Sign up
Sign up

Figure 4 7 – Login and Sign up Screen

4.6.2 Homepage screen


Trending Items

Pick one to buy or saved it to your collection for later Mix n Match

Candy Dress

The new collection of this Summer with hot trending items including: Tops, bottoms, mix with luxury accesories

$300.000

Missout T-shirt

T-shirt best for couple, and friends

$150.000

Missout Hoodie

Warm, cool for gang club, suits for winter

$250.000

Figure 4 8 – Homepage Screen

Homepage screeen contains:

1. Navigtion bar: About – introduce about us, about our team and mission, Home, Virual Look – specific point about web application. Virtual Look for mix and match clothes to create outfit of the day or a new collection
2. Carousel – to advertise our news, event and subscription package
3. Trending item list recommendation – display product to buy or to save for later mix and match
4. Collection list recommendation – inspire user fashionable outfit everyday, how to mix clothes tips
5. Footer – contains our contact info, support and customer service team via social media such as: Facebook, Twitter and Instagram

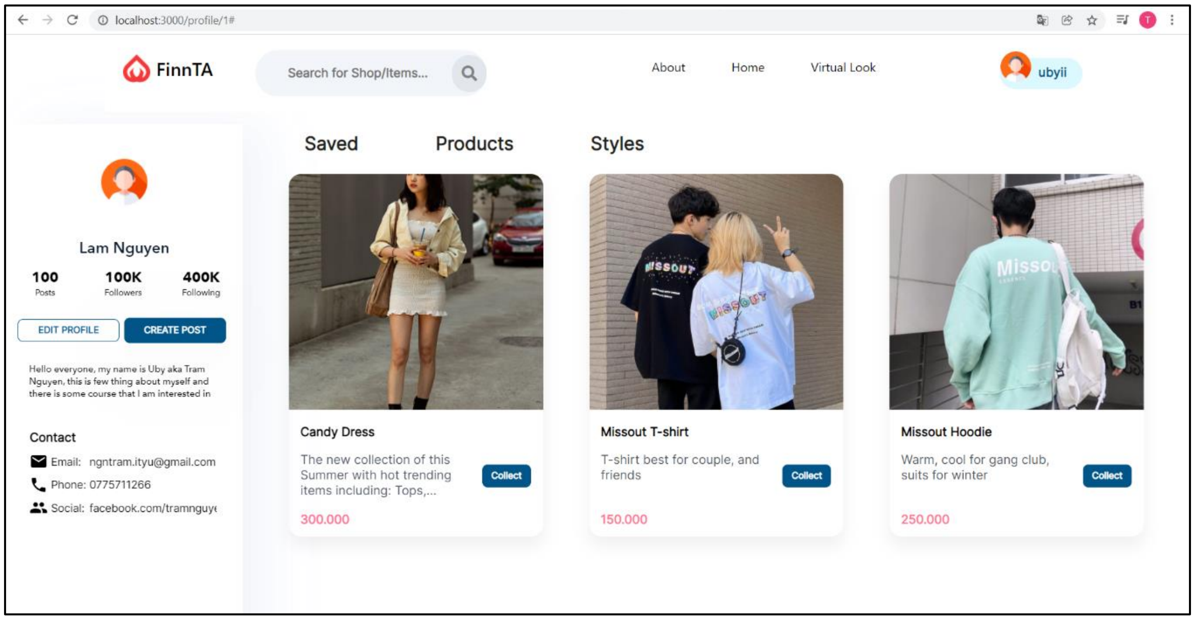
These can be viewed as a visitor – who does not have an account to log in to our website. By doing this, we can create a first impression, first experience to attact more user to our website. When new user click some button to register course, the system will lead them to the sign up screen.

4.6.3 Profile screen

The profile screen contains dashbody: Saved – saved product for later mix and match, Product – upload item of that user and Log out.

Profile displays general information: username, number of post, followers, following account, and their contact information: email address, phone number and social media link.

In the Profile window, user can edit profile, create post to upload their product.



Hello everyone, my name is Uby aka Tram Nguyen, this is few thing about myself and there is some course that I am interested in

Contact
Email: ngntram.ityu@gmail.com
Phone: 0775711266
Social: facebook.com/tramnguy

Candy Dress
The new collection of this Summer with hot trending items including: Tops,...
300.000

Missout T-shirt
T-shirt best for couple, and friends
150.000

Missout Hoodie
Warm, cool for gang club, suits for winter
250.000

Figure 4 9 – Profile Screen

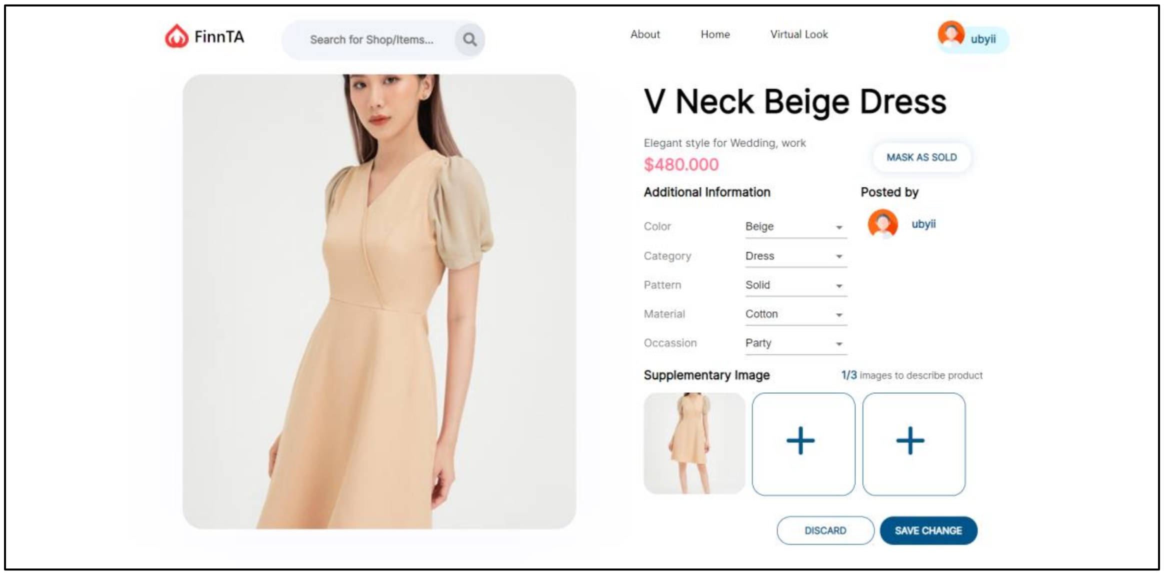
If user views their profile, they are allowed to edit their product's posts. Otherwise, they can select, save product of others for later mix and match by clicking "Collect" button which help to determine whether they should buy that product or not.

4.6.4 Edit Product Screen

In Edit product screen, owner has authority to edit their product post's detail includes: color, category, pattern, material, occasion and add maximum 3 supplementary images for each post.

After uploading supplementary image from their computer, user can re-choose the image by clicking the cross button ('X') to delete image.

In the case that the product is sold through chat, through another social media, or out of stock, user can click "Mark as sold" button to hide the product from available product in the profile and homepage.



V Neck Beige Dress

Elegant style for Wedding, work

$480.000

Additional Information

Color	Beige
Category	Dress
Pattern	Solid
Material	Cotton
Occassion	Party

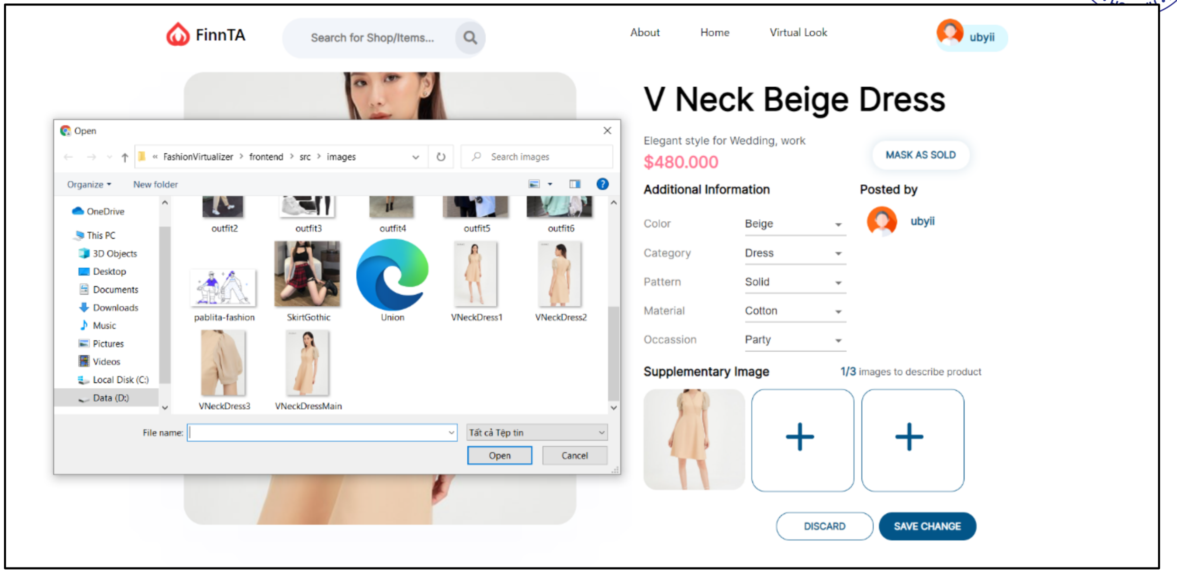
Posted by

ubyii

Supplementary Image

1/3 images to describe product

Figure 4 10 – Edit Product Detail Screen



V Neck Beige Dress

Elegant style for Wedding, work

$480.000

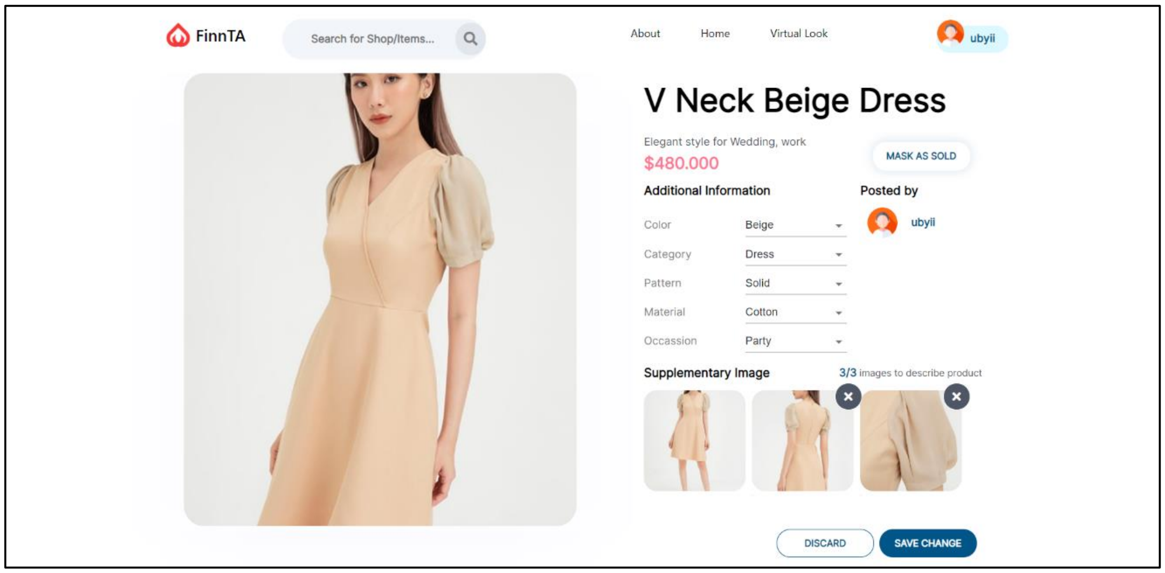
Additional Information

Color	Beige
Category	Dress
Pattern	Solid
Material	Cotton
Occassion	Party

Posted by ubyii

Supplementary Image — 1/3 images to describe product

Figure 4 11 – Upload Image for Product Screen



V Neck Beige Dress

Elegant style for Wedding, work

$480.000

Additional Information

Color	Beige
Category	Dress
Pattern	Solid
Material	Cotton
Occassion	Party

Posted by ubyii

Supplementary Image — 3/3 images to describe product

Figure 4 12 Supplementary Image for Product Screen after update

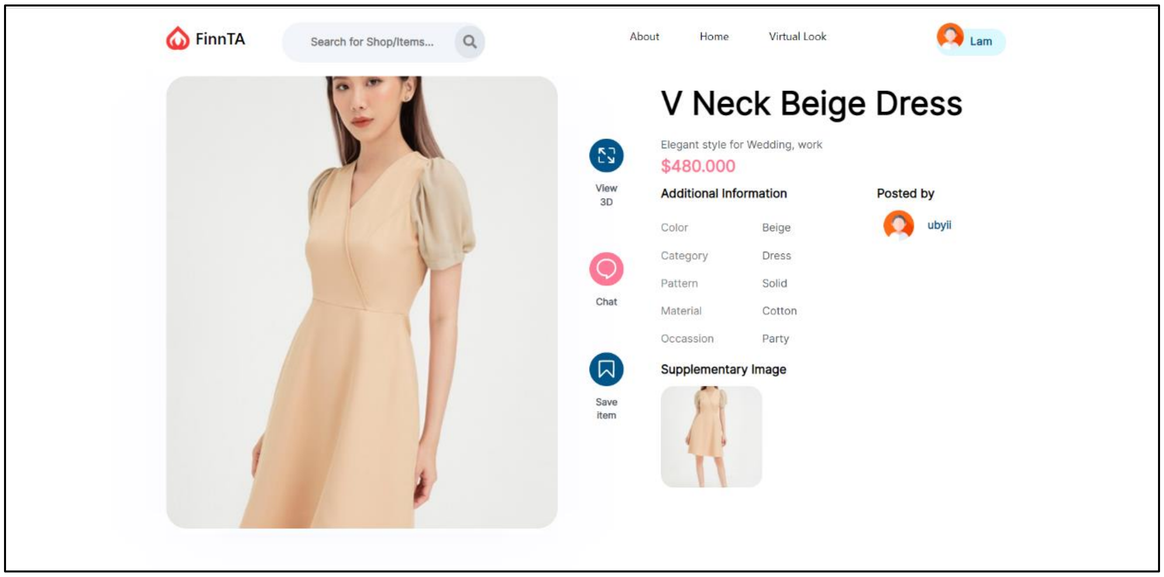
4.6.5 Product Detail Screen

Product Detail Screen display information about the product including the product's name, price, description, owner, and additional information. There are 3 functional buttons: View 3D, Chat, Save item.

View 3D: to display product in 3D. However, there is no 3D generated object for all the product, some product is supported, some is not.

Chat: to support the chatting between users in the community, they can negotiate with each other about the price, the location to view the product, or shipping price, delivery address.

Save item: to collect the item which is owned by another user for later mix and match to generate a new collection.



V Neck Beige Dress

Elegant style for Wedding, work

$480.000

Additional Information

Color	Beige
Category	Dress
Pattern	Solid
Material	Cotton
Occassion	Party

Posted by

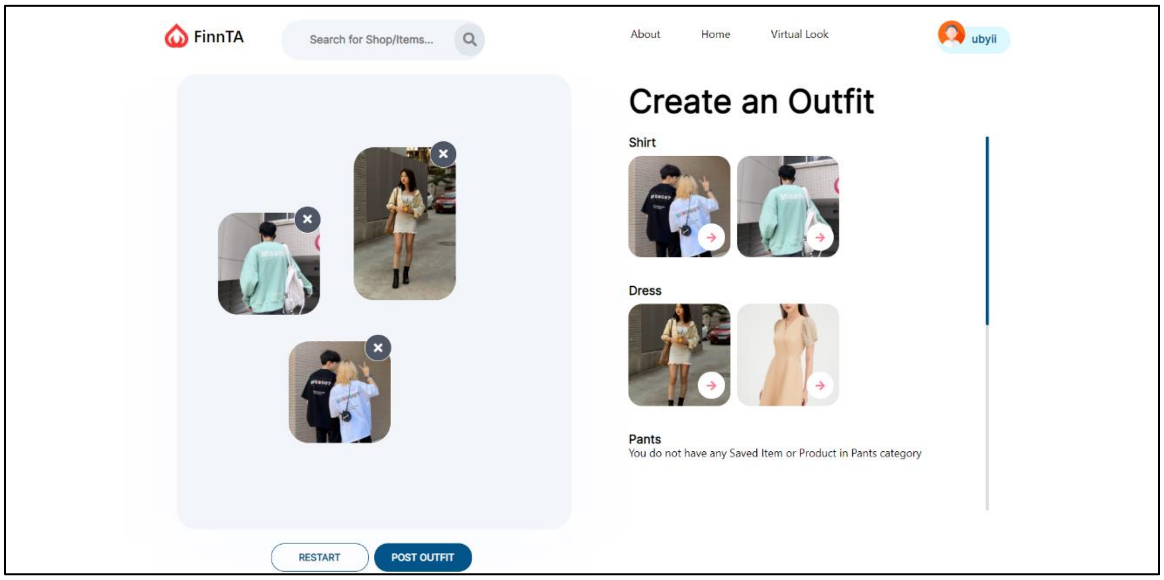
ubyli

Supplementary Image

Figure 4 13 – Product Detail Screen

4.6.6 Create an Outfit Screen

Virtual Look Screen is used for mixing and matching items, arranging the position for an item in the set. Clothes in this virtual wardrobe include users' products – is posted by themselves and saved product – is posted by another user and is collected, saved for later try on in the set.

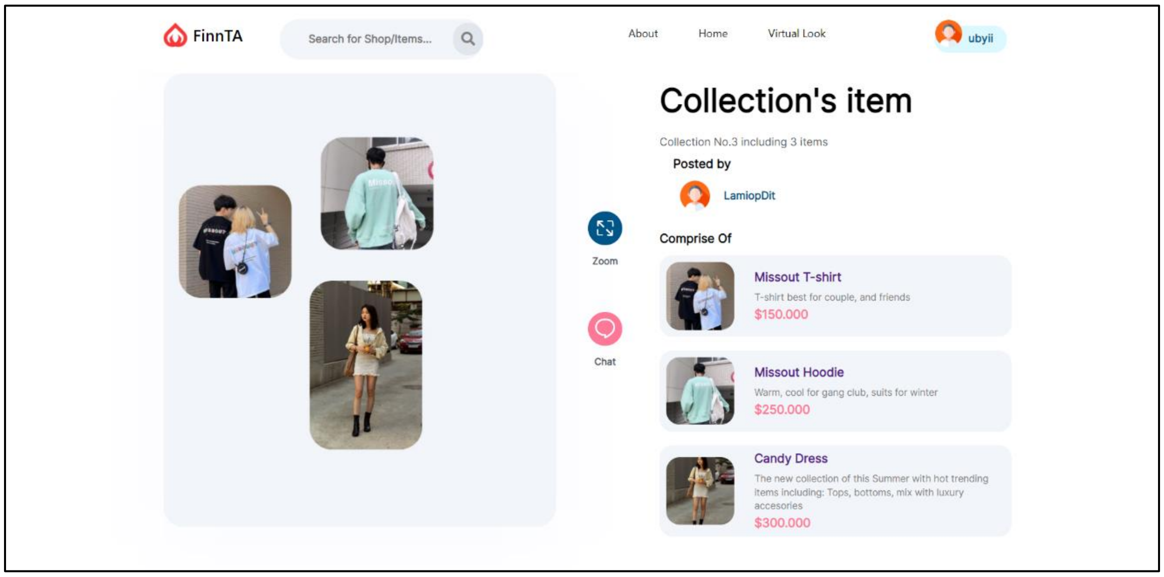


You do not have any Saved Item or Product in Pants category

Figure 4 14 – Create Outfits/Collections by separate item Screen

4.6.7 Virtual Look Detail Screen

Detail Screen of collection displays the number of item in a set, the information contains: product's name, description and price of all the item build up a collection



Collection's item

Collection No.3 including 3 items

Posted by

LamiopDit

Comprise Of

Missout T-shirt
T-shirt best for couple, and friends
$150.000

Missout Hoodie
Warm, cool for gang club, suits for winter
$250.000

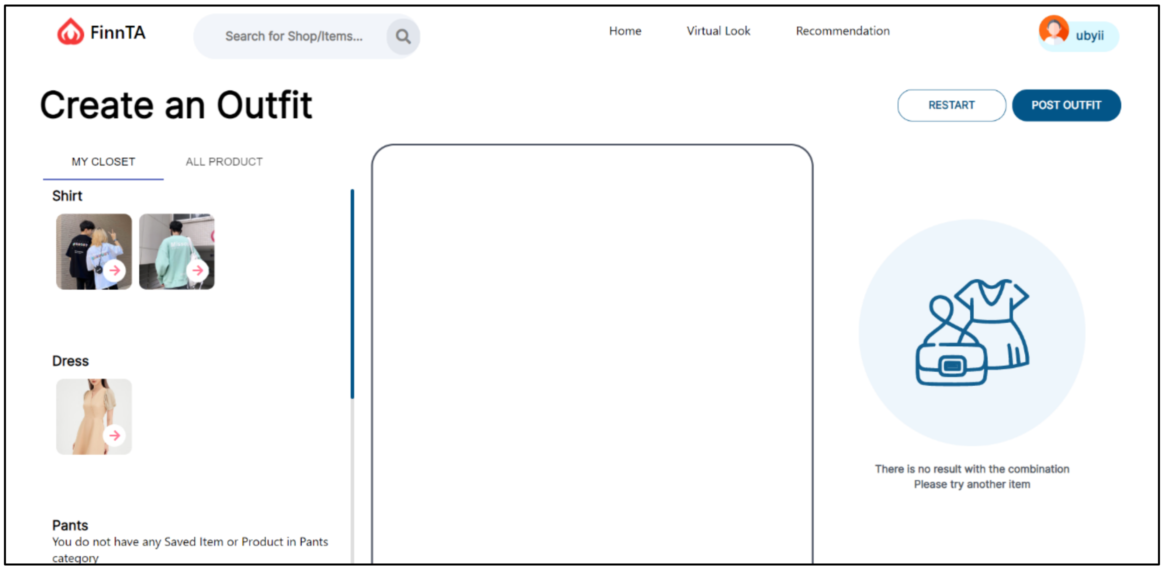
Candy Dress
The new collection of this Summer with hot trending items including: Tops, bottoms, mix with luxury accesories
$300.000

Figure 4 15 – Collections Detail Screen

4.6.8 Recommendation Screen

In this screen, users can both mix and match their clothes together in a personal closet and be able to combine their own clothes which they have uploaded into the website with clothes in the system which is posted by a business account, or clothing store owner in the application.

There are two tabs for choosing the clothes: My Closet refers to the users' personal wardrobe and All Product – refers to all public items in the FinnTA web app. In the All Product tab, the products are contributed to each main category and sub-categories.



You do not have any Saved Item or Product in Pants category

There is no result with the combination
Please try another item

Figure 4 16 – Recommendation Screen with Empty result

With the use of Lazy loading, only the product's images in chosen sub-category will be loaded and displayed in the UI. As the Figure 4.2.7.1, the lazy loading has been applied and let the user waiting for loading products in Top category.

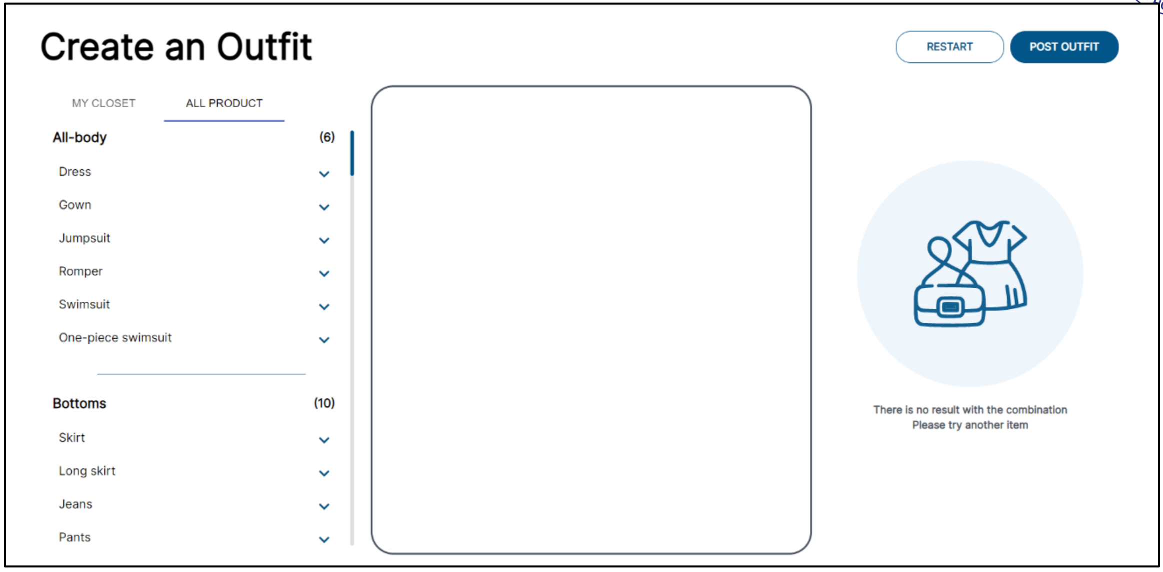


Figure 4 18 – Recommendation Screen in All Product tab

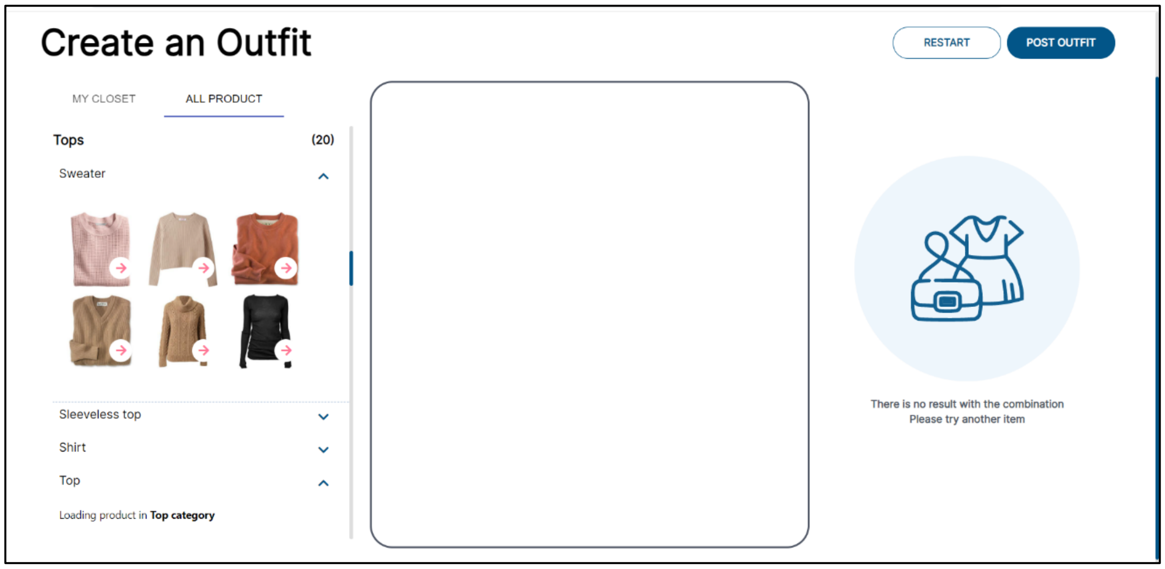


Figure 4 17 – Recommendation Screen apply Lazy Loading

After selecting the item to mix and match, the system will suggest the most suitable item to combine in order to create a complete outfit that people in the app adore when the pieces are put together based on the association rule that has been created in training model process.

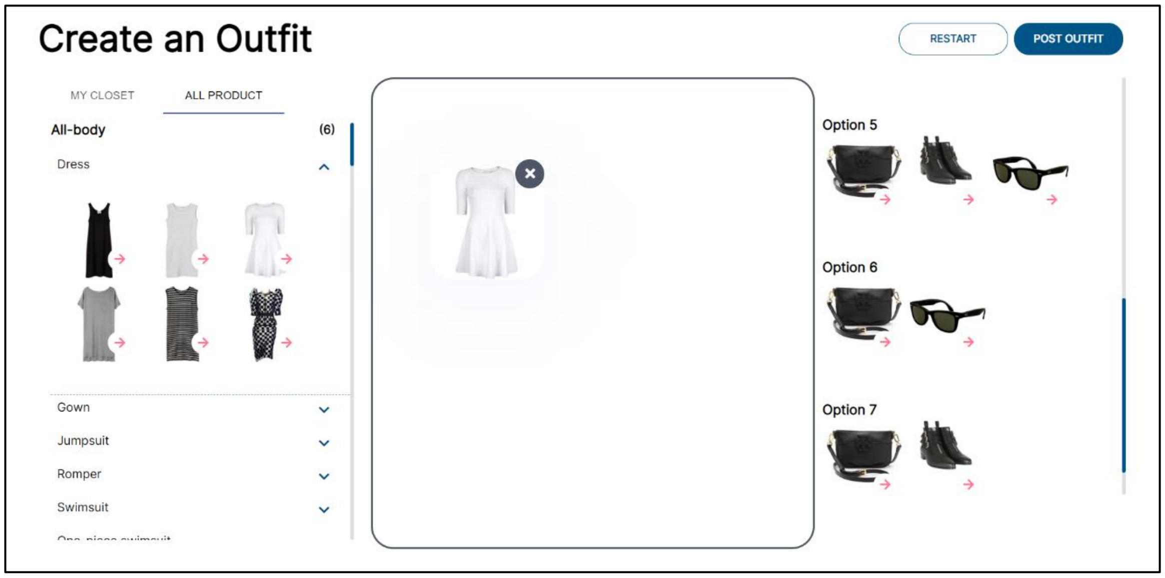


Figure 4 20 – Recommended item based on added item in set

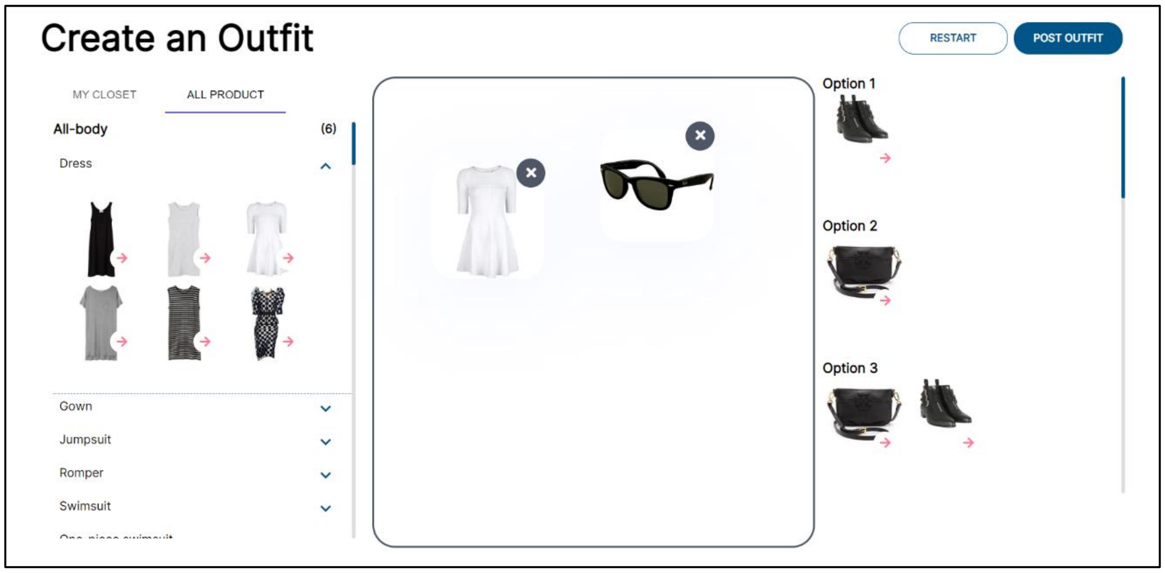
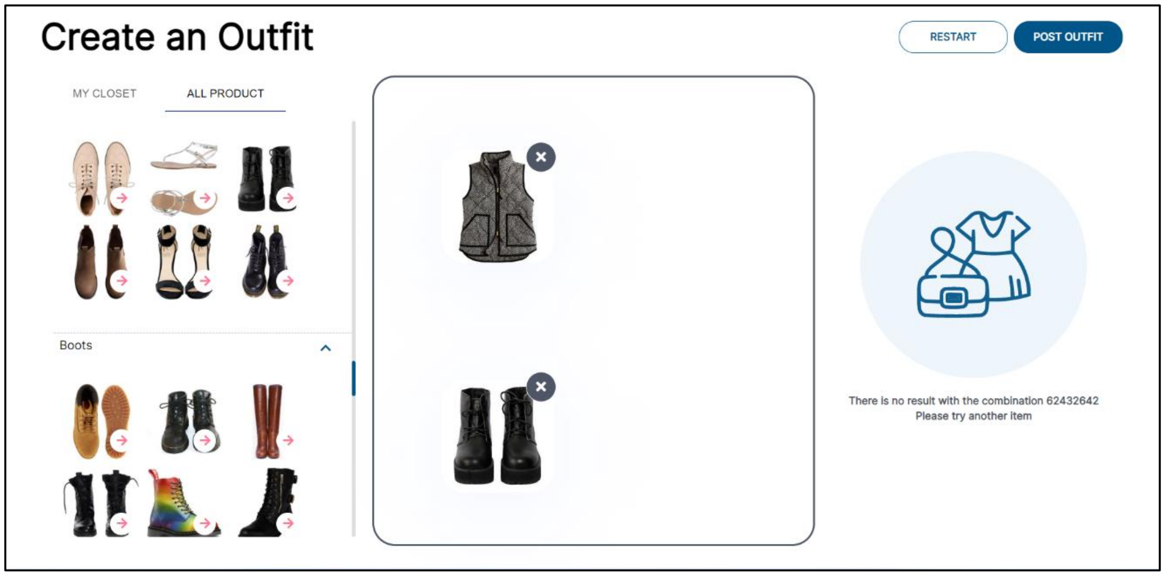


Figure 4 19 – Recommended item change when adding new item to set based on association rule result

When the system cannot find any other item that suits the added item to set, it will display the No Result indicator and show the item id of each product in the set.

Note that, only the item provided by the system can be suggested to mix and match. The item which is uploaded by the user cannot be suggested with any other item in the system.

However, the uploaded item can be suggested which products in the category are missing. For example, the user enters a shirt and a dress, the system will detect that a pair of shoes or a little accessory is needed to complete a set of clothes. From there, the user will be suggested by random shoes, jewelry in the system.



There is no result with the combination 62432642
Please try another item

Figure 4 21 – No result to display in Recommendation Screen

4.7 Discussion

This section may lead to several evaluations. This discussion is related to the problem to be worked on.

Discuss both resources and development:

How long does app development take?, What problems are the members facing when programming? How can the app reach more users?

Thereby, solving with a specific plan for the above problems as quickly as possible, from there, all activities and processes are carried out more smoothly

In the thesis assessment report, I have finished the color scheme, user interface of the web application, some features to support the illustration of the recommendation system.

4.8 Comparison

Compare to the Mix Dress website, we have implemented a clearer layout and user interface more friendly and easy to use for the users. We have a virtual look feature, a profile page to store their product and collect other users' products for mix and match.

Compare to Virtuality Fashion, we have not finished implementing to generate 3D objects automatically and we can only adjust the colour while they can change the pattern of the products.

4.9 Evaluation

Performance of functional and non-functional requirement of Web Application

Almost 80% complete for the interface of the website and some functional requirements of the website such as register an account, log in, browsing and database search, post product to sell, save other products, mix and match separate items to generate a collection.

Other non-functional requirements include:

- Usability and reliability requirements as providing visitors with a friendly user interface that is easy to navigate, security requirements as adjusting the hide/show password by clicking the eye icon,
- Interactive and good performance: users shall be acknowledged in the form of visual changes and the response time and throughput time on the site are minimal. Consistency on the website shall be maintained across all the web pages: the same concept of UI, use same colour scheme: navy blue and pink, responsive web design - that looks good on nearly all devices, all sizes of the screen: 1920px, 1440px, etc. The layout of the site shall be kept simple and must be self-explanatory

Performance of Apriori – a frequent itemset mining algorithm

Frequent pattern mining is a well-studied area of data mining, and there are a variety of methods for mining frequent patterns, each of which performs differently on different datasets. As mentioned the limitation of the Apriori algorithm in section 3.4.5, Apriori is not suitable for large datasets, researching and trying to implement another initial basic algorithm used for frequent pattern mining known as FP Growth is under consideration based on several research about comparative analysis between FP Growth, Apriori and Eclat. [9]

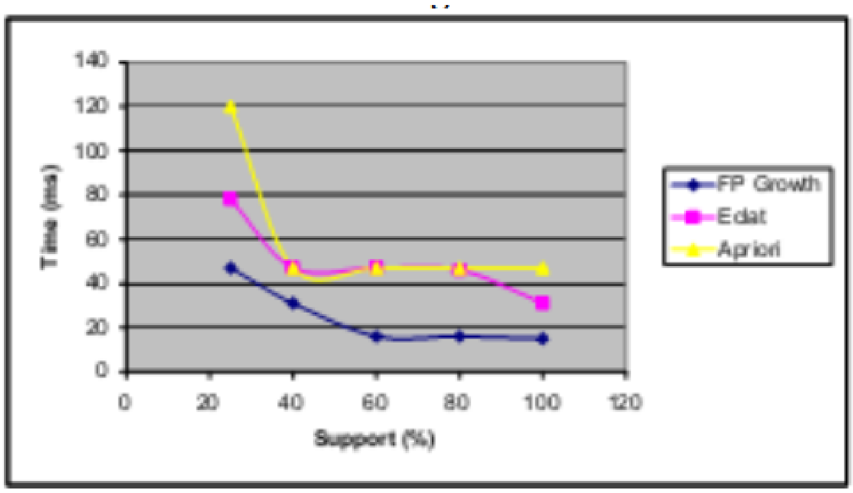


Figure 4 22 – Comparison of Apriori, Eclat and FP Growth algorithm on artificial dataset when the number of transactions is made three times the original. [7]

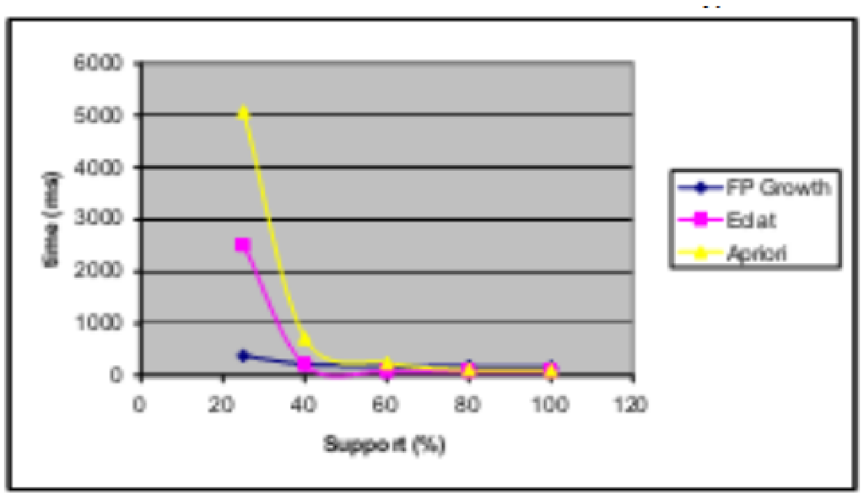


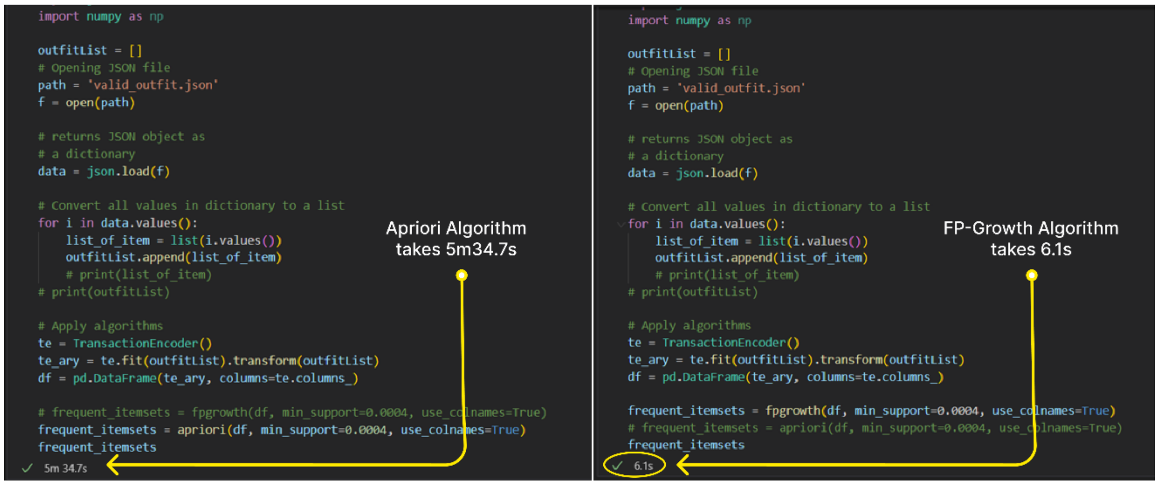
Figure 4 23 – Comparison of Apriori, Eclat and FP Growth algorithm on a dataset with the number of transactions three times the original and number of attributes three times the original.

The following information is derived from the statistics of two comparison analysis figures: FP Growth performs best, and Apriori takes the longest time even on a standard dataset. It was discovered that apriori employs the join and prune approach, Eclat works with vertical datasets, and FP Growth creates a conditional frequent pattern tree that meets the minimal support requirements. The Apriori algorithm's main flaw is that it generates a high number of candidate itemsets and database searches equivalent to a frequent item set's maximum length.

Scanning a huge database is quite costly. The lack of an effective database processing approach is a major factor for apriori failure.

In particular, and what makes it different from the Apriori frequent pattern mining algorithm, FP-Growth is an frequent pattern mining algorithm that does not require candidate generation. Internally, it uses a so-called FP-tree (frequent pattern tree) datastrucure without generating the candidate sets explicitely, which makes is particularly attractive for large datasets. The best of the three methods, FP Growth, is also the most scalable.

The above points are proven when training the model. In fact, for the same data set, it takes only 6.1 seconds for the FP-Growth algorithm to produce a list of frequent itemsets, which is 50 times faster than the Apriori algorithm; this takes 5 minutes and 34.7 seconds to generate.



```
import numpy as np

outfitList = []
# Opening JSON file
path = 'valid_outfit.json'
f = open(path)

# returns JSON object as
# a dictionary
data = json.load(f)

# Convert all values in dictionary to a list
for i in data.values():
    list_of_item = list(i.values())
    outfitList.append(list_of_item)
    # print(list_of_item)
# print(outfitList)

# Apply algorithms
te = TransactionEncoder()
te_ary = te.fit(outfitList).transform(outfitList)
df = pd.DataFrame(te_ary, columns=te.columns_)

# frequent_itemsets = fpgrowth(df, min_support=0.0004, use_colnames=True)
frequent_itemsets = apriori(df, min_support=0.0004, use_colnames=True)
frequent_itemsets
✓ 5m 34.7s
```

```
import numpy as np

outfitList = []
# Opening JSON file
path = 'valid_outfit.json'
f = open(path)

# returns JSON object as
# a dictionary
data = json.load(f)

# Convert all values in dictionary to a list
for i in data.values():
    list_of_item = list(i.values())
    outfitList.append(list_of_item)
    # print(list_of_item)
# print(outfitList)

# Apply algorithms
te = TransactionEncoder()
te_ary = te.fit(outfitList).transform(outfitList)
df = pd.DataFrame(te_ary, columns=te.columns_)

frequent_itemsets = fpgrowth(df, min_support=0.0004, use_colnames=True)
# frequent_itemsets = apriori(df, min_support=0.0004, use_colnames=True)
frequent_itemsets
✓ 6.1s
```

Apriori Algorithm takes 5m34.7s
FP-Growth Algorithm takes 6.1s

Figure 4 24 – Comparison of Time consuming when apply Apriori, FP Growth algorithm on a dataset with the same number of transactions.

CHAPTER 5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

The goal indicated at the beginning of this report has been met because the interface and some features of the Fashion Virtualize Social Media website were developed successfully and visually appealingly.

I received a lot of practical expertise throughout the stages of coming up with ideas to realize them, such as managing databases with JSON and JavaScript, developing websites with ReactJS, visualizing fashion products on the website, and generating a collection combing separate items. The surprising insights I gained from this project assignment were how different development phases of a project fit together and the life cycle of development itself.

The basic knowledge of Machine Learning has been absorbed by me step by step; I have steadily researched, understood and implemented the fundamental concepts in Frequent Itemset Mining and executed the code to test the Apriori algorithm using a dataset from Polyvore.com website and calculate support value and percent of confidence.

Despite the fact that I had never taken a Machine Learning course before and had to gather all of my knowledge from references, scientific articles, and YouTube clips shared on the Internet, I am grateful that, with the help of my academic advisor, I was able to complete the first step in learning, understanding, and using the algorithm - a good start for my thesis the following semester.

5.2 Future work

Many more intriguing features may be included in this system, such as allowing users to connect and communicate as a real messaging platform or a financial-supporting function, such as allowing stakeholders to display their marketing banners on the website. Furthermore, the remove background automatically for product image would be implemented to support the posting product to sell process to have a clearer view in fashion product - which is considered a valuable tool for mix and match in the Virtual Look section.

In next semester, I am going to research more about another algorithm for frequent itemset mining named FP – Growth and apply it in the recommended system of the FinnTA website to receive an effective, faster, satisfying result to visitors. Doing the data processing, connect the model run below the system with web application and display the result to users these suggested items for their collections on the website's interface.

REFERENCES

[1] Atharv Pandit , Kunal Goel , Manav Jain , Neha Katre, 2020, A Review on Clothes Matching and Recommendation Systems based on user Attributes, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 09, Issue 08 (August 2020),

[2] Lai S Q, Zhu J P. A Survey of Association Rule Mining Algorithms in Data Mining[J]. Statistics & Information Tribune, 2005.

[3] Leung K S, Mackinnon R K. Fast Algorithms for Frequent Itemset Mining from Uncertain Data[C]// IEEE International Conference on Data Mining. IEEE, 2015:893-898.

[4] Pham, P. (2020, January 3). Sitecore XC – Automatic Product Recommendation (APR) based on Frequent Itemset Mining (FIM) (part 1). Retrieved November 28, 2021, from https://phuphamtech.wordpress.com/2020/01/03/sitecore-xc-automatic-product-recommendation-apr-based-on-frequent-itemset-mining-fim-part-1/

[5] GeeksforGeeks. (n.d.). Frequent Item set in Data set (Association Rule Mining). Retrieved October 10, 2021, from https://www.geeksforgeeks.org/frequent-item-set-in-data-set-association-rule-mining/

[6] Agrawal, R. (n.d.). Fast Algorithms for Mining Association Rules . Retrieved from http://www.vldb.org/conf/1994/P487.PDF

[7] Rahul Mishra et. al. “Comparative Analysis of Apriori Algorithm and Frequent Pattern Algorithm for Frequent Pattern Mining in Web Log Data.” (IJCSIT)

International Journal of Computer Science and Information Technologies, Vol. 3 (4), 2012, Pp. 4662 – 4665.

[8] Han, Jiawei, Jian Pei, Yiwen Yin, and Runying Mao. "Mining frequent patterns without candidate generation. "A frequent-pattern tree approach." Data mining and knowledge discovery 8, no. 1 (2004): 53-87.

[9] Garg, D. K. (2013). Comparing the Performance of Frequent Pattern . Kurukshetra, Haryana, India: International Journal of Computer Applications.

[10] Docs, M. W. (n.d.). JavaScript – Web technology for developers. Retrieved from Developer Mozilla: https://developer.mozilla.org/en-US/docs/Web/JavaScript

[11] Nicholas C.Zakas, *Professional JavaScript for Web Developers*, Third Edition

[12] HTML Tutorial, H. (n.d.). Retrieved from W3School: https://www.w3schools.com/html/

[13] CSS Tutorial, H. (n.d.). Retrieved from W3School: https://www.w3schools.com/css/default.asp

[14] The React UI library you always wanted. (n.d.). Retrieved from MUI: https://material-ui.com/