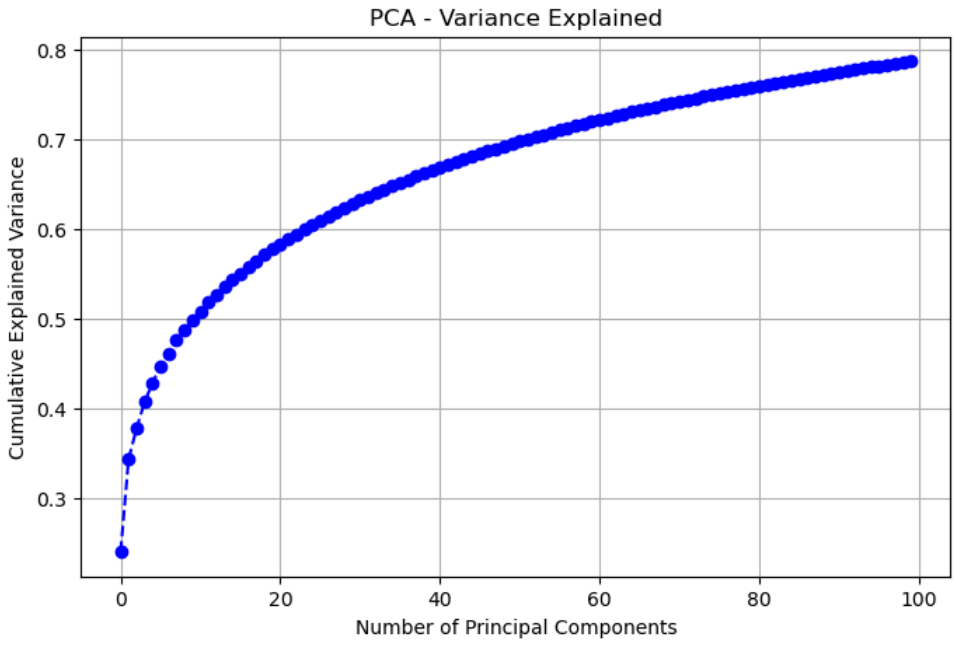
In [1]:
```
import os
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
from sklearn.decomposition import PCA
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import seaborn as sns
```

```
C:\Users\DELL\anaconda3\lib\site-packages\pandas\core\arrays\masked.py:60: UserWarning: Pandas requires versio
n '1.3.6' or newer of 'bottleneck' (version '1.3.5' currently installed).
  from pandas.core import (
```

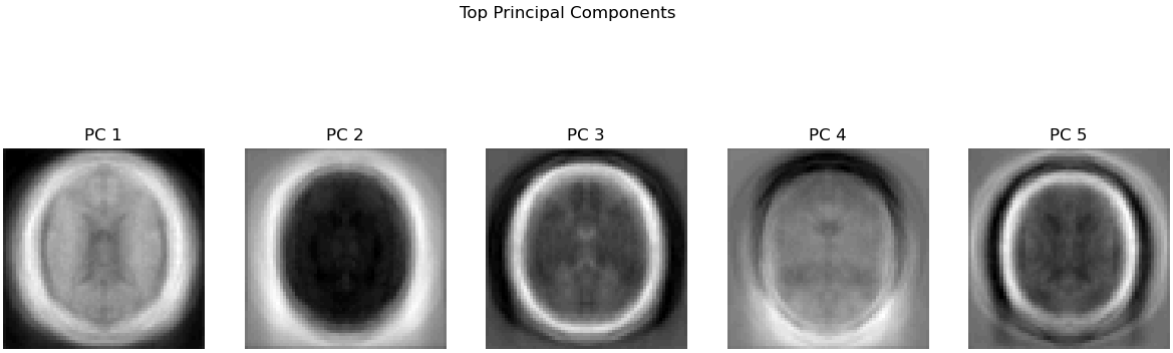
In [2]:
```
dataset_path = r"C:\Users\DELL\Downloads\Tumor_Detection"

def load_mri_images(folder_path):
    images, labels = [], []
    label_dict = {'glioma': 0, 'meningioma': 1, 'pituitary': 2, 'healthy': 3}  # 4 categories

    for folder in os.listdir(folder_path):
        folder_full_path = os.path.join(folder_path, folder)
        for img_file in os.listdir(folder_full_path):
            img_path = os.path.join(folder_full_path, img_file)
            img = Image.open(img_path).convert('L').resize((64, 64))
            img_array = np.array(img).flatten() / 255.0
            images.append(img_array)
            labels.append(label_dict[folder])
    return np.array(images), np.array(labels)

X, y = load_mri_images(dataset_path)
print("Data Loaded! Shape of X:", X.shape, "Shape of y:", y.shape)
```

```
Data Loaded! Shape of X: (7023, 4096) Shape of y: (7023,)
```

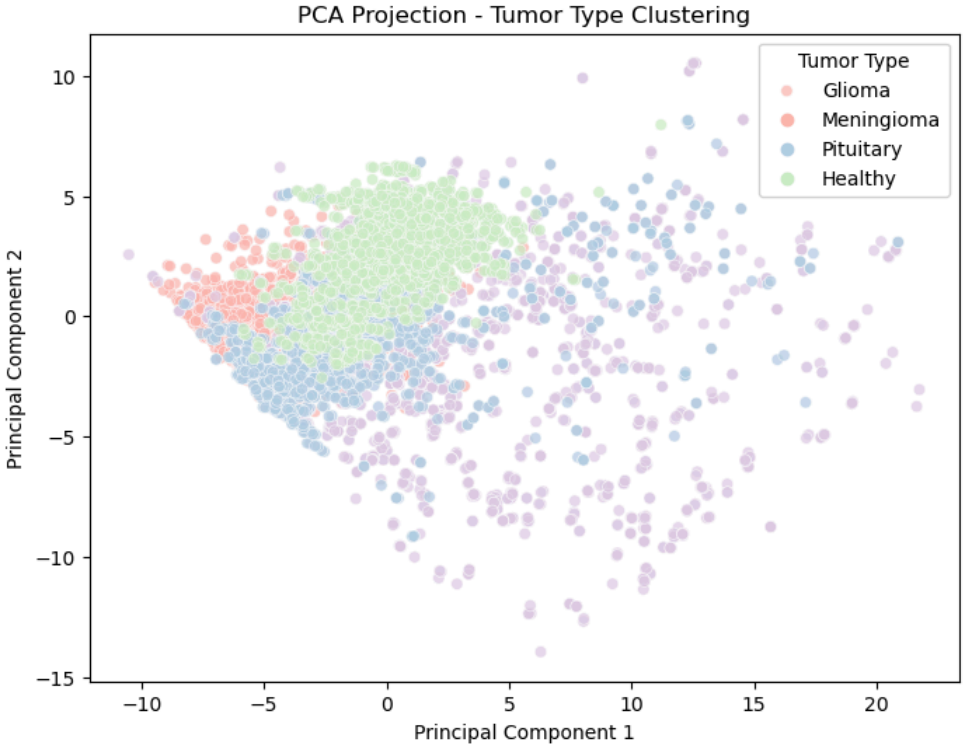
In [3]:
```
# Applying PCA
n_components = 100
pca = PCA(n_components=n_components)
X_pca = pca.fit_transform(X)

plt.figure(figsize=(8,5))
plt.plot(np.cumsum(pca.explained_variance_ratio_), marker='o', linestyle='--', color='b')
plt.xlabel('Number of Principal Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('PCA - Variance Explained')
plt.grid()
plt.show()
```



In [4]:
```
# Visualizing first few principal components as images
def plot_pca_components(pca, img_shape, num_components=5):
    fig, axes = plt.subplots(1, num_components, figsize=(15, 5))
    for i in range(num_components):
        axes[i].imshow(pca.components_[i].reshape(img_shape), cmap='gray')
        axes[i].set_title(f'PC {i+1}')
        axes[i].axis('off')
    plt.suptitle("Top Principal Components")
    plt.show()

plot_pca_components(pca, (64, 64))
```



In [5]:
```
# Scatter Plot of PCA Components
plt.figure(figsize=(8,6))
sns.scatterplot(x=X_pca[:,0], y=X_pca[:,1], hue=y, palette="Pastel1", alpha=0.7)
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.title("PCA Projection - Tumor Type Clustering")
plt.legend(title="Tumor Type", labels=['Glioma', 'Meningioma', 'Pituitary', 'Healthy'])
plt.show()
```

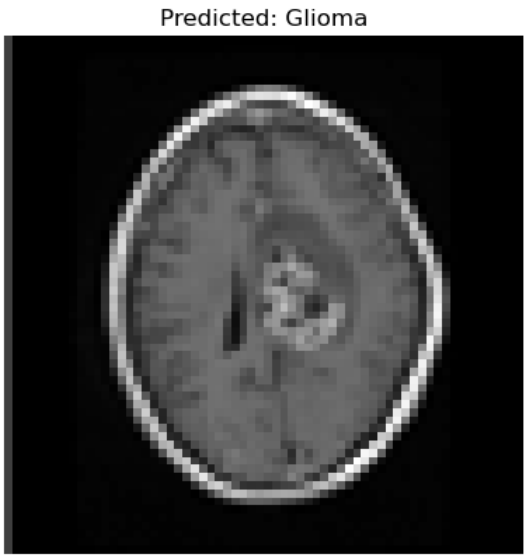


In [6]:
```
# SVM Classification using PCA Features
X_train, X_test, y_train, y_test = train_test_split(X_pca, y, test_size=0.2, random_state=42)
svm_model = SVC(kernel='rbf', C=10, gamma=0.01)
svm_model.fit(X_train, y_train)

y_pred = svm_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("\n SVM Classification Accuracy:", accuracy)
```

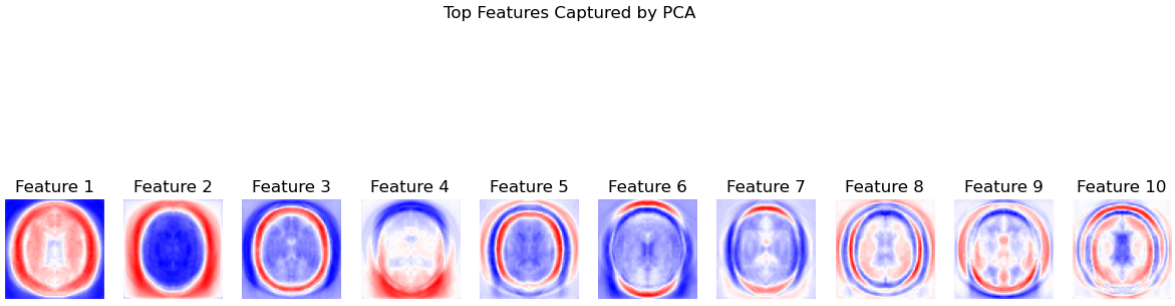
```
 SVM Classification Accuracy: 0.9494661921708185
```

In [7]:
```
def predict_new_scan(image_path, pca_model, svm_model):
    img = Image.open(image_path).convert('L').resize((64, 64))
    img_array = np.array(img).flatten() / 255.0
    img_pca = pca_model.transform([img_array])
    pred_label = svm_model.predict(img_pca)[0]

    tumor_types = {0: 'Glioma', 1: 'Meningioma', 2: 'Pituitary', 3: 'Healthy'}
    print("\n Prediction for New MRI Scan:", tumor_types[pred_label])

    plt.imshow(img, cmap='gray')
    plt.title(f"Predicted: {tumor_types[pred_label]}")
    plt.axis('off')
    plt.show()

new_mri_path = r"C:\Users\DELL\Downloads\trail1.png"
predict_new_scan(new_mri_path, pca, svm_model)
```

```
 Prediction for New MRI Scan: Glioma
```



In [8]:
```
def visualize_pca_features(pca, num_features=10):
    fig, axes = plt.subplots(1, num_features, figsize=(15, 5))
    for i in range(num_features):
        component = pca.components_[i].reshape(64, 64)
        axes[i].imshow(component, cmap='bwr')
        axes[i].set_title(f'Feature {i+1}')
        axes[i].axis('off')
    plt.suptitle("Top Features Captured by PCA")
    plt.show()

visualize_pca_features(pca)
```



In [ ]:
```
```