In [1]:
```
import pandas as pd

# Load dataset
df = pd.read_csv("retail_sales_clean.csv", parse_dates=['order_date'])

# Quick check
print(df.shape)
print(df.head())

# Check for nulls or duplicates
print(df.isnull().sum())
print("Duplicates:", df.duplicated().sum())
```

```
(50000, 16)
    order_id order_date product_id  product_name category  qty  unit_price  \
0  ORD100000 2021-04-13      P1268  Cushion Cover     Home   15        6.84
1  ORD100001 2021-12-15      P6649    Flower Vase     Home   16       87.10
2  ORD100002 2021-09-28      P4078    Photo Album    Decor   11       31.26
3  ORD100003 2021-04-17      P5683    Photo Album    Decor   17       35.36
4  ORD100004 2021-03-13      P2224     Candle Set     Home    6       18.26

   cost_price  profit  profit_margin region  stock_days  customer_id country  \
0        5.62   18.30          0.178  North          89         8725     USA
1       61.19  414.56          0.297   West          66         4697  France
2       25.85   59.51          0.173  South          81         2891  Canada
3       26.08  157.76          0.262  North          73         8634  France
4       14.46   22.80          0.208   West          79         1677  France

   month day_of_week
0      4     Tuesday
1     12   Wednesday
2      9     Tuesday
3      4    Saturday
4      3    Saturday
order_id         0
order_date       0
product_id       0
product_name     0
category         0
qty              0
unit_price       0
cost_price       0
profit           0
profit_margin    0
region           0
stock_days       0
customer_id      0
country          0
month            0
day_of_week      0
dtype: int64
Duplicates: 0
```

In [2]:
```
#Compute Key KPIs
df['sales'] = df['unit_price'] * df['qty']

total_sales = df['sales'].sum()
total_profit = df['profit'].sum()
avg_margin = df['profit_margin'].mean()
```

```
avg_stock_days = df['stock_days'].mean()
unique_products = df['product_id'].nunique()
unique_customers = df['customer_id'].nunique()

print("Total Sales:", round(total_sales,2))
print("Total Profit:", round(total_profit,2))
print("Average Profit Margin:", round(avg_margin,3))
print("Average Stock Days:", round(avg_stock_days,2))
print("Unique Products:", unique_products)
print("Unique Customers:", unique_customers)
```

```
Total Sales: 25626783.16
Total Profit: 5766510.18
Average Profit Margin: 0.225
Average Stock Days: 45.93
Unique Products: 8973
Unique Customers: 8965
```

In [3]:
```
#Category-Level Analysis
category_agg = df.groupby('category').agg(
    total_profit=('profit', 'sum'),
    total_sales=('sales', 'sum'),
    total_qty=('qty', 'sum'),
    avg_margin=('profit_margin', 'mean'),
    avg_stock_days=('stock_days', 'mean')
).reset_index().sort_values('total_profit', ascending=False)

print(category_agg)
```

```
      category  total_profit  total_sales  total_qty  avg_margin  \
2         Home    2601288.27  11550517.65     225608    0.225146
1  Electronics     852501.51   3792334.84      74203    0.224784
4   Stationery     585159.15   2603790.60      50603    0.225000
3      Kitchen     580921.80   2578688.11      50119    0.225521
0        Decor     576525.75   2569645.75      50960    0.224264
5         Toys     570113.70   2531806.21      49513    0.225026

   avg_stock_days
2       46.020585
1       45.885560
4       45.929724
3       46.066868
0       45.677285
5       45.681336
```

In [4]:
```
#Product-Level Analysis
prod = df.groupby(['product_id','product_name','category']).agg(
    total_orders=('order_id','nunique'),
    total_qty=('qty','sum'),
    total_sales=('sales','sum'),
    total_profit=('profit','sum'),
    avg_stock_days=('stock_days','mean'),
    avg_margin=('profit_margin','mean')
).reset_index().sort_values('total_profit', ascending=False)

print(prod.head(10))
```

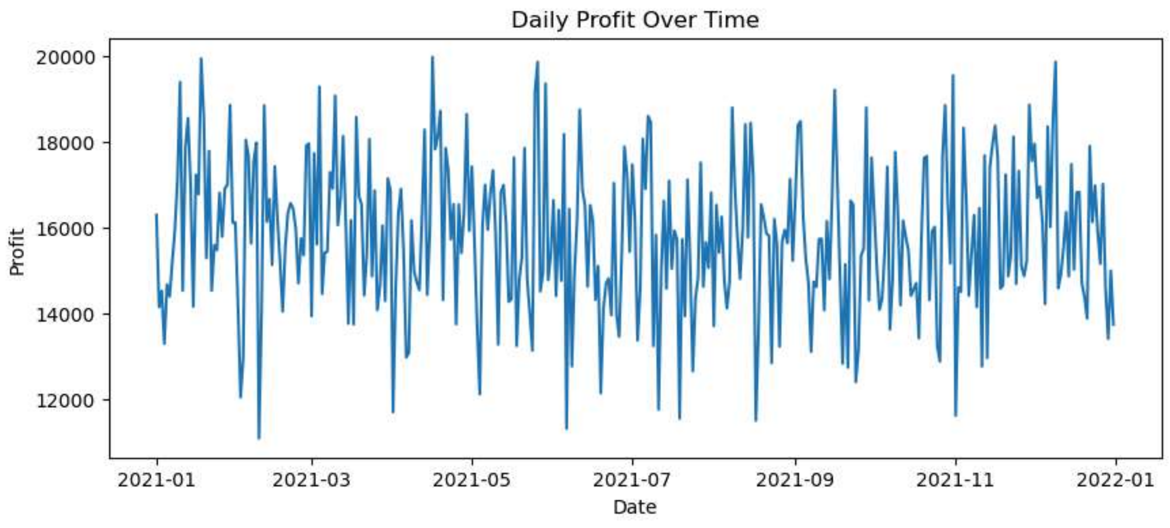
```
      product_id    product_name    category  total_orders  total_qty  \
14240      P3968        LED Lamp  Electronics             4         62
1005       P1208  Desk Organizer  Stationery             2         34
39060      P9095     Flower Vase        Home             3         57
18039      P4767    Cushion Cover        Home             4         49
7780       P2636        Gift Bag        Home             3         49
35050      P8274  Scented Candle        Home             3         43
21434      P5473      Candle Set        Home             3         43
9448       P2974      Wall Clock  Electronics             4         54
12973      P3694       White Mug     Kitchen             3         42
34270      P8118   Handmade Soap        Home             2         37

       total_sales  total_profit  avg_stock_days  avg_margin
14240      4262.27        981.13       47.750000    0.236500
1005       3335.67        952.38       32.000000    0.284000
39060      4379.88        948.10       21.000000    0.213667
18039      4089.73        943.99       34.000000    0.223250
7780       3914.55        925.58       41.666667    0.232333
35050      3976.14        903.60       41.333333    0.235667
21434      3530.82        886.30       58.666667    0.246333
9448       3452.55        861.91       58.500000    0.231750
12973      3175.59        822.24       50.000000    0.253333
34270      3001.55        819.40       31.000000    0.274000
```

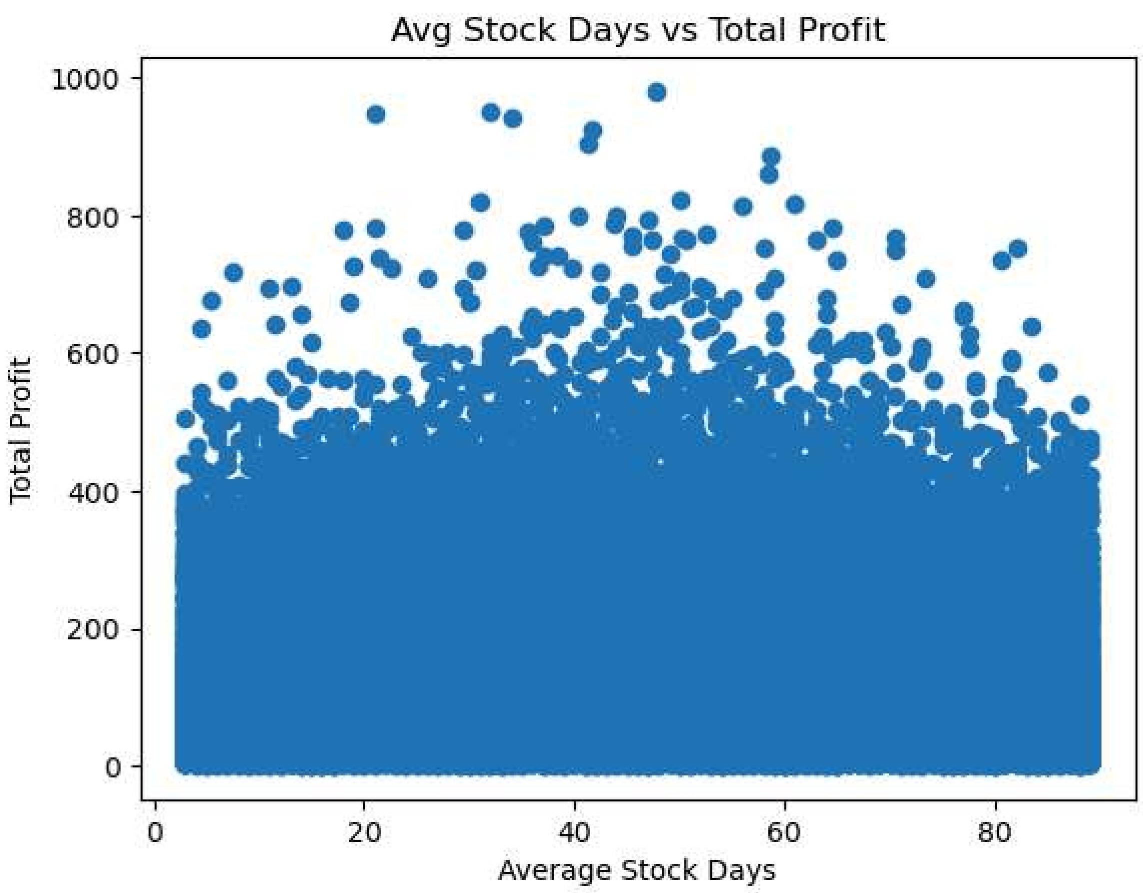
In [5]:
```python
#Time-Series Analysis (Daily Summary)
daily = df.groupby('order_date').agg(
    daily_sales=('sales','sum'),
    daily_profit=('profit','sum'),
    daily_orders=('order_id','nunique')
).reset_index()

import matplotlib.pyplot as plt

plt.figure(figsize=(10,4))
plt.plot(daily['order_date'], daily['daily_profit'])
plt.title('Daily Profit Over Time')
plt.xlabel('Date')
plt.ylabel('Profit')
plt.show()
```



In [6]:
```python
#Correlation & Scatterplots
plt.scatter(prod['avg_stock_days'], prod['total_profit'])
plt.title("Avg Stock Days vs Total Profit")
plt.xlabel("Average Stock Days")
plt.ylabel("Total Profit")
plt.show()
```



In [7]:
```python
# Predictive Modeling
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np

X = prod[['avg_stock_days','total_qty','avg_margin']].fillna(0)
y = prod['total_profit']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
preds = model.predict(X_test)

mae = mean_absolute_error(y_test, preds)
rmse = np.sqrt(mean_squared_error(y_test, preds))

print("MAE:", mae)
print("RMSE:", rmse)
```

```
MAE: 64.18892286245664
RMSE: 88.5054851500455
```

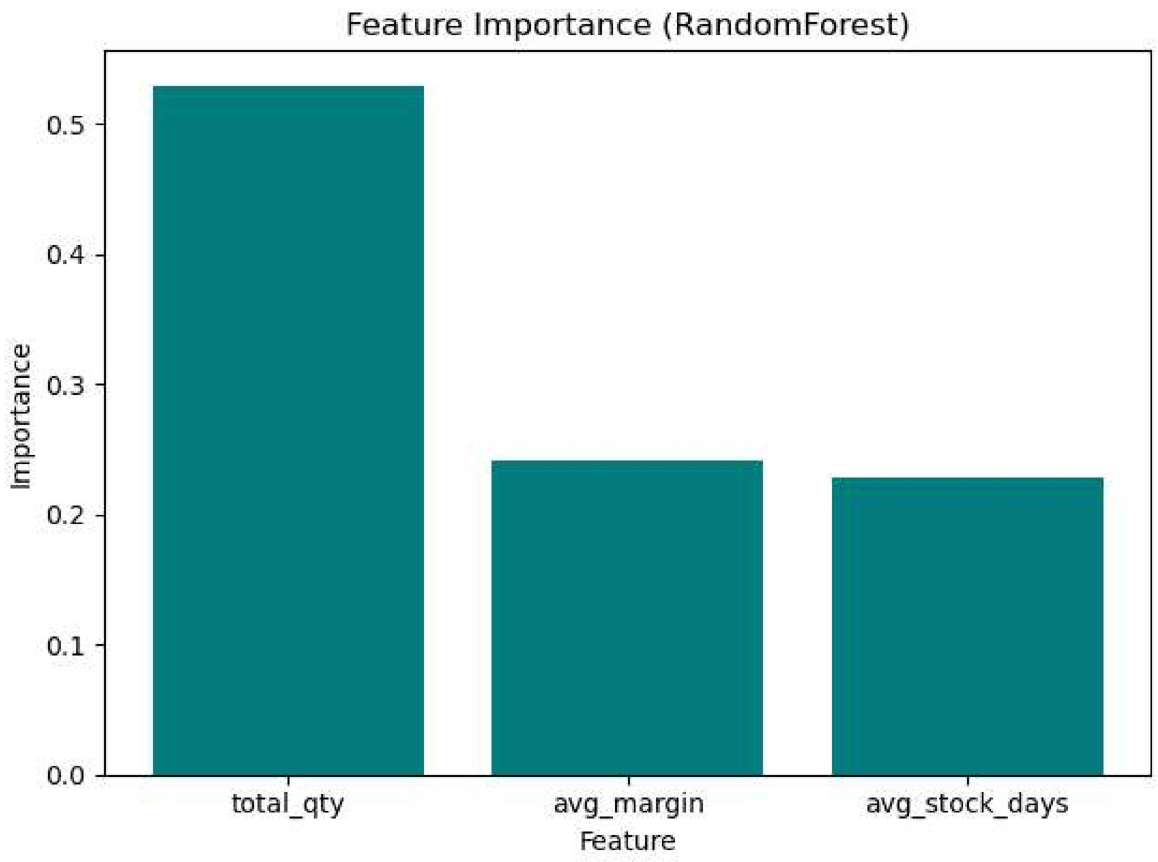
In [8]:
```python
#visualize which features contribute most
import matplotlib.pyplot as plt
import pandas as pd

# Get feature importances
fi = pd.DataFrame({
    'Feature': X.columns,
    'Importance': model.feature_importances_
}).sort_values('Importance', ascending=False)

# Display DataFrame
print("\nFeature Importance:")
print(fi)

# Plot
plt.bar(fi['Feature'], fi['Importance'], color='teal')
plt.title('Feature Importance (RandomForest)')
plt.xlabel('Feature')
plt.ylabel('Importance')
plt.tight_layout()
plt.show()
```

```
Feature Importance:
          Feature  Importance
1       total_qty    0.529512
2      avg_margin    0.241594
0  avg_stock_days    0.228894
```



```
In [ ]:
```