Data Collection: Obtain car images from the CIFAR database and gun images from the internet. Ensure you have a sufficient number of images for both classes. In this case, let's assume we have 4 images of guns.

Data Preprocessing: Resize all images to a uniform size and normalize pixel values. Split the dataset into training and testing sets.

Model Building: Design a convolutional neural network (CNN) model using TensorFlow/Keras. CNNs are well-suited for image classification tasks.

Model Training: Train the CNN model using the training dataset.

Model Evaluation: Evaluate the trained model using the testing dataset. However, since the accuracy is not considered in the evaluation, you can skip this step.

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

# Step 1: Data Collection (Assuming you have the images stored in 'car_images' and
'gun_images' folders)
# Use ImageDataGenerator for loading and preprocessing images
train_datagen = ImageDataGenerator(rescale=1./255)

train_car_images = train_datagen.flow_from_directory(
   'car_images',                   # Path to the directory containing car images
   target_size=(64, 64),           # Resize images to 64x64 pixels
   batch_size=32,
   class_mode='binary'             # Since we have two classes: cars and guns
)

train_gun_images = train_datagen.flow_from_directory(
   'gun_images',                   # Path to the directory containing gun images
   target_size=(64, 64),
   batch_size=32,
   class_mode='binary'
)

# Step 3: Model Building
```

```
model = Sequential([
   Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)),
   MaxPooling2D(pool_size=(2, 2)),
   Conv2D(64, (3, 3), activation='relu'),
   MaxPooling2D(pool_size=(2, 2)),
   Flatten(),
   Dense(128, activation='relu'),
   Dense(1, activation='sigmoid')  # Sigmoid activation for binary classification
])

# Step 4: Model Training
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Combine both car and gun images for training
combined_images = np.concatenate((train_car_images, train_gun_images), axis=0)

# Train the model
model.fit(combined_images, epochs=10, steps_per_epoch=len(combined_images), verbose=1)

# Save the model
model.save('car_gun_classifier.h5')
```