In [13]:

```
import cv2
from matplotlib import pyplot as plt
import numpy as np
import imutils
import easyocr
```

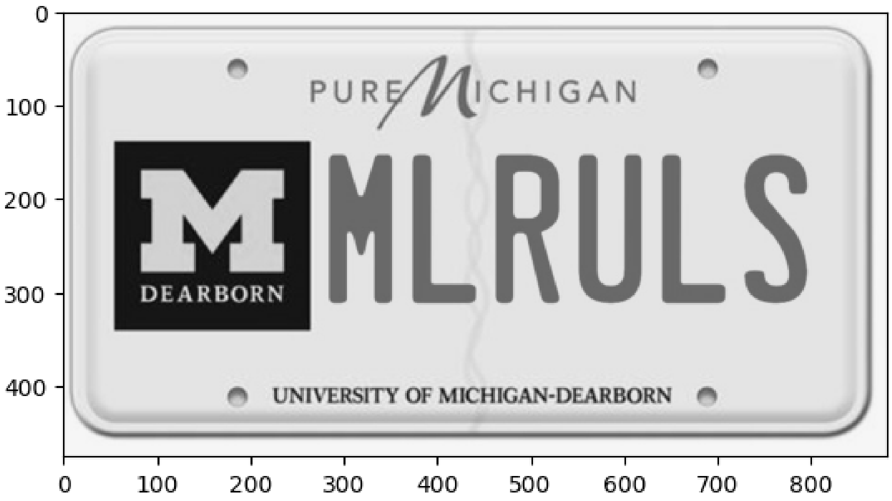
1. Read in Image, Grayscale

In [14]:

```
img = cv2.imread('../data/generated2.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
plt.imshow(cv2.cvtColor(gray,cv2.COLOR_BGR2RGB))
```

Out[14]:

```
<matplotlib.image.AxesImage at 0x28a802ea0>
```



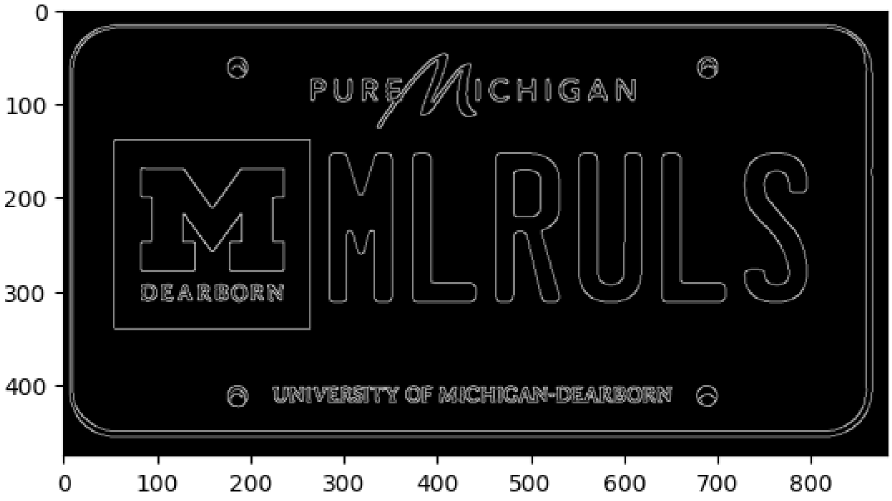
2. Apply filter and find edges for localization

In [15]:

```
bfilter = cv2.bilateralFilter(gray, 11, 17, 17) #Noise reduction
edged = cv2.Canny(bfilter, 20, 200) #Edge detection
plt.imshow(cv2.cvtColor(edged, cv2.COLOR_BGR2RGB))
```

Out[15]:

```
<matplotlib.image.AxesImage at 0x2a21393d0>
```



3. Find contours and apply masks

In [16]:

```
keypoints = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
contours = imutils.grab_contours(keypoints)
contours = sorted(contours, key=cv2.contourArea, reverse=True)[:10]
```

In [17]:

```
location = None
for contour in contours:
    approx = cv2.approxPolyDP(contour, 10, True)
    if len(approx) == 4:
        location = approx
        break
```

In [18]:

```
contour
```

Out[18]:

```
array([[[ 55, 139]],

       [[ 54, 140]],

       [[ 54, 340]],

       [[ 55, 341]],

       [[264, 341]],

       [[264, 139]]], dtype=int32)
```

In [19]:

```
location
```

Out[19]:

```
array([[[ 54, 140]],

       [[ 55, 341]],

       [[264, 341]],

       [[264, 139]]], dtype=int32)
```

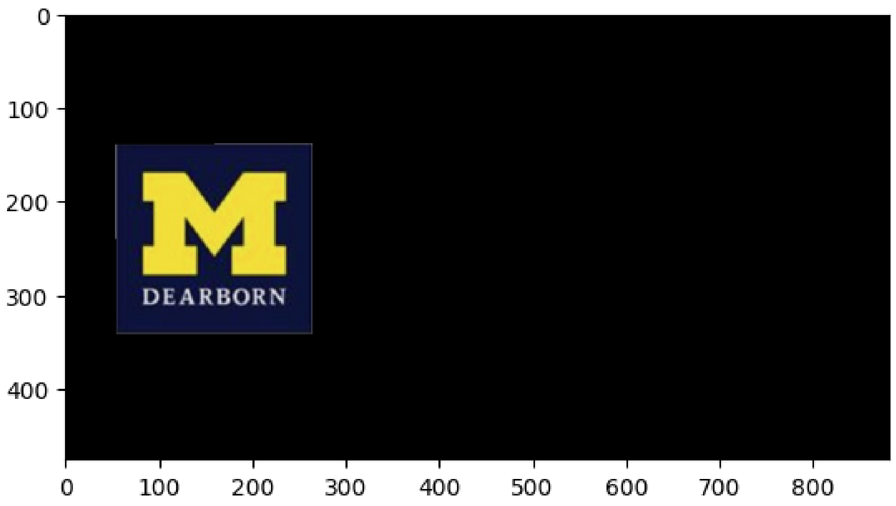
In [20]:

```
mask = np.zeros(gray.shape, np.uint8)
new_image = cv2.drawContours(mask, [location], 0, 255, -1)
new_image = cv2.bitwise_and(img, img, mask=mask)

plt.imshow(cv2.cvtColor(new_image, cv2.COLOR_BGR2RGB))
```

Out[20]:

```
<matplotlib.image.AxesImage at 0x2a21aa5a0>
```



In [21]:

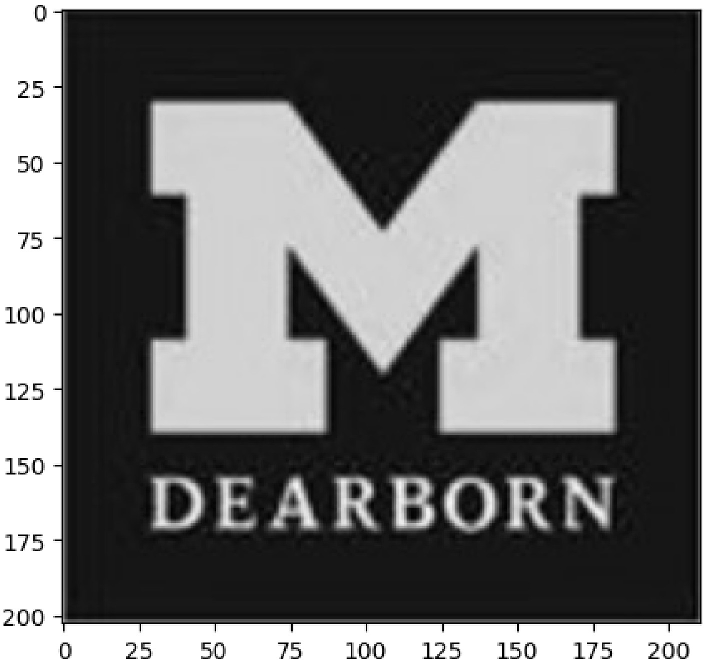
```
(x,y) = np.where(mask==255)
(x1, y1) = (np.min(x), np.min(y))
(x2, y2) = (np.max(x), np.max(y))
cropped_image = gray[x1:x2+1, y1:y2+1]
```

In [22]:

```
plt.imshow(cv2.cvtColor(cropped_image, cv2.COLOR_BGR2RGB))
```

Out[22]:

```
<matplotlib.image.AxesImage at 0x2a21faea0>
```



4. Use Easy OCR to Read Text

In [23]:

```
reader = easyocr.Reader(['en'])
result = reader.readtext(cropped_image)
result
```

Out[23]:

```
[([[24, 12], [192, 12], [192, 156], [24, 156]], 'M', 0.9961248104926632),
 ([[26, 148], [188, 148], [188, 178], [26, 178]],
  'DEARBORN',
  0.887542187273355)]
```

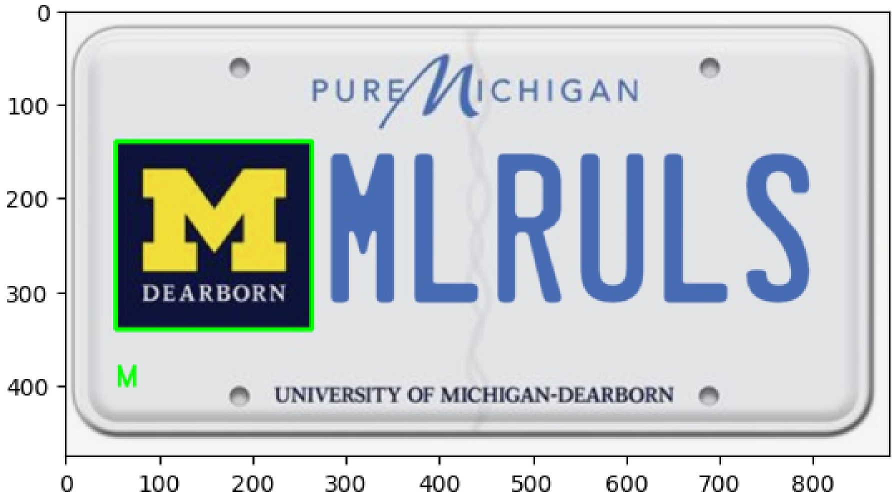
5. Render Result

In [24]:

```
text = result[0][-2]
font = cv2.FONT_HERSHEY_SIMPLEX
res = cv2.putText(img, text=text, org=(approx[0][0][0], approx[1][0][1]+60), fontFace=font, fontSc
                  thickness=2, lineType=cv2.LINE_AA)
res = cv2.rectangle(img, tuple(approx[0][0]), tuple(approx[2][0]), (0,255,0),3)
plt.imshow(cv2.cvtColor(res, cv2.COLOR_BGR2RGB))
```

Out[24]:

```
<matplotlib.image.AxesImage at 0x17fac28a0>
```


0
100
200
300
400
PURE MICHIGAN
M
DEARBORN
MLRULS
M
UNIVERSITY OF MICHIGAN-DEARBORN
0
100
200
300
400
500
600
700
800