In [9]:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
import matplotlib.pyplot as plt
import tensorflow as tf
import numpy as np
import cv2
import os
```

In [33]:

```
# dir_path1 = 'ComputerVision/training/A'

# for i in os.listdir(dir_path1):
#   img = image.load_img(dir_path1 + '//' + i,target_size=(400,400))
#   plt.imshow(img)
#   plt.show()
```

In [6]:

```
# dir_path2 = 'ComputerVision/training/R'

# for i in os.listdir(dir_path2):
#   img = image.load_img(dir_path2 + '//' + i,target_size=(400,400))
#   plt.imshow(img)
#   plt.show()
```

In [7]:

```
# dir_path3 = 'ComputerVision/training/K'

# for i in os.listdir(dir_path3):
#   img = image.load_img(dir_path3 + '//' + i,target_size=(400,400))
#   plt.imshow(img)
#   plt.show()
```

In [ ]:

```
# dir_path3 = 'ComputerVision/training/B'

# for i in os.listdir(dir_path3):
#   img = image.load_img(dir_path3 + '//' + i,target_size=(400,400))
#   plt.imshow(img)
#   plt.show()
```

In [ ]:

In [14]:

```
train = ImageDataGenerator(rescale = 1/255)
validation = ImageDataGenerator(rescale = 1/255)
```

In [16]:
```
ls
```

```
 Volume in drive C is OS
 Volume Serial Number is 1A4A-6571

 Directory of C:\Users\K. RAVITEJA\Downloads

05-11-2022  06:48    <DIR>          .
05-11-2022  06:29    <DIR>          .ipynb_checkpoints
15-10-2022  14:30             2,757 04-using-the-node-modules-system.zip
09-05-2022  16:08           313,844 1.jpg
16-08-2022  15:46           273,247 10.1109ICSCCC.2018.8703316.pdf
13-06-2022  17:51           200,267 134_3_1834546_1655014334_AWS Course Compl
etion Certificate.pdf
25-03-2022  21:59            82,485 1646655437802.jpg
26-09-2022  15:01             1,668 194.CircularLLC++.txt
07-04-2022  21:11         2,941,462 19761A0528.pdf
04-11-2022  09:11         1,605,230 1st Connect Session (Python AI ).pptx
15-05-2022  23:41             1,109 2022_0502-CON (1).ics
15-05-2022  23:40             1,109 2022_0502-CON.ics
21-06-2022  08:45               210 2022_06_21_08_45_02_exportSecurityGroupsT
```

In [17]:
```
cd ComputerVision
```

```
C:\Users\K. RAVITEJA\Downloads\ComputerVision
```

In [19]:
```
ls
```

```
 Volume in drive C is OS
 Volume Serial Number is 1A4A-6571

 Directory of C:\Users\K. RAVITEJA\Downloads\ComputerVision

05-11-2022  06:48    <DIR>          .
05-11-2022  06:48    <DIR>          ..
05-11-2022  06:44    <DIR>          ComputerVision
               0 File(s)              0 bytes
               3 Dir(s)  265,913,503,744 bytes free
```

In [20]:
```
train_dataset = train.flow_from_directory('ComputerVision/training',
                                          target_size=(400,400),
                                          batch_size = 1,
                                          class_mode='categorical')
```

```
Found 316 images belonging to 26 classes.
```

In [21]:
```
validation_dataset = validation.flow_from_directory('ComputerVision/validation',
                                                    target_size = (400,400),
                                                    batch_size = 1,
                                                    class_mode = 'categorical')
```

```
Found 316 images belonging to 26 classes.
```

```
In [22]: train_dataset.class_indices
```

```
Out[22]: {'A': 0,
          'B': 1,
          'C': 2,
          'D': 3,
          'E': 4,
          'F': 5,
          'G': 6,
          'H': 7,
          'I': 8,
          'J': 9,
          'K': 10,
          'L': 11,
          'M': 12,
          'N': 13,
          'O': 14,
          'P': 15,
          'Q': 16,
          'R': 17,
          'S': 18,
          'T': 19,
          'U': 20,
          'V': 21,
          'W': 22,
          'X': 23,
          'Y': 24,
          'Z': 25}
```

```
In [23]: train_dataset.classes
```

```
Out[23]: array([ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,
                 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  2,  2,  2,  2,  2,  2,
                 2,  2,  2,  2,  2,  2,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,
                 3,  4,  4,  4,  4,  4,  4,  4,  4,  4,  4,  4,  4,  5,  5,  5,  5,
                 5,  5,  5,  5,  5,  5,  5,  5,  6,  6,  6,  6,  6,  6,  6,  6,  6,
                 6,  6,  6,  7,  7,  7,  7,  7,  7,  7,  7,  7,  7,  7,  7,  8,  8,
                 8,  8,  8,  8,  8,  8,  8,  8,  8,  8,  9,  9,  9,  9,  9,  9,  9,
                 9,  9,  9,  9,  9, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,
                11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 12, 12, 12, 12, 12,
                12, 12, 12, 12, 12, 12, 12, 13, 13, 13, 13, 13, 13, 13, 13, 13, 13,
                13, 13, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 15, 15, 15,
                15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16,
                16, 16, 16, 16, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 18,
                18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 19,
                19, 19, 19, 19, 19, 19, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20,
                20, 21, 21, 21, 21, 21, 21, 21, 21, 21, 21, 21, 21, 22, 22, 22, 22,
                22, 22, 22, 22, 22, 22, 22, 22, 23, 23, 23, 23, 23, 23, 23, 23, 23,
                23, 23, 23, 24, 24, 24, 24, 24, 24, 24, 24, 24, 24, 24, 24, 25, 25,
                25, 25, 25, 25, 25, 25, 25, 25, 25, 25])
```

```
In [24]: from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Dense,Conv2D,MaxPool2D,Flatten
```

In [25]:

```
def policy_network():
  model = Sequential()
  #model.add(Conv2D(16,(3,3),activation='relu',input_shape=(400,400,3)))

  model.add(Conv2D(16,(3,3),activation='relu',input_shape=(400,400,3)))
  model.add(MaxPool2D(2,2))

  #model.add(Conv2D(32,(3,3),activation='relu'))

  model.add(Conv2D(32,(3,3),activation='relu'))
  model.add(MaxPool2D(2,2))

  #model.add(Conv2D(64,(3,3),activation='relu'))

  model.add(Conv2D(64,(3,3),activation='relu'))
  model.add(MaxPool2D(2,2))

  model.add(Flatten())

  model.add(Dense(512,activation='relu'))

  #model.add(Dense(128,activation='relu'))
  model.add(Dense(26,activation='softmax'))

  model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accura

  return model
```

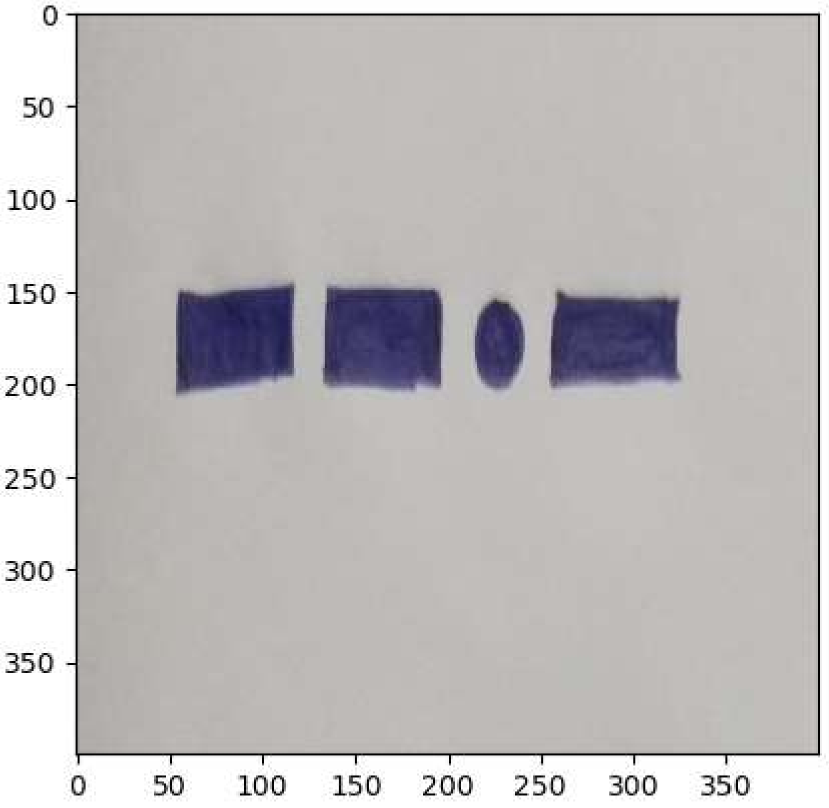
In [26]:

```
model = policy_network()
model.fit(train_dataset,
          steps_per_epoch = 2,
          epochs = 500,
          validation_data = validation_dataset)
```

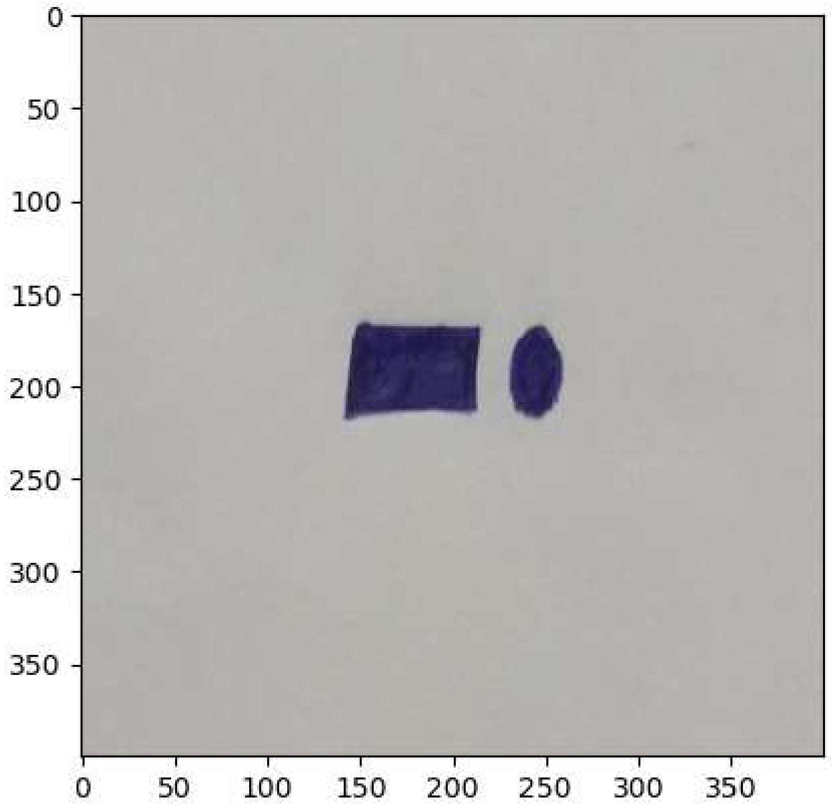
```
2/2 [==============================] - 16s 16s/step - loss: 0.0037 - accurac
y: 1.0000 - val_loss: 0.4478 - val_accuracy: 0.8829
Epoch 485/500
2/2 [==============================] - 16s 16s/step - loss: 0.8090 - accurac
y: 0.5000 - val_loss: 0.4444 - val_accuracy: 0.8861
Epoch 486/500
2/2 [==============================] - 16s 16s/step - loss: 3.5982e-04 - accu
racy: 1.0000 - val_loss: 0.4444 - val_accuracy: 0.8892
Epoch 487/500
2/2 [==============================] - 16s 16s/step - loss: 2.7330e-04 - accu
racy: 1.0000 - val_loss: 0.4459 - val_accuracy: 0.8892
Epoch 488/500
2/2 [==============================] - 16s 16s/step - loss: 3.5617 - accurac
y: 0.5000 - val_loss: 0.4266 - val_accuracy: 0.8861
Epoch 489/500
2/2 [==============================] - 17s 16s/step - loss: 0.1376 - accurac
y: 1.0000 - val_loss: 0.4059 - val_accuracy: 0.8924
Epoch 490/500
2/2 [==============================] - 16s 16s/step - loss: 0.0021 - accurac
y: 1.0000 - val_loss: 0.3981 - val_accuracy: 0.8924
```

In [ ]:

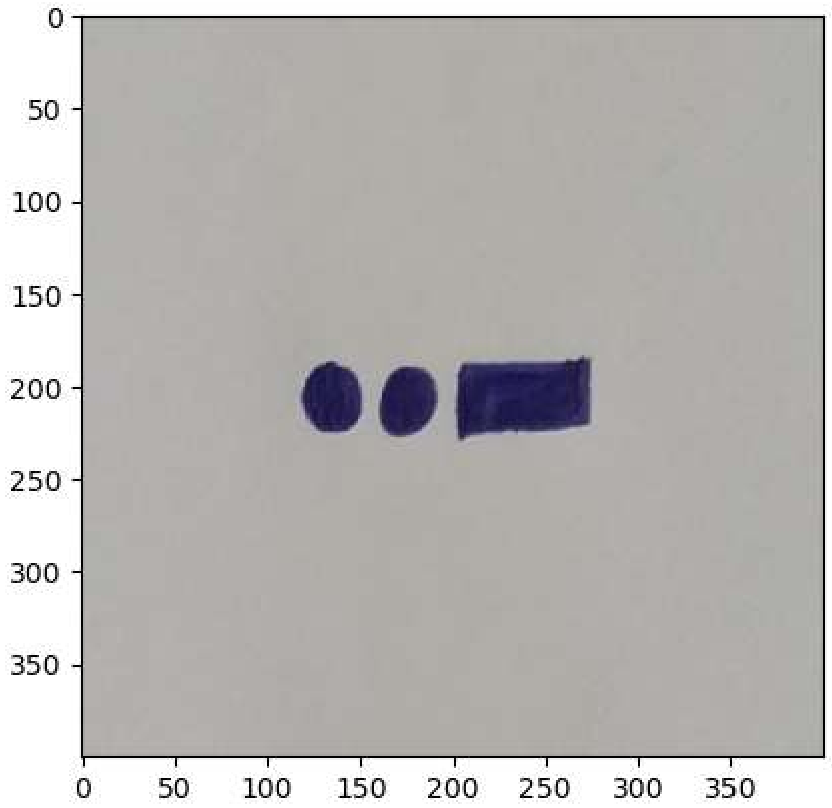
In [32]:
```
dir_path = 'ComputerVision/testing/'

for i in os.listdir(dir_path):
    img = image.load_img(dir_path + '//' + i,target_size=(400,400))
    plt.imshow(img)
    plt.show()

    X = image.img_to_array(img)
    X = np.expand_dims(X,axis=0)
    images = np.vstack([X])
    pred = model.predict(images)
    ans = pred[0]
    alphas = ['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q'
    for i in range(len(ans)):
        if(ans[i] == 1):
            print("Prediction = ",alphas[i])
            break
```



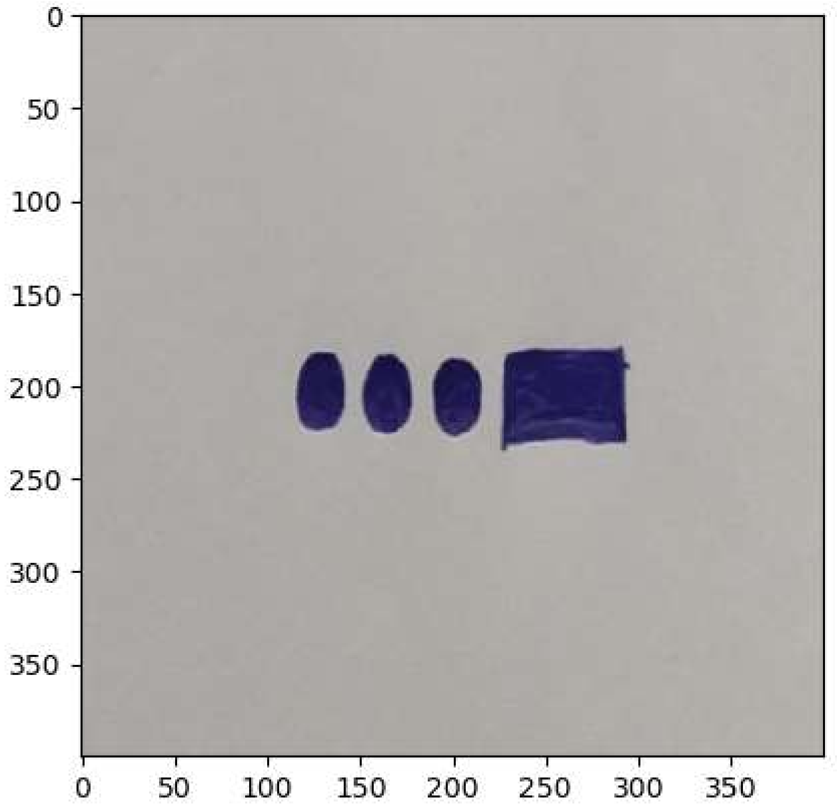
```
1/1 [==============================] - 0s 78ms/step
Prediction =  Q
```



```
1/1 [==============================] - 0s 62ms/step
Prediction =  N
```

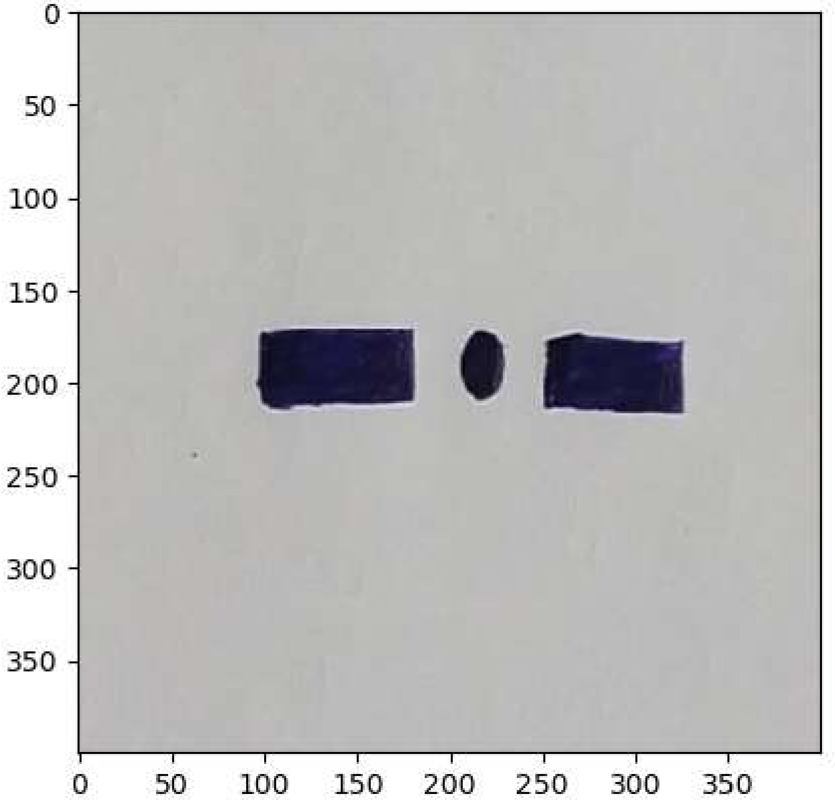


```
1/1 [==============================] - 0s 62ms/step
Prediction =  U
```



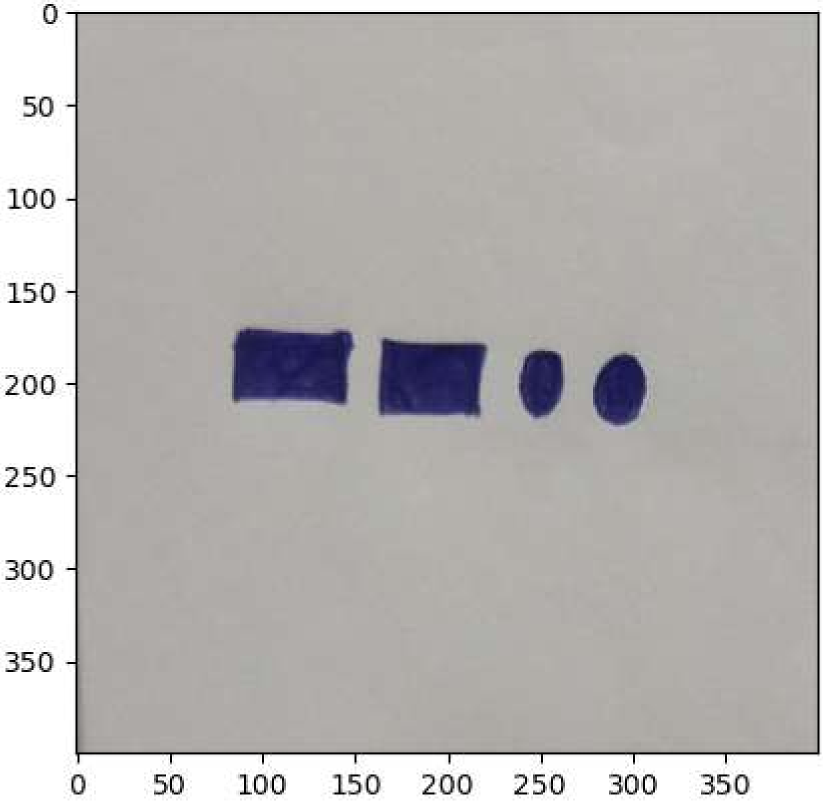
```
1/1 [==============================] - 0s 78ms/step
Prediction =  V
```

```
1/1 [==============================] - 0s 78ms/step
Prediction =  F
```

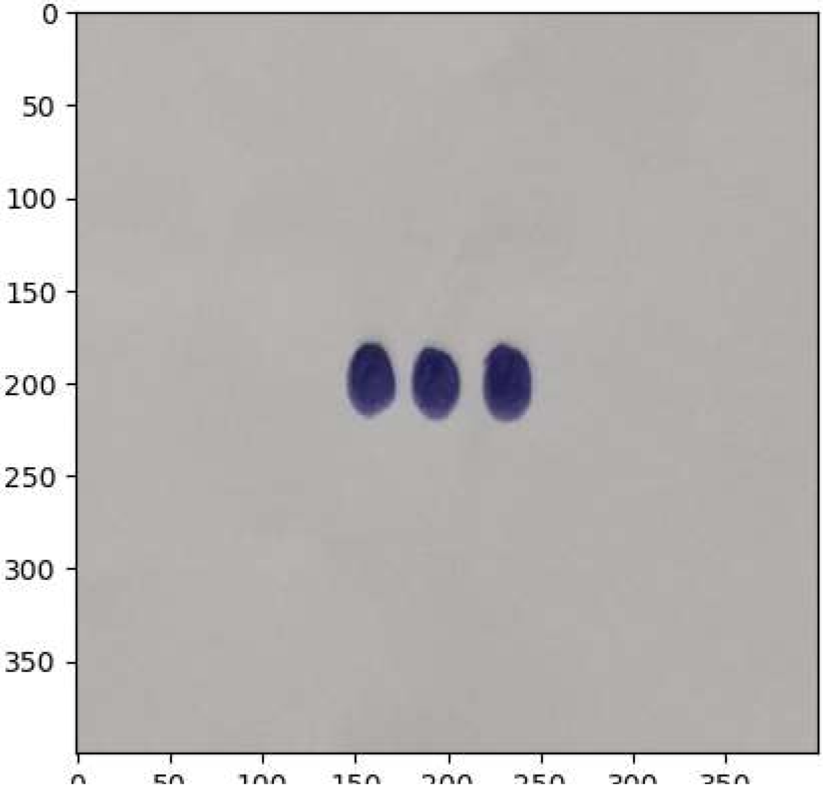


```
1/1 [==============================] - 0s 47ms/step
Prediction =  K
```

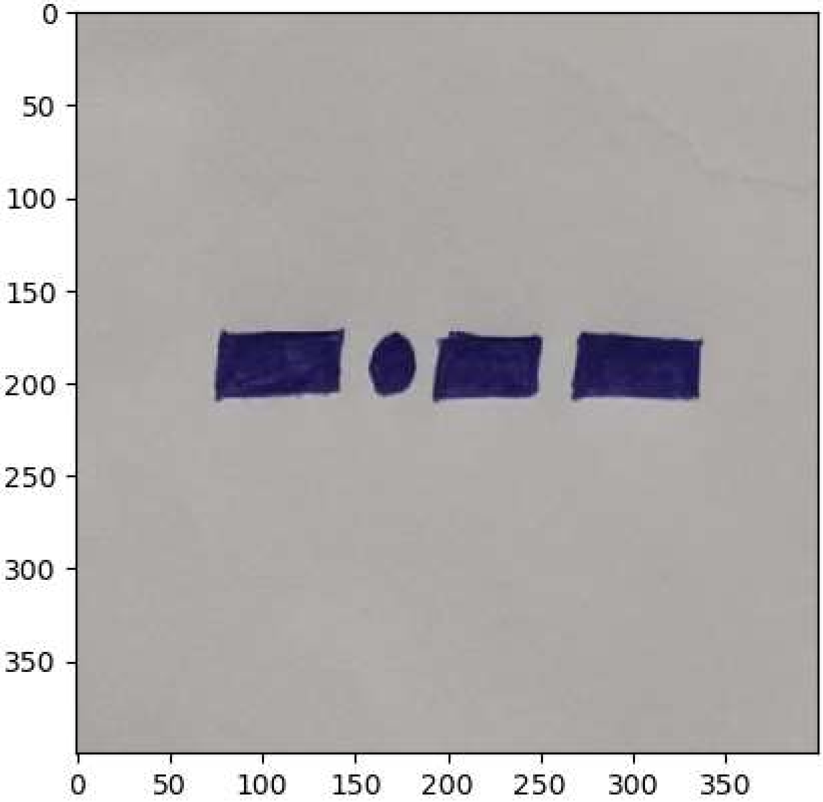
```
1/1 [==============================] - 0s 78ms/step
Prediction =  R
```



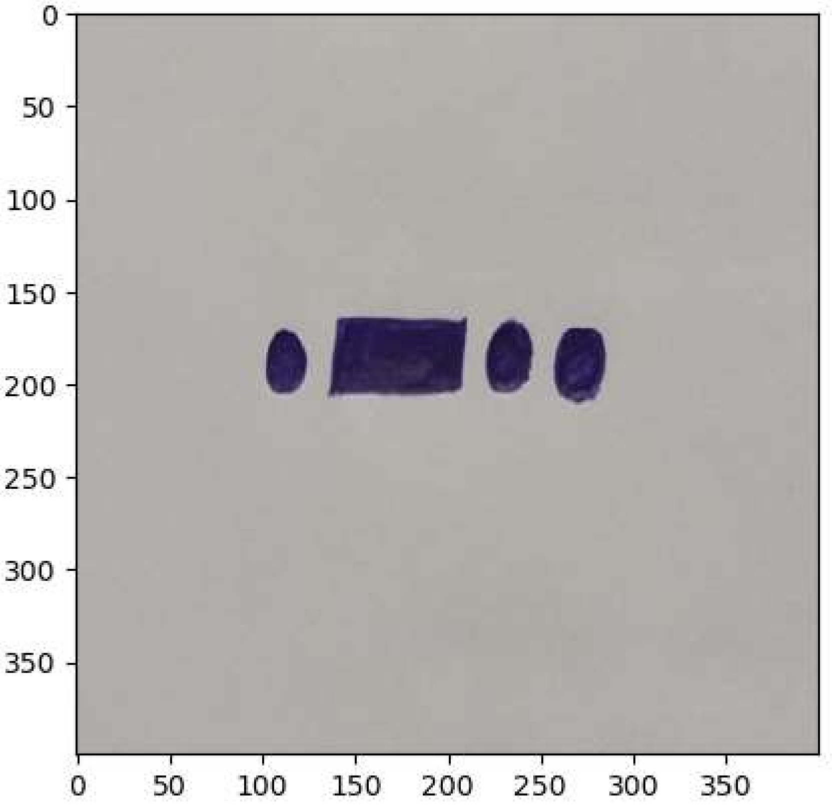
```
1/1 [==============================] - 0s 63ms/step
Prediction =  Z
```



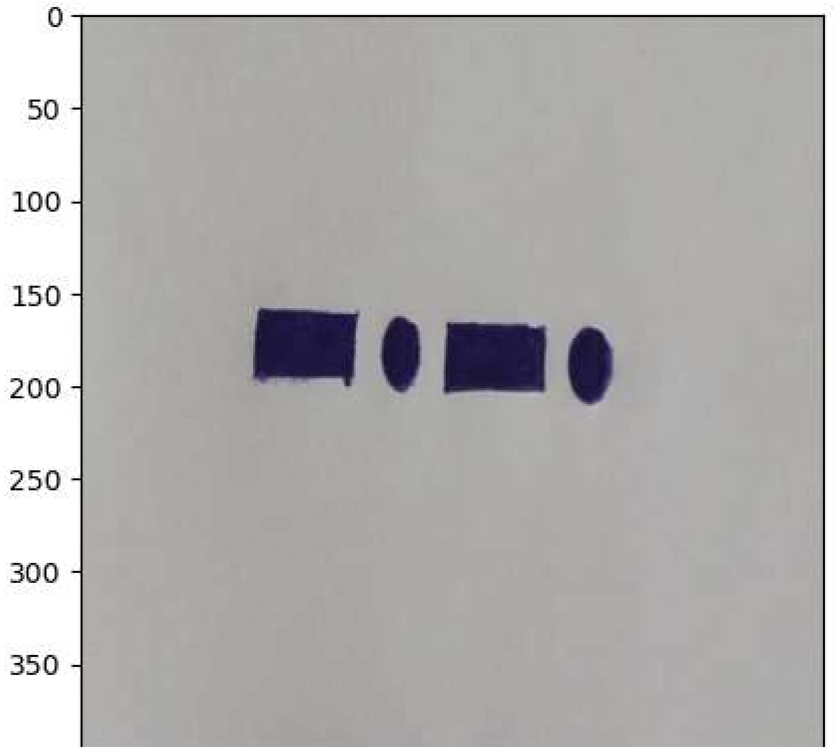
```
1/1 [==============================] - 0s 62ms/step
Prediction =  S
```



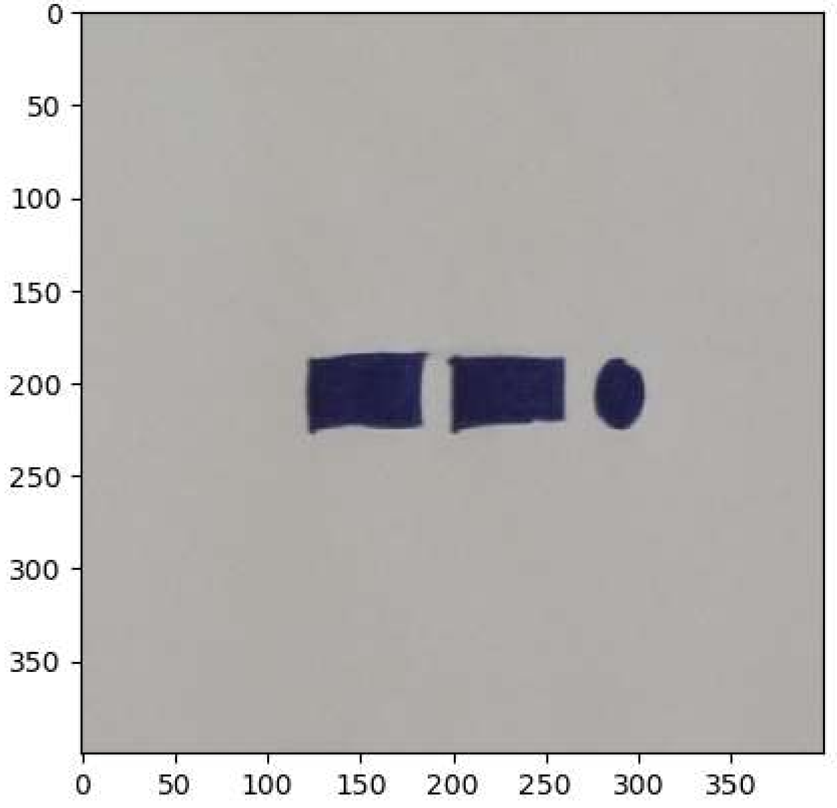
```
1/1 [==============================] - 0s 78ms/step
Prediction =  Y
```



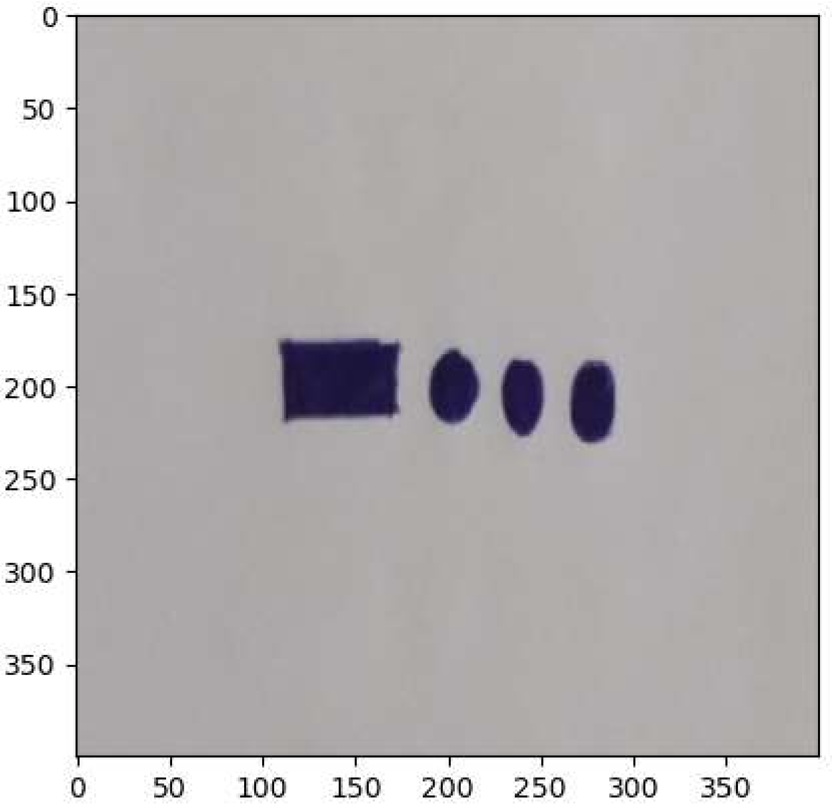
```
1/1 [==============================] - 0s 78ms/step
Prediction =  L
```



```
1/1 [==============================] - 0s 78ms/step
Prediction =  C
```

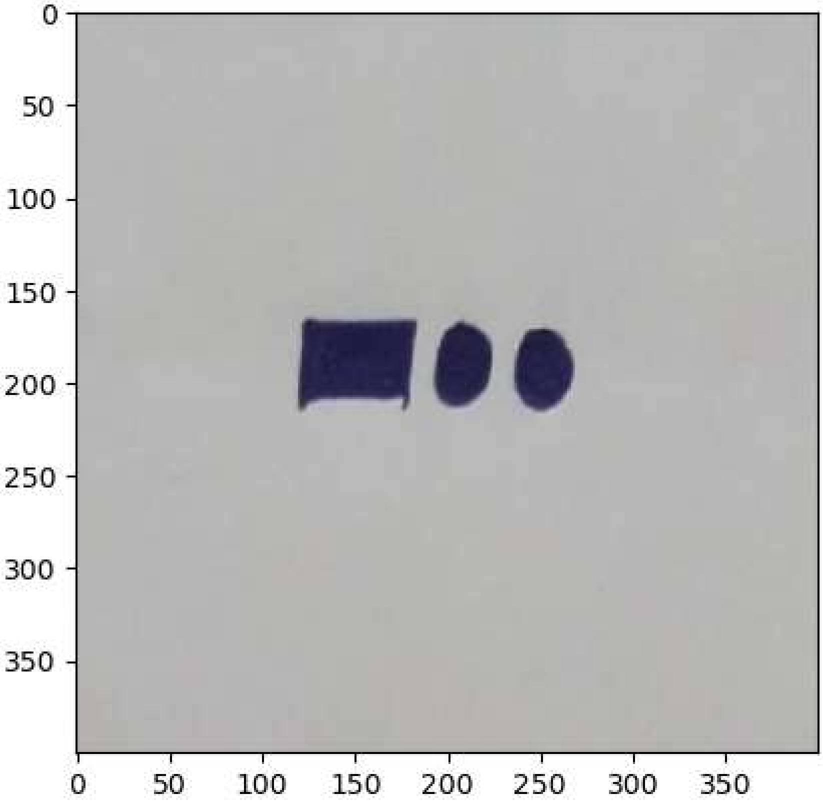


```
1/1 [==============================] - 0s 63ms/step
Prediction =  G
```



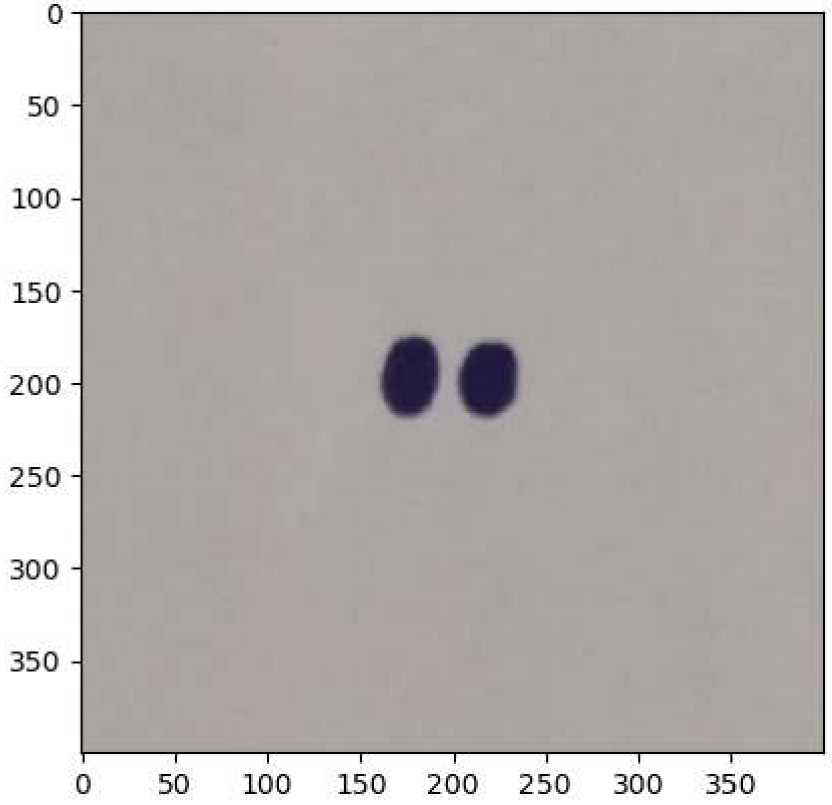
```
1/1 [==============================] - 0s 63ms/step
Prediction =  B
```

```
1/1 [==============================] - 0s 64ms/step
Prediction =  W
```

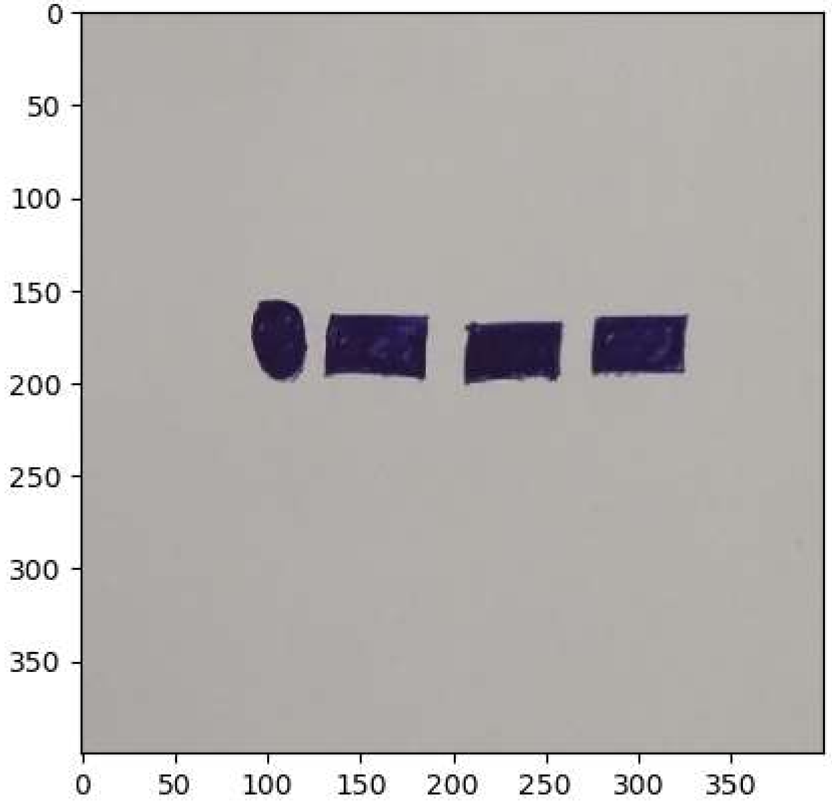


```
1/1 [==============================] - 0s 64ms/step
```

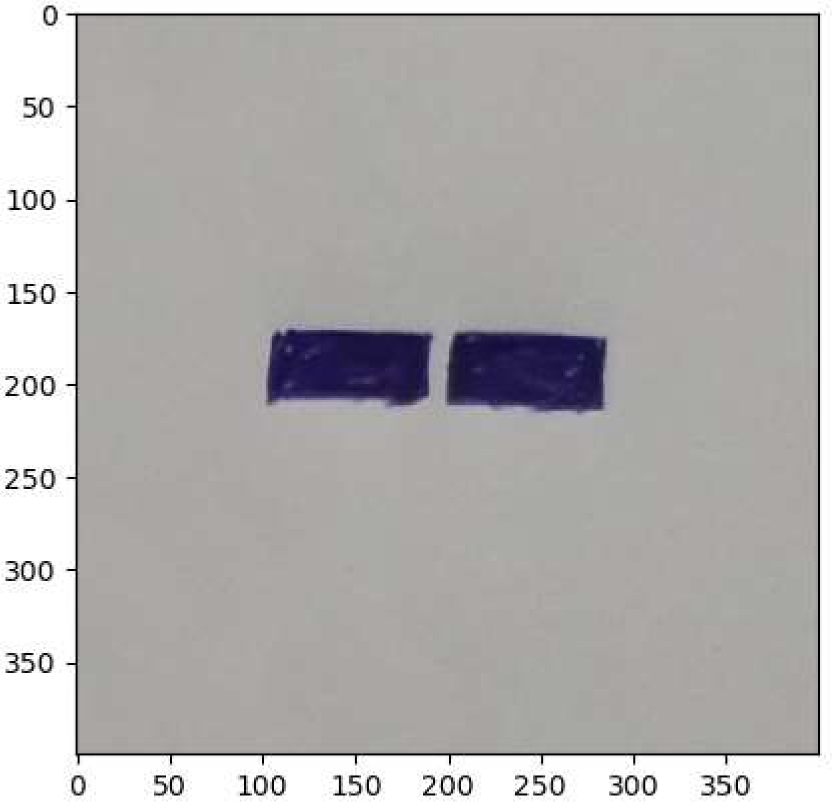
```
Prediction =  D
```



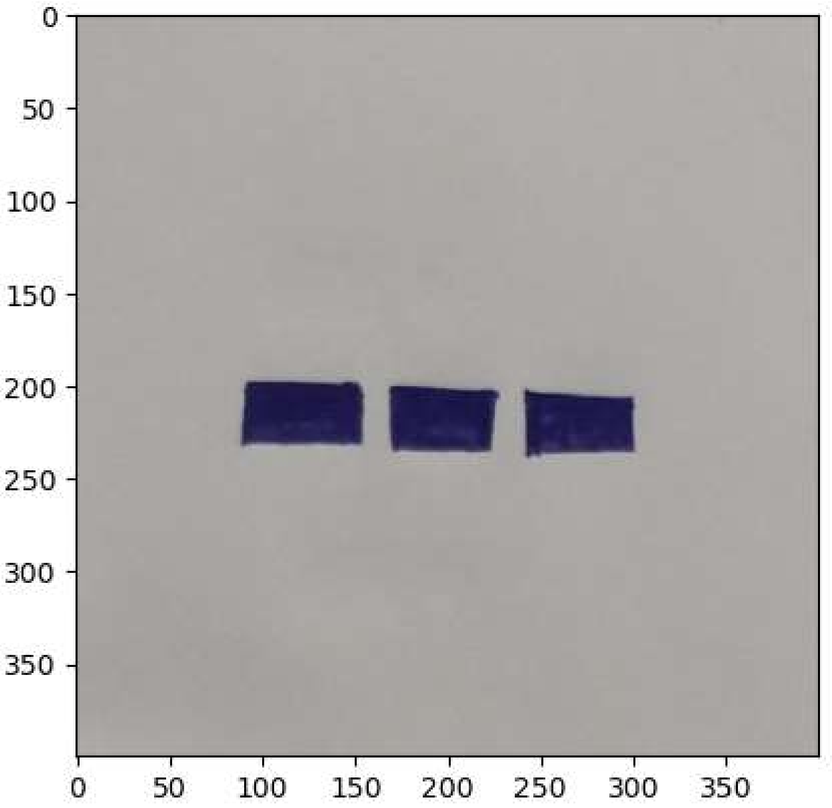
```
1/1 [==============================] - 0s 56ms/step
Prediction =  I
```



```
1/1 [==============================] - 0s 80ms/step
Prediction =  J
```

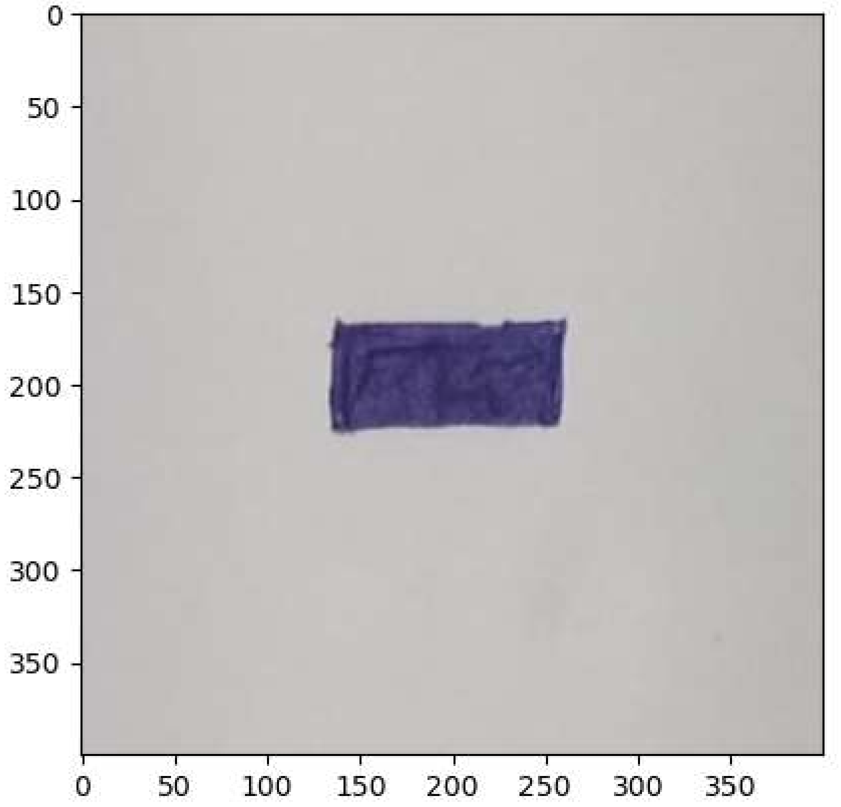


```
1/1 [==============================] - 0s 79ms/step
Prediction =  M
```

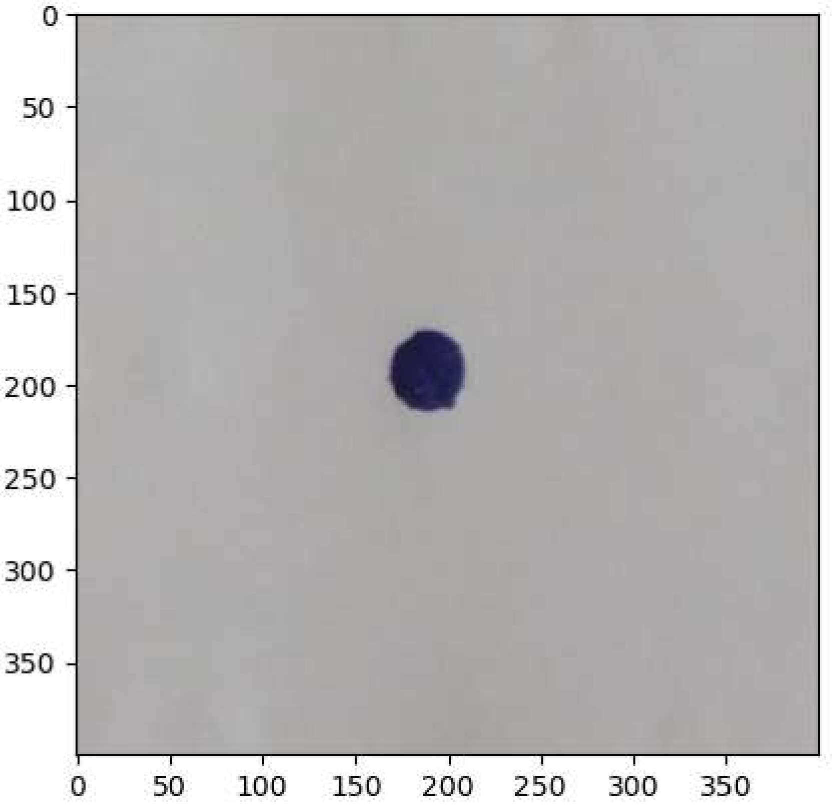


```
1/1 [==============================] - 0s 64ms/step
Prediction =  O
```

```
1/1 [==============================] - 0s 71ms/step
Prediction =  P
```

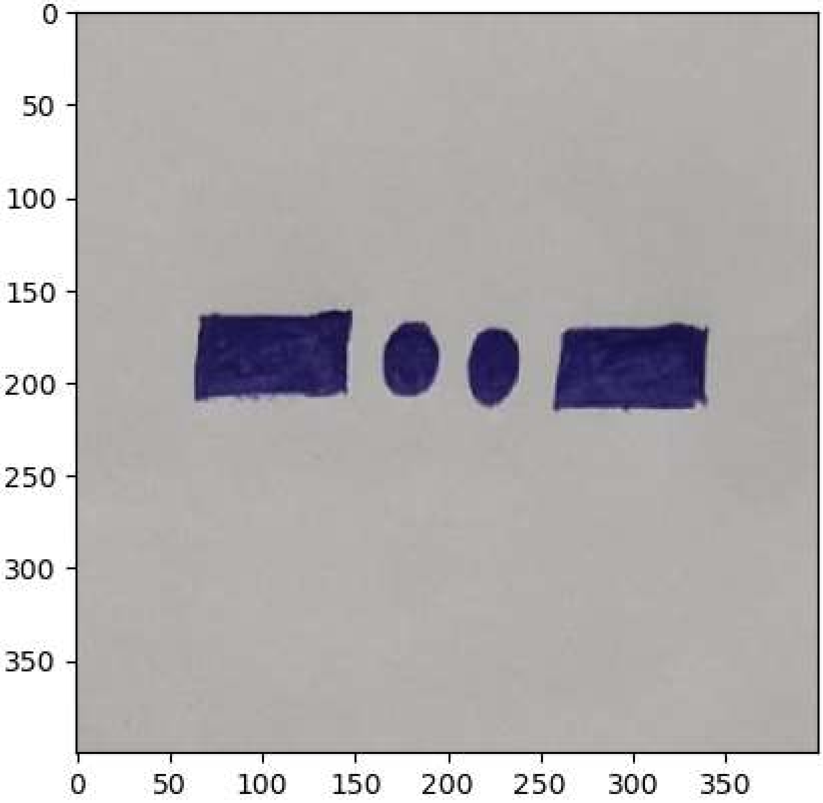


```
1/1 [==============================] - 0s 56ms/step
Prediction =  T
```

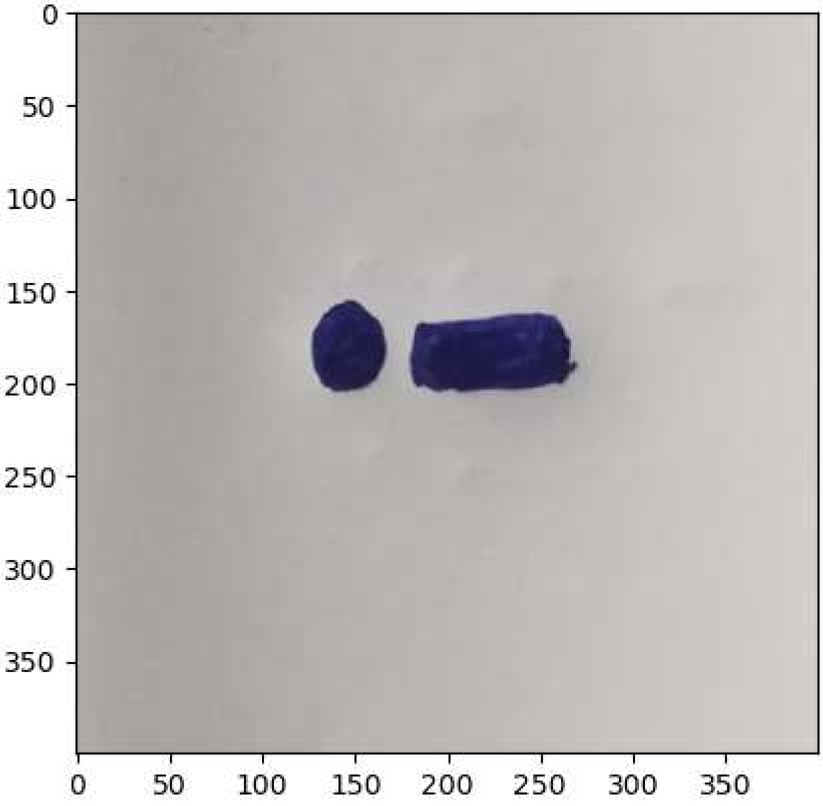


```
1/1 [==============================] - 0s 48ms/step
Prediction =  E
```

```
1/1 [==============================] - 0s 65ms/step
Prediction =  H
```



```
1/1 [==============================] - 0s 64ms/step
Prediction =  X
```



```
1/1 [==============================] - 0s 48ms/step
Prediction =  A
```

```
In [ ]:
```

```
In [ ]:
```