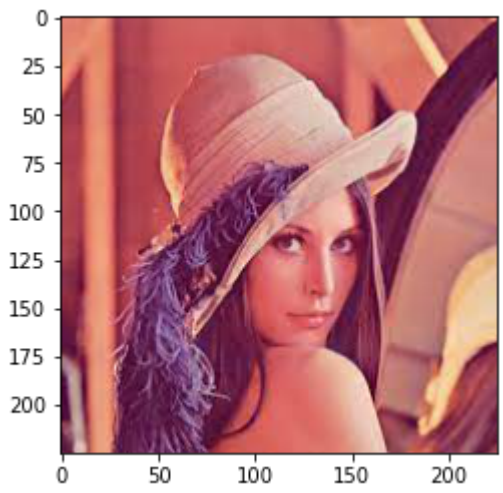
In [45]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
import cv2
```

In [46]:
```
img=plt.imread('lena_new.jpeg')
plt.imshow(img)
```

Out[46]:
```
<matplotlib.image.AxesImage at 0x2b6ccf76130>
```



In [47]:
```
#Splitting into channels
blue,green,red = cv2.split(img)
```

In [48]:
```
df_blue = blue/255
df_green = green/255
df_red = red/255
```

In [49]:
```
def PCA2(X , num_components):

    #Step-1
    X_meaned = X - np.mean(X , axis = 0)

    #Step-2
    cov_mat = np.cov(X_meaned , rowvar = False)

    #Step-3
    eigen_values , eigen_vectors = np.linalg.eigh(cov_mat)

    #Step-4
    sorted_index = np.argsort(eigen_values)[::-1]
    sorted_eigenvalue = eigen_values[sorted_index]
    sorted_eigenvectors = eigen_vectors[:,sorted_index]

    #Step-5
    eigenvector_subset = sorted_eigenvectors[:,0:num_components]

    #Step-6
    X_reduced = np.dot(eigenvector_subset.transpose() , X_meaned.transpose() ).transpose()

    return X_reduced
```

In [50]:
```
op_b = PCA2(df_blue,50)
op_g = PCA2(df_green,50)
op_r = PCA2(df_red,50)
```

In [51]:
```
print(op_b.shape)
print(op_r.shape)
print(op_g.shape)
```

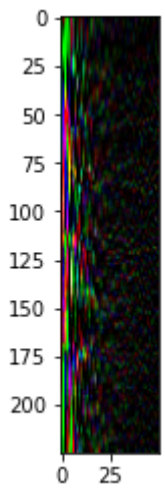
```
(225, 50)
(225, 50)
(225, 50)
```

In [52]:
```
pca_img= (cv2.merge((op_b,op_g,op_r)))
```

In [53]:
```
plt.imshow(pca_img)
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```

Out[53]:
```
<matplotlib.image.AxesImage at 0x2b6cd10df40>
```



In [54]:
```
pca_b = PCA(n_components=50)
pca_b.fit(df_blue)
pca_g = PCA(n_components=50)
pca_g.fit(df_green)
pca_r = PCA(n_components=50)
pca_r.fit(df_green)
b_arr = pca_b.inverse_transform(op_b)
g_arr = pca_g.inverse_transform(op_g)
r_arr = pca_r.inverse_transform(op_r)
```

In [57]:
```
print(f"Blue Channel : {sum(pca_b.explained_variance_ratio_)}")
print(f"Green Channel: {sum(pca_g.explained_variance_ratio_)}")
print(f"Red Channel  : {sum(pca_r.explained_variance_ratio_)}")
```

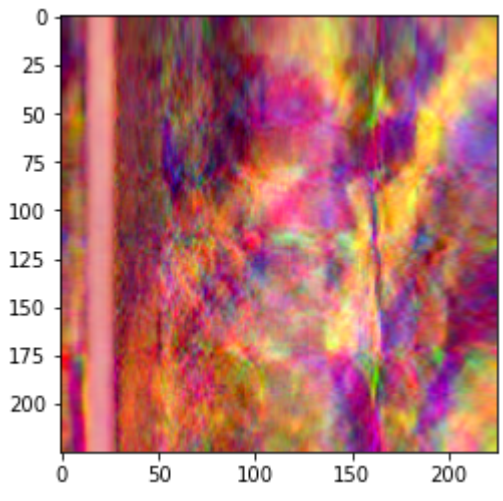
```
Blue Channel : 0.9716693412385506
Green Channel: 0.974376741388708
Red Channel  : 0.974376741388708
```

In [55]:
```
img_reduced= (cv2.merge((b_arr, g_arr, r_arr)))
```

In [56]:
```
plt.imshow(img_reduced)
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```

Out[56]:
```
<matplotlib.image.AxesImage at 0x2b6cd16ac40>
```



In [ ]: