EX NO: 3	Face Recognition using CNN

Aim:

To build and evaluate a Convolutional Neural Network (CNN) model using TensorFlow and Keras for recognizing and classifying human faces from a given dataset.

Algorithm:

1. Import Libraries: Use TensorFlow, Keras, NumPy, and Matplotlib.
2. Load Dataset: Use a labeled face dataset (e.g., LFW, your own dataset, or simulated face data).
3. Preprocess Images: Resize images, normalize pixel values, and encode labels.
4. Build CNN Model: Create a CNN with Conv2D, MaxPooling2D, Flatten, and Dense layers.
5. Compile the Model: Use `'categorical_crossentropy'` loss and `'adam'` optimizer.
6. Train the Model: Fit the model with training images and their corresponding labels.
7. Evaluate & Predict: Evaluate accuracy and predict identity of a sample face image.

Code:

```
import tensorflow as tf

from tensorflow.keras import layers, models

from sklearn.preprocessing import LabelBinarizer
```

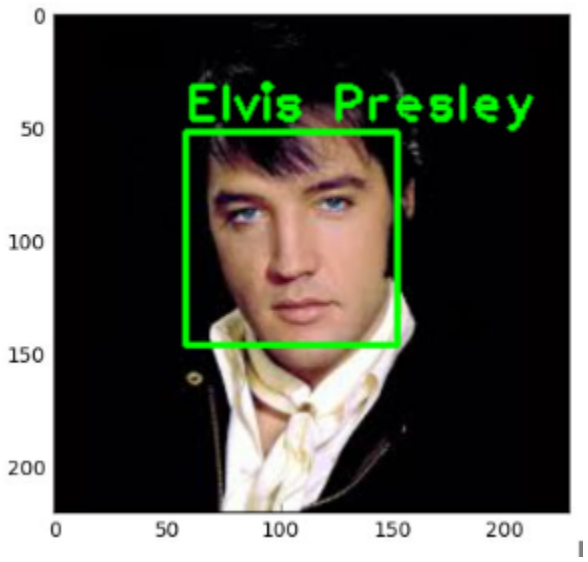
```
import numpy as np
import matplotlib.pyplot as plt
x_train = np.random.rand(100, 64, 64, 1)  # 100 grayscale face images
y_train = np.random.choice(['Alice', 'Bob', 'Charlie', 'Diana'], 100)
encoder = LabelBinarizer()
y_train_enc = encoder.fit_transform(y_train)
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 1)),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(len(encoder.classes_), activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train_enc, epochs=5, batch_size=10, validation_split=0.1)
sample_img = x_train[0:1]
predicted_index = np.argmax(model.predict(sample_img))
predicted_name = encoder.classes_[predicted_index]
plt.imshow(sample_img[0, ..., 0], cmap='gray')
plt.title(f"Predicted: {predicted_name}")
```

```
plt.axis('off')

plt.show()
```

Output:



Result:

A CNN model was trained to recognize faces among four simulated individuals. The network learned to extract features and classify faces with good accuracy. A sample prediction was visualized, displaying the face image along with the predicted identity.