In [1]:

```
import os
current_directory = os.getcwd()
print(current_directory)
```

```
C:\Users\kevin4tx\Saved Games\ICT
```

In [3]:

```
# Answer ti Question 1

import pandas as pd

# Read the csv data file in to dataframe name "df"
df = pd.read_csv('C:/Users/kevin4tx/Saved Games/ICT/fedexdatacoded.csv')

# Printing the first two rows
print(df.head(2))
```

```
   sizecode    age  statecode  typecode  categorycode  frequencycode  anrev  \
0         3  209.0         21         1             3              2   12.0   
1         3  209.0         21         1             3              2   12.0   

   dailyrev  dailyvol  channelcode  purposecode  phasecode  
0       0.1       0.4            0            2          1  
1       0.1       0.4            2            2          1  
```

In [15]:

```
df.shape
```

Out[15]:

```
(115526, 12)
```

In [17]:

```
# Define the file path for the CSV file
file_path = 'C:/Users/kevin4tx/Saved Games/ICT/fedexdatacodednonan.csv'

# Write the DataFrame to a CSV file
df.to_csv(file_path, index=False)  # Use index=False to exclude writing row numbers
```

In [72]:

```
df.columns
```

Out[72]:

```
Index(['id', 'size', 'account', 'city', 'state', 'record', 'status',
       'obstacles', 'outcome', 'risk', 'risk reason', 'age', 'date', 'type',
       'category', 'frequency', 'anrev', 'dailyrev', 'dailyvol', 'phase',
       'flags', 'country', 'channel', 'purpose', 'comments', 'date.1',
       'sizecode', 'statecode', 'typecode', 'categorycode', 'frequencycode',
       'phasecode', 'channelcode', 'purposecode'],
      dtype='object')
```

In [13]:

```
unique_values_count= df['phasecode'].nunique()
print(unique_values_count)

value_counts = df['phasecode'].value_counts()
print(value_counts)
```

```
4
phasecode
2    83172
3    17330
1    14960
0       64
Name: count, dtype: int64
```

In [68]:

```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
df['purposecode'] = label_encoder.fit_transform(df['purpose'])

print(df)
```

```
                 id    size                        account     city state  \
0       111334757.0  Medium  MJCELCO MEXICO, S DE RL DE CV  APODACA    NL   
1       111334757.0  Medium  MJCELCO MEXICO, S DE RL DE CV  APODACA    NL   
2       111334757.0  Medium  MJCELCO MEXICO, S DE RL DE CV  APODACA    NL   
3       111334757.0  Medium  MJCELCO MEXICO, S DE RL DE CV  APODACA    NL   
4       111334757.0  Medium  MJCELCO MEXICO, S DE RL DE CV  APODACA    NL   
...             ...     ...                            ...      ...   ...   
117621  997608755.0   Small     WDM WATER SYSTEMS SA DE CV   GARCIA    MX   
117622  997608755.0   Small     WDM WATER SYSTEMS SA DE CV   GARCIA    MX   
117623  997608755.0   Small     WDM WATER SYSTEMS SA DE CV   GARCIA    MX   
117624  997608755.0   Small     WDM WATER SYSTEMS SA DE CV   GARCIA    MX   
117625  997608755.0   Small     WDM WATER SYSTEMS SA DE CV   GARCIA    MX   

           record           status obstacles  \
0       O-6610205  SMA Prospecting       NaN   
1       O-6610205  SMA Prospecting       NaN   
2       O-6610205  SMA Prospecting       NaN   
3       O-6610205  SMA Prospecting       NaN   
4       O-6610205  SMA Prospecting       NaN   
...           ...              ...       ...   
117621  O-6247849              NaN       NaN   
117622  O-6833372              NaN       NaN   
117623  O-6833372              NaN       NaN   
117624  O-6833372              NaN       NaN   
117625  O-6833372              NaN       NaN   

                                                  outcome risk  ...  \
0                                                     NaN  NaN  ...   
1                                                     NaN  NaN  ...   
2                                                     NaN  NaN  ...   
3                                                     NaN  NaN  ...   
4                                                     NaN  NaN  ...   
...                                                   ...  ...  ...   
117621  ESTAN COMENZANDO CON ESA CUENTA Y NO HAY VOLUM...  NaN  ...   
117622                                          IPLW DESC  NaN  ...   
117623                                          IPLW DESC  NaN  ...   
117624                                          IPLW DESC  NaN  ...   
117625                                          IPLW DESC  NaN  ...   

                                                 comments     date.1 sizecode  \
0       VISITAMOS LA ADUANA OSCAR OSOSRIO, TOMAS HERNA...  4/26/2017        3   
1                                              ME COMENTA   5/3/2017        3   
2       HICIERON IPFS CON GUIA MANUAL, POR QUE EL SIST...   5/4/2017        3   
3       LO REVISO CON CRISTIAN PARA QUE SALGA HOY Y LE...   5/8/2017        3   
4       COMENTA TOMAS QUE YA REALIZO ENVÍO DE IPFS PER...  5/12/2017        3   
...                                                   ...        ...      ...   
117621  YA ESTA EN OFICNAS DE WDM TOTALMENTE SEPARADA ...   6/1/2018        5   
117622  COMENTA JOSE LUIS QUE ESTA PLANTA WDM A COMENZ...   3/9/2018        5   
117623  COMENZARAN A REALIZAR ENVIOS CON GTL AUN NO SA...  3/20/2018        5   
117624  COMENTA QUE NO TIENEN ESE VOLUMEN CON LA CUENT...  5/23/2018        5   
117625  YA ESTA EN OFICNAS DE WDM TOTALMENTE SEPARADA ...   6/1/2018        5   

       statecode typecode categorycode  frequencycode  phasecode  channelcode  \
0             21        1            3              2          1            0   
1             21        1            3              2          1            2   
2             21        1            3              2          1            1   
3             21        1            3              2          1            2   
4             21        1            3              2          1            2   
...          ...      ...          ...            ...        ...          ...   
117621        20        2            2              2          1            1   
```

```
117622       20      2            2             0         2           0
117623       20      2            2             0         2           0
117624       20      2            2             0         2           1
117625       20      2            2             0         2           1

       purposecode
0                2
1                2
2                2
3                2
4                2
...            ...
117621           2
117622           2
117623           2
117624           2
117625           2

[117626 rows x 34 columns]
```

In [74]:

```python
import pandas as pd

# List of 12 columns you want to write to the CSV file
columns_to_write = ['sizecode', 'age', 'statecode', 'typecode', 'categorycode', 'frequ
        'channelcode', 'purposecode','phasecode']

# Create a new DataFrame with only the selected columns
selected_columns_df = df[columns_to_write]

# Specify the file path for the CSV file
csv_file_path = 'fedexdatacoded.csv'

# Write the selected columns to a CSV file
selected_columns_df.to_csv(csv_file_path, index=False)
```

In [16]:

```python
# Answer ti Question 1

import pandas as pd

# Read the csv data file in to dataframe name "df"
#df1 = pd.read_csv('/home/60d01c96-d7b9-4609-a7df-e0974c21daa1/fedexdatacoded.csv')

# Printing the first two rows
#print(df1.head(2))


# List of 12 columns you want to consider
columns_to_check = ['sizecode', 'age', 'statecode', 'typecode', 'categorycode',
       'frequencycode', 'anrev', 'dailyrev', 'dailyvol', 'channelcode',
       'purposecode', 'phasecode']

# Use dropna to remove rows with NaN values in any of the specified columns
df = df.dropna(subset=columns_to_check)

print(df.head(2))

# Now, df contains only the rows without NaN values in the specified columns
```

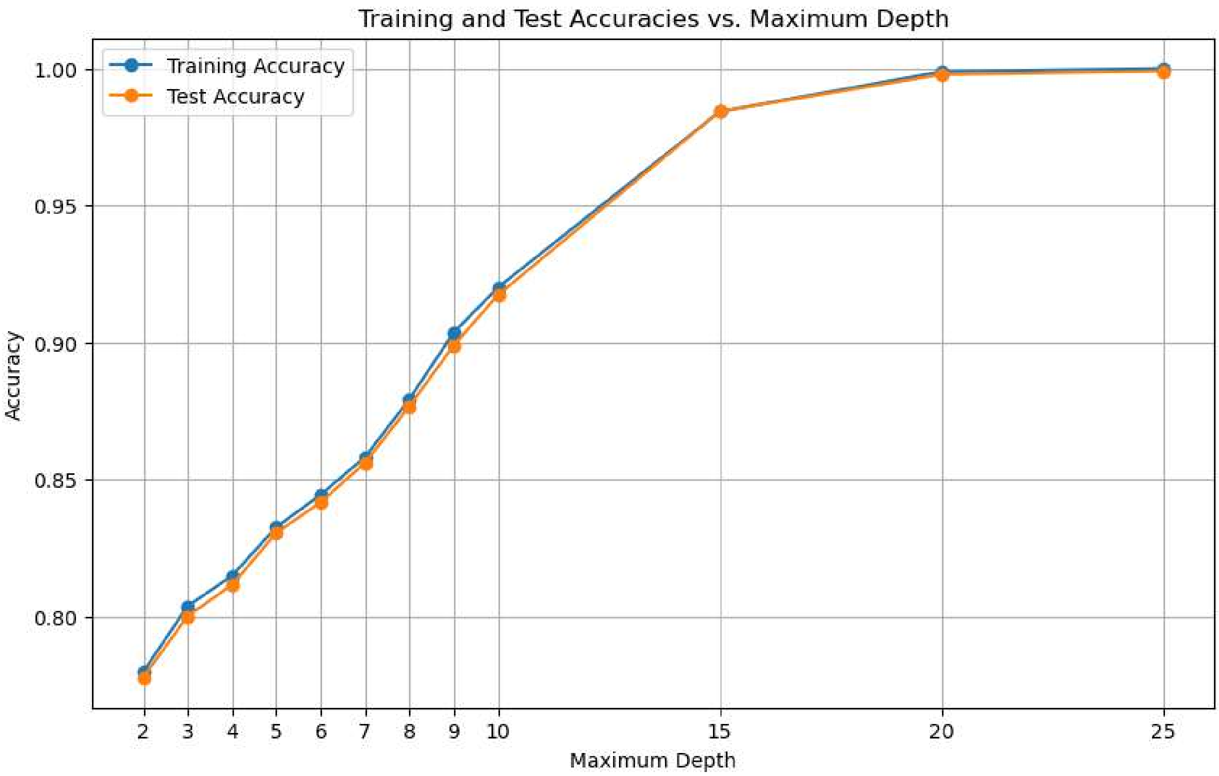
```
   sizecode    age  statecode  typecode  categorycode  frequencycode  anrev  \
0         3  209.0         21         1             3              2   12.0   
1         3  209.0         21         1             3              2   12.0   

   dailyrev  dailyvol  channelcode  purposecode  phasecode  
0       0.1       0.4            0            2          1  
1       0.1       0.4            2            2          1  
```

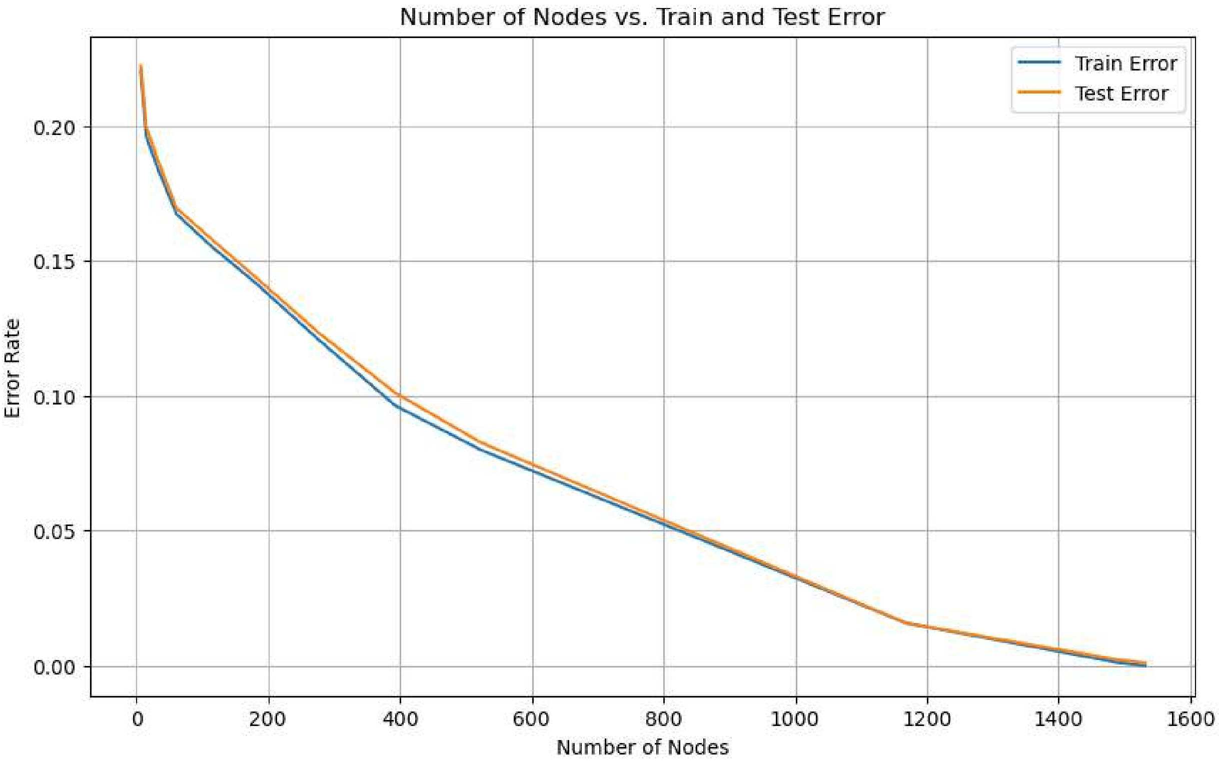
In [8]:

```python
# Answer to Question 1

import numpy as np
from sklearn.model_selection import train_test_split
import pandas as pd

# Split the data into a training set (80%) and a test set (20%)
X = df.drop(columns=['phasecode'])  # Features
# ID is dropped because this attribute is not relevant for our analysis
y = df['phasecode']  # Labels

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state

# Creating dataFrames for training and testing sets
train_data = pd.concat([X_train, y_train], axis=1)
test_data = pd.concat([X_test, y_test], axis=1)

# Export the training and testing sets to CSV files
train_data.to_csv('training.csv', index=False)
test_data.to_csv('testing.csv', index=False)
```

In [9]:

```python
# Answer to Question 2

import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# Load the training and testing sets that we exported in the previous step.
train_data = pd.read_csv('training.csv')
test_data = pd.read_csv('testing.csv')

# Separate features and labels for training and testing sets
X_train = train_data.drop(columns=['phasecode'])
y_train = train_data['phasecode']
X_test = test_data.drop(columns=['phasecode'])
y_test = test_data['phasecode']

# Defining the list of maximum depths to experiment with
max_depths = [2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25]

# Initializing lists to store training and test accuracy values and node counts
train_accuracies = []
test_accuracies = []
node_counts = []

# Fitting decision trees with different maximum depths and calculate accuracy values
for depth in max_depths:
    # Creating and fitting the decision tree classifier
    dt_classifier = DecisionTreeClassifier(criterion='entropy', max_depth=depth, rando
    dt_classifier.fit(X_train, y_train)
    # Predictting on training and test sets
    train_predictions = dt_classifier.predict(X_train)
    test_predictions = dt_classifier.predict(X_test)

    # Calculating training and test accuracy values
    train_accuracy = accuracy_score(y_train, train_predictions)
    test_accuracy = accuracy_score(y_test, test_predictions)

    # Append accuracies and node count
    train_accuracies.append(train_accuracy)
    test_accuracies.append(test_accuracy)
    node_counts.append(dt_classifier.tree_.node_count)

# Plotting training and test accuracy values vs. maximum depth
plt.figure(figsize=(10, 6))
plt.plot(max_depths, train_accuracies, label='Training Accuracy', marker='o')
plt.plot(max_depths, test_accuracies, label='Test Accuracy', marker='o')
plt.title('Training and Test Accuracies vs. Maximum Depth')
plt.xlabel('Maximum Depth')
plt.ylabel('Accuracy')
plt.xticks(max_depths)
plt.legend()
plt.grid(True)
plt.show()


# Plot number of nodes vs. test error and train error
plt.figure(figsize=(10, 6))
plt.plot(node_counts, 1 - np.array(train_accuracies), label='Train Error')
plt.plot(node_counts, 1 - np.array(test_accuracies), label='Test Error')
plt.xlabel('Number of Nodes')
plt.ylabel('Error Rate')
plt.title('Number of Nodes vs. Train and Test Error')
plt.legend()
plt.grid(True)
plt.show()
```



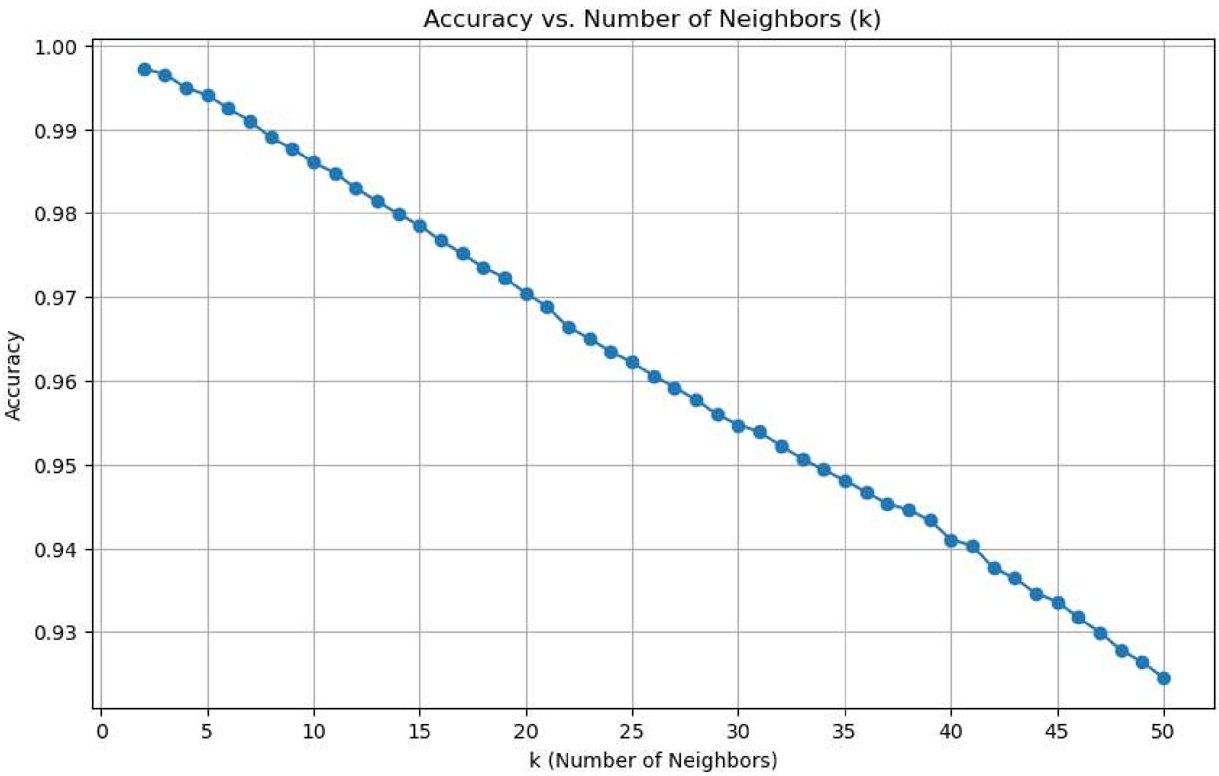


In [11]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt  # Import Matplotlib for plotting
from sklearn.model_selection import KFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

train_df = pd.read_csv('training.csv')
test_df = pd.read_csv('testing.csv')
```

```python
X_train = train_df.drop(columns=['phasecode'])
y_train = train_df['phasecode']

X_test = test_df.drop(columns=['phasecode'])
y_test = test_df['phasecode']

def knn_classification(k, X_train, y_train, X_test):
    knn_classifier = KNeighborsClassifier(n_neighbors=k)
    knn_classifier.fit(X_train, y_train)
    y_pred = knn_classifier.predict(X_test)
    return y_pred

kf = KFold(n_splits=5, shuffle=True, random_state=42)
k_values = list(range(2, 51))
best_k = None
best_accuracy = 0
accuracies = []  # List to store accuracy values

for k in k_values:
    fold_accuracies = []

    for train_index, val_index in kf.split(X_train):
        X_train_fold, X_val_fold = X_train.iloc[train_index], X_train.iloc[val_index]
        y_train_fold, y_val_fold = y_train.iloc[train_index], y_train.iloc[val_index]

        y_pred = knn_classification(k, X_train_fold, y_train_fold, X_val_fold)
        fold_accuracy = accuracy_score(y_val_fold, y_pred)
        fold_accuracies.append(fold_accuracy)

    mean_accuracy = np.mean(fold_accuracies)
    accuracies.append(mean_accuracy)  # Store mean accuracy for this k

    if mean_accuracy > best_accuracy:
        best_k = k
        best_accuracy = mean_accuracy

print(f"Best k value: {best_k}")


y_pred_test = knn_classification(best_k, X_train, y_train, X_test)
accuracy_test = accuracy_score(y_test, y_pred_test)
print(f"Accuracy on the test set with k = {best_k}: {accuracy_test}")

# Plot the accuracy values
plt.figure(figsize=(10, 6))
plt.plot(k_values, accuracies, marker='o', linestyle='-')
plt.title('Accuracy vs. Number of Neighbors (k)')
plt.xlabel('k (Number of Neighbors)')
plt.ylabel('Accuracy')
plt.xticks(np.arange(0, 51, step=5))  # Set x-axis ticks from 0 to 50 with a step of 5
plt.grid(True)
plt.show()

# Find and print the best k value
best_k = k_values[np.argmax(accuracies)]
print(f"Best k value: {best_k}")
```

```
Best k value: 2
Accuracy on the test set with k = 2: 0.9983986843244179
```



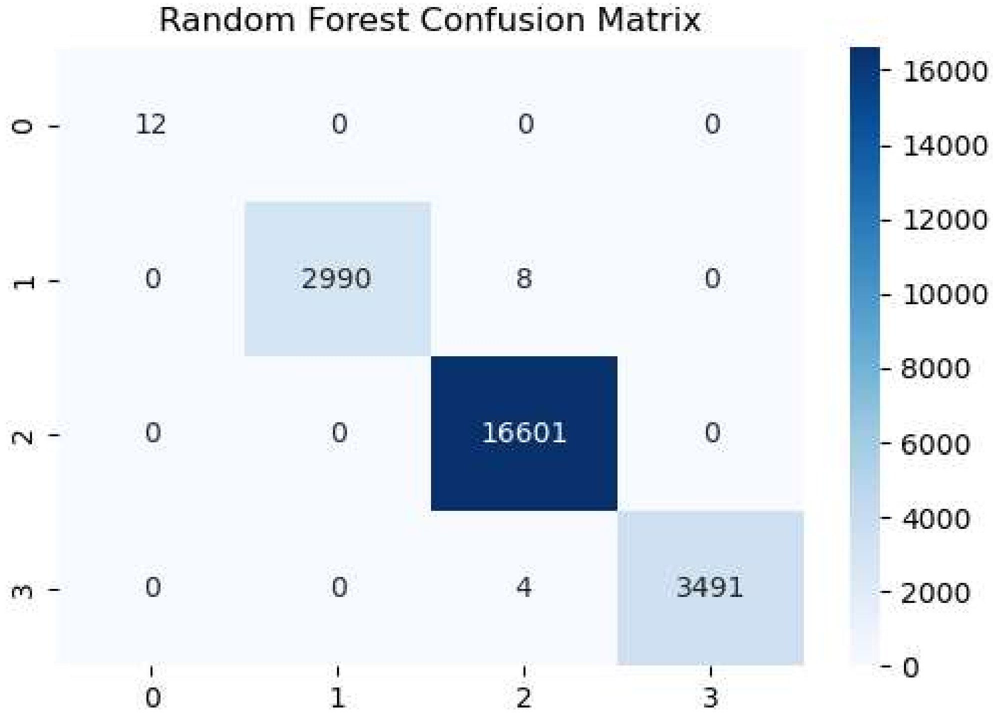
```
Best k value: 2
```

In [18]:

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Load your dataset (replace 'your_dataset.csv' with the actual filename)
data = pd.read_csv('C:/Users/kevin4tx/Saved Games/ICT/fedexdatacodednonan.csv')

# Split the dataset into features (X) and the target variable (y)
X = data.drop(columns=['phasecode'])  # Features
y = data['phasecode']  # Target variable

# Split the dataset into training (80%) and testing (20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=

# Training the Random Forest classifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
rf_classifier.fit(X_train, y_train)

# Predict with the Random Forest classifier
rf_predictions = rf_classifier.predict(X_test)

# Calculating the accuracy
rf_accuracy = accuracy_score(y_test, rf_predictions)

# Generating the confusion matrix
labels = [0, 1, 2, 3]  # Replace with your specific labels for phasecode
rf_cm = confusion_matrix(y_test, rf_predictions, labels=labels)

# Printing the accuracy
print(f'Random Forest Accuracy: {rf_accuracy}')
```

```
# Plotting the confusion matrix
plt.figure(figsize=(6, 4))
sns.heatmap(rf_cm, annot=True, fmt='d', cmap='Blues', xticklabels=labels, yticklabels=
plt.title('Random Forest Confusion Matrix')

plt.show()
```

```
Random Forest Accuracy: 0.9994806543754868
```


Random Forest Confusion Matrix