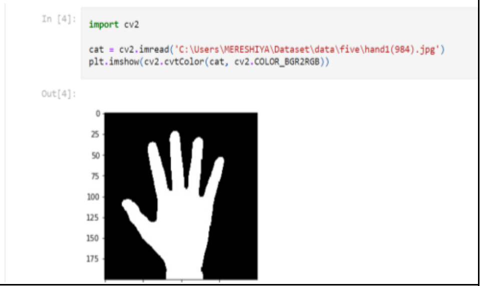
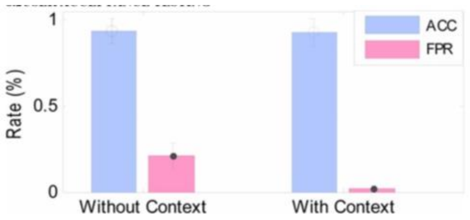
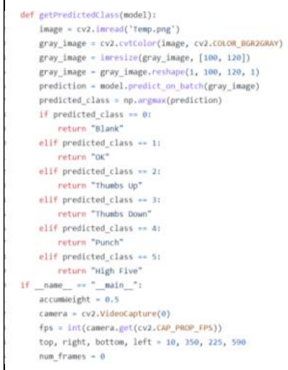
Project Development Phase
Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID28409
Project Name	A Gesture-Based Tool For Sterile Browsing Of Radiology Images
Maximum Marks	10 Marks

Model Performance Testing:

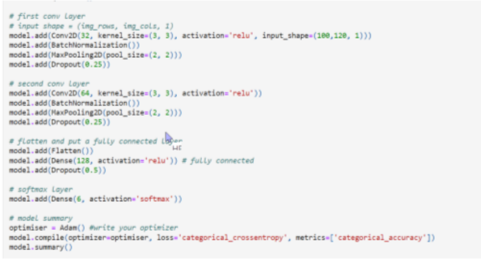
Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Detects the hand gesture shown by the surgeons during the surgery	
	Accuracy	Training Accuracy – 92% Validation Accuracy -89%	 

```
In [4]: import cv2

        cat = cv2.imread('C:\Users\MERESHIYA\Dataset\data\five\hand1(984).jpg')
        plt.imshow(cv2.cvtColor(cat, cv2.COLOR_BGR2RGB))
Out[4]:
```

```
def getPredictedClass(model):
    image = cv2.imread('Temp.png')
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    gray_image = imresize(gray_image, [100, 120])
    gray_image = gray_image.reshape(1, 100, 120, 1)
    prediction = model.predict_on_batch(gray_image)
    predicted_class = np.argmax(prediction)
    if predicted_class == 0:
        return "Blank"
    elif predicted_class == 1:
        return "OK"
    elif predicted_class == 2:
        return "Thumbs Up"
    elif predicted_class == 3:
        return "Thumbs Down"
    elif predicted_class == 4:
        return "Punch"
    elif predicted_class == 5:
        return "High Five"
if __name__ == "__main__":
    accumWeight = 0.5
    camera = cv2.VideoCapture(0)
    fps = int(camera.get(cv2.CAP_PROP_FPS))
    top, right, bottom, left = 10, 350, 225, 590
    num_frames = 0
```

			
			# first conv Layer
			# input shape = (img_rows, img_cols, 1)
			model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(100,120, 1)))
			model.add(BatchNormalization())
			model.add(MaxPooling2D(pool_size=(2, 2)))
			model.add(Dropout(0.25))
			
			# second conv Layer
			model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
			model.add(BatchNormalization())
			model.add(MaxPooling2D(pool_size=(2, 2)))
			model.add(Dropout(0.25))
			
			# flatten and put a fully connected Layer
			model.add(Flatten())
			model.add(Dense(128, activation='relu')) # fully connected
			model.add(Dropout(0.5))
			
			# softmax Layer
			model.add(Dense(6, activation='softmax'))
			
			# model summary
			optimiser = Adam() #write your optimizer
			model.compile(optimizer=optimiser, loss='categorical_crossentropy', metrics=['categorical_accuracy'])
			model.summary()

3.	Confidence Score	Class Detected - Confidence Score -	NA