Wear Your Mood

Big Data, The Self and Social Platforms

Prepared by Xingyu Pu
(23024979)

Table of Conten

Introduction

With the rapid growth of Internet technology and rising living standards, personalized fashion recommendation systems have become a popular research topic. Color, as a key element of fashion, can directly reflect and influence emotions. However, traditional color recommendation systems often focus on objective factors like season and skin tone, while overlooking users' emotional states.

This project presents an emotion-based fashion color recommendation system. It analyzes user-input emotional keywords, crawls matching color schemes from professional color websites, and uses deep learning to generate personalized outfit suggestions.

The system addresses three main challenges:

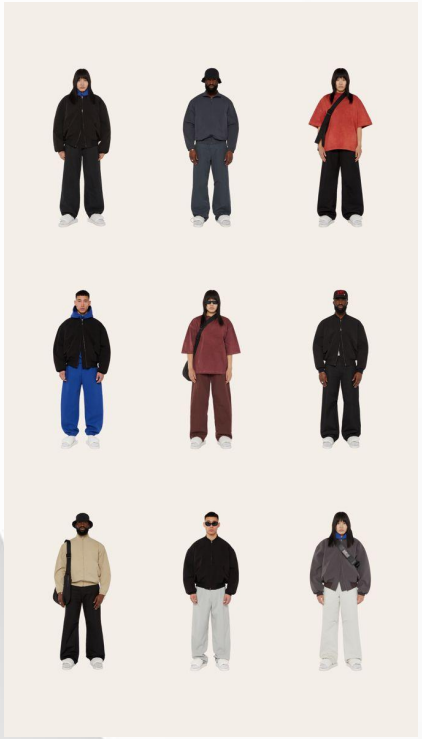
- How to find emotion-related color combinations online
- How to present these color palettes with interactive features
- How to turn color palettes into practical fashion advice

Built with Python Flask, the system combines web crawling (from sites like Coolors and ColorHunt) with DeepSeek API for fashion suggestion generation. A responsive front-end displays the results, making the system useful both for everyday users and fashion designers seeking color inspiration.

Introduction

User Group

• Emotion–sensitive youth: who are aware of their mental states and use clothing as self expression.
• Fashion–forward users: interested in trying color–based outfit suggestions tied to moods.
• Creative technologists: seeking interactive digital artworks merging data with aesthetics.
Especially relevant for Gen–Z, this platform combines fashion psychology, color theory, and interaction.



The primary users for “Wear Your Mood” are young, digital–native individuals—primarily GenZ—who are highly self–aware and accustomed to expressing their emotions through visual mediums. These users are luent in digital aesthetics, fashion culture, and emotional language.

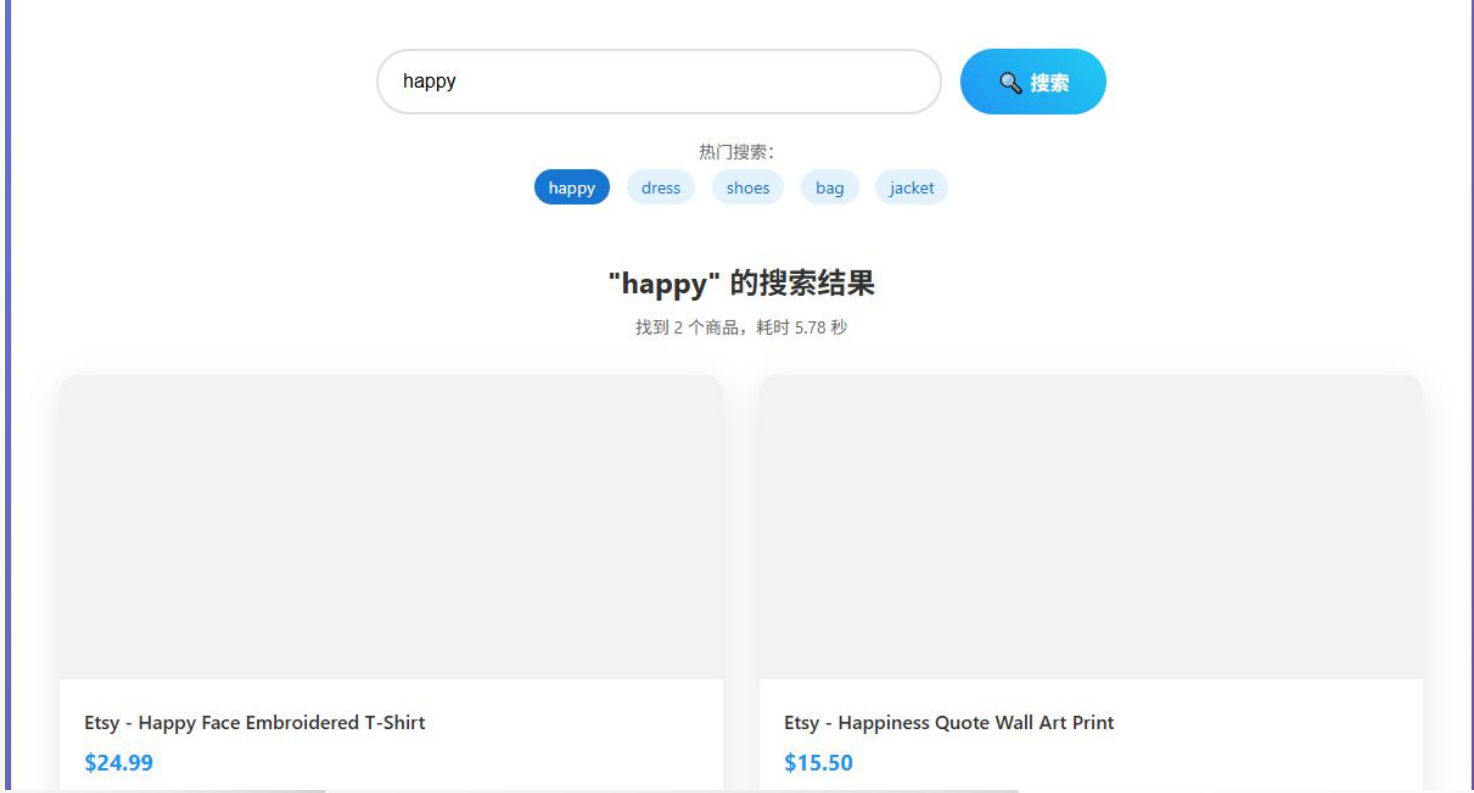
In an era of mood trackers, social filters, and Instagram aesthetics, fashion has become a key medium of emotional storytelling. Many users find comfort and empowerment by dressing in a way that reflects their mental state. However, there’s currently a gap between emotional data and fashion recommendations in an interactive, visual form.

This project bridges that gap—offering an intuitive, playful way for users to translate their inner feelings into wearable color combinations and artistic visuals. It enhances emotional awareness, supports mental wellness, and inspires creative dressing based on mood

Problem & Research

Initially, my idea was to collect product images from fashion websites based on users' emotional input. I targeted platforms such as ASOS.com and Pinterest, hoping to directly crawl fashion-related visuals linked to specific emotions. However, during implementation, I discovered that most of these websites actively block bots, preventing automated data collection.

For example, when scraping ASOS or Etsy, the responses only included product names and prices, but no valid image URLs. I tested many similar platforms, and all presented the same issue. On the other hand, websites without anti-scraping measures often lacked emotional keyword coverage, making the available images too limited or irrelevant.



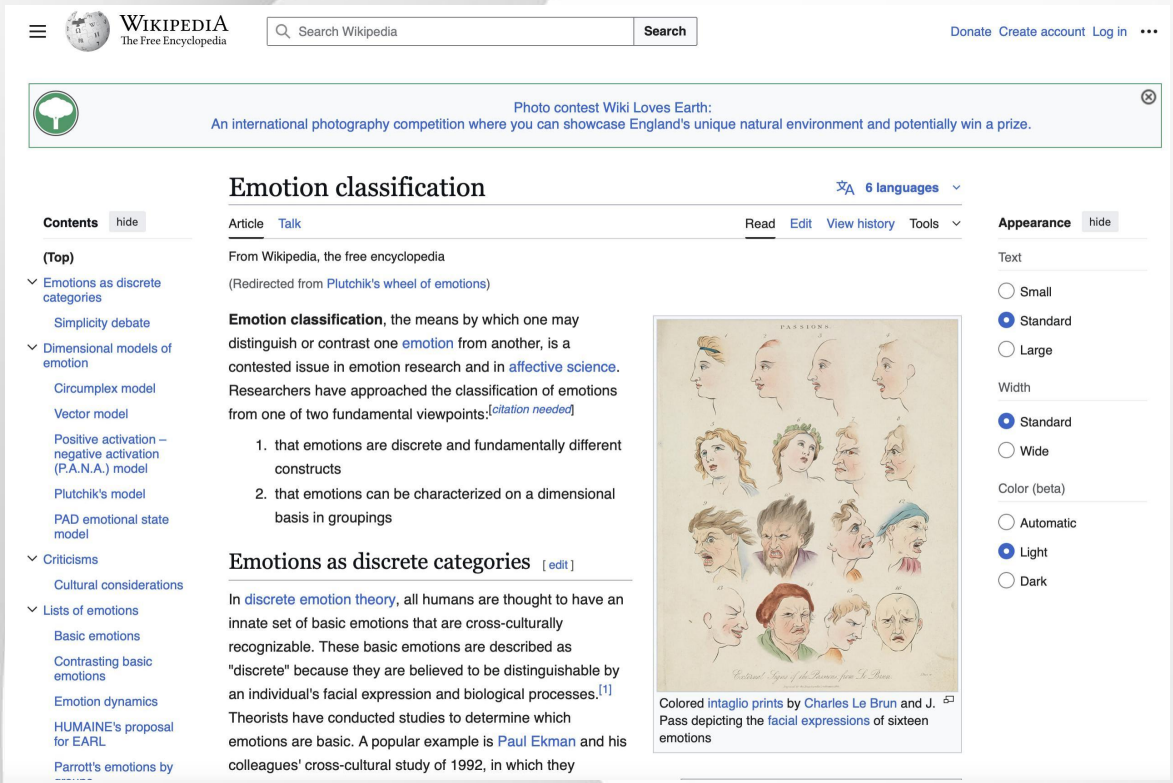
So I decided to get some color matching website color, because most of the color can get text, numbers, symbols (such as #334455 or rgb (255,255,255), etc.) can be generated in the front-end, compared to the picture crawling will be a little simpler, but in practice found that some of the site also has an anti-crawl mechanism, I decided to use a mixed collection strategy, combined with direct access to the target website and Google search in two ways, has enhanced the scope of crawling, and finally finally let most of the emotional keywords are able to get the corresponding color matching suggestions

Color Psychology

https://www.verywellmind.com/color-psychology-2795824

Color psychology suggests that colors have a significant impact on human emotions and behavior. Warm colors such as red, orange, and yellow are often associated with feelings of energy, excitement, and urgency, while cool colors like blue, green, and purple tend to evoke calmness, relaxation, or sometimes sadness. Although cultural background and personal experiences can influence individual responses to color, many of these associations are widely recognized. The theory is frequently applied in real-world settings—from hospital walls painted blue to reduce anxiety, to retail environments using red to stimulate a ction. While the scientific basis for color-emotion links is still evolving, color psychology remains a practical and intuitive framework for designing mood-based systems. This research directly informs the foundation o f ‘Wear Your Mood’, where users are recommended outfits and color palettes based on emotional input

https://en.wikipedia.org/wiki/Plutchik%27s_wheel_of_emotions



Wikipedia
The Free Encyclopedia

Search Wikipedia | Search

Donate | Create account | Log in

Photo contest Wiki Loves Earth:
An international photography competition where you can showcase England's unique natural environment and potentially win a prize.

Emotion classification

6 languages

Contents hide

(Top)

- Emotions as discrete categories
  - Simplicity debate
- Dimensional models of emotion
  - Circumplex model
  - Vector model
  - Positive activation – negative activation (P.A.N.A.) model
  - Plutchik's model
  - PAD emotional state model
- Criticisms
  - Cultural considerations
- Lists of emotions
  - Basic emotions
  - Contrasting basic emotions
  - Emotion dynamics
  - HUMAINE's proposal for EARL
  - Parrott's emotions by

Article | Talk | Read | Edit | View history | Tools

From Wikipedia, the free encyclopedia

(Redirected from Plutchik's wheel of emotions)

Emotion classification, the means by which one may distinguish or contrast one emotion from another, is a contested issue in emotion research and in affective science. Researchers have approached the classification of emotions from one of two fundamental viewpoints:[citation needed]

1. that emotions are discrete and fundamentally different constructs
2. that emotions can be characterized on a dimensional basis in groupings

Emotions as discrete categories [edit]

In discrete emotion theory, all humans are thought to have an innate set of basic emotions that are cross-culturally recognizable. These basic emotions are described as "discrete" because they are believed to be distinguishable by an individual's facial expression and biological processes.[1] Theorists have conducted studies to determine which emotions are basic. A popular example is Paul Ekman and his colleagues' cross-cultural study of 1992, in which they

Colored intaglio prints by Charles Le Brun and J. Pass depicting the facial expressions of sixteen emotions

Appearance hide

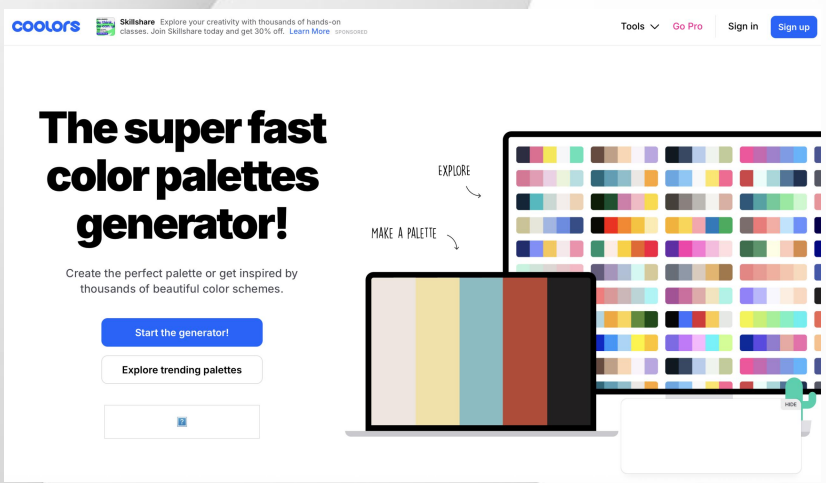
Text
- Small
- Standard
- Large

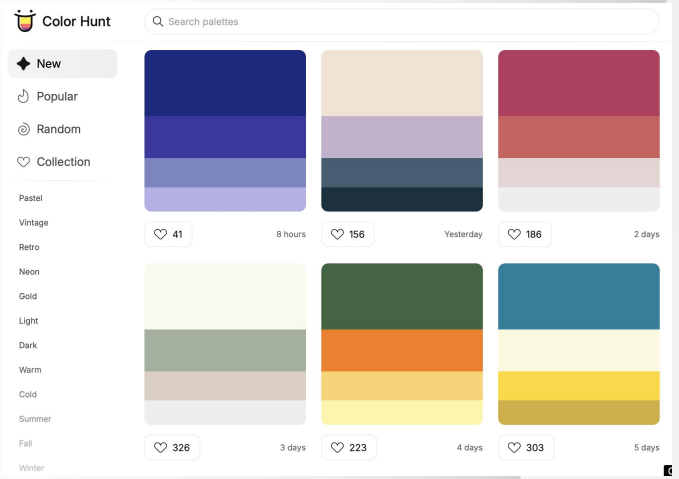
Width
- Standard
- Wide

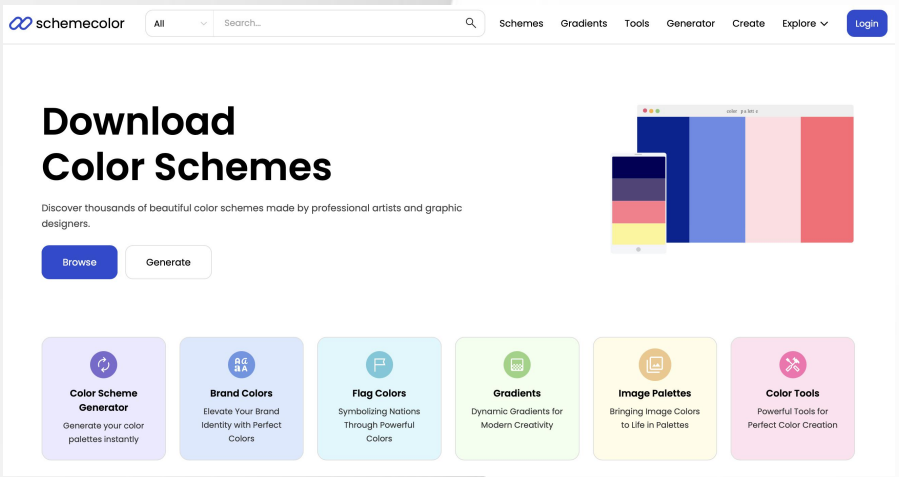
Color (beta)
- Automatic
- Light
- Dark

Psychological models like Plutchik's Wheel link emotions to colors—yellow for joy, red for anger, blue for sadness, and green for fear. Warm colors often represent high-energy emotions, while cool tones suggest calm or introspection. These associations help visualize emotional states and are widely used in mood-based design.

Data Source







The system primarily sources color palette data from the following professional color websites:

- Coolors.co — Offers a wide range of designer-created color palettes and supports keyword-based search.
- ColorHunt.co — A popular color scheme platform featuring daily curated palettes.
- SchemeColor.com — A professional site with a broad variety of categorized color schemes.
- ColorHexa.com — Provides detailed color information along with related palette suggestions.
- Design-Seeds.com — Extracts color inspiration from nature and everyday life scenes.
- ColorCombos.com — Focuses on commonly used color combinations in commercial design.

In addition, the system uses Google search to retrieve more relevant resources, ensuring a diverse and comprehensive data source.

Scraping Methods

Direct Crawling from Target Websites:

- Construct specific URLs based on emotional keywords (e.g., coolors.co/palettes/search/happy)
- Use random User-Agent headers to simulate browser access
- Parse HTML pages to extract color palette data

Google Search Enhancement:

- Build search queries including the target website domain (e.g., happy color palette site:coolors.co)
- Parse search results to obtain relevant links
- Perform secondary crawling on the retrieved URLs

This hybrid strategy leverages structured data from professional color websites while using search engines to expand data coverage, enhancing both the scope and accuracy of the system.

Data Processing

Color Data Extraction Process

The system adopts a multi-layered approach to color extraction:

Page Structure Analysis:

- Identify common color palette containers (e.g., elements with classes like "palette" or "color-group")
- Extract HTML elements that contain background color or inline color styles

Color Value Extraction:

- Extract background-color or color values from the style attribute
- Retrieve color codes (e.g., data-hex) from data attributes
- Parse color names or hexadecimal codes from the element's text content

```
# 匹配十六进制颜色代码
hex_colors = re.findall(r'#(?:[0-9a-fA-F]{3}){1,2}\b', text)
# 匹配rgb/rgba颜色
rgb_colors = re.findall(r'rgba?\(\s*\d+\s*,\s*\d+\s*,\s*\d+\s*(?:,\s*[\d.]+\s*)?\)', text)
# 匹配颜色名称
color_names = re.findall(r'\b(?:red|green|blue|yellow|...)\b', text, re.IGNORECASE)
```

Data Deduplication and Validation

To ensure data quality, the system applies the following processing steps to the collected color palettes:

Deduplication:

- Convert each color list into a tuple
- Use a set to remove duplicate entries

```
unique_palettes = []
seen = set()
for palette in palettes:
    palette_tuple = tuple(palette)
    if palette_tuple not in seen:
        seen.add(palette_tuple)
        unique_palettes.append(palette)
```

System Architecture

Backend Implementation

The system's backend is built using the Python Flask framework, with the following core modules:

1. Route Controller:
   - / — Returns the front-end HTML page
   - /get_palettes — Handles requests for color palette generation

2. Crawler Scheduler:
   - Manages crawling tasks across multiple websites
   - Controls request frequency and timeout settings
   - Handles exceptions during the crawling process

3. Data Processor:
   - Cleans raw data
   - Validates color formats
   - Removes duplicate palettes

4. API interface:
Receive front-end requests
Return JSON format data

```python
@app.route('/get_palettes', methods=['POST'])
def get_palettes():
    keyword = request.form.get('keyword', '').strip().lower()
    if not keyword:
        return jsonify({'error': 'Please enter a keyword'})
    try:
        palettes = crawl_palette_sites(keyword)
        if not palettes:
            return jsonify({'error': f'No color palettes found for "{keyword}"'})
        return jsonify({
            'keyword': keyword,
            'palettes': unique_palettes[:5]
        })
    except Exception as e:
        return jsonify({'error': str(e)})
```

API Integration

Integration with DeepSeek API

Considering the need for logical, expressive, and stylistically coherent fashion suggestions, I initially explored both ChatGPT and DeepSeek. After comparing factors such as performance and pricing, I ultimately chose the DeepSeek API to generate personalized fashion recommendations for users.

With the API integrated, the system can still provide outfit suggestions even when the input color values cannot be retrieved through web scraping.

1. Request construction

```
const response = await fetch(DEEPSEEK_API_URL, {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
        'Authorization': `Bearer ${DEEPSEEK_API_KEY}`
    },
    body: JSON.stringify({
        model: 'deepseek-chat',
        messages: [{role: 'user', content: prompt}],
        temperature: 0.7,
        max_tokens: 1000
    })
});
```

2.Cue word design

```
let prompt;
if (colors.length > 0) {
    prompt = `Provide a detailed fashion advice based on the emotion "${emotion}"
    and these color palette: ${colorList}. Suggest what color to use for tops,
    what colors would work well for bottoms, and what accessories would complement
    this look. Also suggest suitable styles and materials...`;
} else {
    prompt = `Provide a detailed fashion advice based on the emotion "${emotion}".
    Suggest color combinations, styles, and materials that would best represent...`;
}
```

Output Design

Front-end technical implementation

1. Responsive design:
Using CSS Flexbox and Grid layouts
Adapting to different screen sizes

```css
@media (max-width: 768px) {
    .palettes {
        grid-template-columns: 1fr;
    }
}
```

2. Interactive function, color click to copy

```javascript
colorElement.addEventListener('click', () => {
    navigator.clipboard.writeText(color).then(() => {
        colorElement.textContent = 'Copied!';
        setTimeout(() => {
            colorElement.textContent = color;
        }, 1000);
    });
});
```

Technology Stack and Dependencies

The system is developed using Python 3.x as the programming language. The main dependencies are as follows:

- **beautifulsoup4==4.12.3**
- **requests==2.31.0**
- **flask==3.0.2**

All dependencies are managed through a requirements.txt file.

Key Technical Implementations

Web Parsing Technology

The system uses the BeautifulSoup4 library to perform multi-mode HTML parsing.

```
def extract_coolors_palettes(soup):
    palettes = []
    for container in soup.select('.palette_container'):
        colors = [f"#{color['data-hex']}" for color in container.select('.palette_color')]
        if len(colors) >= 3:  # 有效配色方案至少包含3种颜色
            palettes.append(colors)
    return palettes
```

Web Request Optimization
Anti-crawl strategy via Requests library:

```python
# 随机User-Agent生成器
USER_AGENTS = [
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64)...",
    "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)..."
]

def get_with_retry(url, max_retries=3):
    for attempt in range(max_retries):
        try:
            response = requests.get(
                url,
                headers={'User-Agent': random.choice(USER_AGENTS)},
                timeout=10
            )
            response.raise_for_status()
            return response
        except Exception as e:
            if attempt == max_retries - 1:
                raise
            time.sleep(2 ** attempt)
```

Final Display

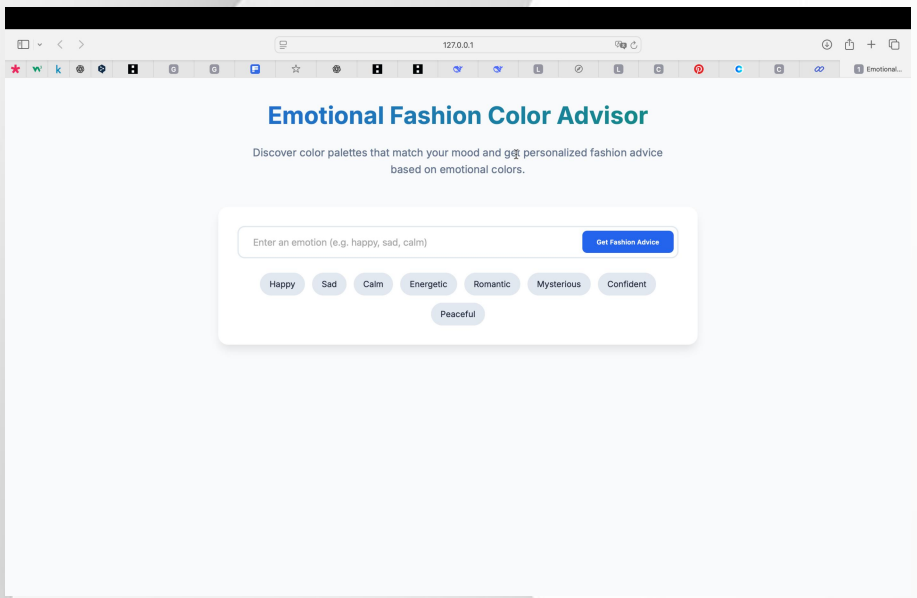


Figure 1. Emotion input interface with keyword buttons and text field.

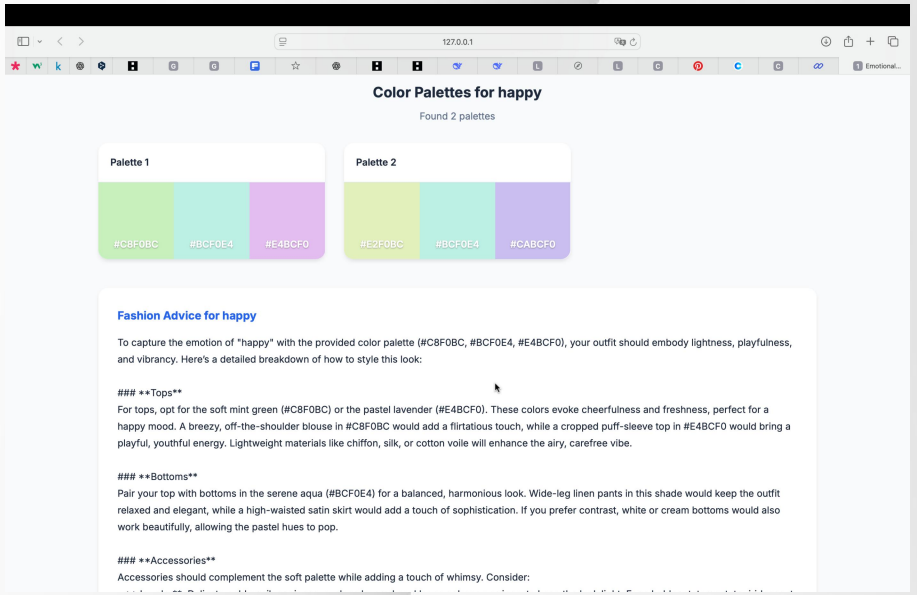


Figure 2. Display of matching color palettes and generated fashion advice.

The final interface of the system is clean and intuitive, guiding users from emotional input to outfit suggestions with ease.

On the homepage, users can either type in an emotion (e.g., happy, sad, calm) or select from preset keywords. Once submitted, the system retrieves matching color palettes from various online sources and displays them as color swatches.

Each palette includes its corresponding hex codes for clear visual reference. Below the palettes, personalized fashion advice is generated via the DeepSeek API, offering styling suggestions for tops, bottoms, and accessories based on the selected emotion and color mood.

Since the system fetches data in real-time through web scraping, the results may take a few seconds to load. This ensures the color palettes and recommendations are accurate and up-to-date.

This final display not only provides practical fashion guidance, but also effectively translates abstract emotions into visual and color-driven outfit solutions.

Reflection

This project involved several technical and strategic challenges that led to valuable learning outcomes.

At first, I planned to crawl product images from fashion websites like ASOS and Pinterest based on emotional keywords. However, most sites had anti-scraping measures, preventing access to valid image data. To overcome this, I shifted to collecting color palettes from professional color websites and enhanced the process with Google search queries. This hybrid approach improved both coverage and accuracy.

The system was built using Flask for backend logic and DeepSeek API for generating outfit suggestions. The API ensured that even without color data, the system could still provide results.

Overall, this project improved my skills in data collection, system design, and API integration. It also laid a solid foundation for future development, such as adding image generation or emotion recognition features.

References

1. Coolors. Color palette generator & inspiration. Retrieved from https://coolors.co
2. Color Hunt. Curated color palettes for designers and artists. Retrieved from https://colorhunt.co
3. SchemeColor. Color schemes and combinations. Retrieved from https://www.schemecolor.com
4. ColorHexa. Color encyclopedia and palette generator. Retrieved from https://www.colorhexa.com
5. Design Seeds. Color inspiration from nature and lifestyle. Retrieved from https://www.design-seeds.com
6. ColorCombos. Commercial color combinations and design tools. Retrieved from https://www.colorcombos.com
7. Google Search. Used to expand data coverage through keyword-based queries. Retrieved from https://www.google.com
8. DeepSeek API. AI-powered text generation for fashion recommendations. Retrieved from https://deepseek.com
9. BeautifulSoup4. Python library for parsing HTML and XML. Retrieved from https://www.crummy.com/software/BeautifulSoup/
10. Requests. HTTP library for Python. Retrieved from https://requests.readthedocs.io/
11. Flask. Web development framework for Python. Retrieved from https://flask.palletsprojects.com/
12. https://www.verywellmind.com/color-psychology-2795824
13. https://en.wikipedia.org/wiki/Plutchik%27s_wheel_of_emotions