Team : She Squad

Myntra

<hacker-ramp/>

weforshe

Team Members

1. Aadya Dewangan
2. Aastha Tembhare
3. Revati Goyal

Problem Statement

tech stack: Python, TensorFlow, React, HTML5, CSS3, JS

- Fast fashion brands **struggle to keep up with rapidly changing trends and varied customer preferences**. Customers have diverse and evolving tastes, making it **challenging to predict individual interests accurately**. This leads to **sub-optimal customer engagement and conversion rates**, highlighting the need for more advanced and precise recommendation systems.

The Mentioned Data Percentage Are Taken From Survey Performed By Us On Age Group Of 20-35years.

83.3% Of Users Prefer Cosmetics To Be Initially Tested Based On Their Specific Skin Characteristics, Such As Complexion And Skin Type.

62.5% People Plan Occasion Specific Outfits.
70.8% Of Frequent Users Express Neutral Satisfaction With Myntra's Recommendation System Regarding Suggestions On Current Trending Fashion.

USER JOURNEY

Arrives To Home Screen

Search
Myntra allows users to search query by typing, voice search or visual search.

Filter/Sort
The user taps the filter icon to refine search results by size, color, brand, price, and can sort results by popularity, newest arrivals, price, etc.

Viewing Product Details
Selecting a Product aligning with user's style.

Adding to wishlists/cart.

Receiving Recommendations:
- Similar Products
- People Also Viewed
- Personalized Recommendations

Users don't receive suggestions for trending fashion aligned with their choices, making it difficult to stay updated with the latest styles and trends.

Users find difficulty to keep track of their favourite items for different occasions without a centralized place to save and categorize them.

Exploring Recommendations
The user scrolls through recommended products that align with their style and preferences and taps on one to view its detailed product page, to see reviews, qualities, etc.

Adding to Cart and Checkout
The user adds chosen products to their cart, reviews the items by tapping the cart icon, enters shipping information, selects a payment method, reviews the order summary, and places the order.

For cosmetic products, there's no try-on feature to suggest options based on your complexion or to recommend which cosmetics would complement your outfit color

According To **Statista** In 2023, 47% Of Gen-Z, 46% Of Millennials, And 41% Of Gen-X Shoppers Worldwide Desired Personalized Product Recommendations When Making Online Purchases.

Solution 01: Recommendation Engine

A **recommendation engine** based on **Collaborative Filtering (CF) architecture** for **personalized mood based and dynamic recommendations** based on **user's personal style** and **themes of recent browsing history.**

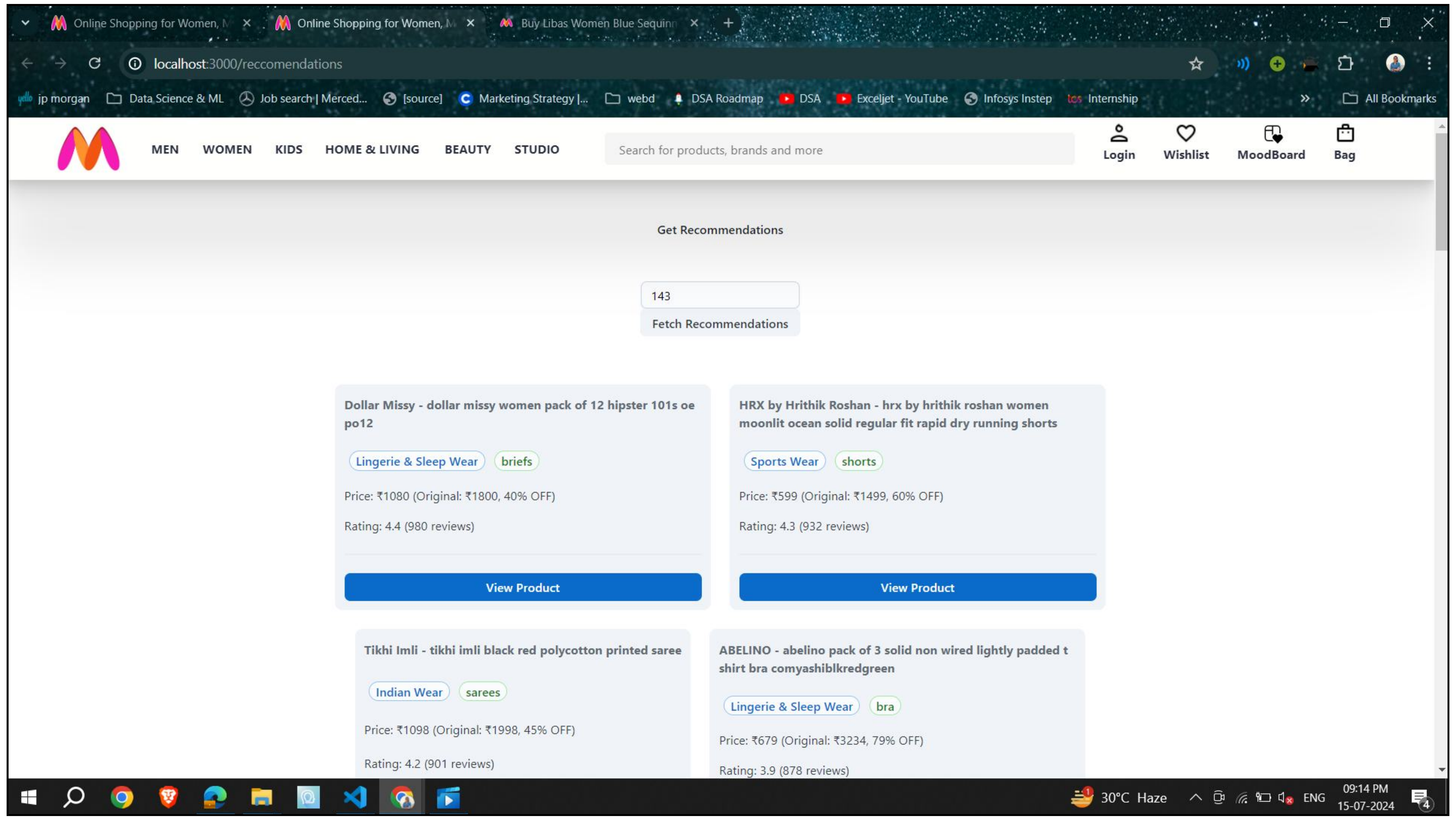
```javascript
const fetchRecommendations = async () => {
      try {
          const response = await axios.post('http://127.0.0.1:8000/recommend', { user_id: userId });
          setRecommendations(response.data.recommendations);
          setError(null);
      } catch (error) {
          console.error('Error fetching recommendations:', error);
          setError('Error fetching recommendations');
      }
  };
```

```python
def recommend_products(X_T_dict, user_id, loaded_svd_model):

    X_T = pd.DataFrame(X_T_dict).T
    decomposed_matrix = loaded_svd_model.transform(X_T.T)
    correlation_matrix = np.corrcoef(decomposed_matrix)
    user_index = list(X_T.index).index(user_id)
    correlation_user_ID = correlation_matrix[user_index]
    Recommend = list(X_T.columns[correlation_user_ID > 0.90])

    return Recommend
```

Video Link

Collab Link

Data Link



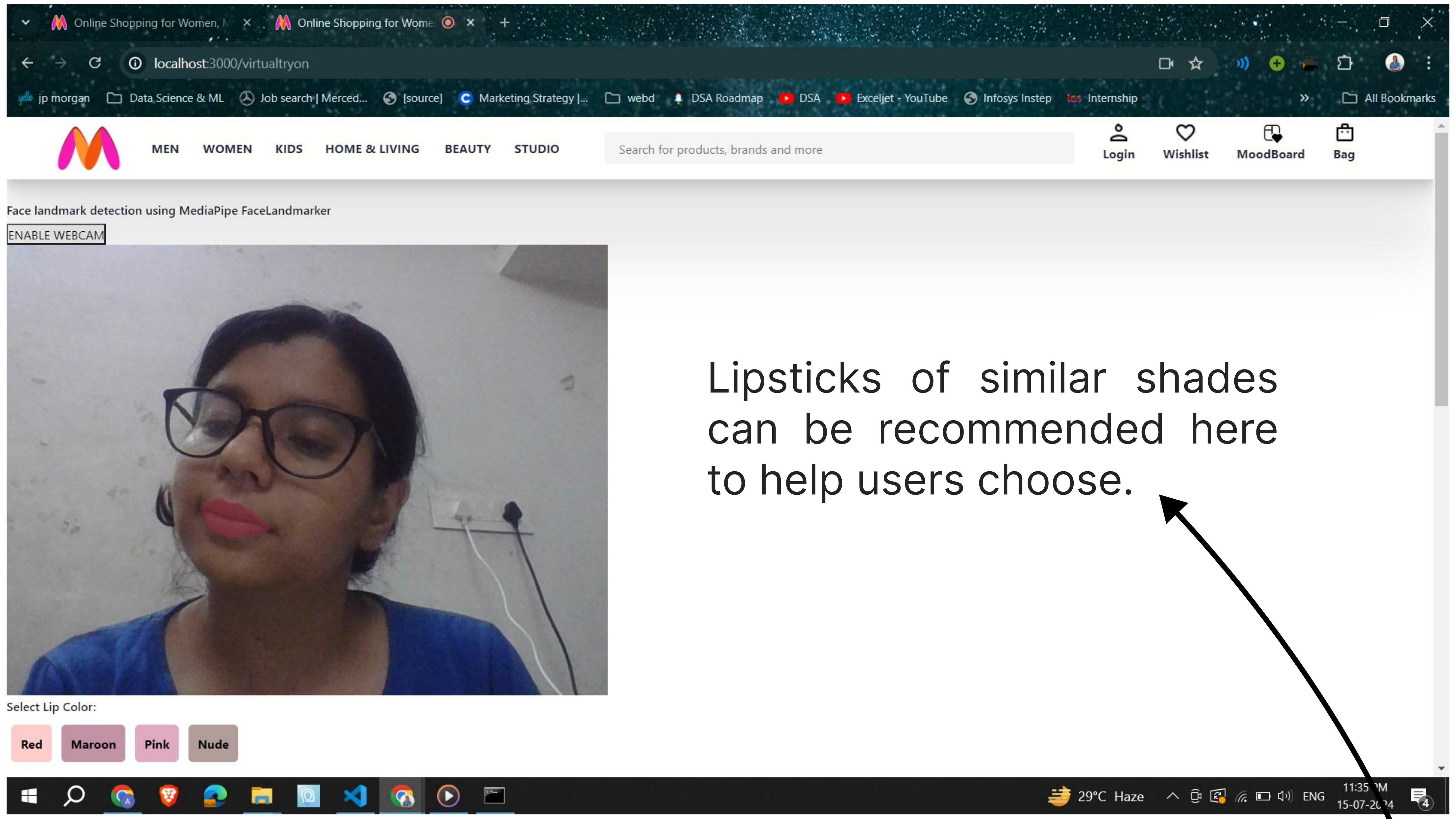
Recommendation engine demo by entering User ID
(User ID from 0-390)

Creating User-Item Interaction Matrix	Matrix Factorization via SVD	Calculating reccomendations	Connecting to front-end
Rows represent users and columns represent items. Entries indicate user-item interactions, such as ratings.	Matrix factorization decomposes the user-item interaction matrix into two lower-dimensional matrices: one for users and one for items using Single Value Decomposition (SVD)	Product recommendations are calculated by finding highly correlated users based on the transformed data using SVD and suggesting similar items.	An api was created using Fast API to fetch the recommendations from the browser and display the product details accordingly based on the user id.

Solution 02 : Virtual Try On

```
for (const landmarks of results.faceLandmarks) {
     const outerLipsIndices = [
                 61, 185, 40, 39, 37, 0, 267, 269, 270, 409, 291,
                 375, 321, 405, 314, 17, 84, 181, 91, 146, 61

     ];

      const innerLipsIndices = [
                  78, 191, 80, 81, 82, 13, 312, 311, 310, 415, 308,
                  324, 318, 402, 317, 14, 87, 178, 88, 95, 78

      ];
}
```

```
// Draw and fill the outer lips with overlay mode
  canvasCtx.globalCompositeOperation = 'overlay';
  canvasCtx.fillStyle = selectedColor;
  canvasCtx.beginPath();
  for (let i = 0; i < outerLips.length; i++) {
      const { x, y } = outerLips[i];
      if (i === 0) {
          canvasCtx.moveTo(x, y);
      } else {
          canvasCtx.lineTo(x, y);
      }
  }
  canvasCtx.closePath();
  canvasCtx.fill();
```



Lipsticks of similar shades can be recommended here to help users choose.

Identifying facial features

faces are composed of various features like eyes, ears, nose cheeks and lips. we need to be able to **identify the lip's boundary** to work with it

creating the mask

segmenting the lip area using the landmarks identified and **creating a mask for the lip region**

color application

using the mask to **apply and change colors** to the masked area and **blending** it well for a more natural look

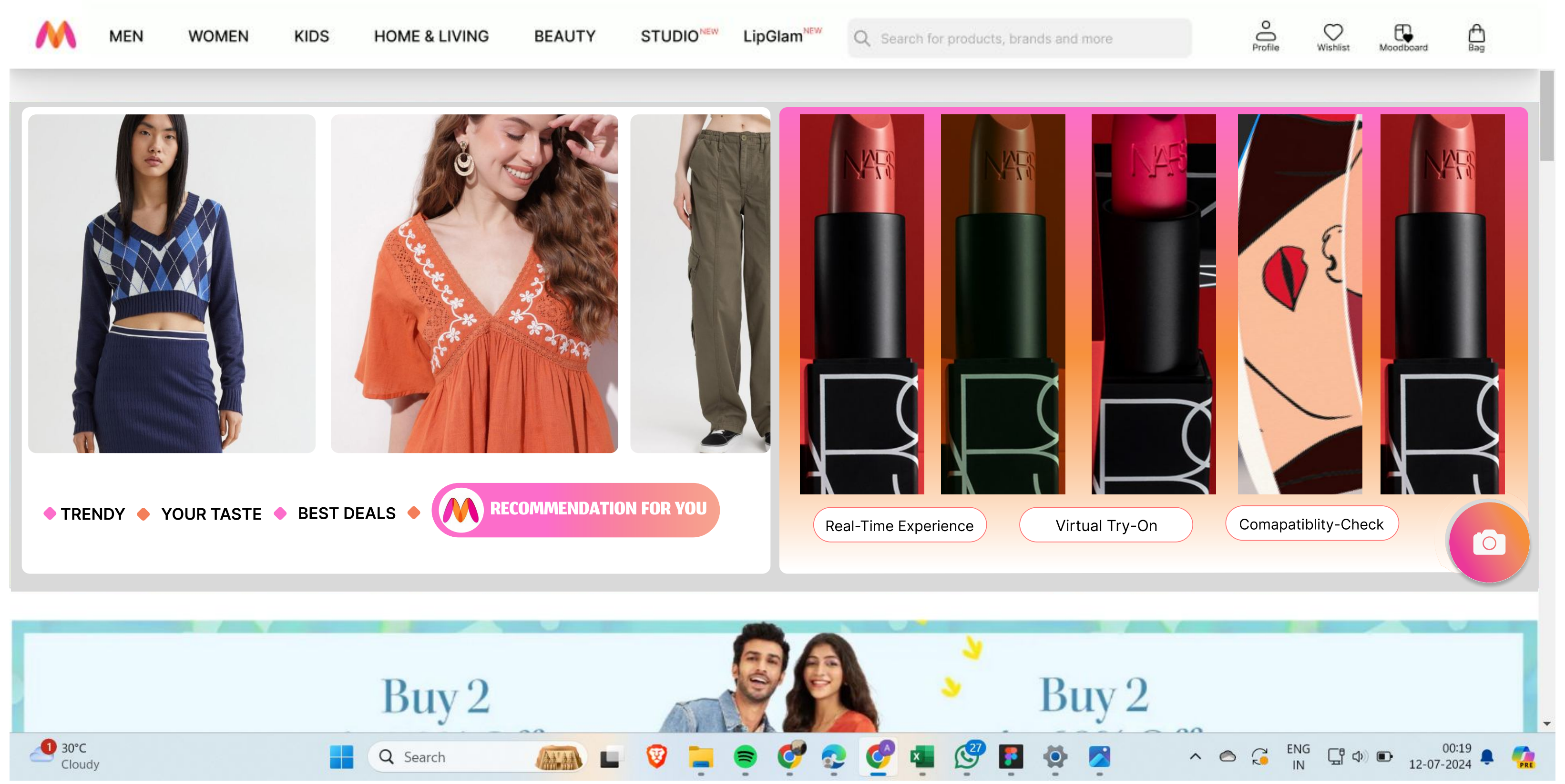
changing shades

user's should easily be able to **switch between different shades** to **compare and analyze** which looks better on them.

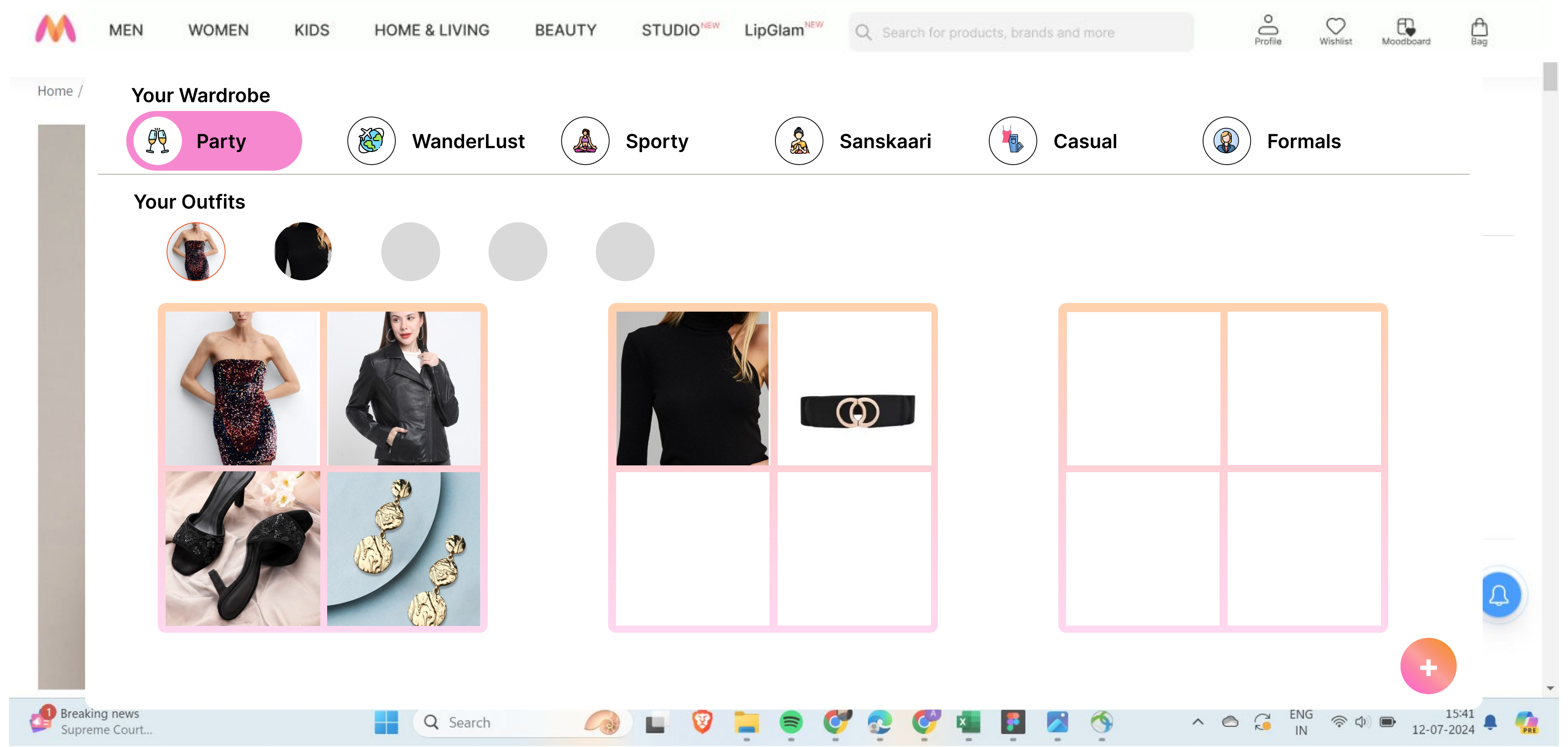
recommending lipsticks

based on the selected shade, **lipsticks available** in that shade **can be shown to the user** to help them select

Wireframes



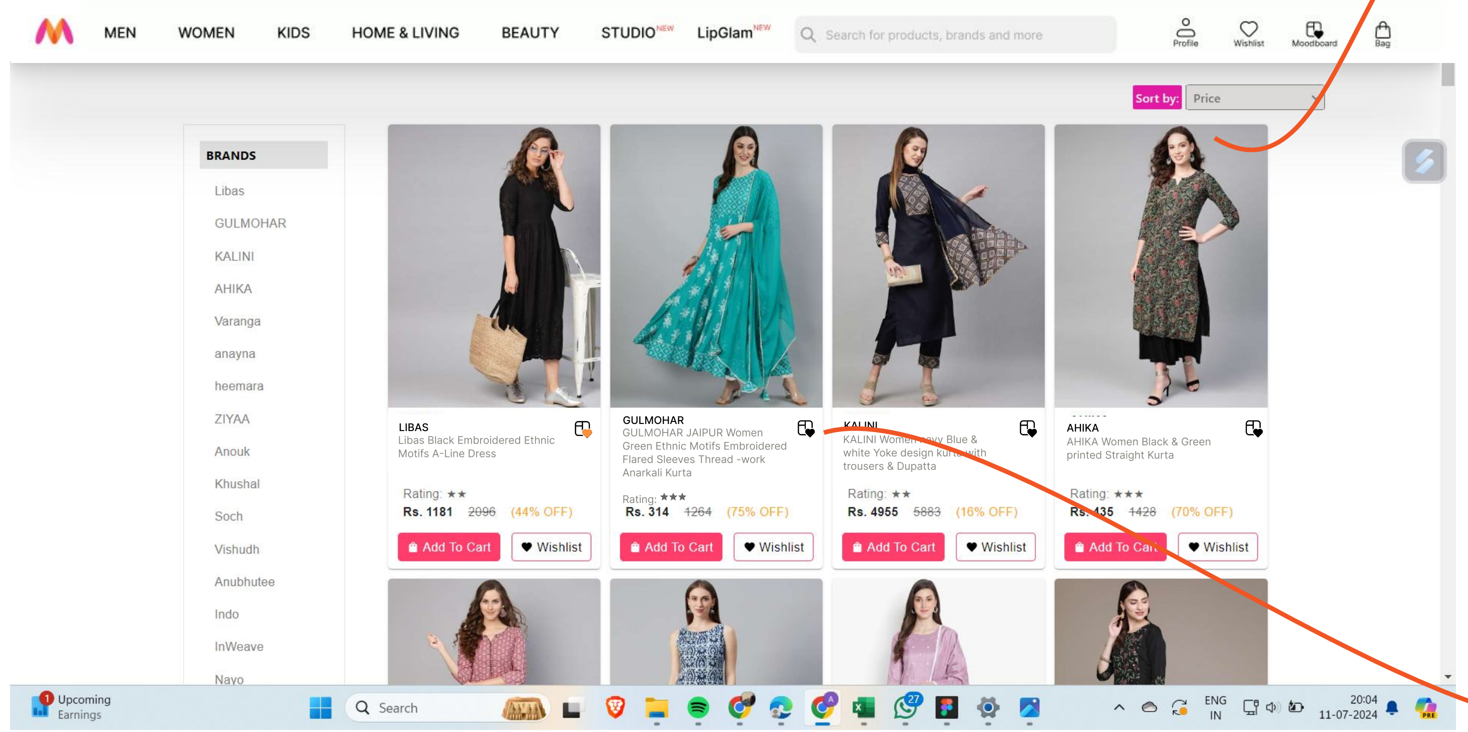
Home Screen

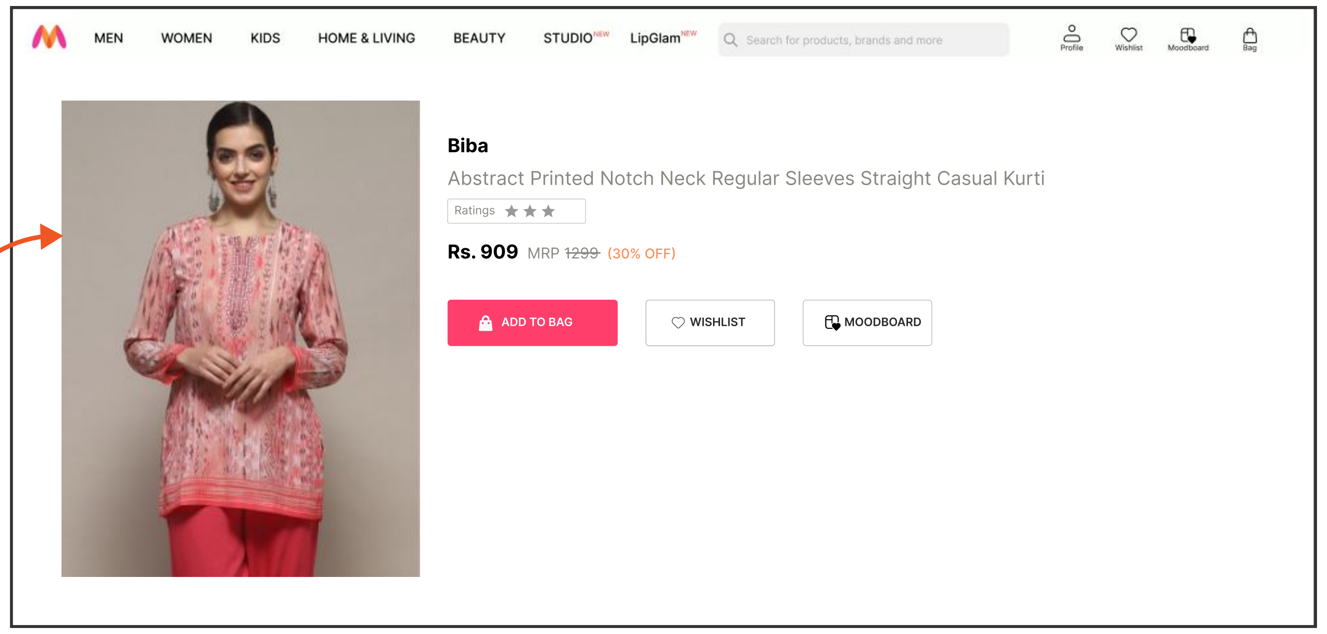


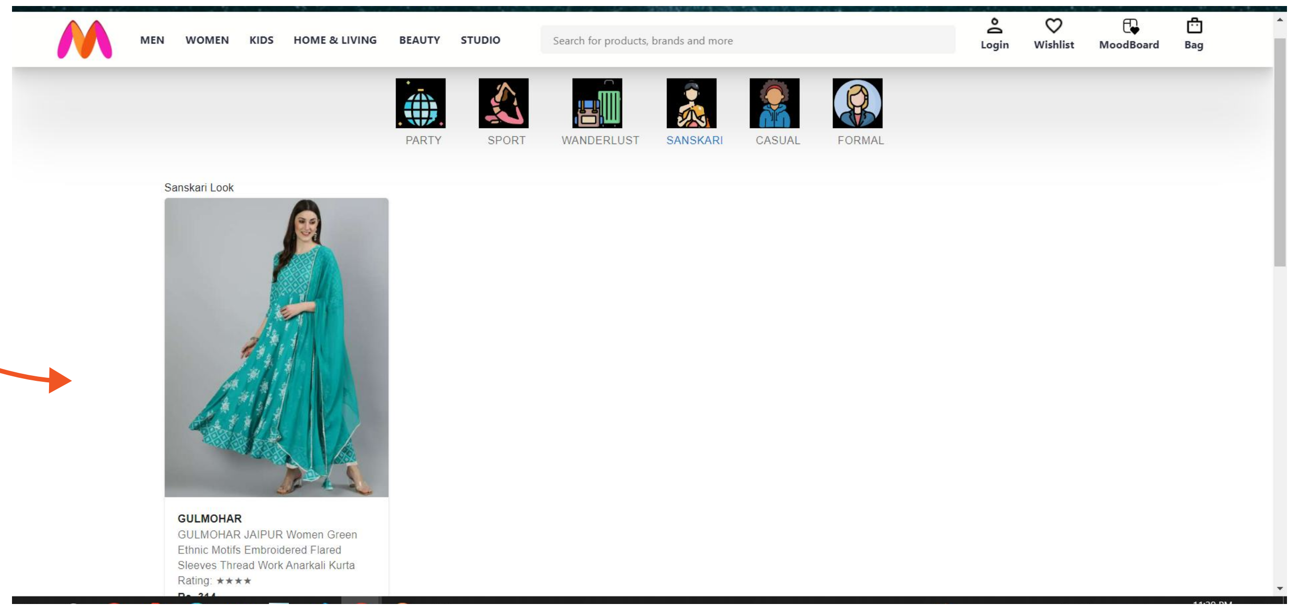
Mood Boards

MoodBoards

Clicking the product open the product's detailed page







Benefits

Personalized Suggestions

help in curating a **more engaging shopping experience** by helping customers **easily find clothes** as per their needs and styles

Novelty & Uniqueness

facilitates **greater engagement** by creating a **new & unique experience** that can **differentiate Myntra** from its competitors **enabling users to come back**

Style Boards and Mood Boards

facilitates **quick comparisons** based on different looks, helping user **quickly match different items** to create their outfits.

Challenges

Cold Start

Collaborative filtering needs data to identify similarities, so, **when either new users or new items are added,** recommendations might be erroneous due to lack of data.

Model Life

Myntra is an ever growing app with **new users and products everyday**, so just creating **one model is not enough** as it will **not be able to work with new data.**

Performance

Both recommendation engines and AR products **require high end hardware resources**, making it a challenging to deploy it without affecting performance too much.

Overcoming Them

Cold Start

This can be reduced by **using clustering techniques** from demographic data to identify similar users, coupled with **content-based recommendation engine** to create a backup solution in case of data-deficiency.

Model Life

The **model should be retrained frequently**, every few days, to ensure that it is **updated with the latest data and using dynamic weights**, so it can effectively capture recent trends and give optimal recommendations.

Performance

Caching of data browsed via the engine can be implemented to ensure **quick reloads**.
For the virtual try-on, user can be given the **option of uploading photo** too in case their phone cannot process the video part efficiently.