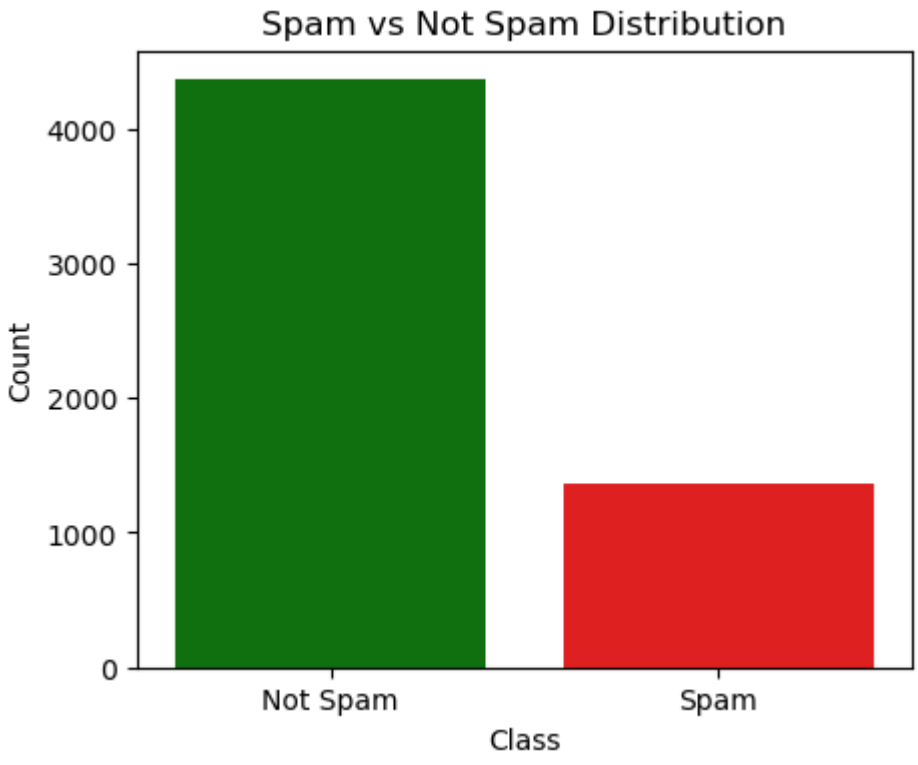
In [95]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
```

In [97]:
```
df = pd.read_csv("emails.csv")

# Rename column if needed
if 'spam_or_not' in df.columns:
    df.rename(columns={'spam_or_not': 'spam'}, inplace=True)

df.head()
```

Out[97]:

	text	spam
0	Subject: naturally irresistible your corporate...	1
1	Subject: the stock trading gunslinger fanny i...	1
2	Subject: unbelievable new homes made easy im ...	1
3	Subject: 4 color printing special request add...	1
4	Subject: do not have money , get software cds ...	1

In [99]:
```
plt.figure(figsize=(5,4))
sns.countplot(x='spam', data=df, hue='spam', palette=['green', 'red'], legend=False)
plt.xticks([0,1], ['Not Spam', 'Spam'])
plt.title("Spam vs Not Spam Distribution")
plt.xlabel("Class")
plt.ylabel("Count")
plt.show()
```



In [101…
```
X = df['text']
y = df['spam']
```

In [103…
```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=0
)
```

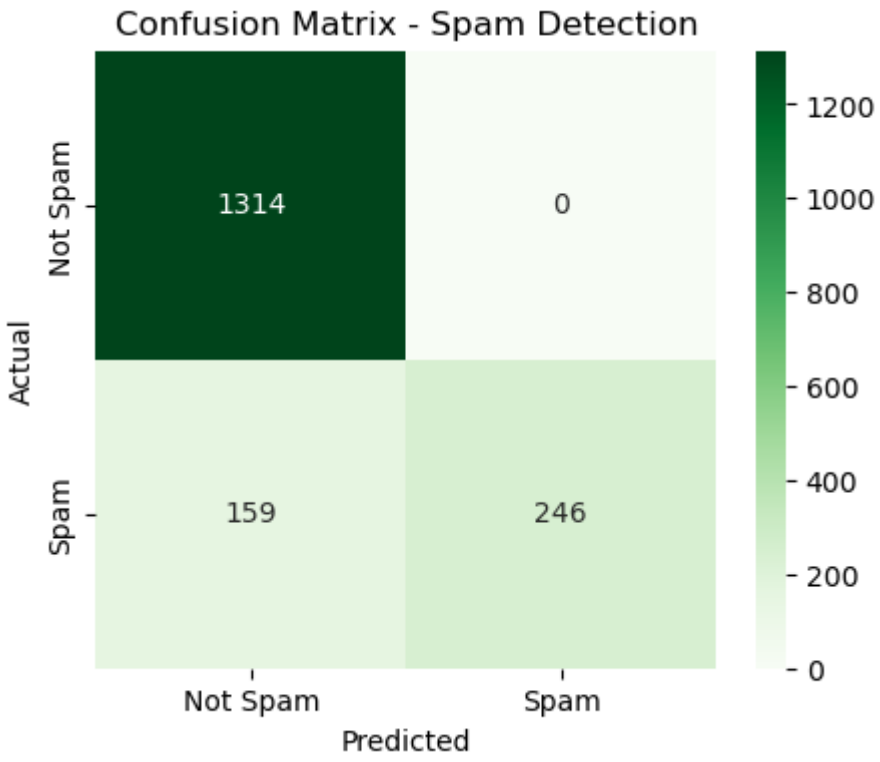
In [105…
```
vectorizer = TfidfVectorizer(stop_words='english')

X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)
```

In [106…
```
model = MultinomialNB()
model.fit(X_train_vec, y_train)

y_pred = model.predict(X_test_vec)
```

In [109…
```
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[1314    0]
 [ 159  246]]
```

In [111…
```
plt.figure(figsize=(5,4))
sns.heatmap(
    cm,
    annot=True,
    fmt='d',
    cmap='Greens',
    xticklabels=['Not Spam', 'Spam'],
    yticklabels=['Not Spam', 'Spam']
)

plt.title("Confusion Matrix - Spam Detection")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```



In [113…
```
TN, FP, FN, TP = cm.ravel()

print("True Positive (TP)  - Spam correctly predicted:", TP)
print("False Positive (FP) - Not Spam predicted as Spam:", FP)
```

```
print("True Negative (TN)  - Not Spam correctly predicted:", TN)
print("False Negative (FN) - Spam predicted as Not Spam:", FN)
```

```
True Positive (TP)  - Spam correctly predicted: 246
False Positive (FP) - Not Spam predicted as Spam: 0
True Negative (TN)  - Not Spam correctly predicted: 1314
False Negative (FN) - Spam predicted as Not Spam: 159
```

In [115…

```
accuracy = accuracy_score(y_test, y_pred)
error_rate = 1 - accuracy

print("\nAccuracy:", accuracy)
print("Error Rate:", error_rate)
```

```
Accuracy: 0.9075043630017452
Error Rate: 0.0924956369982548
```

In [117…

```
print("\nClassification Report:\n")
print(classification_report(
    y_test,
    y_pred,
    target_names=['Not Spam', 'Spam']
))
```

```
Classification Report:

              precision    recall  f1-score   support

    Not Spam       0.89      1.00      0.94      1314
        Spam       1.00      0.61      0.76       405

    accuracy                           0.91      1719
   macro avg       0.95      0.80      0.85      1719
weighted avg       0.92      0.91      0.90      1719
```

In [ ]: