To solve this problem, we will be writing a web scraper using Python that can scrape images from the website provided in the prompt. Here is the process for coding the solution:

Import Required Libraries: First, we need to import the required libraries, including Requests and BeautifulSoup, which are commonly used for web scraping.

Define the URL: Next, we will define the URL of the website we want to scrape, which is <https://www.ralphlauren.nl/en/men/clothing/hoodies-sweatshirts/10204?webcat=men%7Cclothing%7Cmen-clothing-hoodies-sweatshirts>

Send a Request: Then, we will send a request to the website using the requests.get() method to get the HTML content of the website.

Parse HTML: After getting the HTML content, we will parse it using BeautifulSoup to extract the relevant information.

Find Images: We will then use BeautifulSoup to find the images on the website. This can be done using the find\_all() method and specifying the tag name as 'img'.

Filter Images: Next, we will filter the images based on the criteria mentioned in the prompt, i.e., images of people wearing a particular fabric and images of fabrics (not worn by someone). This can be done using a combination of tags and classes.

Scrap Image URLs: Once we have filtered the images, we will extract the URLs of the images using the 'src' attribute.

Save Images: Finally, we will save the images on the local machine using the requests.get() method and the open() method.

Handle Blocking: To handle blocking, we will use a proxy server to make the requests. This can be done by specifying the 'proxies' parameter in the requests.get() method.

Convert to .jpg Format: To convert the scrapped images into .jpg format, we will use the Python Imaging Library (PIL) to load the images and then save them in .jpg format using the save() method.

Part 2

The code uses the OpenCV library to process the original cloth image and the cloth mask to produce a new image with a blue background. The steps involved in the implementation are as follows:

Importing required libraries: cv2 and numpy libraries are imported for use in the code.

Defining the preprocess function: The preprocess function takes two arguments - cloth and cloth\_mask, which are the file paths for the original cloth image and the cloth mask image, respectively.

Loading the original cloth image: cv2.imread function is used to load the original cloth image, which is stored in the img variable.

Loading the cloth mask image: cv2.imread function is used to load the cloth mask image, which is stored in the mask variable. The second argument, cv2.IMREAD\_GRAYSCALE, is used to specify that the image should be read as grayscale.

Creating a blue background image: np.zeros\_like function is used to create an array with the same shape as the img array and fill it with zeros. This array represents the blue background. The blue color is specified using the tuple (255, 0, 0), which represents the blue color in the BGR color space used by OpenCV.

Converting the cloth mask image to a binary image: cv2.threshold function is used to convert the cloth mask image to a binary image, where all pixels with a value greater than or equal to 127 are set to 255 (white) and all other pixels are set to 0 (black). The binary mask is stored in the binary\_mask variable.

Replacing the background with the cloth: blue\_background array is updated using indexing. All pixels in the blue background that are masked with 0 in the binary\_mask are replaced with the corresponding pixels from the img array. This results in the blue background being replaced with the cloth in the areas where the binary\_mask is 0.

Saving the blue background cloth image: The cv2.imwrite function is used to save the blue\_background array as an image file named blue\_background\_cloth.png.

Returning the blue background cloth image: The blue\_background array is returned from the function as the output.

This code can be run in a Python environment with OpenCV installed. To realize the code, you need to provide the file paths for the original cloth image and the cloth mask and call the preprocess function with those arguments. The function will return the blue background cloth image, and also save it to a file named "blue\_background\_cloth.png".

The steps involved in coding this program are as follows:

Import the required libraries: Start by importing the cv2 and numpy libraries, which will be used in the code.

Define the function: Define the preprocess function, which will take two arguments - the file path for the original cloth image and the file path for the cloth mask image.

Load the original cloth image: Use the cv2.imread function to load the original cloth image and store it in a variable.

Load the cloth mask image: Use the cv2.imread function to load the cloth mask image and store it in a variable. Specify the cv2.IMREAD\_GRAYSCALE flag to ensure that the image is read as grayscale.

Create a blue background image: Use the np.zeros\_like function to create an array with the same shape as the original cloth image, filled with zeros. Fill this array with blue color using the blue\_background[:] assignment.

Convert the cloth mask image to binary: Use the cv2.threshold function to convert the cloth mask image to a binary image. Store the binary image in a variable.

Replace the background with the cloth: Use the binary cloth mask to replace the blue background with the cloth. Do this by using indexing to select the part of the blue background to replace with the cloth.

Save the blue background cloth image: Use the cv2.imwrite function to save the blue background cloth image to a file named blue\_background\_cloth.png.

Return the blue background cloth image: Return the blue background cloth image from the preprocess function as the output.

By following these steps, you can implement a program that will take the original cloth image and cloth mask as inputs and output the blue background cloth image.

The website <https://colab.research.google.com/drive/18RenTYhuPVip9SHdMLn-vnK0K57B--um#scrollTo=D0h2Y-oOCnXJ> is a Google Colab notebook that contains information and code related to a pre-trained machine learning model for generating cloth masks. It is likely that the model has been trained to segment cloth images and generate black and white masks that separate the cloth from its background.

The exact details of the model and its training data are not specified on the website, but the website provides a code snippet that allows users to input their own cloth images, run them through the pre-trained model, and obtain the generated cloth masks. The website also provides a description of how to use the code and how to download the generated masks.