





*Problem Statement Title: Personalized Product Recommendations*  
*Team Name: CodeX*



# TEAM MEMBERS DETAILS



Team Name	CodeX		
Institute Name/Names	Chaitanya Bharathi Institute of Technology		
Team Members >	1 (Leader)	2	3
Name	Hima Charan Gangula	Vishwanth Barma	Bhasuri Polaki
Batch	2024	2024	2024



# Problem Statement:

## Personalized Product Recommendations

The aim is to enhance user experience by implementing a personalized product ranking system. Your task is to develop an algorithm or model that can generate accurate and relevant product rankings for individual users. The ranking system should consider factors such as user preferences, past interactions, product popularity, and user similarity. It should be able to predict the most suitable products for a user based on their unique characteristics and preferences.

You are not provided with a specific dataset for this challenge. Instead, you are expected to design and implement a solution that simulates user interactions and generates personalized rankings. You can define user profiles, product categories, and interaction patterns within your solution.

To evaluate the effectiveness of your solution, you should define appropriate metrics for measuring the accuracy and relevance of the rankings. You should also provide a report explaining your approach, describing the algorithms or techniques used, and discussing the strengths and limitations of your solution.

---

# USE-CASES

P0

- **New User Recommendations** - Provide personalized product recommendations for new users who have not yet interacted with any products. To engage and convert new users into active customers.
- **User Preference Updates** - Continuously update user preferences based on their interactions, such as likes, wishlists, and orders. This ensures that the recommendations remain relevant and adaptive to user behavior changes.
- **Popular Products Recommendations**- Recommend popular products that are currently trending or have high overall user engagement. This can help in maximizing conversions for products with high demand.
- **Similar User Recommendations** - Utilize hybrid filtering to suggest products that users with similar preferences have interacted with. This enhances the personalization aspect by leveraging collective intelligence.

P1

- **Diverse Product Recommendations**- Ensure that recommendations are not overly repetitive and introduce diversity in product suggestions. This prevents users from getting bored and helps them discover new products.
- **Seasonal and Trend-based Recommendations** - Incorporate temporal patterns and trends to recommend products that are relevant to the current season, holidays, or emerging trends.
- **Wishlist Conversion - Recommend products** from users' wishlists that have a high likelihood of conversion, taking into account factors like price drops or availability.

P2

- **Cross-Selling and Upselling** - Suggest related or complementary products to users who have already shown interest in a particular product. This can help in increasing the average order value.
- **Customer Segmentation** - Segment users into different groups based on their behavior, demographics, and preferences. Tailor recommendations to each segment to improve the accuracy of suggestions.
- **Long-Tail Product Recommendations** - Recommend niche or less-popular products that might align with specific user preferences, providing exposure to products they might not have discovered otherwise.

# Solution statement/ Proposed approach

## Data Collection and Preprocessing:

- Collecting user interaction data, product details , and user profiles.
- Preprocess data by handling missing values, data formatting, and cleaning.

## Algorithm Design:

- Designing an algorithm that considers collaborative(user based and item based), content-based, and rule-based filtering techniques to get personalized recommendations.
- Integrating dimensionality reduction techniques to enhance scalability and performance.

## User Interface:

- Developing a user interface that allows users to view recommendations, wishlist, cart, and interact with products.
- Implementing user registration

## Backend Implementation:

- Implementing the backend using a Flask to handle API requests and interact with the database. As Algorithms are written in python Flask is optimal backend choice

# Data Collection and Preprocessing

## **User Interaction Data Collection:**

Collect user interaction data including likes, orders, searches, and wishlist actions.

Store data in the `userProductRelation`, `productRatingsDb`, `wishlistDb`, and `cartDB` tables.

## **Product Details Collection:**

Gather detailed product information including categories, prices, ratings, etc.

Store data in the `productDB` table.

## **User Profile Collection:**

Collect user details such as names, emails, mobile numbers, and images.

Store data in the `userDb` table.

## **Generating Data :**

Using Random Function and Faker module, data sets are created and made them realistic by defining probability(probability of a product being liked is 0.3 etc)

## DATA BASE DETAILS :

No.of User = 100  
 No.of Products = 2816  
 Number of Categories = 9  
 Number of Subcategories = 455

## SAMPLE DATA :

user_id	product_id	liked_or_not	wishlisted_or_not	ordered_or_not	search_count
68	1413	0	0	0	3
85	38	0	0	0	1
71	1813	1	1	1	10

userProductDB

userId	productIds
3	280
5	190

cartDB

user_id	product_id	rating
9	2602	4.0
41	1182	3.5

ratingsDB

userid	productIds
1	869
1	2053
1	2319

wishlistDB

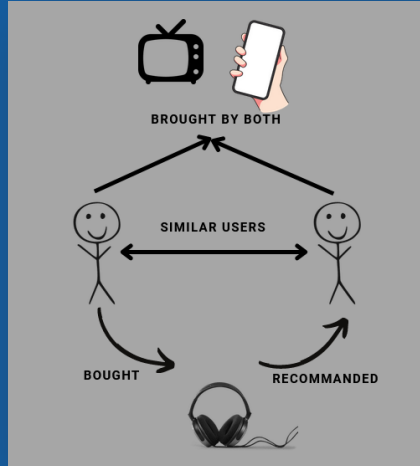
user_id	user_name	user_email	user_mobile	user_image
1	Holly_Acosta	holly@gmail.com	693-863-0127x801	https://randomuser.me/api/portraits/men/1.jpg
2	Heather_Santos	heather@gmail.com	3732205878	https://randomuser.me/api/portraits/men/2.jpg
3	Pamela_Garcia	pamela@outlook.com	(214)265-1566x274	https://randomuser.me/api/portraits/women/3.jpg

userDB

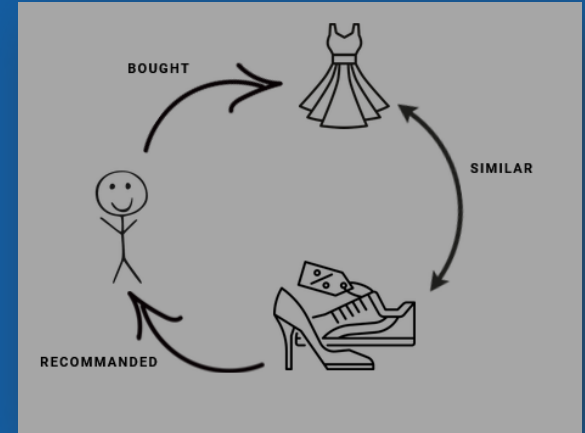
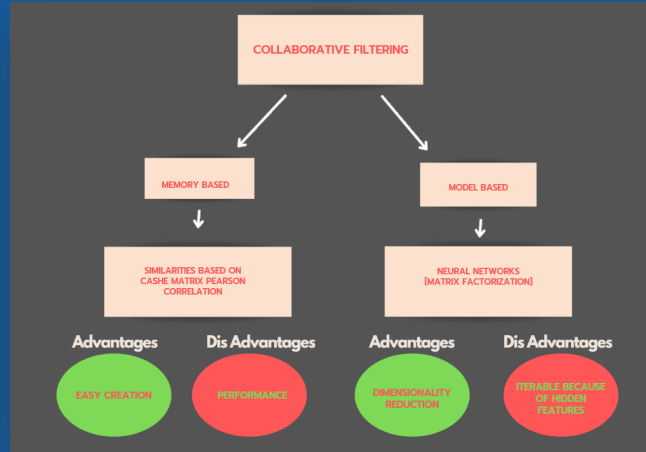
product_id	category	sub_category	product_name	price	original_price	discount	reviews_count	brand	images	stock	f_assured	seller	seller_rating	return_policy	description	specifications	avg_rating\
1	accessories	Autres	\c9tui \e0 lunett	292.53	662.24	56	173	Baban	https://in	Hurry, C	1	Turner	1.5	10 Days Replac	Enhance you	Wide variety of	4.49\
2	accessories	Baseball Caps	Casquette de ba	864.36	1894.1	54	15	NUZADA	https://in	Hurry, C	0	Johns	0.5	7 Days Replace	Enhance you	Wide variety of	2.43\
3	accessories	Baseball Caps	Casquettes de B	926.1	1426.64	35	51	NUZADA	https://in	Hurry, C	0	Booke	2.0	7 Days Replace	Enhance you	Wide variety of	2.59\
4	accessories	Baseball Caps	NUZADA Casque	827.61	1750.77	53	36	NUZADA	https://in	Hurry, C	1	Wagne	1.0	7 Days Replace	Enhance you	Wide variety of	3.54\

productDB

# Algorithm Design



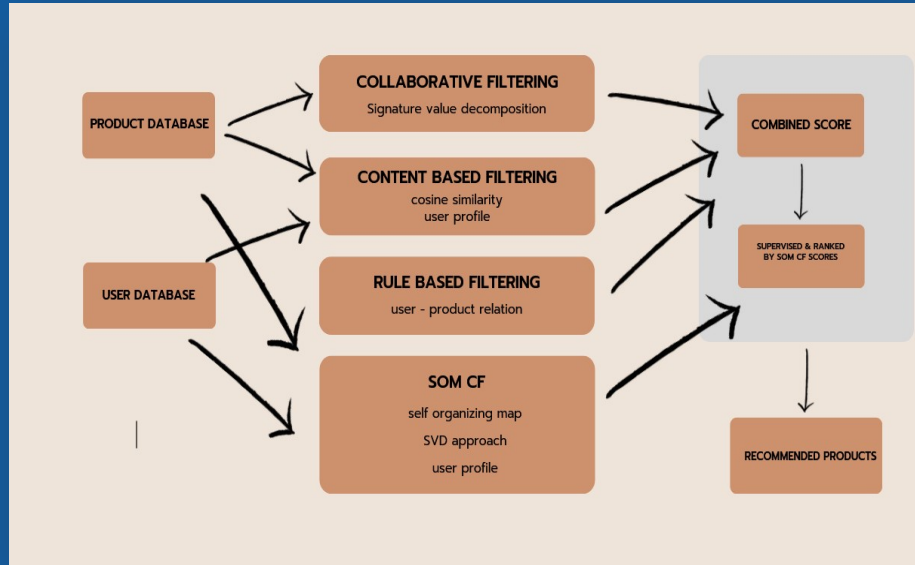
Collaborative Filtering



Content - Based Filtering



# Hybrid Model



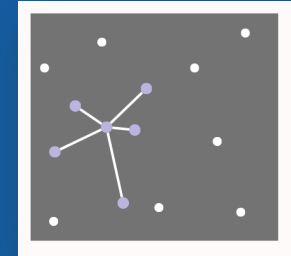
- Using **HYBRID MODEL** makes the recommending system even more accurate
- Calculating user-user similarity and item-item similarity are calculated Using collaborative filtering
- Interaction matrix is generated for both content based and collaborative filtering
- Rule based approach is used, priority of (wishlist > liked > searched) product
- **COMBINED RANKING** is calculated to get product recommendations to the particular users

# Filtering Process

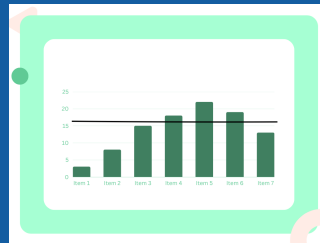
## STEP-1 Collecting User-Product Interactions Data Matrix

USERS \ PRODUCTS	PRODUCTS			
	PRODUCT 1	PRODUCT 2	PRODUCT 3	PRODUCT 4
USER 1	$R_{11}$	$R_{12}$	$R_{13}$	$R_{14}$
USER 2	$R_{21}$	$R_{22}$	$R_{23}$	$R_{24}$
USER 3	$R_{31}$	$R_{32}$	$R_{33}$	$R_{34}$
USER 4	$R_{41}$	$R_{42}$	$R_{43}$	$R_{44}$

## STEP-2 Selecting similar neighbours by measuring parameter – Statistical techniques

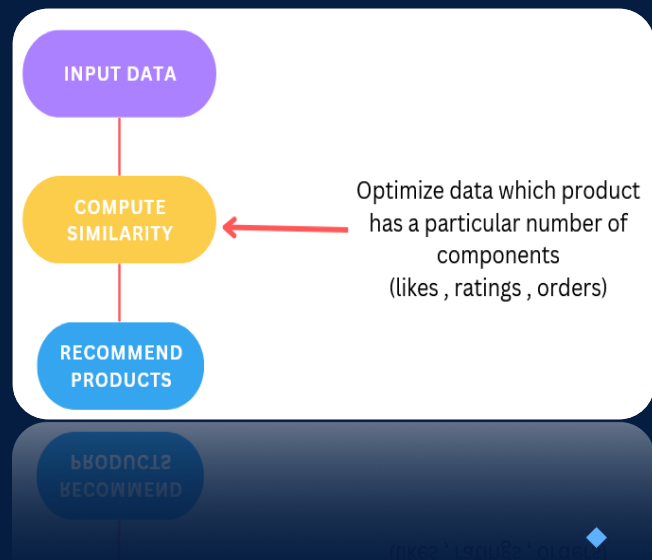


## STEP-3 Generation



# Dimensionality Reduction

- To overcome scalability and performance issues due to collaborative filtering  
dimensionality reduction is the process of mapping a high-dimensional input space into a reduced level of latent space.
- Matrix factorization is a subset of dimensionality reduction in which a data matrix is reduced to the product of many low-rank matrices.
- SVD is a powerful dimensionality reduction technique that is a specialization of the MF approach.
- Apply dimensionality reduction techniques (e.g., SVD) to handle sparse interaction matrices.
- Reducing the complexity of the recommendation algorithm
- Reducing noise of dimensional data



## Metrics for measuring the accuracy and relevance of the rankings

### Recall at K (R@K):

- It measures the proportion of relevant items that are successfully recommended within the top K items in the recommendation list

$$R@K = \frac{\text{Number of relevant items in top K recommendations}}{\text{Total number of relevant items}}$$

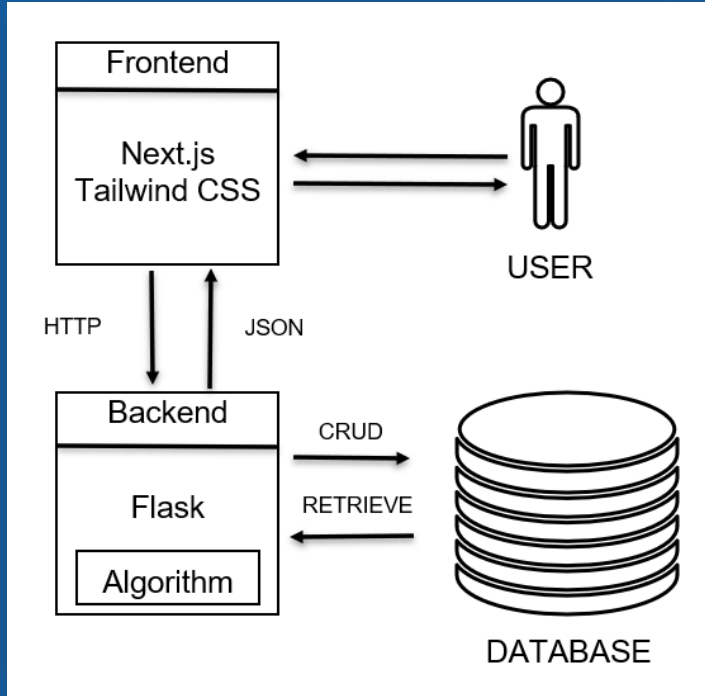
```
12 RECOMMENDATIONS TO USER (ID=59):  
[636, 9229, 9835, 7933, 9151, 12011, 3853, 4473, 7911, 9333, 10198, 10537]  
Recall at K (K=12): 0.57
```

### Mean Average Precision at K (MAP@K):

- MAP@K calculates the average precision at each position for a set of users and then takes the mean. It provides an overall measure of the system's performance across multiple users.

```
12 RECOMMENDATIONS TO USER (ID=34) USING SVD:  
[1140, 947, 1649, 1700, 2102, 1657, 2510, 2738, 168, 206, 833, 1546]  
Mean Average Precision at K = 0.39
```

# User Interface and Backend Implementation:



- **User has access to order ,like Wishlist and manage cart etc**
- **DeCoupled Frontend & Backend (cross-platform feasible)**
- **Using react context in order to obtain stable State management**
- **Algorithm is intact with backend using Flask framework**
- **User interaction with product is constantly updated in the database in order to get accurate recommendations**



# Limitations

## Data Quality and Quantity

- As data sets are created using random and Faker module recommendations or wide ranged but if we use realistic data algorithm will recommended accurate products
- Presently 2 level Categorization is done due to lack of datasets
- Browsing history, timestamps, user reviews can also be used in this algorithm but dataset doesn't have those parameters



## Algorithm Complexity

- Combining multiple filtering techniques with dimensionality reduction can lead to increased algorithm complexity, making it harder to maintain and optimize.

## Data Sparsity

- Dimensionality reduction might exacerbate sparsity issues, potentially leading to less accurate recommendations.
- 

# Future Scope

## **Advanced Machine Learning Techniques:**

- Incorporating more advanced machine learning techniques such as deep learning, and natural language processing to enhance the accuracy of content-based filtering and user profiling.

## **Hybrid Model Optimization:**

- Experimenting with different hybrid model combinations and weighting mechanisms for combining results from different filtering techniques. Optimizing the hybrid model to achieve better performance.

## **Reinforcement Learning:**

- Exploring the integration of reinforcement learning to continuously adapt the recommendation strategy based on user feedback and interactions.

## **A/B Testing and Experimentation:**

- Implement A/B testing to compare the effectiveness of different recommendation strategies and fine-tune the system based on user feedback and performance metrics.

## **Social Collaborative Filtering:**

- Incorporating social network data to enhance collaborative filtering, taking into account users' social connections and their preferences.



*Thank You*