In [31]:
```
#IMPLEMENT THE K-NEARESET NEIGHBOUR CLASSIFIER ON LABELLED DATASET AND EVALUATE PER
#24MCAR0188 FAHEEM ABDURAHIMAN SAIDALAVI
# Step 1: Import Libraries
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import matplotlib.pyplot as plt
import seaborn as sns  # For heatmap
```

In [3]:
```
# Step 2: Load dataset (Iris dataset)
iris = load_iris()
X = iris.data  # Features
y = iris.target  # Labels
```

In [5]:
```
# Convert to DataFrame for better display
df = pd.DataFrame(X, columns=iris.feature_names)
df['target'] = y
```

In [7]:
```
# Show dataset info
print("=== Dataset Information ===")
print(f"Shape: {df.shape[0]} rows, {df.shape[1]} columns")
print("\nColumn Names:", df.columns.tolist())
print("\nFirst 5 Rows:\n", df.head())
print("\nSummary Statistics:\n", df.describe())
```

```
=== Dataset Information ===
Shape: 150 rows, 5 columns

Column Names: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal
width (cm)', 'target']

First 5 Rows:
    sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   target
0       0
1       0
2       0
3       0
4       0

Summary Statistics:
        sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000
mean            5.843333          3.057333           3.758000
std             0.828066          0.435866           1.765298
min             4.300000          2.000000           1.000000
25%             5.100000          2.800000           1.600000
50%             5.800000          3.000000           4.350000
75%             6.400000          3.300000           5.100000
max             7.900000          4.400000           6.900000

       petal width (cm)      target
count        150.000000  150.000000
mean           1.199333    1.000000
std            0.762238    0.819232
min            0.100000    0.000000
25%            0.300000    0.000000
50%            1.300000    1.000000
75%            1.800000    2.000000
max            2.500000    2.000000
```

In [9]:
```
# Step 3: Split dataset into training & testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta
```

In [11]:
```
# Step 4: Initialize KNN model
knn = KNeighborsClassifier(n_neighbors=5)  # k=5
```

In [13]:
```
# Step 5: Train the model
knn.fit(X_train, y_train)
```

Out[13]:

▾ KNeighborsClassifier

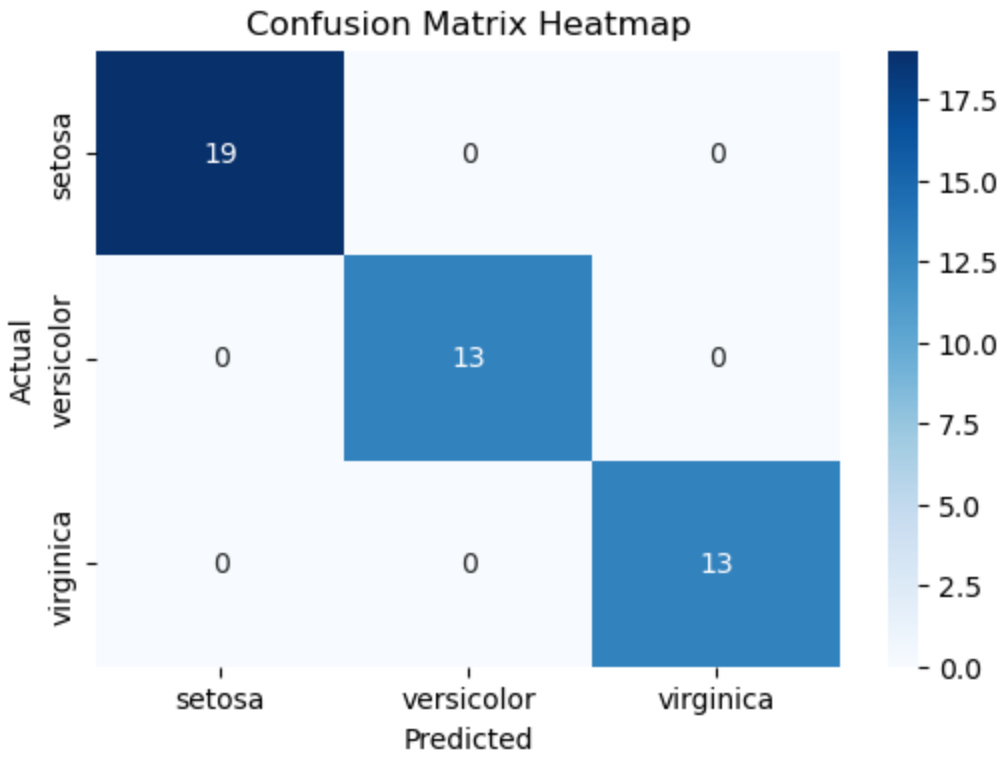
```
KNeighborsClassifier()
```

In [15]:
```
# Step 6: Make predictions
y_pred = knn.predict(X_test)
```

In [17]:
```
# Step 7: Evaluate performance
print("\n=== Model Performance ===")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred, target_na
```

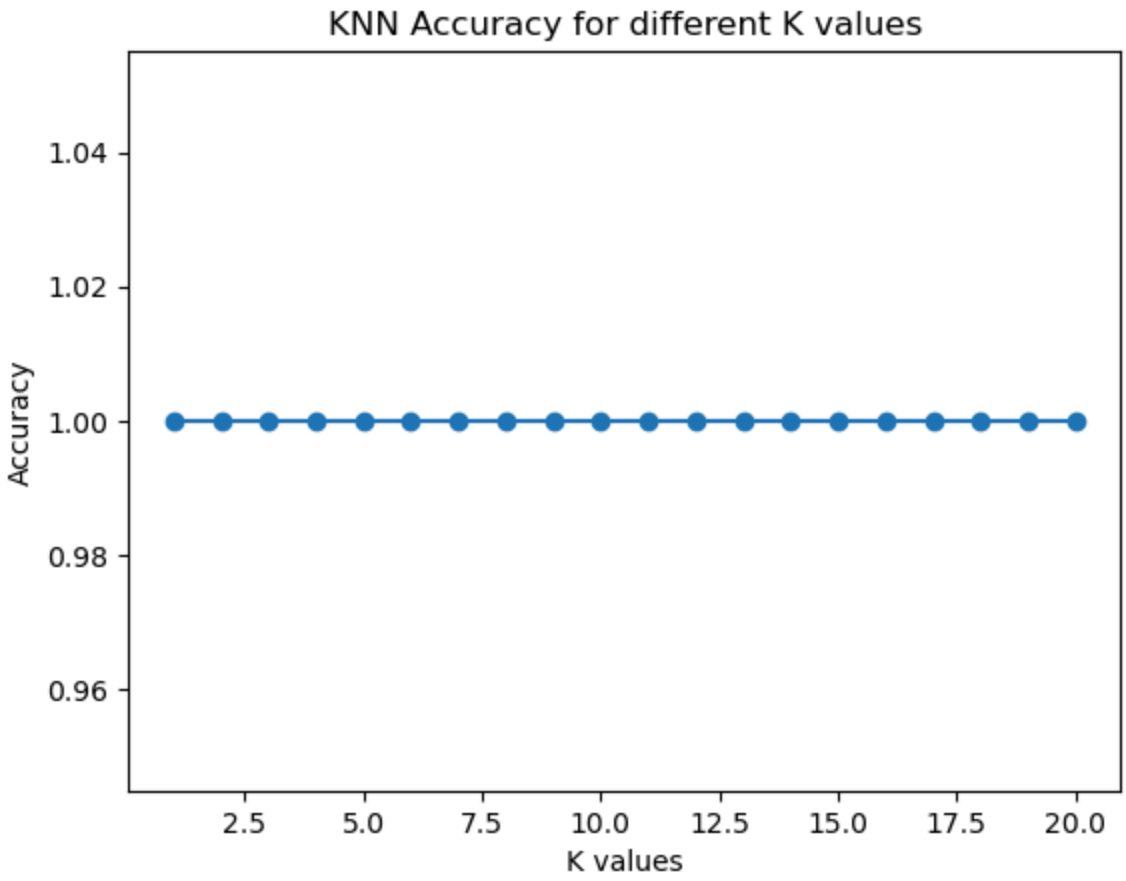
```
=== Model Performance ===
Accuracy: 1.0

Classification Report:
               precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        19
  versicolor       1.00      1.00      1.00        13
   virginica       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45
```

In [19]:
```
# Visualize Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=iris.target_names,
            yticklabels=iris.target_names)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
plt.show()
```



In [21]:
```python
# Step 8: Optional - visualize accuracy for different K values
accuracy_list = []
for k in range(1, 21):
    knn_temp = KNeighborsClassifier(n_neighbors=k)
    knn_temp.fit(X_train, y_train)
    y_temp_pred = knn_temp.predict(X_test)
    accuracy_list.append(accuracy_score(y_test, y_temp_pred))
```

In [23]:
```python
plt.plot(range(1, 21), accuracy_list, marker='o')
plt.xlabel('K values')
plt.ylabel('Accuracy')
plt.title("KNN Accuracy for different K values")
plt.show()
```



In [27]:
```
df.head()
```

Out[27]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

In [ ]: