In [1]:
```
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.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.preprocessing import label_binarize
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
```

In [6]:
```
df = pd.read_csv(r"C:\Users\Admin\Downloads\diabetes_012_health_indicators_BRFSS2015.csv.zip"
df.head()
```

Out[6]:

	Diabetes_012	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	HeartDiseaseorAttack	PhysAct
0	0.0	1.0	1.0	1.0	40.0	1.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	25.0	1.0	0.0	0.0	
2	0.0	1.0	1.0	1.0	28.0	0.0	0.0	0.0	
3	0.0	1.0	0.0	1.0	27.0	0.0	0.0	0.0	
4	0.0	1.0	1.0	1.0	24.0	0.0	0.0	0.0	

5 rows × 22 columns

In [7]:
```
X = df.drop("Diabetes_012", axis=1)
y = df["Diabetes_012"]
```

In [8]:
```
X_train, X_test, y_train, y_test = train_test_split(
X, y, test_size=0.2, random_state=42, stratify=y
)
```

In [9]:
```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Logistic Regression

In [15]:
```
lr = LogisticRegression(max_iter=2000)
lr.fit(X_train_scaled, y_train)
pred_lr = lr.predict(X_test_scaled)
```

KNN

In [11]:
```
knn = KNeighborsClassifier(n_neighbors=7,weights='uniform')
knn.fit(X_train_scaled, y_train)
pred_knn = knn.predict(X_test_scaled)
```

RandomForestClassifier

In [12]:
```
rf = RandomForestClassifier(n_estimators=100, max_depth=8, random_state=42)
rf.fit(X_train, y_train)
```

```
pred_rf = rf.predict(X_test)
```

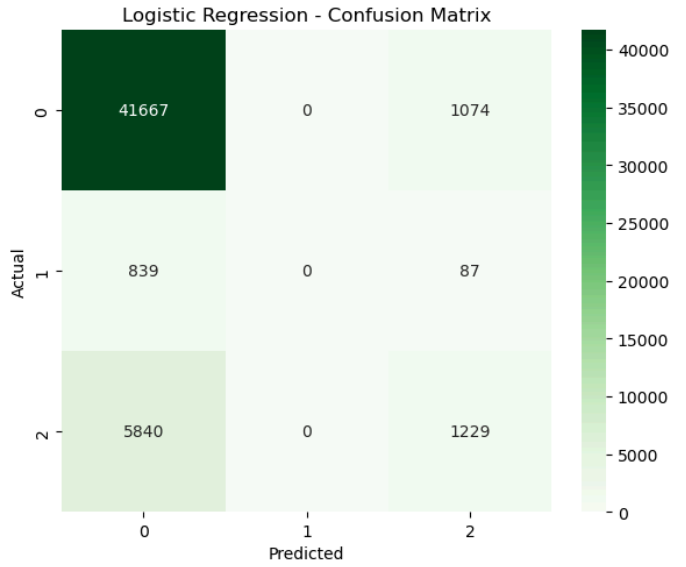
Decision Tree

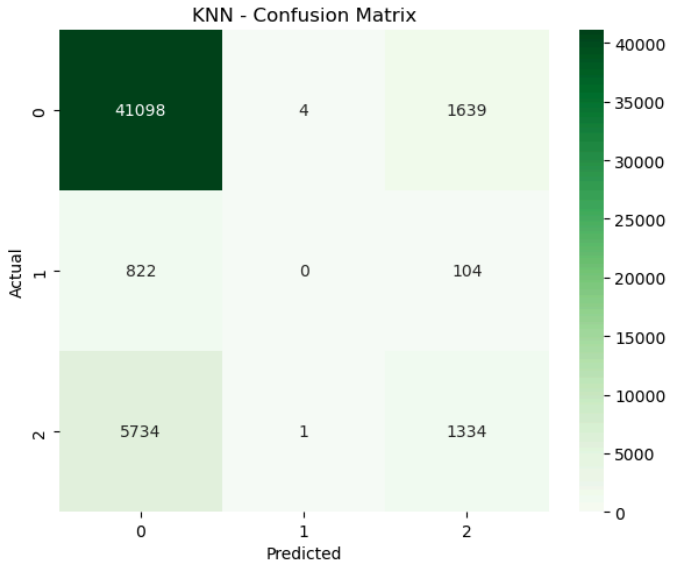
In [13]:
```
dt = DecisionTreeClassifier(max_depth=8, random_state=42)
dt.fit(X_train, y_train)
pred_dt = dt.predict(X_test)
```

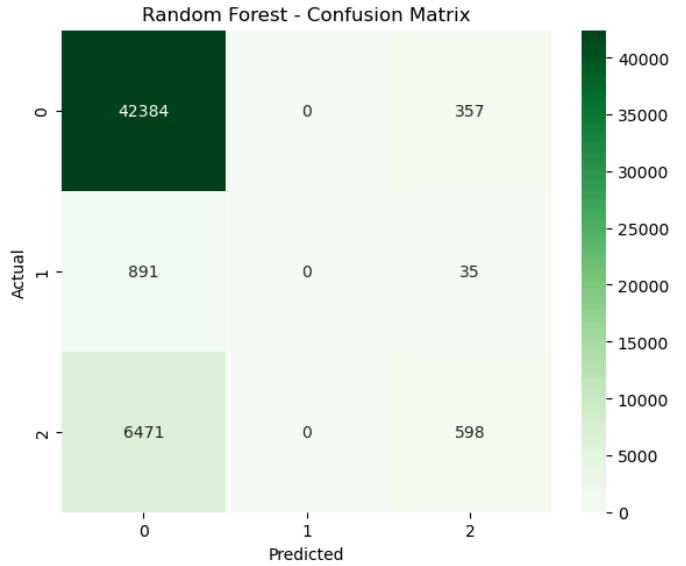
In [16]:
```
print("Logistic Regression Accuracy:", accuracy_score(y_test, pred_lr))
print("KNN Accuracy:", accuracy_score(y_test, pred_knn))
print("Random Forest Accuracy:", accuracy_score(y_test, pred_rf))
print("Decision Tree Accuracy:", accuracy_score(y_test, pred_dt))
```

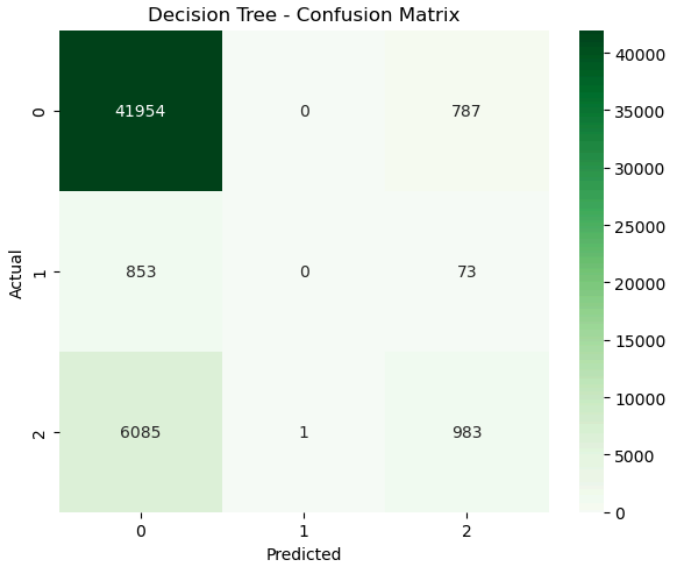
```
Logistic Regression Accuracy: 0.8454746136865342
KNN Accuracy: 0.836329233680227
Random Forest Accuracy: 0.8471696625670135
Decision Tree Accuracy: 0.8462827183853674
```

In [21]:
```
models = {
"Logistic Regression": pred_lr,
"KNN": pred_knn,
"Random Forest": pred_rf,
"Decision Tree": pred_dt
}
plt.figure(figsize=(12, 10))
for i, (name, preds) in enumerate(models.items(), 1):
    plt.subplot(2, 2, i)
    cm = confusion_matrix(y_test, preds)
    sns.heatmap(cm, annot=True, fmt="d", cmap="Greens")
    plt.title(f"{name} - Confusion Matrix")
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
plt.tight_layout()
plt.show()
```


Logistic Regression - Confusion Matrix
41667
0
1074
839
0
87
5840
0
1229
Actual
Predicted
0
1
2
40000
35000
30000
25000
20000
15000
10000
5000
0


KNN - Confusion Matrix
41098
4
1639
822
0
104
5734
1
1334
Actual
Predicted
0
1
2
40000
35000
30000
25000
20000
15000
10000
5000
0


Random Forest - Confusion Matrix
42384
0
357
891
0
35
6471
0
598
Actual
Predicted
0
1
2
40000
35000
30000
25000
20000
15000
10000
5000
0


Decision Tree - Confusion Matrix
41954
0
787
853
0
73
6085
1
983
Actual
Predicted
0
1
2
40000
35000
30000
25000
20000
15000
10000
5000
0

In [ ]: