In [16]:
```
import cv2
import os
from matplotlib import pyplot as plt
import numpy as np
import imutils
import easyocr
```

In [17]:
```
import dropbox

ACCESS_TOKEN = "sl.By84k9bomI3lUQGufdqQZgttyiPlnB5kANoNOjYi_vXTO5CXs7N_1E2o
HQvoP0wm6KXtt-KServEJk4eC7MuSQASvf4E_RvoSvdPIMhflEMZh8LkK_l6JGxnpvHrci0BwLt
xTiGE9ykZaPumMqSD"
dropbox_folder_path = "/home"
local_download_path = "C:/Users/"

def dropbox_connect():
    """Create a connection to Dropbox."""
    try:
        dbx = dropbox.Dropbox(ACCESS_TOKEN)
        print("Connected to Dropbox")
        return dbx
    except Exception as e:
        print('Error connecting to Dropbox with access token:', str(e))
        return None

def dropbox_download_file(dropbox_file_path, local_file_path):
    """Download a file from Dropbox to the local machine."""
    try:
        dbx = dropbox_connect()
        if dbx:
            with open(local_file_path, 'wb') as f:
                metadata, response = dbx.files_download(path=dropbox_file_p
ath)
                f.write(response.content)
                print("File downloaded successfully")
    except Exception as e:
        print('Error downloading file from Dropbox:', str(e))

# Example usage:
dropbox_file_path = "/home/image1.jpg" # Provide your Dropbox file path
local_file_path = "C:/Users/rohan/img2.jpg" # Provide the path where you wa
nt to save the downloaded file
dropbox_download_file(dropbox_file_path, local_file_path)
```

```
Connected to Dropbox
File downloaded successfully
```

In [18]:
```
access_token = 'sl.By84k9bomI3lUQGufdqQZgttyiPlnB5kANoNOjYi_vXTO5CXs7N_1E2o
HQvoP0wm6KXtt-KServEJk4eC7MuSQASvf4E_RvoSvdPIMhflEMZh8LkK_l6JGxnpvHrci0BwLt
xTiGE9ykZaPumMqSD'
file_to_delete = '/output/image1.png'


def delete_file(file_to_delete):
    try:
        dbx = dropbox.Dropbox(access_token)
        dbx.files_delete(file_to_delete)
        print("File deleted successfully")
    except dropbox.exceptions.ApiError as e:
        print(f"Error deleting file: {e}")

# Example usage

delete_file(file_to_delete)
```

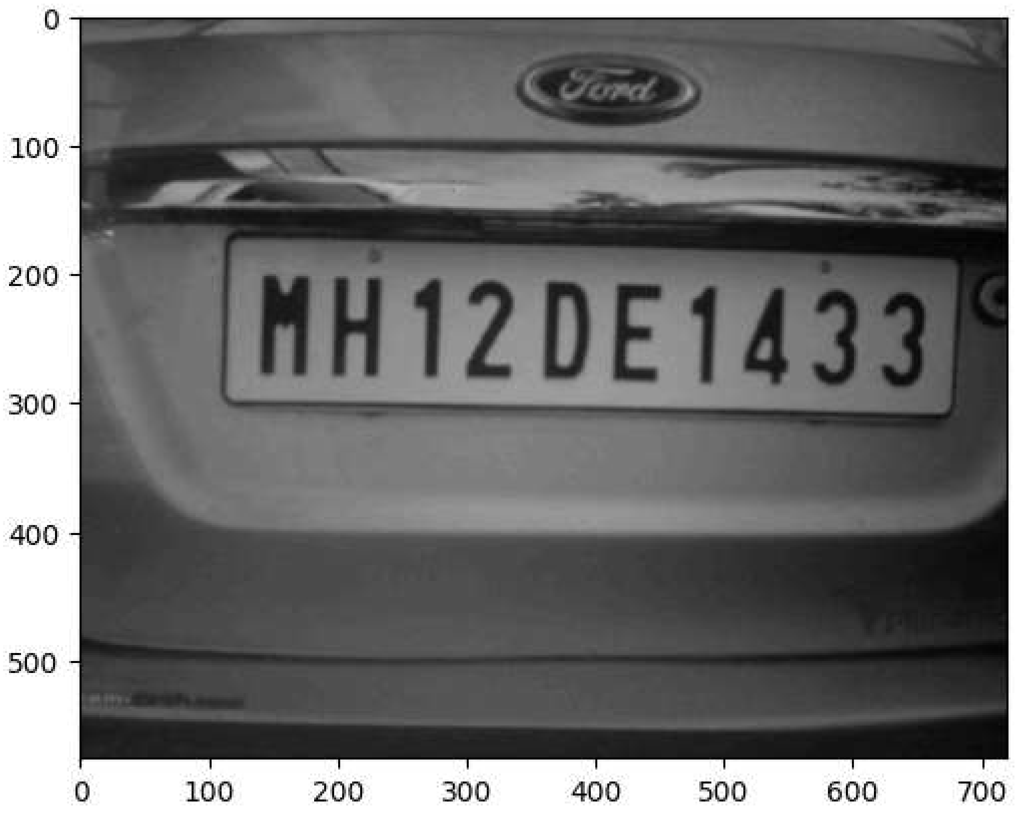
```
Error deleting file: ApiError('98f08307bf2a4fee8157f1c86e221dda', DeleteErr
or('path_lookup', LookupError('not_found', None)))
```

In [19]:
```
access_token = 'sl.By84k9bomI3lUQGufdqQZgttyiPlnB5kANoNOjYi_vXTO5CXs7N_1E2o
HQvoP0wm6KXtt-KServEJk4eC7MuSQASvf4E_RvoSvdPIMhflEMZh8LkK_l6JGxnpvHrci0BwLt
xTiGE9ykZaPumMqSD'
file_to_delete = '/home/image1.jpg'

def delete_file(file_to):
    dbx = dropbox.Dropbox(access_token)
    dbx.files_delete(file_to)
    print("deleted")

delete_file(file_to_delete)
```

```
deleted
```

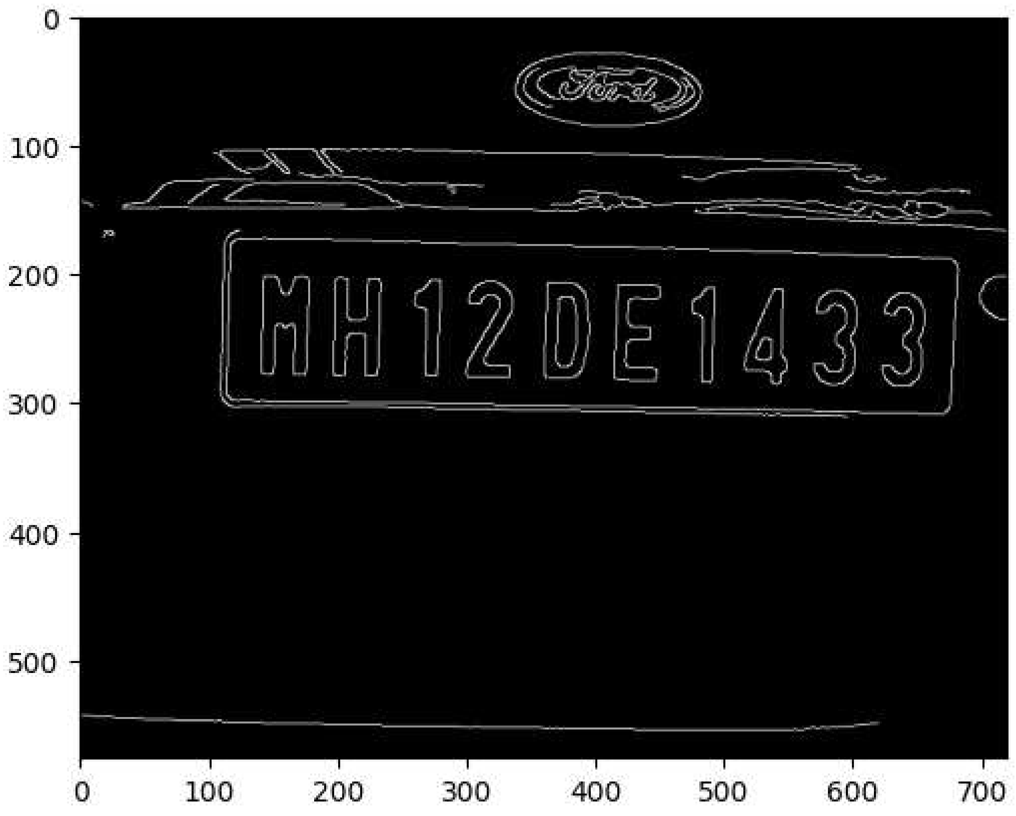
In [20]:
```
img = cv2.imread('img2.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
plt.imshow(cv2.cvtColor(gray, cv2.COLOR_BGR2RGB))
```

Out[20]:
```
<matplotlib.image.AxesImage at 0x180bbe7f810>
```



In [21]:
```
bfilter=cv2.bilateralFilter(gray, 11, 17, 17) #Noise reduction
edged= cv2.Canny(bfilter, 30, 200) #Edge detection
plt.imshow(cv2.cvtColor(edged, cv2.COLOR_BGR2RGB))
```

Out[21]:
```
<matplotlib.image.AxesImage at 0x180be4c1850>
```



In [22]:
```
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 [23]:
```
location = None
for contour in contours:
    approx = cv2.approxPolyDP(contour, 10, True)
    if len(approx) == 4:
        location = approx
        break
```

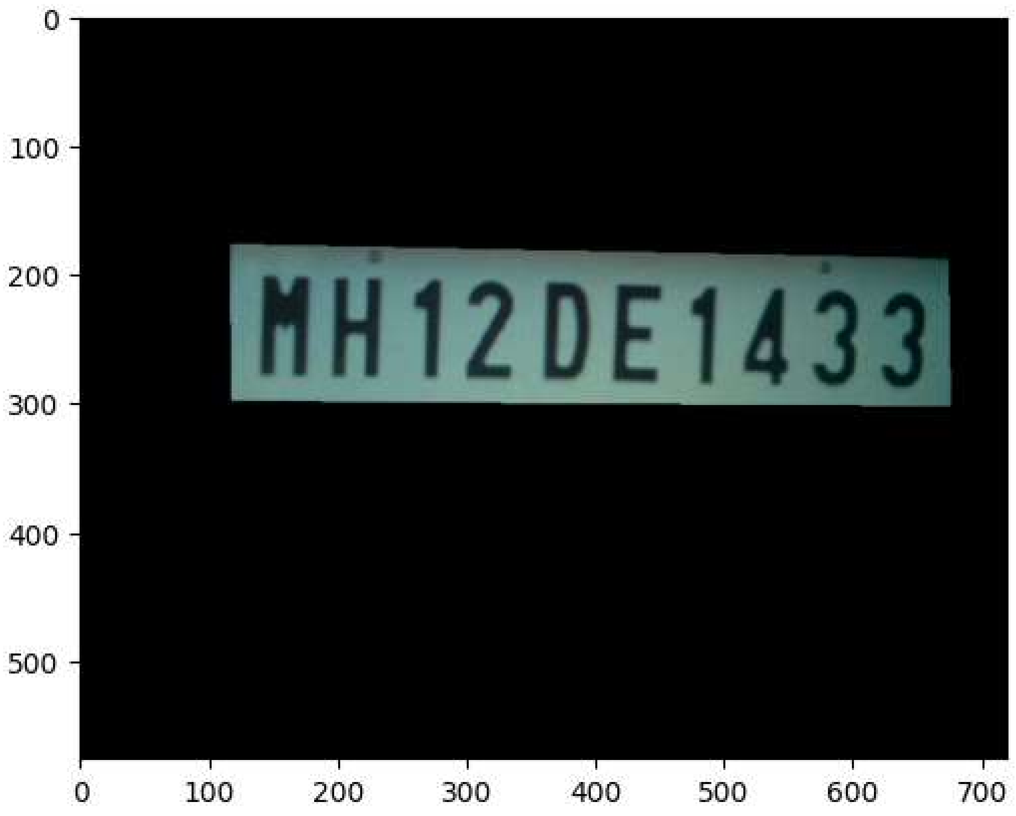
In [24]:
```
location
```

Out[24]:
```
array([[[117, 177]],

       [[118, 298]],

       [[676, 302]],

       [[674, 188]]], dtype=int32)
```

In [25]:
```
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)
```

In [26]:
```
plt.imshow(cv2.cvtColor(new_image, cv2.COLOR_BGR2RGB))
```

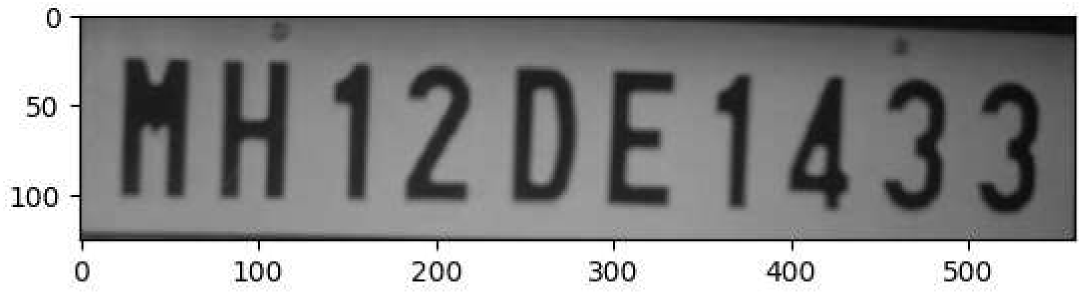
Out[26]:
```
<matplotlib.image.AxesImage at 0x180bbec7810>
```



In [27]:
```
(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 [28]:
```
plt.imshow(cv2.cvtColor(cropped_image, cv2.COLOR_BGR2RGB))
```

Out[28]:
```
<matplotlib.image.AxesImage at 0x180be923d90>
```



In [29]:
```
reader = easyocr.Reader(['en'])
result = reader.readtext(cropped_image)
result
```

```
Neither CUDA nor MPS are available - defaulting to CPU. Note: This module i
s much faster with a GPU.
```

Out[29]:
```
[([[8, 7], [557, 7], [557, 126], [8, 126]], 'MHI2 DE1433', 0.26974458548563
74)]
```

In [ ]:

```
```

In [30]:

```
text = result[0][-2]
font = cv2.FONT_HERSHEY_SIMPLEX
res = cv2.putText(img, text=text, org=(approx[0][0][0], approx[1][0][1]+6
0), fontFace=font, fontScale=1, color=(0,255,0), thickness=2, lineType=cv2.
LINE_AA)
res = cv2.rectangle(img, tuple(approx[0][0]), tuple(approx[2][0]), (0,255,
0),3)
outp=cv2.cvtColor(res, cv2.COLOR_BGR2RGB)
```

In [31]:

```
recognized_text = result[0][1]

# Print the recognized text
print(recognized_text)
```

```
MHI2 DE1433
```

In [32]:

```
# Assuming 'res' is your image variable
# Convert to RGB format
rgb_image = cv2.cvtColor(res, cv2.COLOR_BGR2RGB)

# Define the output folder path
output_folder = r'C:\Users\rohan'

# Check if the folder exists, if not, create it
if not os.path.exists(output_folder):
    os.makedirs(output_folder)

# Define the output file path
output_path = os.path.join(output_folder, 'output_image.jpg')

# Save the image as JPG
cv2.imwrite(output_path, rgb_image)
```

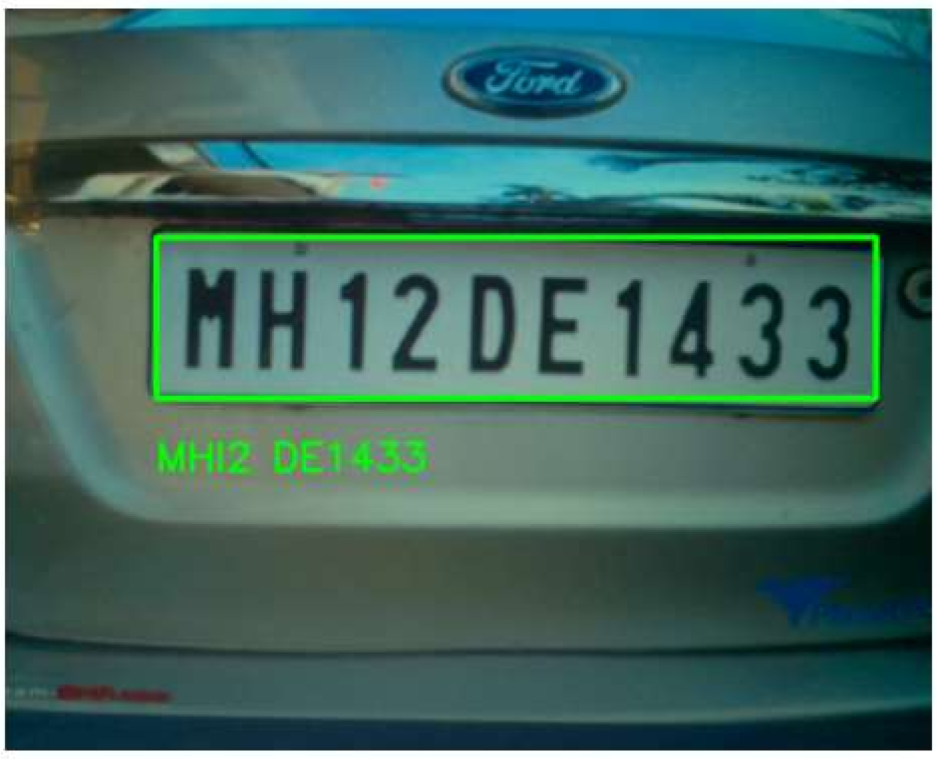
Out[32]:

```
True
```

In [33]:
```
# Print the recognized text
print(recognized_text)

# Display the image with the recognized text as the title
plt.imshow(rgb_image)
plt.title(text)
plt.axis('off')  # Hide the axis

# Save the plot as another image
plt.savefig('image_with_title.png')

# Show the plot
plt.show()
```

```
MHI2 DE1433
```

MHI2 DE1433



In [34]:
```
file_from = 'image_with_title.png'
file_to = '/output/image1.png'
def upload_file(file_from, file_to):
    dbx = dropbox.Dropbox(access_token)
    f = open(file_from, 'rb')
    dbx.files_upload(f.read(), file_to)
    print("uploaded")
upload_file(file_from,file_to)
```

```
uploaded
```