

# CS-898 BA- Image Analysis and Comp Vision

## Assignment-1

*Chakradhar Reddy Donuri*

*E949F496*

### **Task-1**

Apply affine transformations to the set of 10 images collected from the web to augment the dataset. (50 points)

Below attached are the screenshots of Affine Transformations

- Identity
- Scaling
- Rotation
- Transform
- Shear Vertical
- Shear Horizontal

All the transformations are done for 10 different images.

# Image-1

JupyterLab interface showing two code cells and their corresponding image outputs.

**Cell [51]:**

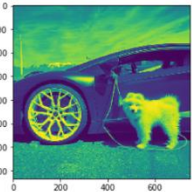
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

**Cell [52]:**

```
#Affine transformations
img = cv2.imread('Image_01.jpg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
plt.imshow(dst,cmap='gray')
```

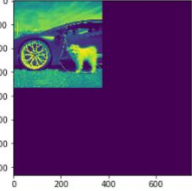
**Output [52]:** `<matplotlib.image.AxesImage at 0x11820179388>`



**Cell [53]:**

```
# Scaling
M = np.float32([[1.5,0,0],[0,1.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```


**Output [53]:** `<matplotlib.image.AxesImage at 0x118201d5f88>`



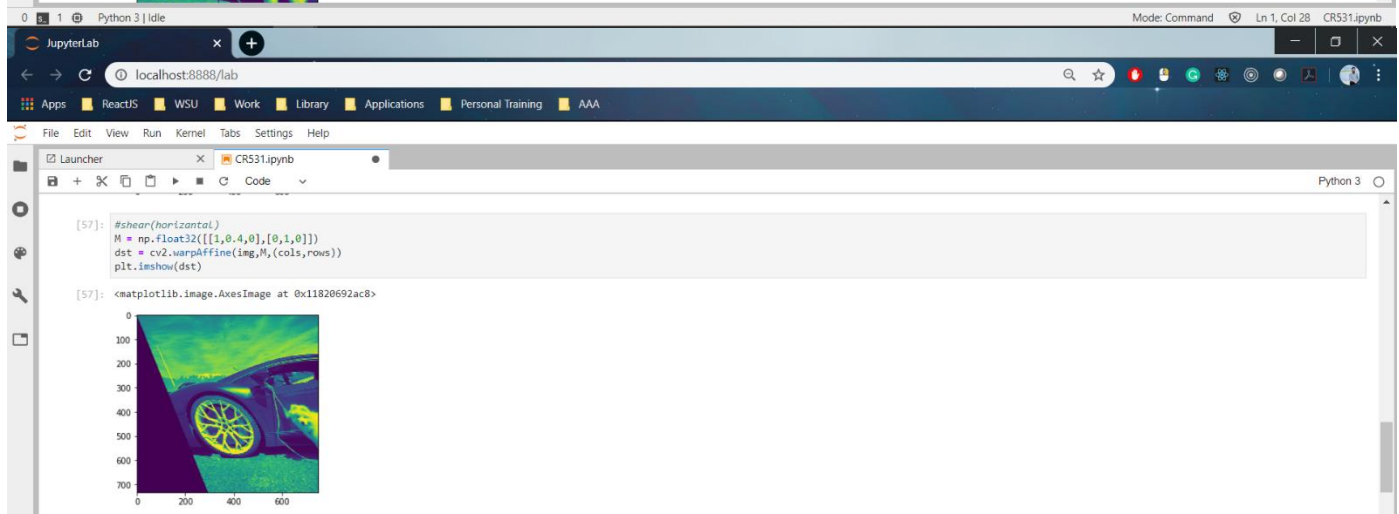
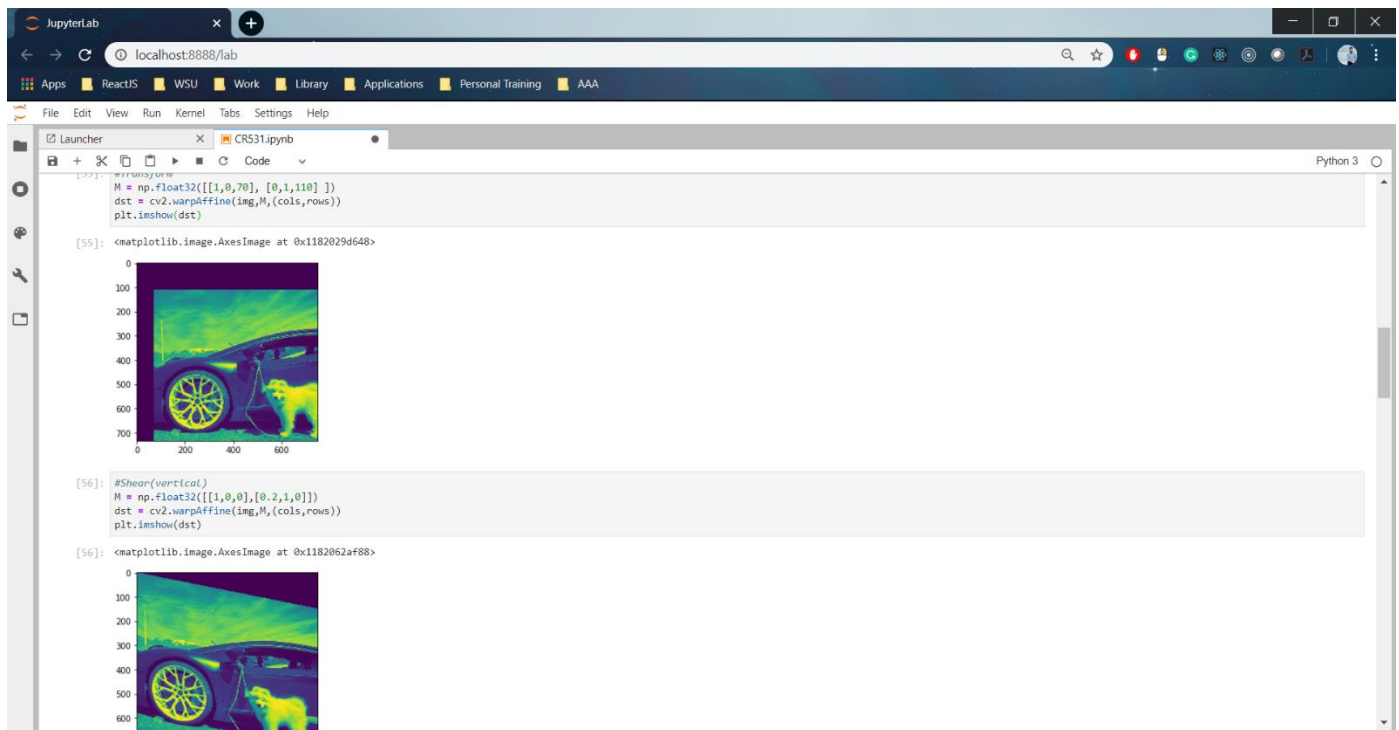
**Cell [54]:**

```
#Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```

**Output [54]:** `<matplotlib.image.AxesImage at 0x11820239c48>`



0 1 Python 3 | Idle Mode: Command Ln 1, Col 28 CRS31.ipynb



## Image – 2

JupyterLab interface showing a sequence of image transformations applied to a portrait of a man in a suit.


The interface displays the following code and results:

```
[38]: import cv2
import matplotlib.pyplot as plt
import numpy as np

[39]: #Affine transformations
img = cv2.imread('Image_02.jpg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
#plt.imshow(dst, cmap='gray')

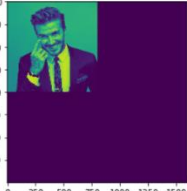
[39]: <matplotlib.image.AxesImage at 0x1181ff29e48>
```



The image is displayed with axes ranging from 0 to 1500 on the x-axis and 0 to 1400 on the y-axis.

```
[40]: # Scaling
M = np.float32([[1.5,0,0],[0,.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)


[40]: <matplotlib.image.AxesImage at 0x11820357388>
```



The image is displayed with axes ranging from 0 to 1500 on the x-axis and 0 to 1400 on the y-axis.

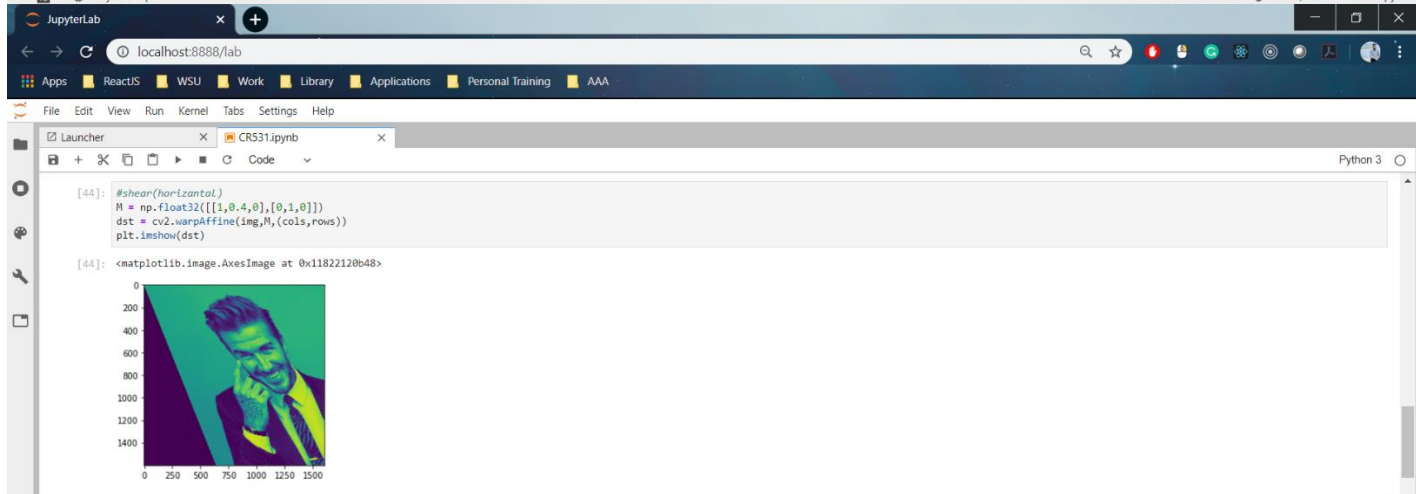
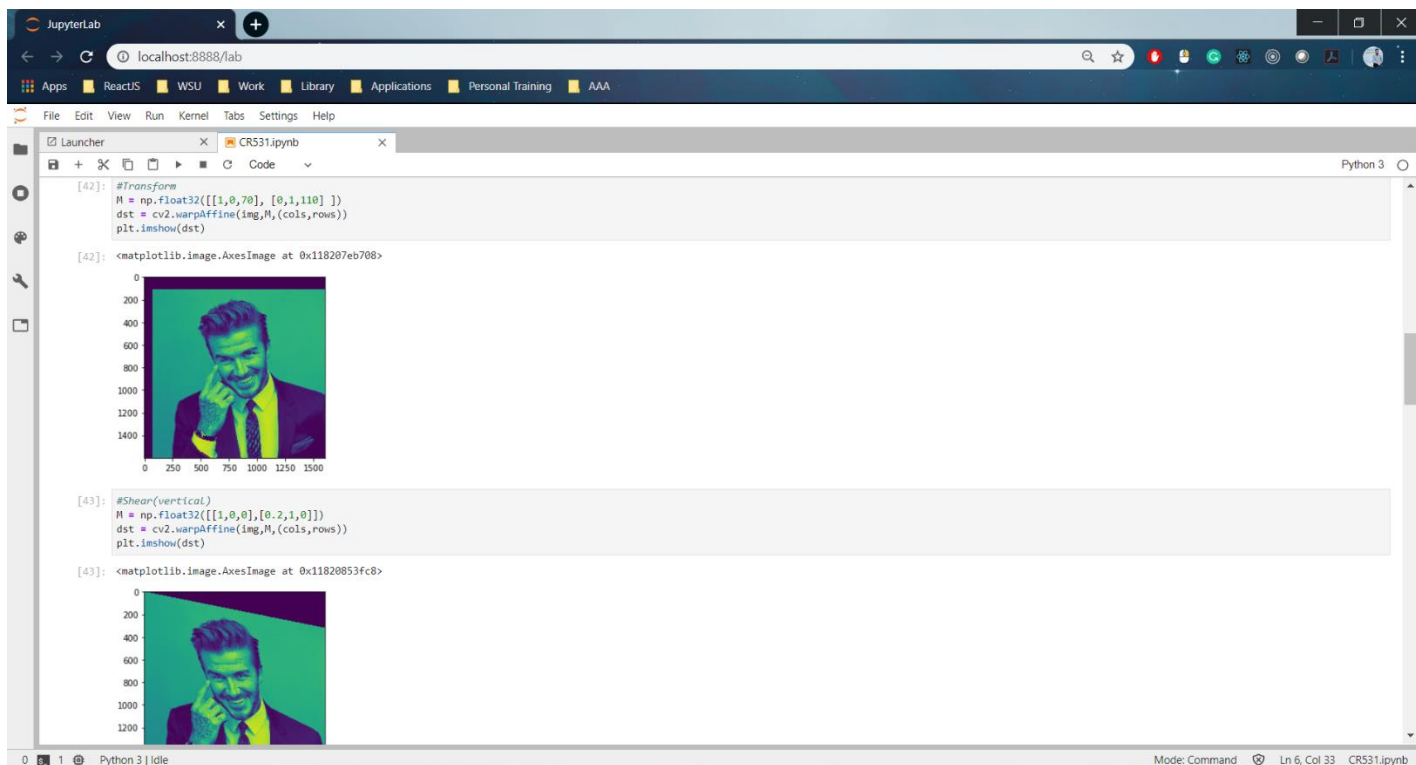
```
[41]: #Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)

[41]: <matplotlib.image.AxesImage at 0x1182077ed88>
```



The image is displayed with axes ranging from 0 to 1500 on the x-axis and 0 to 1400 on the y-axis.

At the bottom of the JupyterLab interface, the status bar indicates: Python 3 | Idle, Mode: Command, Ln 6, Col 33, CR531.ipynb.



## Image – 3

JupyterLab interface showing image processing results.

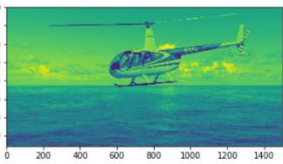
**Top Screenshot:**

```
[58]: import cv2
import matplotlib.pyplot as plt
import numpy as np

[59]: #Affine transformations
img = cv2.imread('Image_03.jpg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
#plt.imshow(dst,cmap='gray')
```

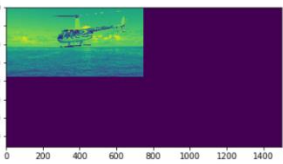
[59]: <matplotlib.image.AxesImage at 0x118206f6b08>



**Bottom Screenshot:**

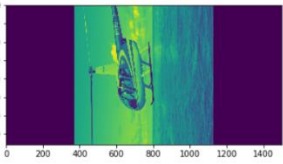
```
[60]: # Scaling
M = np.float32([[.5,0,0],[0,.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```

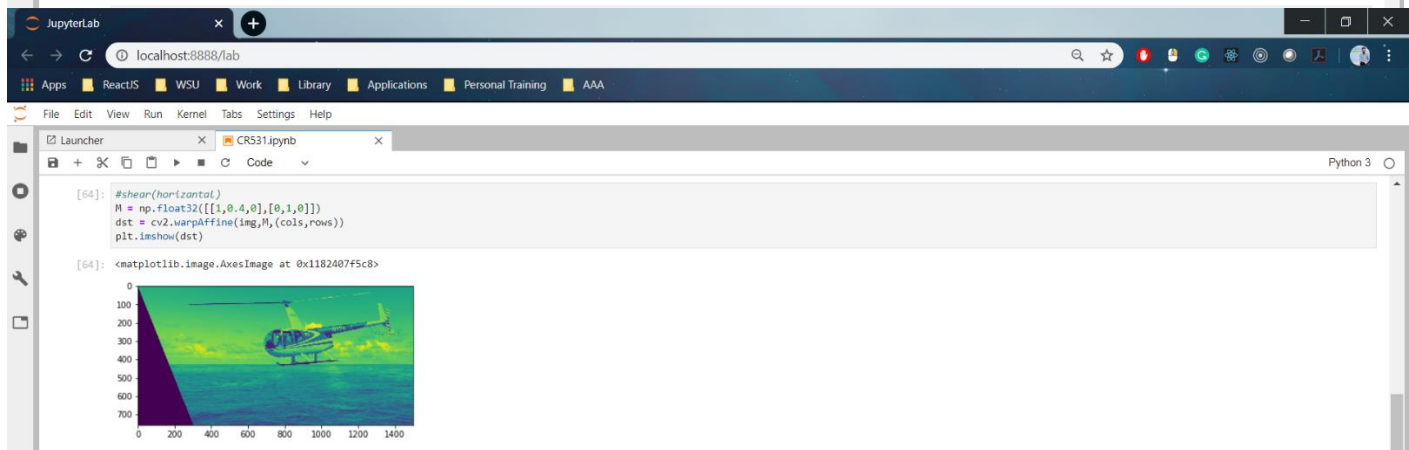
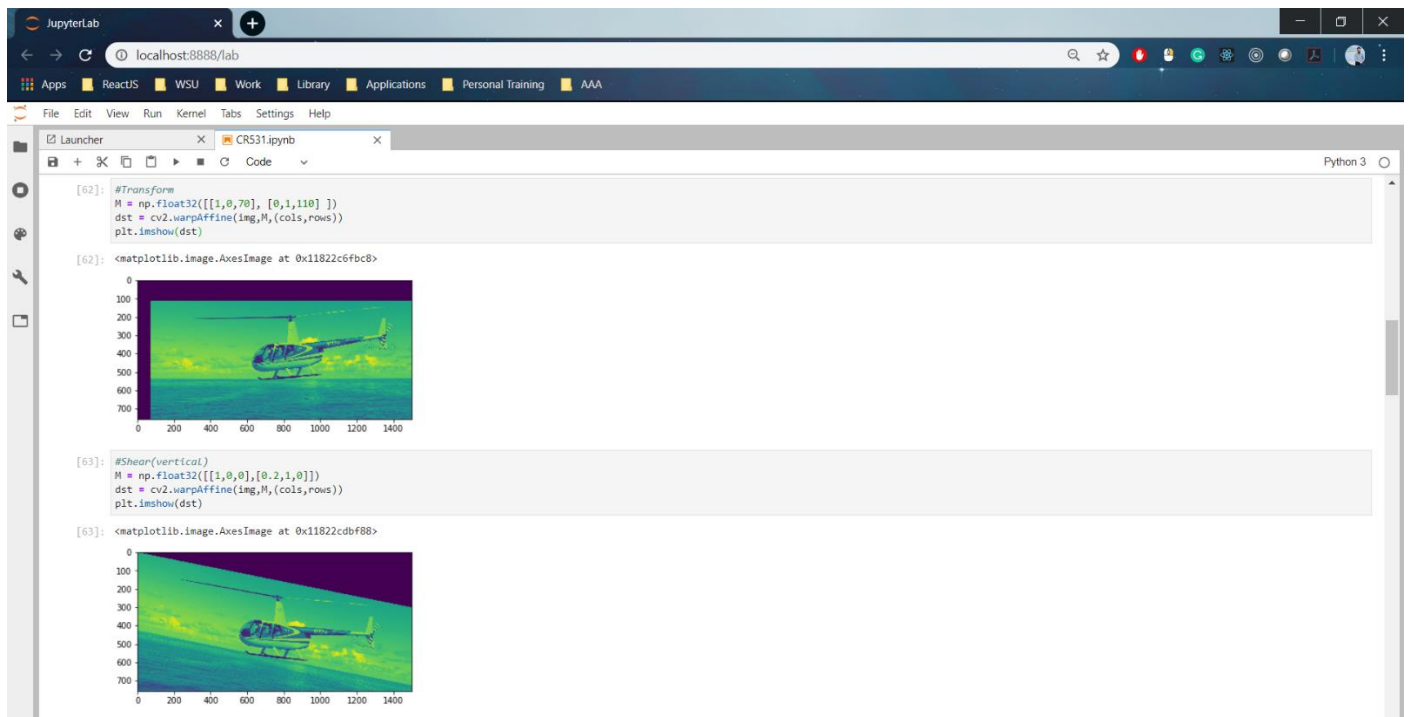
[60]: <matplotlib.image.AxesImage at 0x11822562fc8>



```
[61]: #Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```

[61]: <matplotlib.image.AxesImage at 0x118225d4848>





## Image – 4

JupyterLab interface showing two code cells and their outputs.

**Cell [67]:**

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

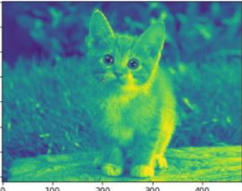
**Cell [68]:**

```
#Affine transformations
img = cv2.imread('Image_04.jpeg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
#plt.imshow(dst, cmap='gray')
```

**Output [68]:**

<matplotlib.image.AxesImage at 0x118222d9688>

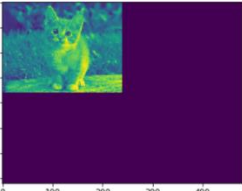


**Cell [69]:**

```
# Scaling
M = np.float32([[1.5,0,0],[0,.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```

**Output [69]:**

<matplotlib.image.AxesImage at 0x11822338d08>




**Cell [70]:**

```
#Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```

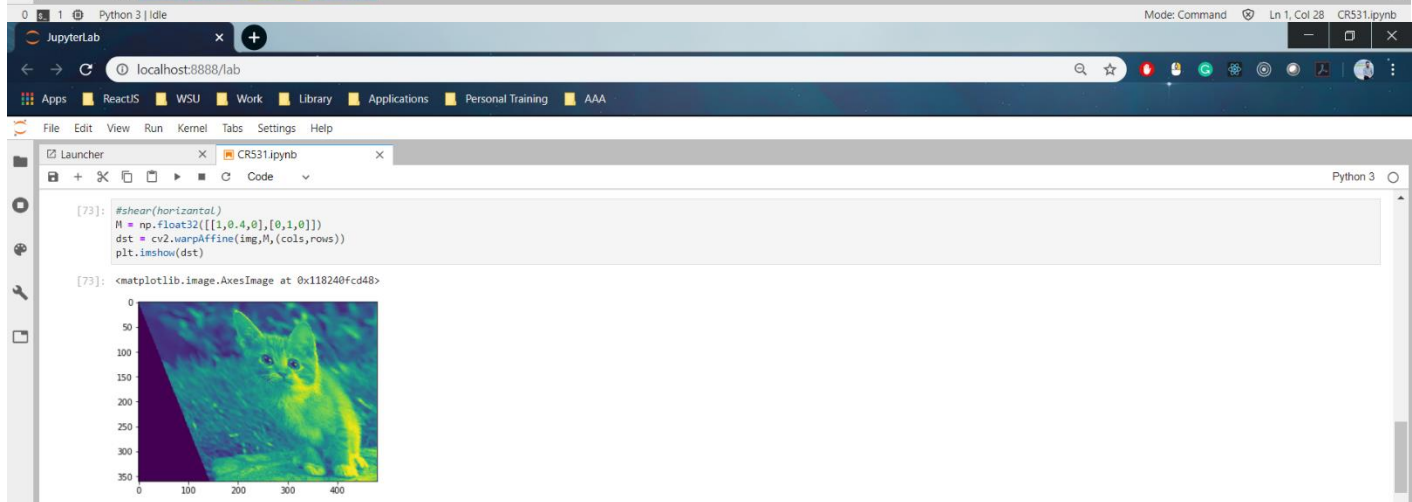
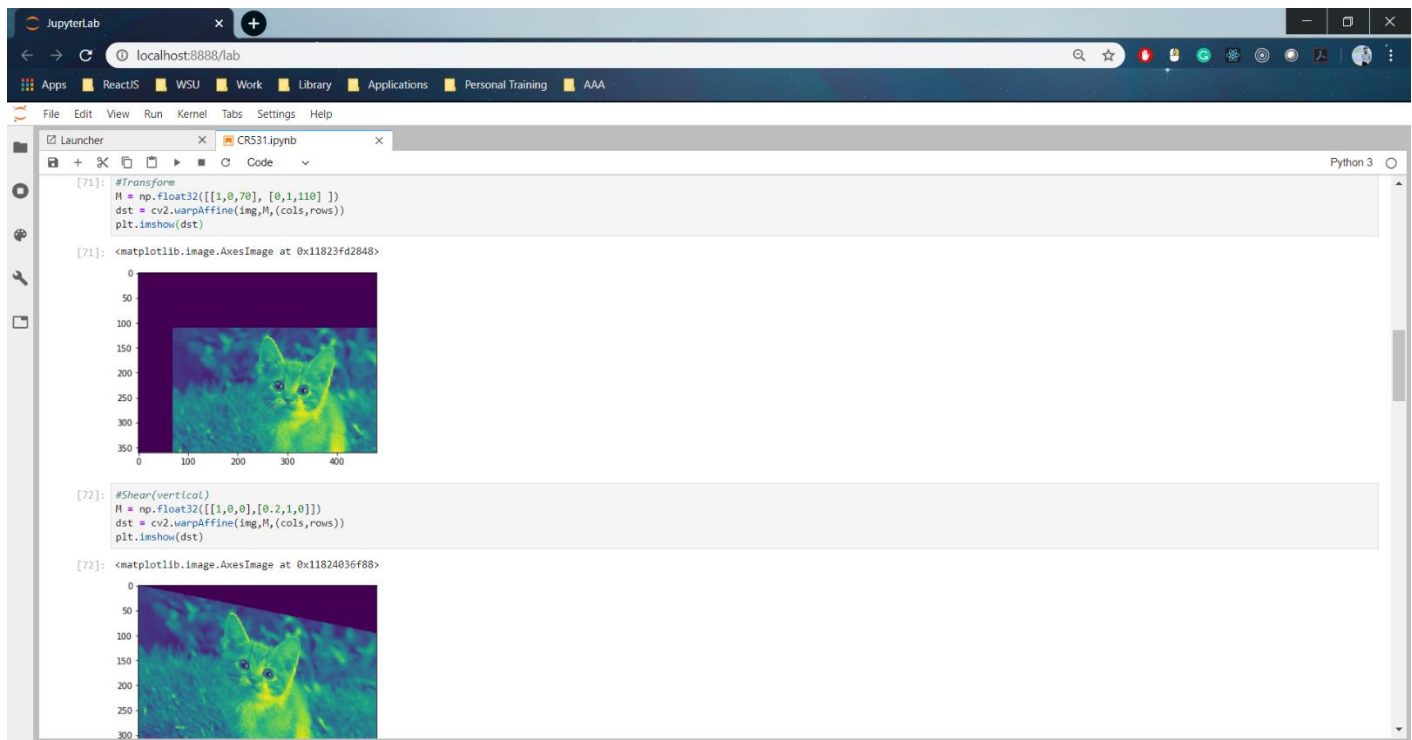
**Output [70]:**

<matplotlib.image.AxesImage at 0x118223a1688>

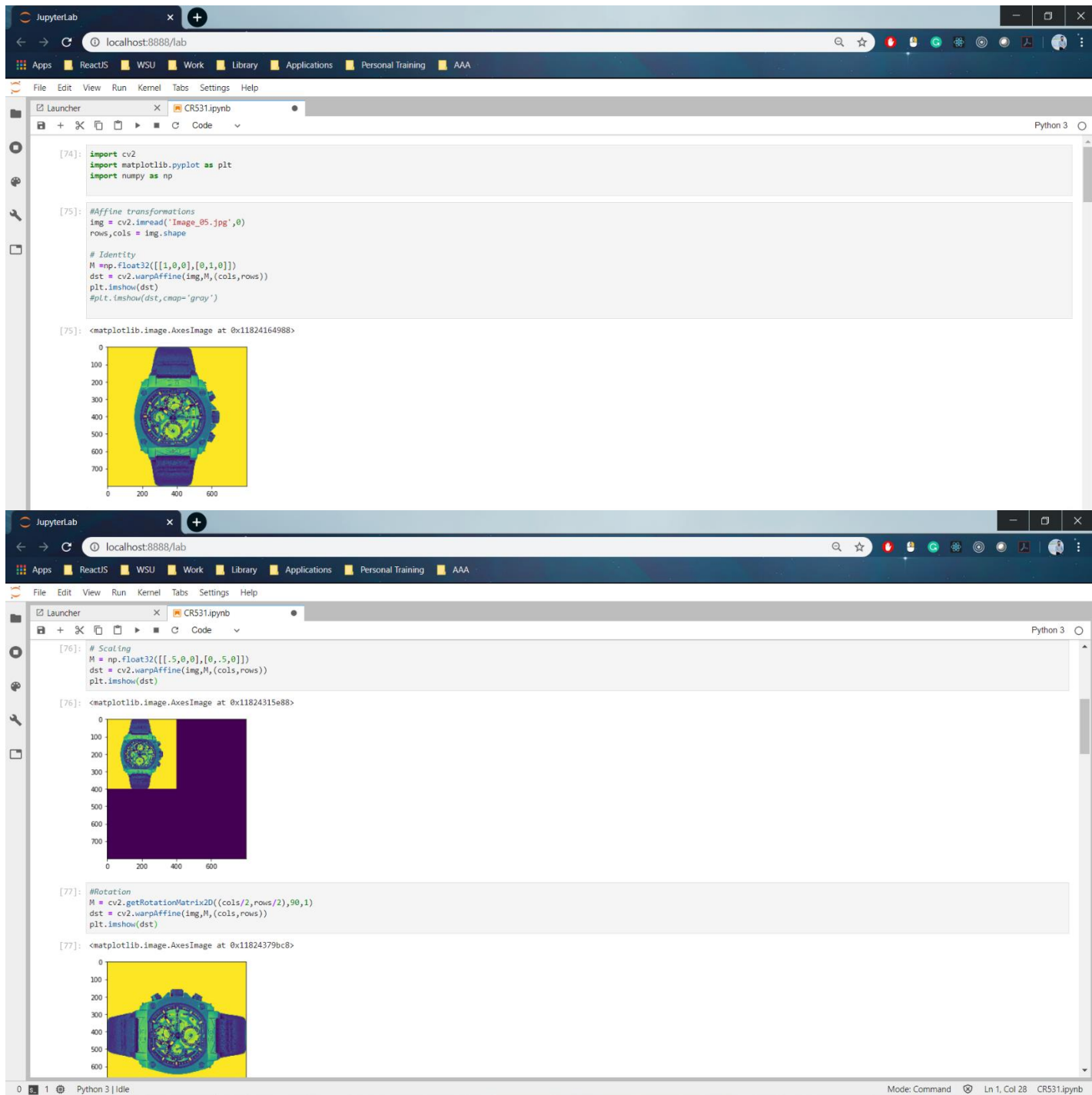


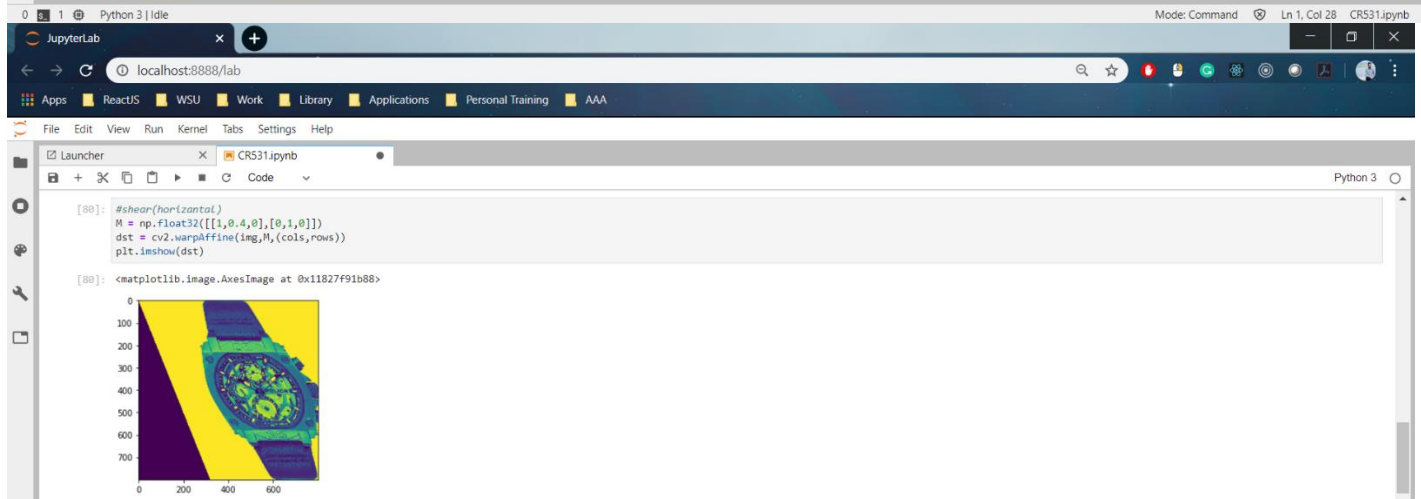
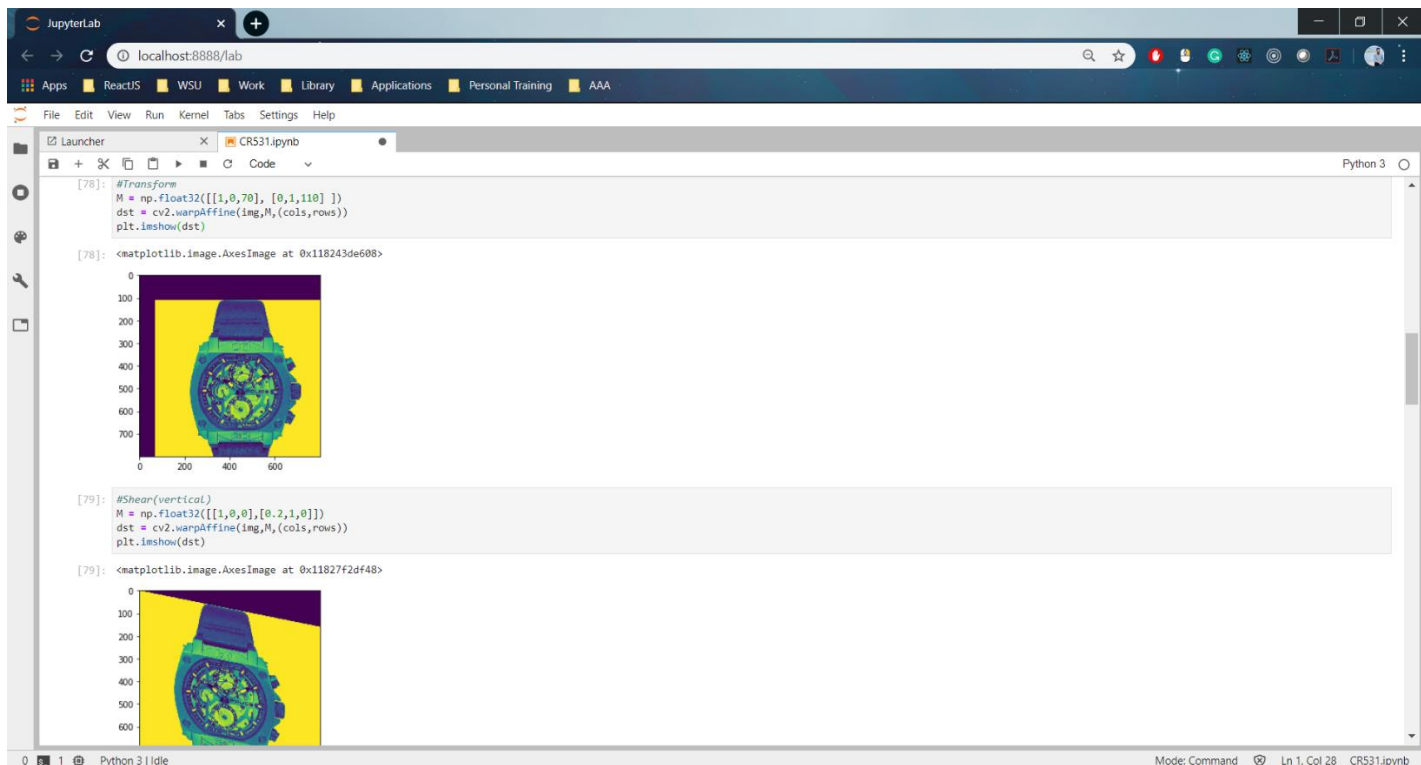
Python 3 | Idle Mode: Command Ln 1, Col 28 CR531.ipynb



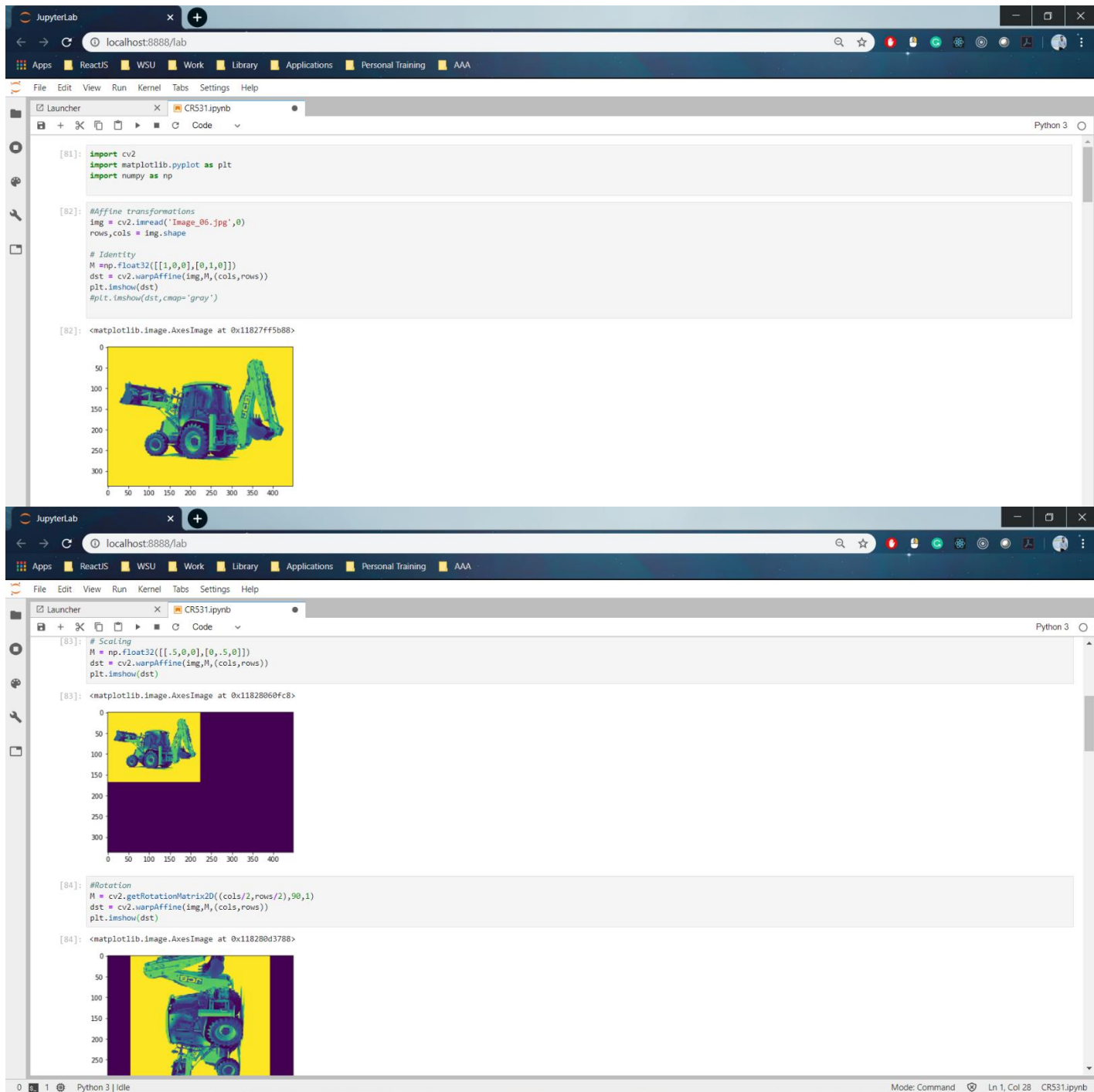


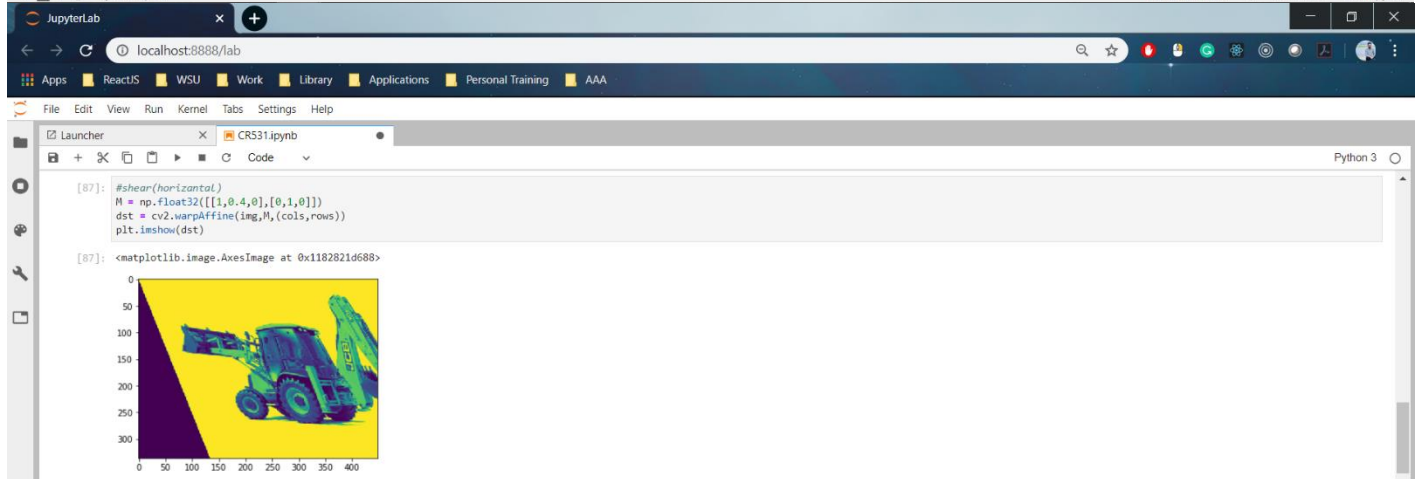
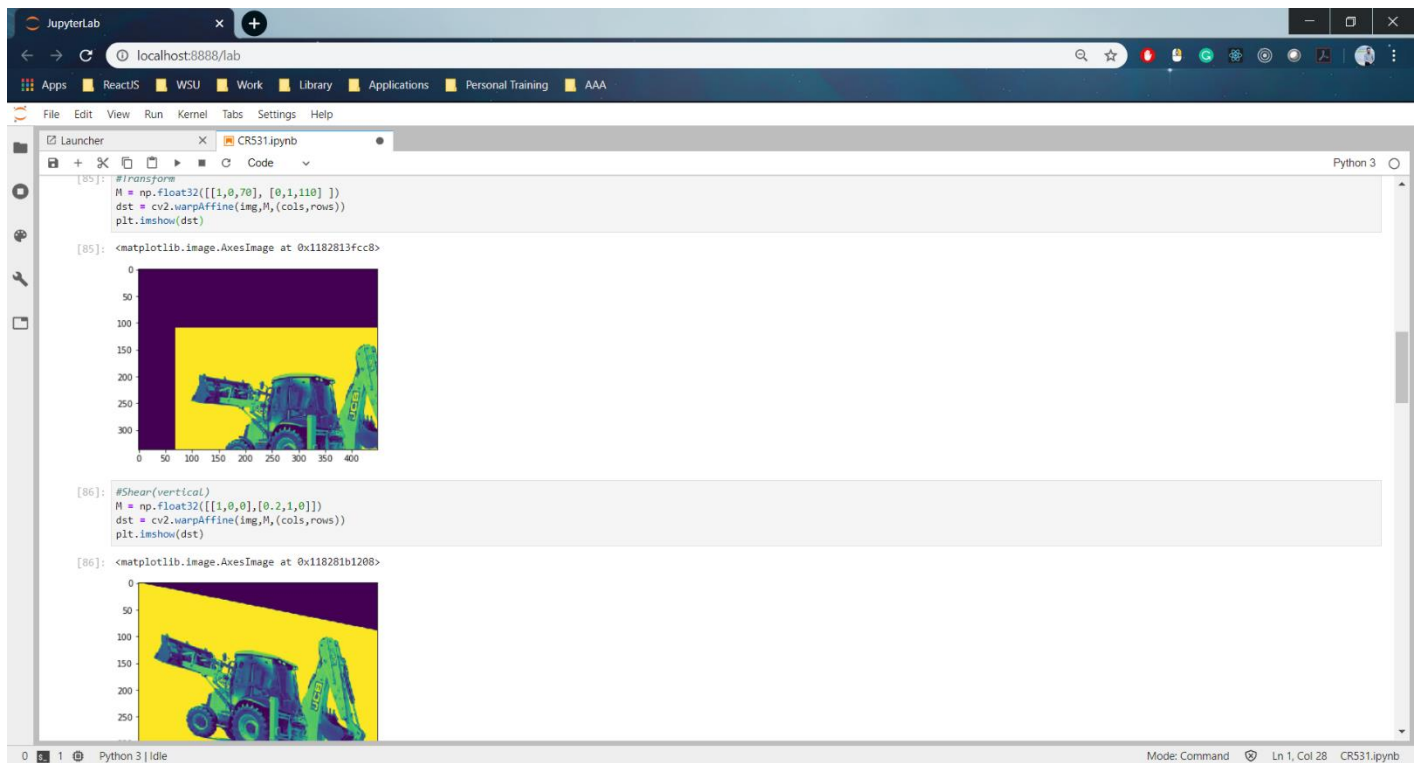
## Image – 5





## Image – 6





## Image – 7

JupyterLab interface showing image processing results.

**Top Screenshot:**

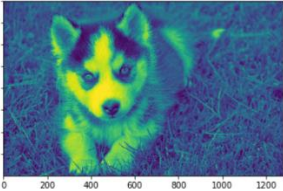
- Code cell [88]:

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```
- Code cell [89]:

```
#Affine transformations
img = cv2.imread('Image_07.jpg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
plt.imshow(dst,cmap='gray')
```
- Output [89]:

```
<matplotlib.image.AxesImage at 0x118282bf48>
```

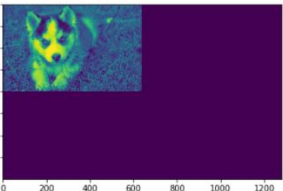


**Bottom Screenshot:**

- Code cell [90]:


```
M = np.float32([[1.5,0,0],[0,-.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [90]:

```
<matplotlib.image.AxesImage at 0x118282fc6c8>
```

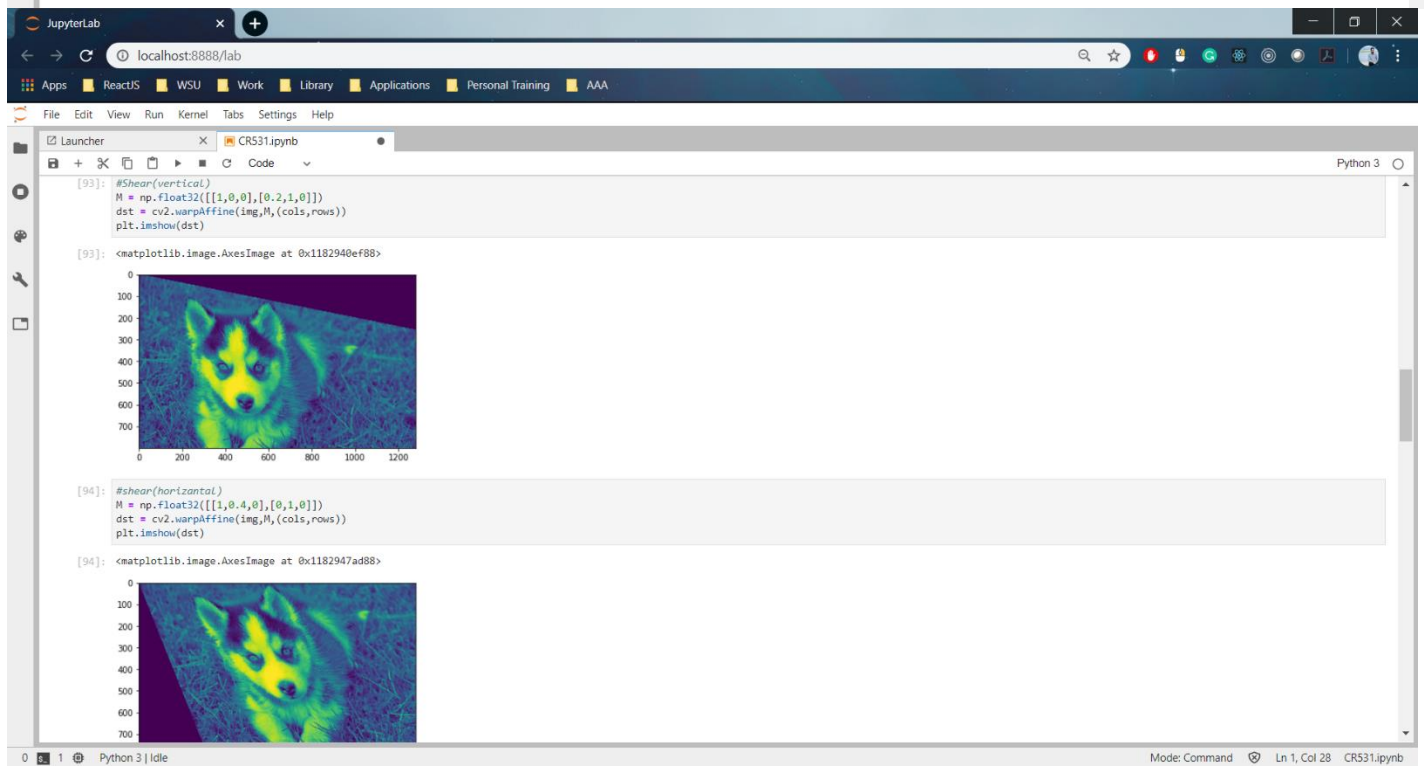
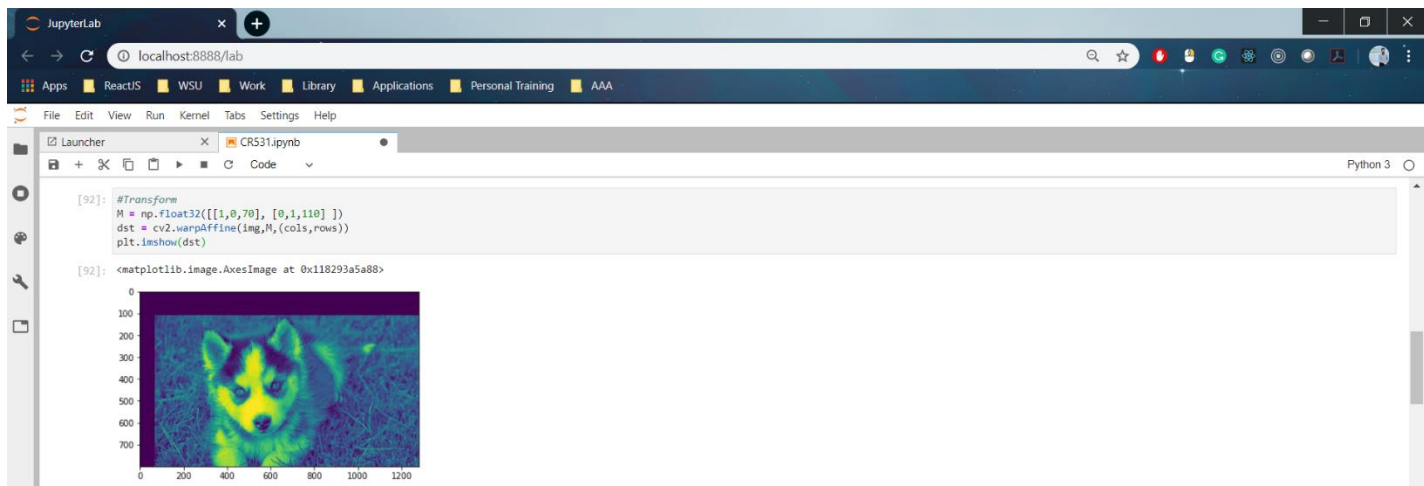

- Code cell [91]:

```
#Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [91]:

```
<matplotlib.image.AxesImage at 0x1182935fc8>
```



Mode: Command | Ln 1, Col 28 | CR531.ipynb





## Image – 8

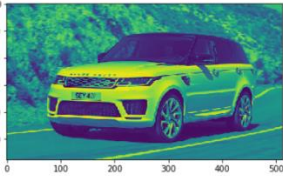
JupyterLab interface showing image processing results.

**Top Screenshot:**

- Code cell [95]:

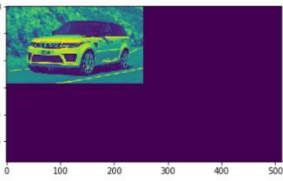
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```
- Code cell [96]:

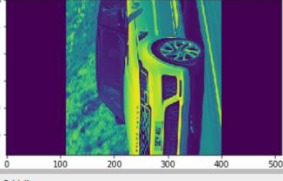
```
#Affine transformations
img = cv2.imread('Image_08.jpg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
#plt.imshow(dst,cmap='gray')
```
- Output [96]: 

**Bottom Screenshot:**

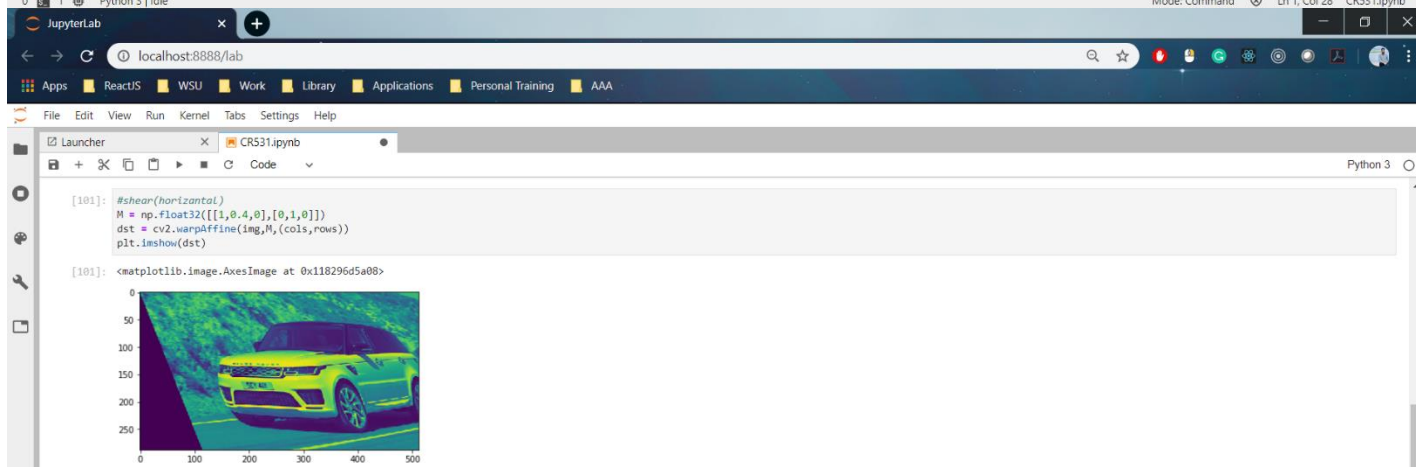
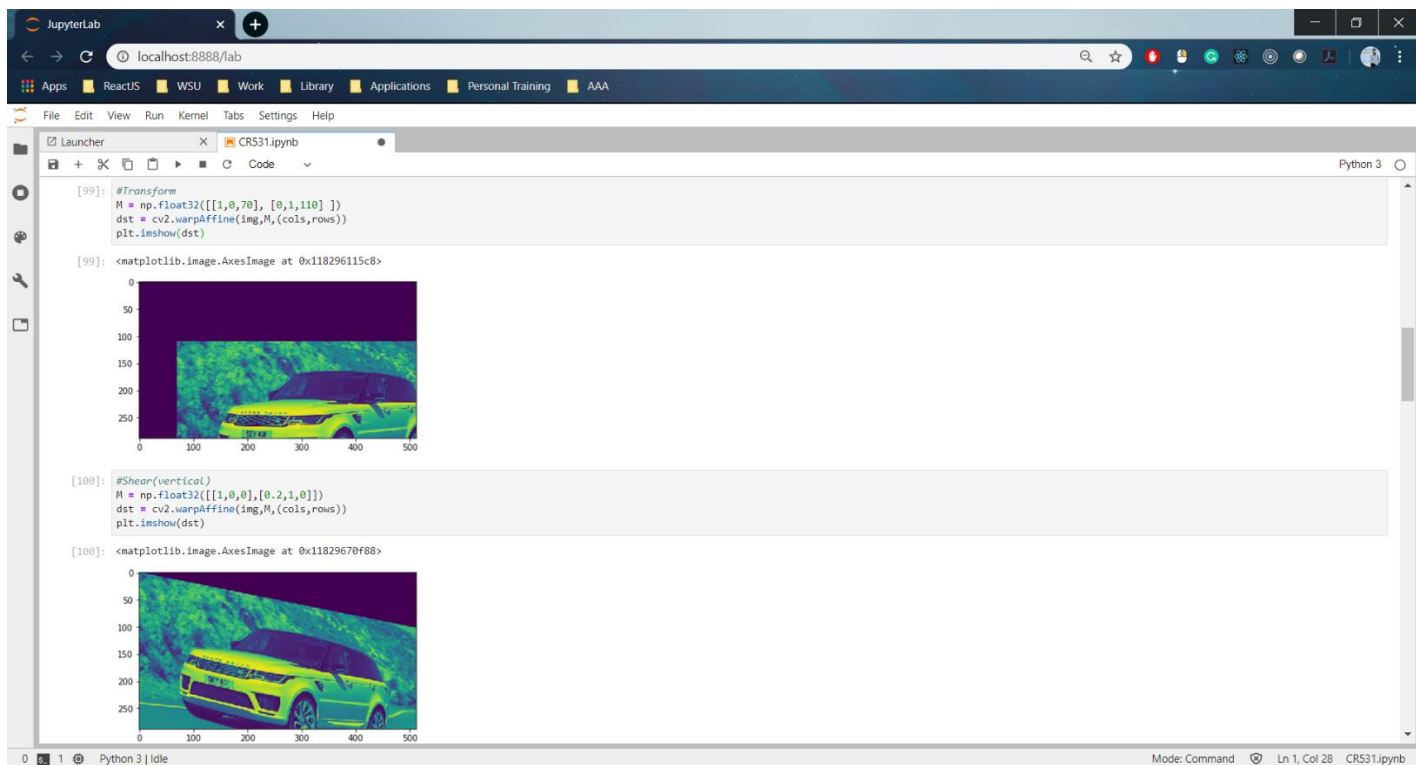
- Code cell [97]:

```
# Scaling
M = np.float32([[.5,0,0],[0,.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [97]: 
- Code cell [98]:

```
#Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [98]: 

Mode: Command | Ln 1, Col 28 | CR531.ipynb





## Image – 9

JupyterLab interface showing image processing results.

**Top Screenshot:**


- Code cell [102]:

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```
- Code cell [103]:

```
#Affine transformations
img = cv2.imread('Image_09.jpg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
#plt.imshow(dst,cmap='gray')
```
- Output [103]:

```
<matplotlib.image.AxesImage at 0x11829978988>
```

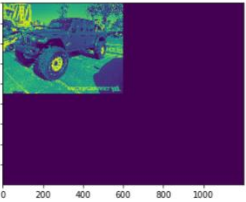


**Bottom Screenshot:**

- Code cell [104]:


```
# Scaling
M = np.float32([[1.5,0,0],[0,1.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [104]:

```
<matplotlib.image.AxesImage at 0x11829b07708>
```

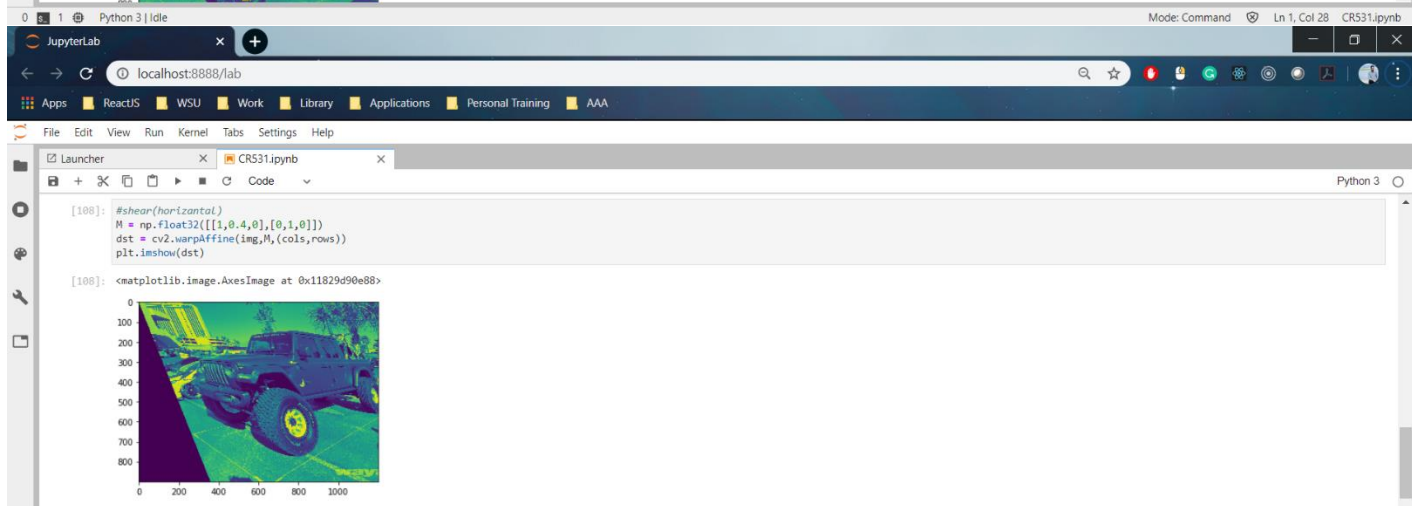
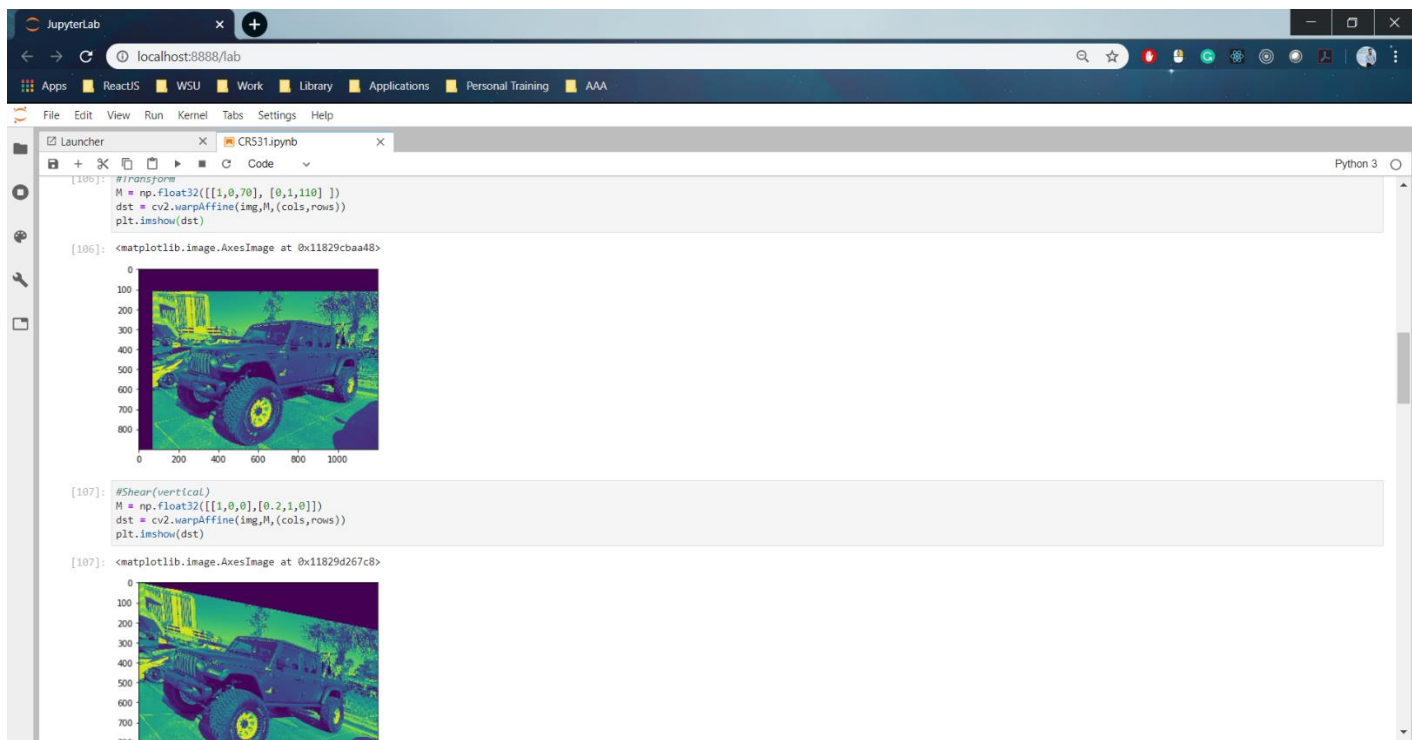

- Code cell [105]:

```
#Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [105]:

```
<matplotlib.image.AxesImage at 0x11829c0bfc8>
```



Mode: Command | Ln 1, Col 28 | CR531.ipynb



## Image – 10

JupyterLab interface showing image processing results.

**Top Screenshot:**

- Code cell [109]:

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```
- Code cell [110]:

```
#Affine transformations
img = cv2.imread('Image_10.jpg',0)
rows,cols = img.shape

# Identity
M = np.float32([[1,0,0],[0,1,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
#plt.imshow(dst,cmap='gray')
```
- Output [110]:  
- Message: `<matplotlib.image.AxesImage at 0x1182985cf08>`  
- Image: A grayscale plot of a car on a road, showing the original image after an identity transformation.

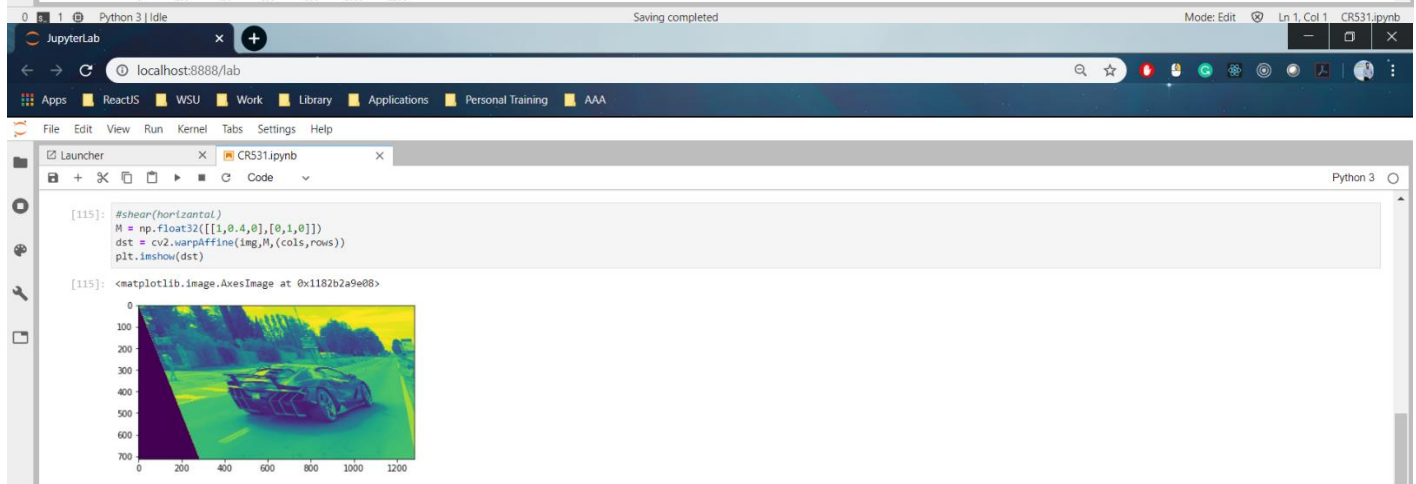
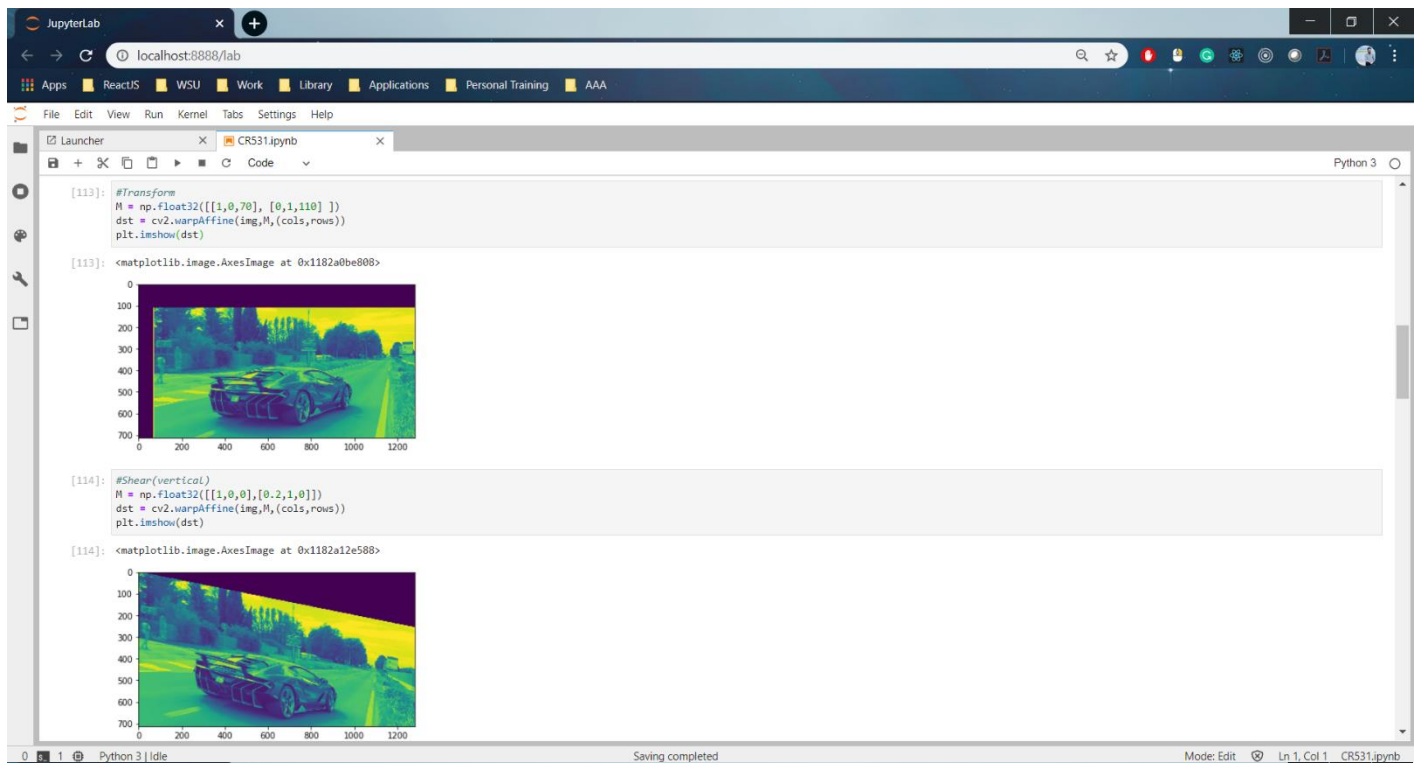
**Bottom Screenshot:**

- Code cell [111]:

```
# Scaling
M = np.float32([[.5,0,0],[0,.5,0]])
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [111]:  
- Message: `<matplotlib.image.AxesImage at 0x118298cb5c8>`  
- Image: A grayscale plot of the car image scaled down to half its original size.
- Code cell [112]:

```
#Rotation
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.imshow(dst)
```
- Output [112]:  
- Message: `<matplotlib.image.AxesImage at 0x11829931ec8>`  
- Image: A grayscale plot of the car image rotated 90 degrees clockwise.

0 1 Python 3 | Idle Mode Command Ln 1, Col 28 CR531.ipynb



## Task -2

On a randomly selected image compute (a) image statistics, (b) histogram, (c) histogram equalization and (d) binarize it.

