In [7]:
```
# Implementation of Support Vector Machines (SVM) for classifying images of handwri
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import datasets, svm, metrics
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

In [8]:
```
digits = datasets.load_digits()
digits.keys()
```

Out[8]:
```
dict_keys(['data', 'target', 'frame', 'feature_names', 'target_names', 'images',
'DESCR'])
```

In [9]:
```
X = digits.data   # Flattened pixel values (64 features for 8x8 images)
y = digits.target # Labels (0–9)

# X, y are from digits.data and digits.target
indices = np.arange(len(X))                # [0,1,2,...,n-1]

# Split X, y and indices together so idx_test aligns with X_test
X_train, X_test, y_train, y_test, idx_train, idx_test = train_test_split(
    X, y, indices, test_size=0.2, random_state=42
)
```

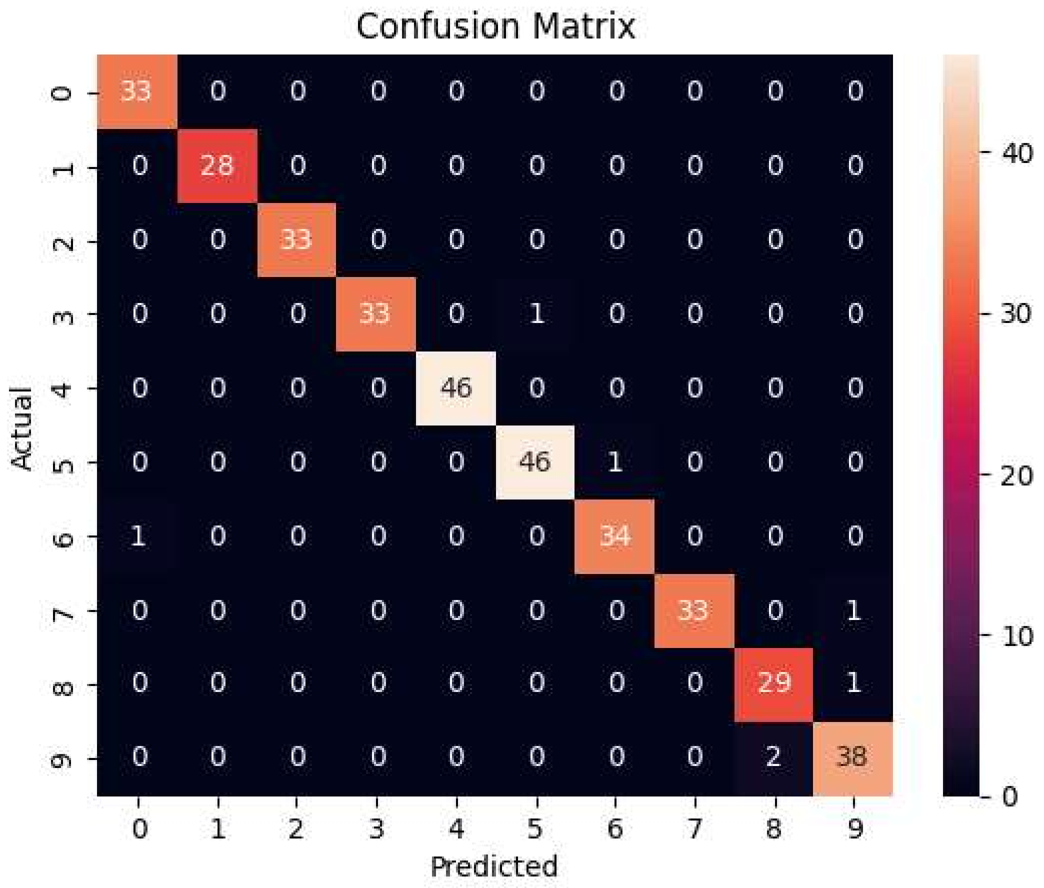
In [10]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [11]:
```
classifier = svm.SVC(gamma=0.001, C=10)  # C tuned a bit for better accuracy
classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9805555555555555
```

In [12]:
```
print("\nConfusion Matrix:")
print(metrics.confusion_matrix(y_test, y_pred))

sns.heatmap(metrics.confusion_matrix(y_test, y_pred), annot=True)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

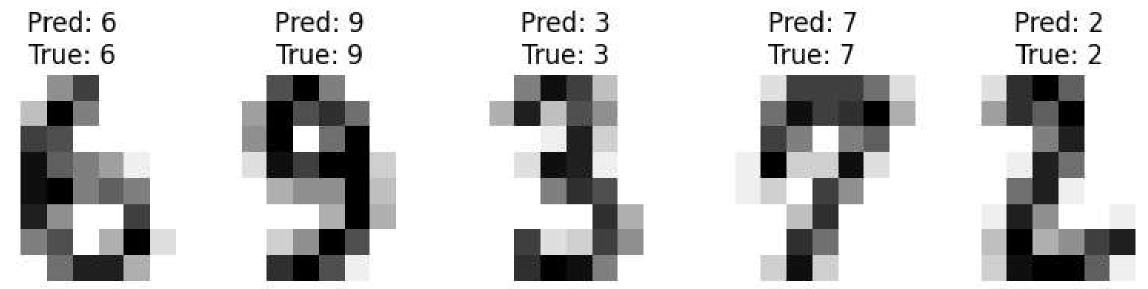
```
Confusion Matrix:
[[33  0  0  0  0  0  0  0  0  0]
 [ 0 28  0  0  0  0  0  0  0  0]
 [ 0  0 33  0  0  0  0  0  0  0]
 [ 0  0  0 33  0  1  0  0  0  0]
 [ 0  0  0  0 46  0  0  0  0  0]
 [ 0  0  0  0  0 46  1  0  0  0]
 [ 1  0  0  0  0  0 34  0  0  0]
 [ 0  0  0  0  0  0  0 33  0  1]
 [ 0  0  0  0  0  0  0  0 29  1]
 [ 0  0  0  0  0  0  0  0  2 38]]
```



In [13]:
```python
print("\nClassification Report:")
print(metrics.classification_report(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99        33
           1       1.00      1.00      1.00        28
           2       1.00      1.00      1.00        33
           3       1.00      0.97      0.99        34
           4       1.00      1.00      1.00        46
           5       0.98      0.98      0.98        47
           6       0.97      0.97      0.97        35
           7       1.00      0.97      0.99        34
           8       0.94      0.97      0.95        30
           9       0.95      0.95      0.95        40

    accuracy                           0.98       360
   macro avg       0.98      0.98      0.98       360
weighted avg       0.98      0.98      0.98       360
```

In [14]:
```python
images_and_predictions = list(zip(digits.images[idx_test], y_pred, y_test))

plt.figure(figsize=(10, 3))
for index, (image, pred, true) in enumerate(images_and_predictions[:5]):
    plt.subplot(1, 5, index + 1)
    plt.axis('off')
    plt.imshow(image, cmap=plt.cm.gray_r, interpolation='nearest')
    plt.title(f'Pred: {pred}\nTrue: {true}')
plt.show()
```



In [ ]: