In [1]:
```python
# nag_optimizer.py

import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.datasets import cifar10
from tensorflow.keras import layers, models
from tensorflow.keras.optimizers import SGD
from sklearn.metrics import confusion_matrix
import seaborn as sns
import time
from sklearn.model_selection import train_test_split
```

In [2]:
```python
# Load CIFAR-10 dataset
(X_train, y_train), (X_test, y_test) = cifar10.load_data()

# Normalize the images
X_train =X_train.astype('float32') / 255.0
X_test = X_test.astype('float32') / 255.0

# Ensure labels are integers (no one-hot encoding)
y_train = y_train.astype('int')
y_test = y_test.astype('int')

X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2,
print (X_train.shape )
print (X_test.shape )
```

```
(40000, 32, 32, 3)
(10000, 32, 32, 3)
```

In [3]:
```python
# Create model function
def create_model(optimizer):
    model = models.Sequential()
    model.add(layers.Flatten(input_shape=(32, 32, 3)))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(10, activation='softmax'))
    model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy', metr
    return model

# Nesterov Accelerated Gradient (NAG) optimizer
optimizer = SGD(learning_rate=0.001, momentum=0.9, nesterov=True)
```

In [4]:
```python
# Train and evaluate model
start_time = time.time()
model = create_model(optimizer)
history = model.fit(X_train, y_train, epochs=50, batch_size=64, validation_data=(X_
end_time = time.time()

# Record training time
training_time = end_time - start_time
print(f"Training time: {training_time:.2f} seconds")
```

```
c:\Users\Omar Wessam\AppData\Local\Programs\Python\Python312\Lib\site-packages\keras
\src\layers\reshaping\flatten.py:37: UserWarning: Do not pass an `input_shape`/`inpu
t_dim` argument to a layer. When using Sequential models, prefer using an `Input(sha
pe)` object as the first layer in the model instead.
  super().__init__(**kwargs)
```

```
Epoch 1/50
625/625 3s 4ms/step - accuracy: 0.2546 - loss: 2.0745 - val_acc
uracy: 0.3472 - val_loss: 1.8482
Epoch 2/50
625/625 2s 4ms/step - accuracy: 0.3603 - loss: 1.8288 - val_acc
uracy: 0.3725 - val_loss: 1.7757
Epoch 3/50
625/625 2s 3ms/step - accuracy: 0.3849 - loss: 1.7535 - val_acc
uracy: 0.3878 - val_loss: 1.7283
Epoch 4/50
625/625 2s 3ms/step - accuracy: 0.3970 - loss: 1.7230 - val_acc
uracy: 0.3974 - val_loss: 1.7064
Epoch 5/50
625/625 2s 3ms/step - accuracy: 0.4138 - loss: 1.6864 - val_acc
uracy: 0.4126 - val_loss: 1.6696
Epoch 6/50
625/625 2s 4ms/step - accuracy: 0.4251 - loss: 1.6526 - val_acc
uracy: 0.4228 - val_loss: 1.6389
Epoch 7/50
625/625 2s 2ms/step - accuracy: 0.4357 - loss: 1.6196 - val_acc
uracy: 0.4154 - val_loss: 1.6394
Epoch 8/50
625/625 2s 3ms/step - accuracy: 0.4477 - loss: 1.5967 - val_acc
uracy: 0.4229 - val_loss: 1.6167
Epoch 9/50
625/625 1s 2ms/step - accuracy: 0.4536 - loss: 1.5674 - val_acc
uracy: 0.4332 - val_loss: 1.6122
Epoch 10/50
625/625 1s 2ms/step - accuracy: 0.4565 - loss: 1.5578 - val_acc
uracy: 0.4424 - val_loss: 1.5832
Epoch 11/50
625/625 1s 2ms/step - accuracy: 0.4646 - loss: 1.5407 - val_acc
uracy: 0.4464 - val_loss: 1.5714
Epoch 12/50
625/625 1s 2ms/step - accuracy: 0.4694 - loss: 1.5300 - val_acc
uracy: 0.4375 - val_loss: 1.5907
Epoch 13/50
625/625 1s 2ms/step - accuracy: 0.4745 - loss: 1.5131 - val_acc
uracy: 0.4471 - val_loss: 1.5622
Epoch 14/50
625/625 2s 2ms/step - accuracy: 0.4806 - loss: 1.4962 - val_acc
uracy: 0.4546 - val_loss: 1.5566
Epoch 15/50
625/625 1s 2ms/step - accuracy: 0.4835 - loss: 1.4878 - val_acc
uracy: 0.4519 - val_loss: 1.5536
Epoch 16/50
625/625 1s 2ms/step - accuracy: 0.4843 - loss: 1.4747 - val_acc
uracy: 0.4649 - val_loss: 1.5234
Epoch 17/50
625/625 1s 2ms/step - accuracy: 0.4971 - loss: 1.4481 - val_acc
uracy: 0.4674 - val_loss: 1.5213
Epoch 18/50
625/625 1s 2ms/step - accuracy: 0.5016 - loss: 1.4469 - val_acc
uracy: 0.4695 - val_loss: 1.5129
Epoch 19/50
625/625 1s 2ms/step - accuracy: 0.5032 - loss: 1.4323 - val_acc
```

```
uracy: 0.4650 - val_loss: 1.5104
Epoch 20/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5052 - loss: 1.4397 - val_acc
uracy: 0.4686 - val_loss: 1.5133
Epoch 21/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5086 - loss: 1.4177 - val_acc
uracy: 0.4754 - val_loss: 1.4975
Epoch 22/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.5088 - loss: 1.4151 - val_acc
uracy: 0.4693 - val_loss: 1.5004
Epoch 23/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.5147 - loss: 1.4025 - val_acc
uracy: 0.4711 - val_loss: 1.4966
Epoch 24/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.5162 - loss: 1.3948 - val_acc
uracy: 0.4723 - val_loss: 1.4912
Epoch 25/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.5207 - loss: 1.3838 - val_acc
uracy: 0.4761 - val_loss: 1.4868
Epoch 26/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.5286 - loss: 1.3660 - val_acc
uracy: 0.4726 - val_loss: 1.4926
Epoch 27/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.5269 - loss: 1.3618 - val_acc
uracy: 0.4832 - val_loss: 1.4686
Epoch 28/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5322 - loss: 1.3498 - val_acc
uracy: 0.4818 - val_loss: 1.4772
Epoch 29/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.5275 - loss: 1.3553 - val_acc
uracy: 0.4850 - val_loss: 1.4636
Epoch 30/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5335 - loss: 1.3443 - val_acc
uracy: 0.4828 - val_loss: 1.4633
Epoch 31/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.5329 - loss: 1.3462 - val_acc
uracy: 0.4887 - val_loss: 1.4559
Epoch 32/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5395 - loss: 1.3287 - val_acc
uracy: 0.4689 - val_loss: 1.4944
Epoch 33/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5358 - loss: 1.3303 - val_acc
uracy: 0.4855 - val_loss: 1.4658
Epoch 34/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5356 - loss: 1.3278 - val_acc
uracy: 0.4902 - val_loss: 1.4537
Epoch 35/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5446 - loss: 1.3124 - val_acc
uracy: 0.4910 - val_loss: 1.4487
Epoch 36/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5451 - loss: 1.3122 - val_acc
uracy: 0.4868 - val_loss: 1.4587
Epoch 37/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5451 - loss: 1.3093 - val_acc
uracy: 0.4936 - val_loss: 1.4547
Epoch 38/50
```

```
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5454 - loss: 1.3015 - val_acc
uracy: 0.4954 - val_loss: 1.4311
Epoch 39/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5480 - loss: 1.2925 - val_acc
uracy: 0.4942 - val_loss: 1.4473
Epoch 40/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5501 - loss: 1.2975 - val_acc
uracy: 0.4968 - val_loss: 1.4382
Epoch 41/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5543 - loss: 1.2814 - val_acc
uracy: 0.4899 - val_loss: 1.4461
Epoch 42/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5554 - loss: 1.2743 - val_acc
uracy: 0.4919 - val_loss: 1.4480
Epoch 43/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5591 - loss: 1.2694 - val_acc
uracy: 0.4947 - val_loss: 1.4438
Epoch 44/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.5589 - loss: 1.2700 - val_acc
uracy: 0.4962 - val_loss: 1.4372
Epoch 45/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5621 - loss: 1.2511 - val_acc
uracy: 0.4968 - val_loss: 1.4287
Epoch 46/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5654 - loss: 1.2501 - val_acc
uracy: 0.4959 - val_loss: 1.4376
Epoch 47/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5653 - loss: 1.2583 - val_acc
uracy: 0.4947 - val_loss: 1.4305
Epoch 48/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5625 - loss: 1.2559 - val_acc
uracy: 0.4970 - val_loss: 1.4428
Epoch 49/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.5721 - loss: 1.2386 - val_acc
uracy: 0.4953 - val_loss: 1.4392
Epoch 50/50
625/625 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.5660 - loss: 1.2385 - val_acc
uracy: 0.4987 - val_loss: 1.4374
Training time: 78.25 seconds
```

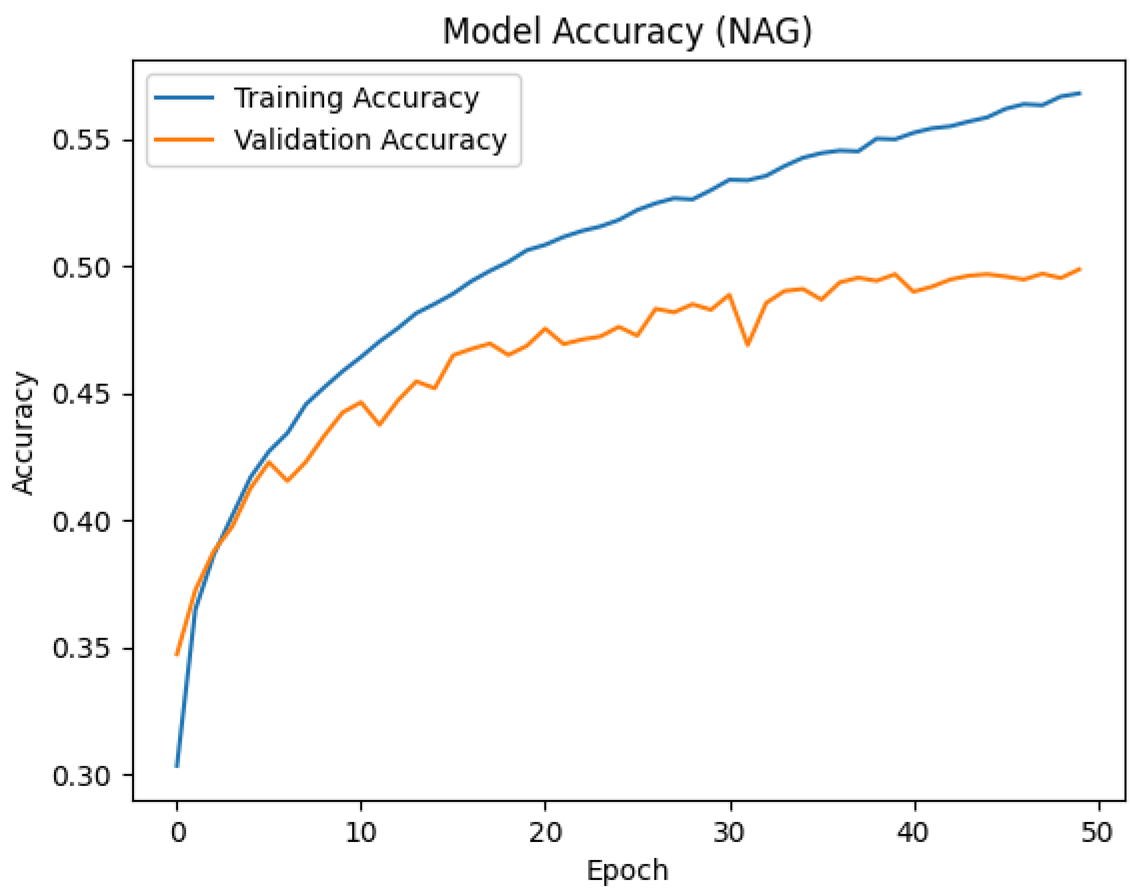
In [5]:

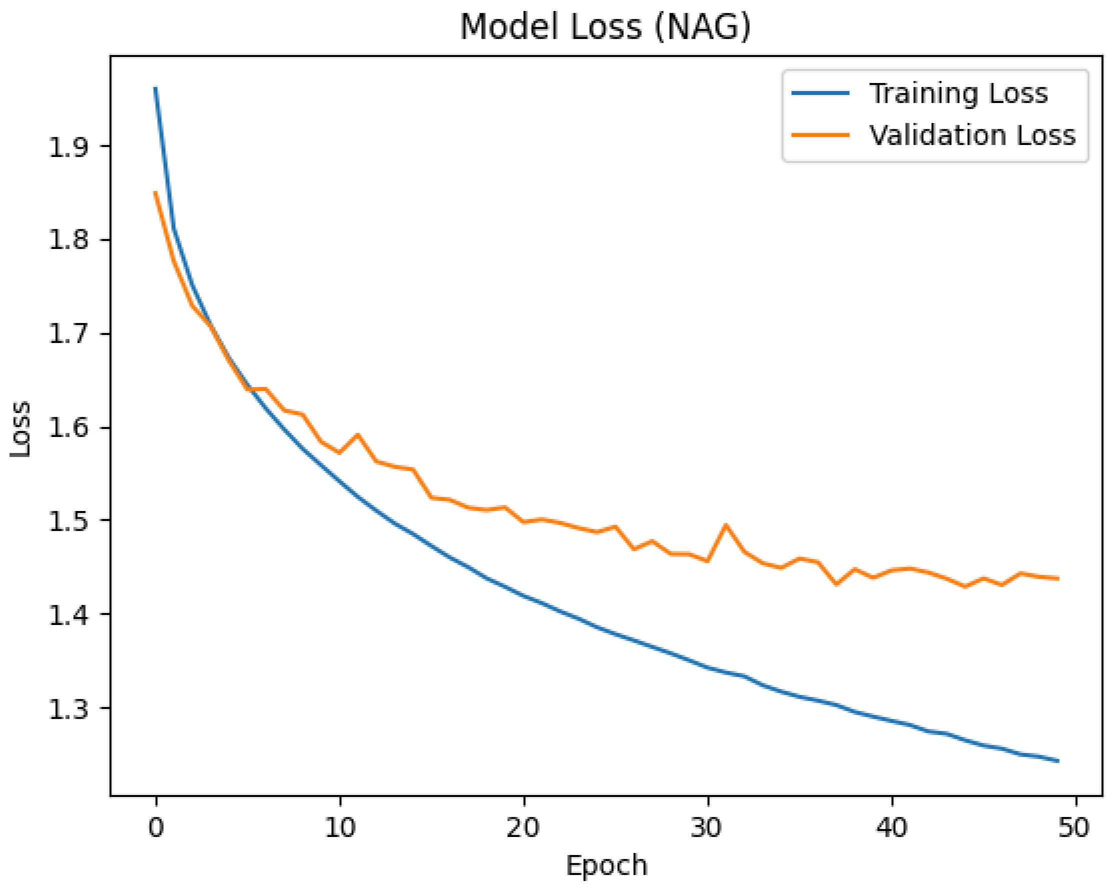
```python
# Plot training and validation accuracy
accuracy = history.history['accuracy']
val_accuracy = history.history['val_accuracy']
plt.plot(accuracy, label='Training Accuracy')
plt.plot(val_accuracy, label='Validation Accuracy')
plt.title('Model Accuracy (NAG)')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

# Plot training and validation accuracy
accuracy = history.history['loss']
val_accuracy = history.history['val_loss']
plt.plot(accuracy, label='Training Loss')
plt.plot(val_accuracy, label='Validation Loss')
```

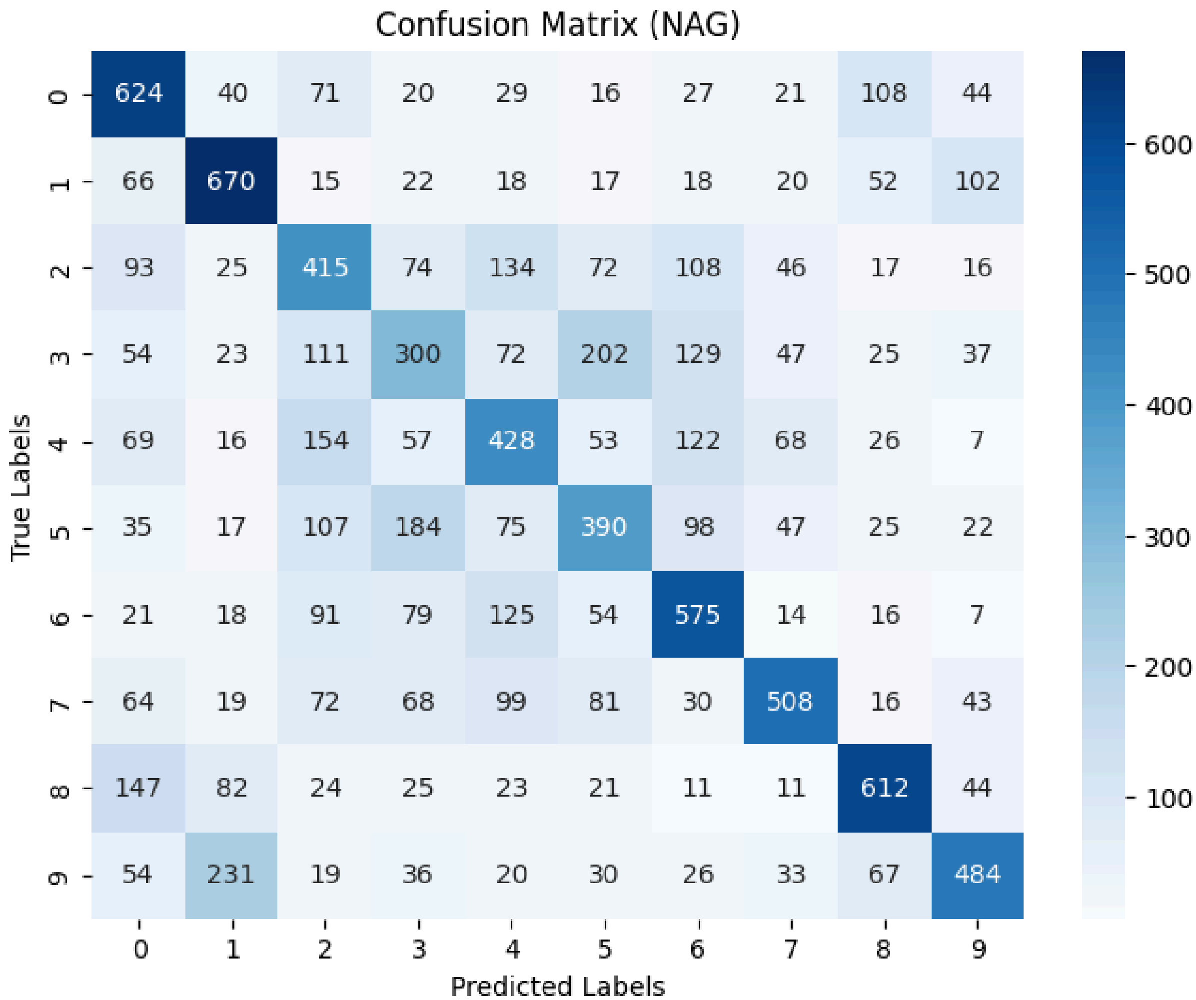
```
plt.title('Model Loss (NAG)')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()

# Confusion Matrix
y_pred = np.argmax(model.predict(X_test), axis=1)
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=np.arange(10), ytick
plt.title('Confusion Matrix (NAG)')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```






```
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```


Confusion Matrix (NAG)
True Labels
Predicted Labels
0 1 2 3 4 5 6 7 8 9
624 40 71 20 29 16 27 21 108 44
66 670 15 22 18 17 18 20 52 102
93 25 415 74 134 72 108 46 17 16
54 23 111 300 72 202 129 47 25 37
69 16 154 57 428 53 122 68 26 7
35 17 107 184 75 390 98 47 25 22
21 18 91 79 125 54 575 14 16 7
64 19 72 68 99 81 30 508 16 43
147 82 24 25 23 21 11 11 612 44
54 231 19 36 20 30 26 33 67 484
600
500
400
300
200
100

In [ ]: