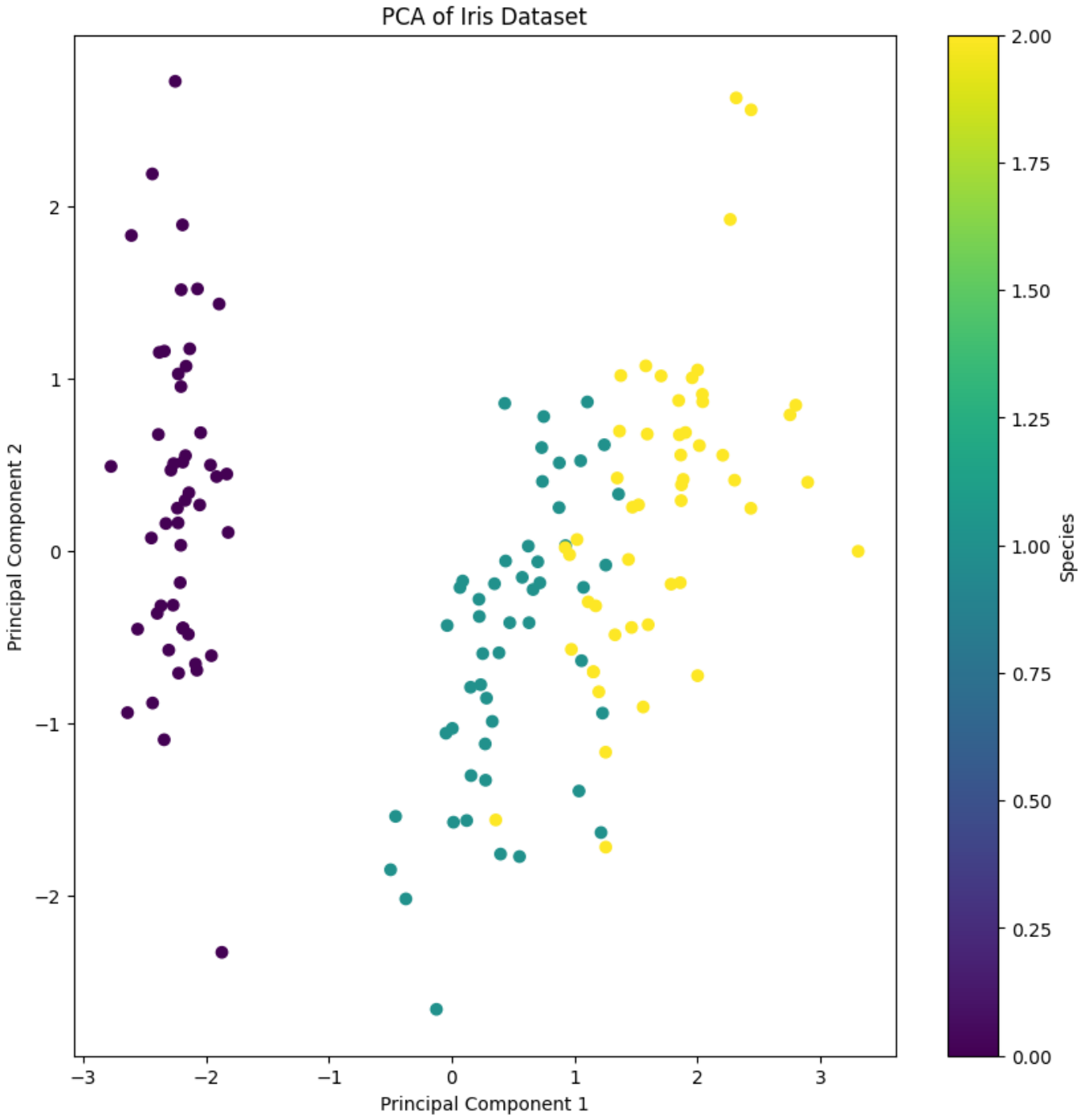
In [6]:

```
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

# Load the dataset
df = pd.read_csv("IRIS_DATASET.csv")

# Define feature columns and target column
features = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
targetcolumn = 'species'

# Extract features and target from the dataframe
x = df[features].values
y = df[targetcolumn].values

# Standardize the data
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)

# Perform PCA
pca_model = PCA(n_components=2)
x_pca = pca_model.fit_transform(x_scaled)

# Check the shape of the transformed data
print(f"Original shape: {x.shape}")
print(f"Transformed shape: {x_pca.shape}")

# Plotting the PCA results
plt.figure(figsize=(10, 10))
scatter = plt.scatter(x_pca[:, 0], x_pca[:, 1], c=pd.factorize(y)[0], cmap='viridis')
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.colorbar(scatter, label='Species')
plt.title("PCA of Iris Dataset")
plt.show()

# Explained variance
explained_variance = pca_model.explained_variance_ratio_
print(f"Explained variance by each component: {explained_variance}")
```

```
Original shape: (150, 4)
Transformed shape: (150, 2)
```



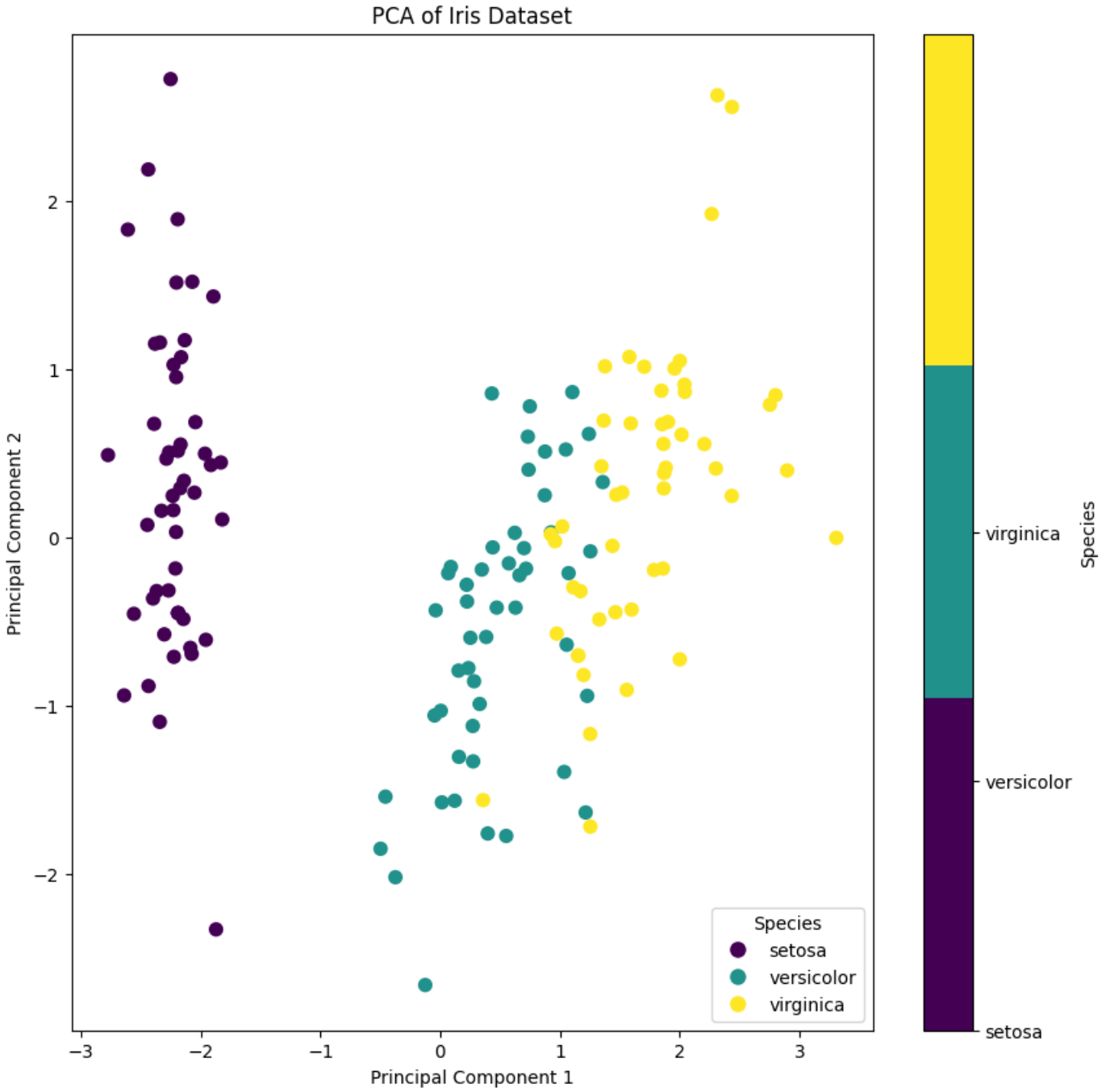

```
Explained variance by each component: [0.72770452 0.23030523]
```

In [8]:

```python
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

# Load the dataset
df = pd.read_csv("IRIS_DATASET.csv")

# Define feature columns and target column
features = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
targetcolumn = 'species'

# Extract features and target from the dataframe
```

```
x = df[features].values
y = df[targetcolumn].values

# Standardize the data
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)

# Perform PCA
pca_model = PCA(n_components=2)
x_pca = pca_model.fit_transform(x_scaled)

# Check the shape of the transformed data
print(f"Original shape: {x.shape}")
print(f"Transformed shape: {x_pca.shape}")

# Create a mapping of species to integers
species_mapping = {species: idx for idx, species in enumerate(np.unique(y))}
# Map the target values to integers
y_mapped = np.array([species_mapping[species] for species in y])

# Define the colormap
cmap = plt.get_cmap('viridis', len(species_mapping))

# Plotting the PCA results
plt.figure(figsize=(10, 10))
scatter = plt.scatter(x_pca[:, 0], x_pca[:, 1], c=y_mapped, cmap=cmap, s=50)
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.title("PCA of Iris Dataset")

# Create a legend manually with specific colors
handles = []
for species, idx in species_mapping.items():
    handles.append(plt.Line2D([0], [0], marker='o', color='w', markerfacecolor=cmap(idx

plt.legend(handles=handles, title='Species')

# Colorbar to show the color mapping
cbar = plt.colorbar(scatter)
cbar.set_label('Species')
cbar.set_ticks(np.linspace(0, 1, len(species_mapping)))
cbar.set_ticklabels(list(species_mapping.keys()))

plt.show()

# Explained variance
explained_variance = pca_model.explained_variance_ratio_
print(f"Explained variance by each component: {explained_variance}")
```

```
Original shape: (150, 4)
Transformed shape: (150, 2)
```




```
Explained variance by each component: [0.72770452 0.23030523]
```