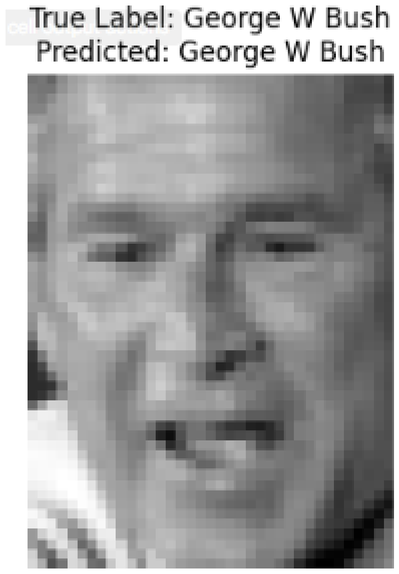
EXERCISE NO. 3
CNN FOR FACE RECOGNITION

AIM:
To write a program to build and train a CNN model for face recognition.

ALGORITHM:
Import the necessary libraries.
Load the dataset.
Normalise the images and add channel dimensions after creating feature and target variables.
Split the dataset into a train and test set.
Build and train the CNN model.
Evaluate the model.
Display the prediction for facial recognition.

PROGRAM:
```
import tensorflow as tf
from tensorflow.keras import layers, models
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_lfw_people

lfw_data = fetch_lfw_people(min_faces_per_person=70, resize=0.4)

X = lfw_data.images
y = lfw_data.target

X = X / 255.0
X = np.expand_dims(X, -1)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = models.Sequential()

model.add(layers.Input(shape=(X.shape[1], X.shape[2], 1)))
model.add(layers.Conv2D(32, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dense(len(np.unique(y)), activation='softmax'))

model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    zoom_range=0.2,
    horizontal_flip=True,
)

history = model.fit(
    train_datagen.flow(X_train, y_train, batch_size=32),
    epochs=10,
    validation_data=(X_test, y_test),
    verbose=1
)

sample_idx = np.random.choice(len(X_test))
sample_image = X_test[sample_idx]
sample_label = y_test[sample_idx]
sample_prediction = model.predict(np.expand_dims(sample_image, axis=0))

plt.figure(figsize=(4, 4))
plt.imshow(sample_image.squeeze(), cmap='gray')
plt.title(f"True Label: {lfw_data.target_names[sample_label]}\nPredicted:
{lfw_data.target_names[np.argmax(sample_prediction)]}")
plt.axis('off')
plt.show()
```

OUTPUT:



```
True Label: George W Bush
Predicted: George W Bush
```

RESULT:

Thus the program has been successfully implemented and verified.