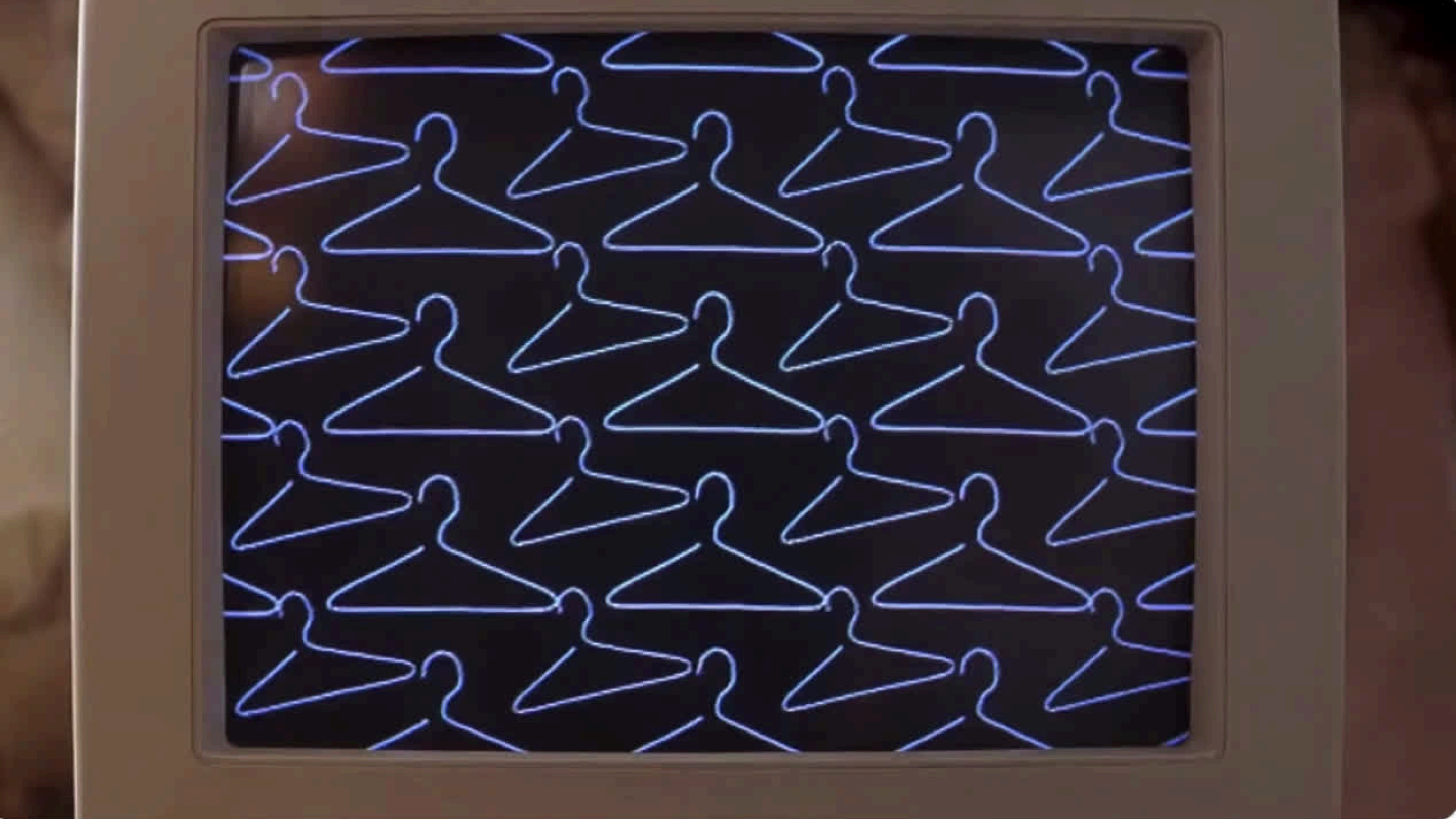
REDRESS

A FASHION RECOMMENDER
INSPIRED BY CHER'S CLOSET

Is There A Problem
Here?



OUR CODE

tensorflow, keras, sklearn, cv2

```
[ ]  data = fashion_mnist.load_data()
```

```
[ ]  #fashion_mnist = tf.keras.datasets.fashion_mnist

     (train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

```
[ ]  class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
                    'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']
```

```
plt.figure(figsize=(10,10))
for i in range(25):
    plt.subplot(5,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(train_images[i], cmap=plt.cm.binary)
    plt.xlabel(class_names[train_labels[i]])
plt.show()
```

LABEL

Ankle boot

T-shirt/top

THE FIRST MODEL

```
from tensorflow.keras import layers
import tensorflow as tf
#from keras_complex import ComplexFlatten, ComplexDense

model = tf.keras.Sequential([
    layers.Flatten(input_shape=(28, 28)),
    layers.Dense(128, activation='relu', dtype=np.float32),
    layers.Dense(10, dtype=np.float32)
])
```

THE FIRST MODEL

```
from tensorflow.keras import layers
import tensorflow as tf
#from keras_complex import ComplexFlatten, ComplexDense

model = tf.keras.Sequential([
    layers.Flatten(input_shape=(28, 28)),
    layers.Dense(128, activation='relu', dtype=np.float32),
    layers.Dense(10, dtype=np.float32)
])
```

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy
               (from_logits=True),
              metrics=['accuracy'])
```

TRAIN & MAKE PREDICTIONS


```
model.fit(train_images, train_labels, epochs=10)
```

```
test_loss, test_acc = model.evaluate(test_images,  test_labels, verbose=2

print('\nTest accuracy:', test_acc)
```

```
probability_model = tf.keras.Sequential([model, tf.keras.layers.Softmax()])
predictions = probability_model.predict(test_images)
```

LET'S VISUALIZE

```
def plot_image(i, predictions_array, true_label, img):
    true_label, img = true_label[i], img[i]
    plt.grid(False)
    plt.xticks([])
    plt.yticks([])

    plt.imshow(img, cmap=plt.cm.binary)

    predicted_label = np.argmax(predictions_array)
    if predicted_label == true_label:
        color = 'blue'
    else:
        color = 'red'

    plt.xlabel("{} {:2.0f}% ({})".format(class_names[predicted_label],
                                         100*np.max(predictions_array),
                                         class_names[true_label]),
                                         color=color)
```

LET'S VISUALIZE

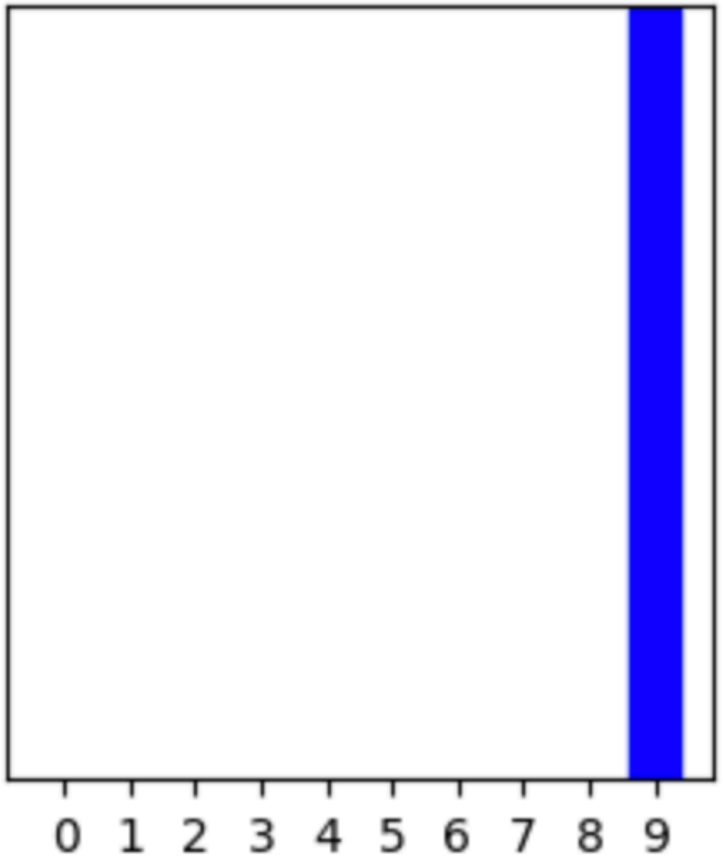
```
def plot_value_array(i, predictions_array, true_label):
    true_label = true_label[i]
    plt.grid(False)
    plt.xticks(range(10))
    plt.yticks([])
    thisplot = plt.bar(range(10), predictions_array, color="#7777
    plt.ylim([0, 1])
    predicted_label = np.argmax(predictions_array)

    thisplot[predicted_label].set_color('red')
    thisplot[true_label].set_color('blue')
```

LET'S VISUALIZE

```
i = 0
plt.figure(figsize=(6,3))
plt.subplot(1,2,1)
plot_image(i, predictions[i], test_labels, test_images)
plt.subplot(1,2,2)
plot_value_array(i, predictions[i],  test_labels)
plt.show()
```

Ankle boot 100% (Ankle boot)



THE SECOND MODEL

Part I - Content based recommender

THE SECOND MODEL

Part 1 - Item recommender

Step 1 - Import libraries and load dataset

```
import numpy
import sklearn
import tensorflow
import pandas as pd
```

	id	gender	masterCategory	subCategory	articleType	baseColour	season	year	usage	productDisplayName
0	15970	Men	Apparel	Topwear	Shirts	Navy Blue	Fall	2011.0	Casual	Turtle Check Men Navy Blue Shirt
1	39386	Men	Apparel	Bottomwear	Jeans	Blue	Summer	2012.0	Casual	Peter England Men Party Blue Jeans
2	59263	Women	Accessories	Watches	Watches	Silver	Winter	2016.0	Casual	Titan Women Silver Watch
3	21379	Men	Apparel	Bottomwear	Track Pants	Black	Fall	2011.0	Casual	Manchester United Men Solid Black Track Pants
4	53759	Men	Apparel	Topwear	Tshirts	Grey	Summer	2012.0	Casual	Puma Men Grey T-shirt

THE SECOND MODEL

Part I - Item recommender

Step 2 - Preprocess data

	id	gender	masterCategory	subCategory	articleType	baseColour	season
0	15970	2	1	38	103	25	0
1	39386	2	1	6	56	2	2
2	59263	4	0	42	139	37	3
3	21379	2	1	6	127	1	0
4	53759	2	1	38	133	13	2

THE SECOND MODEL

Part I - Item recommender

EXAMPLE: `'Catwalk Women Leather Flats'`

THE SECOND MODEL

Part I - Item recommender

Step 3 - Similarity matrix and scores

	Leather flats	Leather sandals	Black flats	...	Gold sandals	Grey sneakers
Leather flats	array([[1.	, 0.99953563,	0.99982317,	...,	0.99982433,	0.99983015,
Leather sandals	[0.99953563,	1.	, 0.99885046,	...,	0.99914361,	0.99984409,
Black flats	[0.99982317,	0.99885046,	1.	, ...,	0.99984267,	0.99945453,
...	...,					
Gold sandals	[0.99982433,	0.99914361,	0.99984267,	...,	1.	, 0.99969462,
Grey sneakers	[0.99983015,	0.99984409,	0.99945453,	...,	0.99969462,	1. ,

ITEM SIMILARITY SCORE:

Mean:	0.9997	0.9995	0.9996	0.9991	0.9989

THE SECOND MODEL

Part I - Item recommender

Step 4 - Function to recommend similar items

```python
def recommend_similar_items(item_name, top_n=5):
    # Find the index of the item in the dataset
    item_index = data[data['productDisplayName'] == item_name].index[0]

    # Get the similarity scores for the given item
    item_similarities = similarity_matrix[item_index]

    # Get indices of top similar items
    similar_indices = item_similarities.argsort()[::-1][1:top_n+1]  # Exclude itself

    # Get the names of top similar items
    similar_items = data.iloc[similar_indices]['productDisplayName'].values

    return similar_items
```

Step 5 - Show results

THE SECOND MODEL

Part I - Item recommender

```
Recommended items similar to 'Catwalk Women Leather Flats':
```

```
Grendha Women Jola Cigana Black Sandals
```

```
Catwalk Women Red Flats
```

THE SECOND MODEL

Part II - Outfit building

THE SECOND MODEL

Part II - Outfit building

Step 1 - Function to randomly select items

```
# Function to randomly select an item based on user input
def select_item(subcategory, gender, season, usage):
    items = original_data[(original_data['subCategory'] == subcategory) &
                          (original_data['gender'] == gender) &
                          (original_data['season'] == season) &
                          (original_data['usage'] == usage)]
    return items.sample(1)
```

THE SECOND MODEL

Part II - Outfit building

Step 2 - User inputs preferences

```
# Specify characteristics
gender = input("Enter gender (Men/Women): ")
season = input("Enter season (Summer/Fall/Winter/Spring): ")
usage = input("Enter usage (Casual/Formal/Party): ")
```

EXAMPLE:

Gender: Female

Season: Spring

Usage: Formal

THE SECOND MODEL

Part II - Outfit building

EXAMPLE:

- Shoes
- Bag
- Jewellery
- 50% Dress
- 50% Topwear + Bottomwear

Step 3 - Define which items to choose

```python
# Randomly select one item from each subcategory
selected_items = {}
chance = random.randint(0,1)
if gender == 'Women' and chance==0:
    subcategories = ['Topwear', 'Bottomwear', 'Shoes', 'Bags', 'Jewellery']
elif gender == 'Women' and chance==1:
    subcategories = ['Dress', 'Shoes', 'Bags', 'Jewellery']
elif gender == 'Men':
    subcategories = ['Topwear', 'Bottomwear', 'Shoes', 'Watches']
for subcategory in subcategories:
    selected_items[subcategory] = select_item(subcategory, gender, season, usage)
```

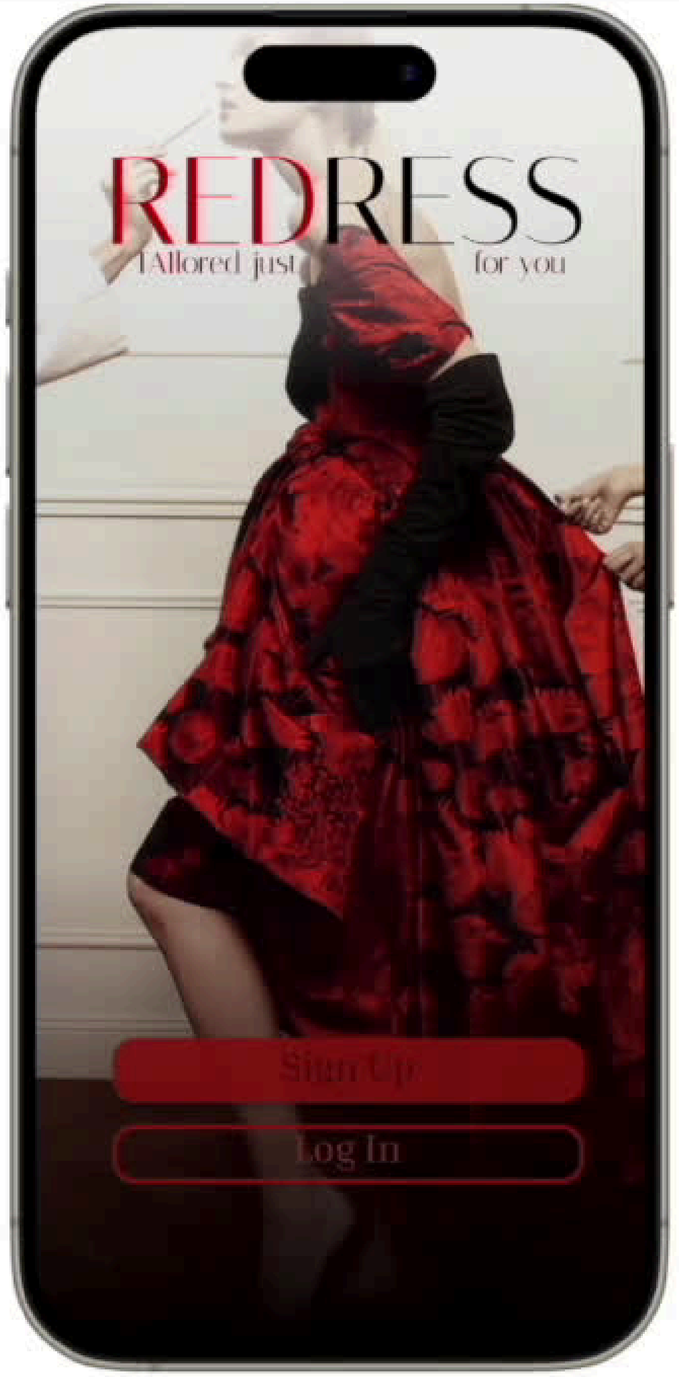
THE SECOND MODEL

Part II - Outfit building

Step 4 - Show outfit combination

Selected items:
Dress: United Colors of Benetton Women Solid Grey Dresses
Shoes: Enroute Women Casual Taupe Wedges
Bags: Puma Women Core Lite Shopper White Bags
Jewellery: Lencia Sterling Silver Pendant

USER INTERFACE:
OUR APP


REDRESS
Tailored just
for you
Sign Up
Log In

CONCLUSION

Current app:

- Recognizes items
- Recommends similar items
- Builds outfit combinations

Future implementations:

- Recognize characteristics like color or texture
- Users rate clothes for collaborative filtering recommendation
- Connect weather and calendar app