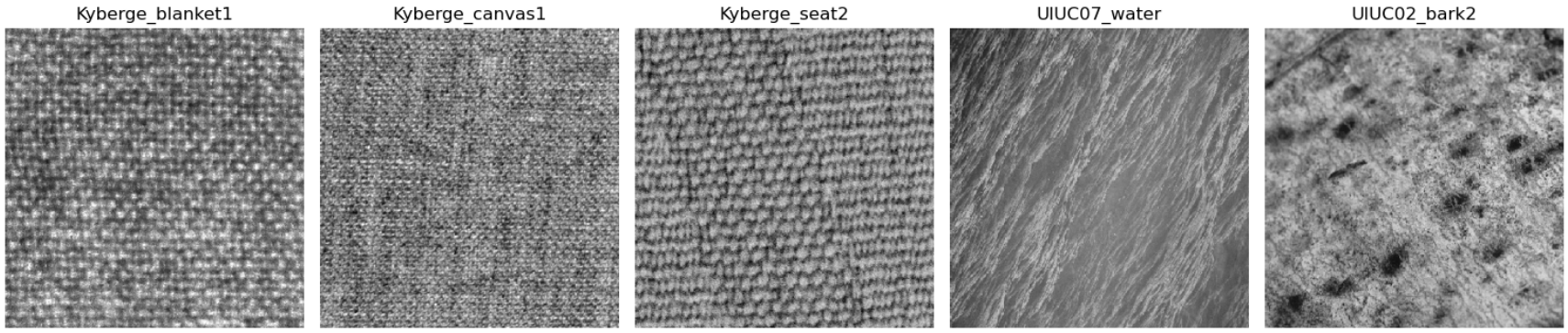
In [1]:

```
import os
import matplotlib.pyplot as plt
from PIL import Image

# Set the path to the directory where the dataset is stored in Google Drive
dataset_directory = '/Users/samkhatri/Downloads/Splited/train'

# Initialize a counter for the number of classes processed
num_classes = 0

# Initialize a figure for plotting
plt.figure(figsize=(15, 10))

# Display one image from each class in both vertical and horizontal orientations
for class_name in os.listdir(dataset_directory):
    class_dir = os.path.join(dataset_directory, class_name)

    # Check if it's a directory
    if os.path.isdir(class_dir):
        # Get the first image in the directory
        image_name = os.listdir(class_dir)[0]
        image_path = os.path.join(class_dir, image_name)

        # Load the image
        image = Image.open(image_path)

        # Increment the class counter
        num_classes += 1

        # Add a subplot for the original image
        plt.subplot(2, 5, num_classes)
        plt.imshow(image)
        plt.title(class_name)
        plt.axis('off')

        # Check if we have processed 5 classes
        if num_classes == 5:
            break
```

```
# Adjust subplot parameters for a nicer layout
plt.tight_layout()
plt.show()
```



In [4]:
```
import numpy as np
from PIL import Image
from tensorflow.keras.applications.vgg16 import VGG16, preprocess_input
from tensorflow.keras.models import Model
from sklearn.metrics.pairwise import cosine_similarity


train_path = '/Users/samkhatri/Downloads/Splited/train'
valid_path = '/Users/samkhatri/Downloads/Splited/valid'

def extract_features(img_path):
    img = Image.open(img_path)
    img = img.resize((224, 224))
    img_array = np.expand_dims(np.array(img), axis=0)
    img_array = preprocess_input(img_array)

    model = VGG16(weights='imagenet')
    features_extract = Model(inputs=model.input, outputs=model.get_layer('fc2').output)

    features = features_extract.predict(img_array)
    return features.flatten()

def helper1(features1, features2):
    return cosine_similarity([features1], [features2])[0][0]
```

```
def helper2(query_image_path, similar_images_paths):
    plt.figure(figsize=(12, 6))


    plt.subplot(1, len(similar_images_paths) + 1, 1)
    plt.imshow(Image.open(query_image_path))
    plt.title("Query Image")
    plt.axis('off')


    for i, (img_path, _) in enumerate(similar_images_paths):
        plt.subplot(1, len(similar_images_paths) + 1, i + 2)
        plt.imshow(Image.open(img_path))
        plt.title(f"Similar {i+1}")
        plt.axis('off')

plt.show()
```

In [ ]:
```
train_f = []
train_loc = []

for class_name_train in class_names_train:
    class_dir = os.path.join(train_loc, os.listdir(train_path))

    if os.path.isdir(class_dir):
        for i in os.listdir(class_dir):
            i_location = os.path.join(class_dir, i)

            f_train = extract_features(i_location)

            train_f.append(f_train)
            train_loc.append(i)

train_f = np.array(train_features)
```

```
1/1 [==============================] - 0s 183ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 241ms/step
1/1 [==============================] - 0s 182ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 306ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 239ms/step
1/1 [==============================] - 0s 166ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 289ms/step
1/1 [==============================] - 0s 497ms/step
1/1 [==============================] - 0s 180ms/step
1/1 [==============================] - 0s 247ms/step
1/1 [==============================] - 0s 180ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 179ms/step
```

In [6]:
```
#Running the model on a sample image
valid_image = '/Users/samkhatri/Downloads/Splited/valid/Kyberge_blanket1/1.jpg'

features_valid = extract_features(valid_image)
similarities = [helper1(features_valid, j) for j in train_f]
similar_images = np.argsort(similarities)[-5:][::-1]
helper2(valid_image, [(train_paths[k], similarities[k]) for k in similar_images])
```

```
1/1 [==============================] - 0s 193ms/step
```