PROJECT : MDS Business Analytics using Python, Power BI

Made by : Abhishek Goyal (Abhi)

Good Morning Everyone!! Hope you all are doing well.

Here is a project I made. I undertook a detailed Python, Power BI business analytics project focused on analyzing pharmaceutical data to derive insights that would support the Medicines Development and Supply (MDS) organization. The project's primary objectives were to assist in the :

1. Collation and analysis of data
2. Leverage data visualization
3. Create dynamic business dashboards to communicate key business insights effectively.

Note - Data used is not real data. I wanted the project to be realistic such that I can present real scenarious to you to draw common grounds.I have also attached the CSV format data file and Power BI dasboard file.

In [179]:
```
"""Importing necessary modules and combining data files received
from different departments into one file which will further """

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

files = [
    'Medicines_Cardiovascular_Data.csv',
    'Medicines_Diabetes_Data.csv',
    'Medicines_Infectious_Diseases_Data.csv',
    'Medicines_Neurology_Data.csv',
    'Medicines_Oncology_Data.csv'
]
combined_df = pd.concat((pd.read_csv(f) for f in files), ignore_index=
combined_df.to_csv('Medicines_Development&Supply__Data.csv', index=Fal
```

In [180]:
```
data0 = pd.read_csv("Medicines_Development&Supply__Data.csv")
data0
```

Out[180]:

	Drug_ID	Drug_Name	Phase	Therapeutic Area	Start_Date	End_Date	Regulatory Status	Re
0	8	Ceftriaxone	Phase II	Cardiovascular	2015-07-30	2016-02-18	Approved	
1	10	Clindamycin	Approval	Cardiovascular	2015-09-28	2016-04-12	Pending	
2	11	Clopidogrel	Phase III	Cardiovascular	2015-10-28	2016-05-14	Rejected	
3	14	Dexamethasone	Phase I	Cardiovascular	2016-01-26	2016-09-05	Rejected	
4	15	Diazepam	Phase III	Cardiovascular	2016-02-25	2016-03-30	Approved	
5	17	Digoxin	Phase III	Cardiovascular	2016-04-	2016-11-	Pending	

In [181]:
```
# Sorting data by Drug Id and then re - setting index
data0_sorted = data0.sort_values(by='Drug_ID', ascending=True)
data0_sorted = data0_sorted.reset_index(drop=True)
data0_sorted.index = data0_sorted.index + 1
data0_sorted
data = data0_sorted
data.head()
```

Out[181]:

	Drug_ID	Drug_Name	Phase	Therapeutic Area	Start_Date	End_Date	Regulatory Status	Research Site
1	1	Abacavir	Phase III	Neurology	2015-01-01	2015-09-26	Rejected	China
2	2	Acetaminophen	Approval	Neurology	2015-01-31	2015-05-16	Approved	Germany
3	3	Albuterol	Phase II	Neurology	2015-03-02	2015-04-09	Pending	Germany
4	4	Amlodipine	Approval	Infectious Diseases	2015-04-01	2015-07-13	Approved	Brazil
5	5	Amoxicillin	Approval	Diabetes	2015-05-01	2016-02-07	Rejected	USA

In [182]:
```
""" Converting 'Start_Date', 'End_Date', 'Patent Expiry Date', 'Develo
the DataFrame to datetime format, and calculates the duration of devel
the Start_Date from the End_Date for each entry."""


data['Start_Date'] = pd.to_datetime(data['Start_Date'])
data['End_Date'] = pd.to_datetime(data['End_Date'])
data['Patent Expiry Date'] = pd.to_datetime(data['Patent Expiry Date']
data['Development_Duration'] = (data['End_Date'] - data['Start_Date'])
```

In [183]:
```
""" This code aggregates the data by the Phase of a pharmaceutical stu
millions of USD (Cost_Million_USD) and the average number of patients
each phase. """

summary = data.groupby('Phase').agg({
    'Cost_Million_USD': 'mean',
    'Number of Patients Enrolled': 'mean'
}).reset_index()
summary
```

Out[183]:

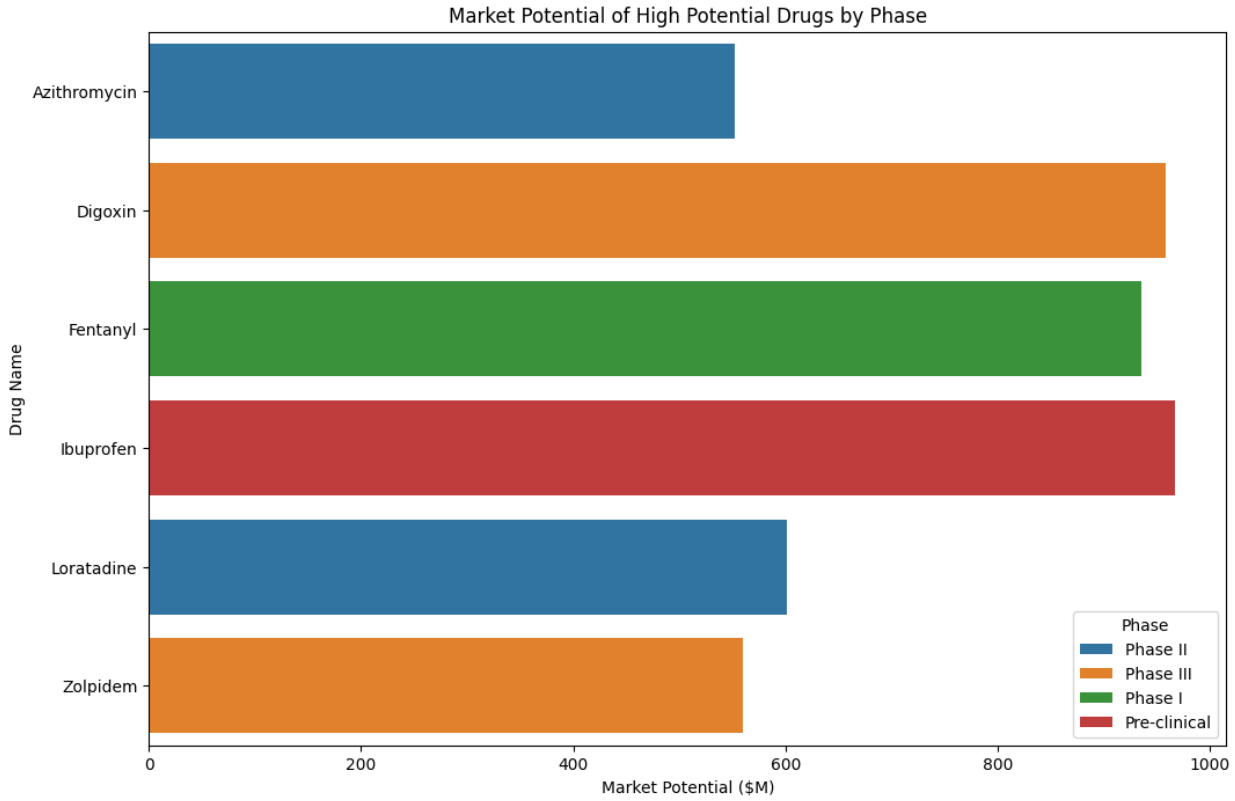
	Phase	Cost_Million_USD	Number of Patients Enrolled
0	Approval	103.542738	6382.000000
1	Phase I	108.740599	4216.200000
2	Phase II	81.820645	5428.700000
3	Phase III	120.471740	5224.307692
4	Pre-clinical	89.147004	5552.142857

In [137]:
```
# This code identifies drugs with pending regulatory approval and mark

high_potential_drugs = data[(data['Regulatory Status'] == 'Pending') &
high_potential_drugs[['Drug_Name', 'Phase', 'Market Potential $M']]
```

Out[137]:

	Drug_Name	Phase	Market Potential $M
7	Azithromycin	Phase II	552.37
17	Digoxin	Phase III	958.54
21	Fentanyl	Phase I	935.49
24	Ibuprofen	Pre-clinical	967.26
31	Loratadine	Phase II	601.12
50	Zolpidem	Phase III	560.21

In [86]:
```
plt.figure(figsize=(12, 8))
sns.barplot(x='Market Potential $M', y='Drug_Name', data=high_potentia
plt.title('Market Potential of High Potential Drugs by Phase')
plt.xlabel('Market Potential ($M)')
plt.ylabel('Drug Name')
plt.legend(title='Phase')
plt.show()
```

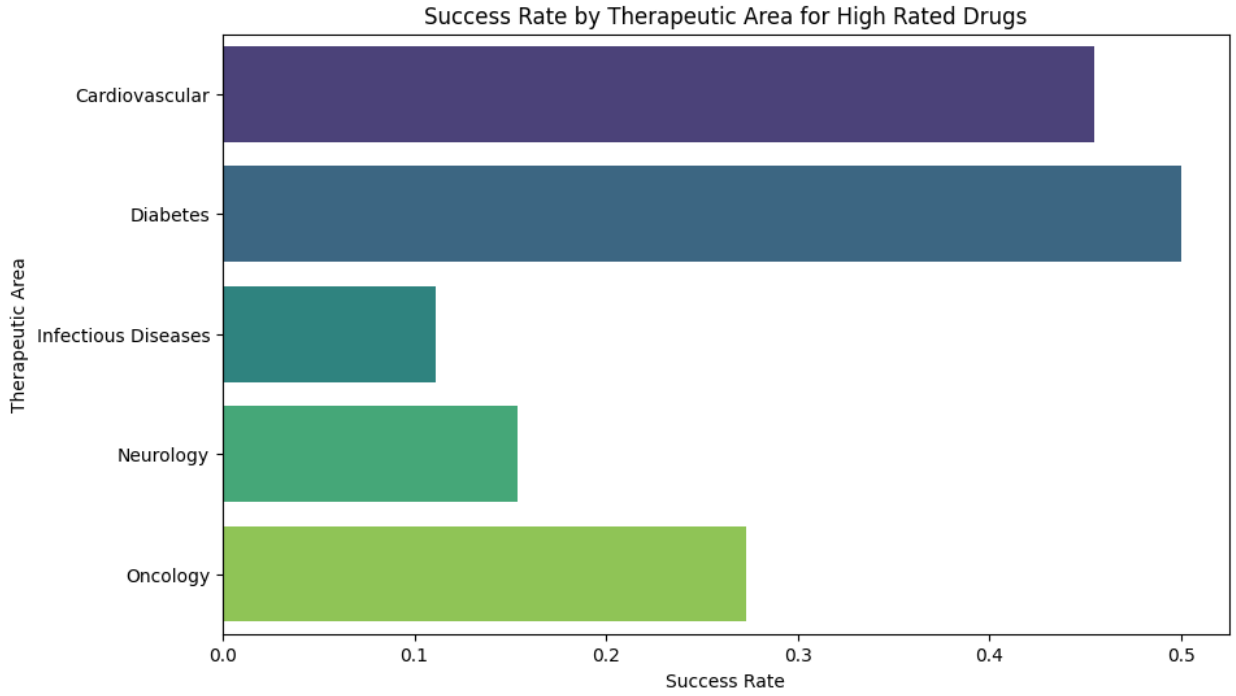


In [139]:
```
# This code calculates the percentage of high-success-rate drugs by th

success_rate_summary = data.groupby('Therapeutic Area').apply(lambda x
                                                              ).reset_i
success_rate_summary
```

Out[139]:

	Therapeutic Area	Success_Rate
0	Cardiovascular	0.454545
1	Diabetes	0.500000
2	Infectious Diseases	0.111111
3	Neurology	0.153846
4	Oncology	0.272727

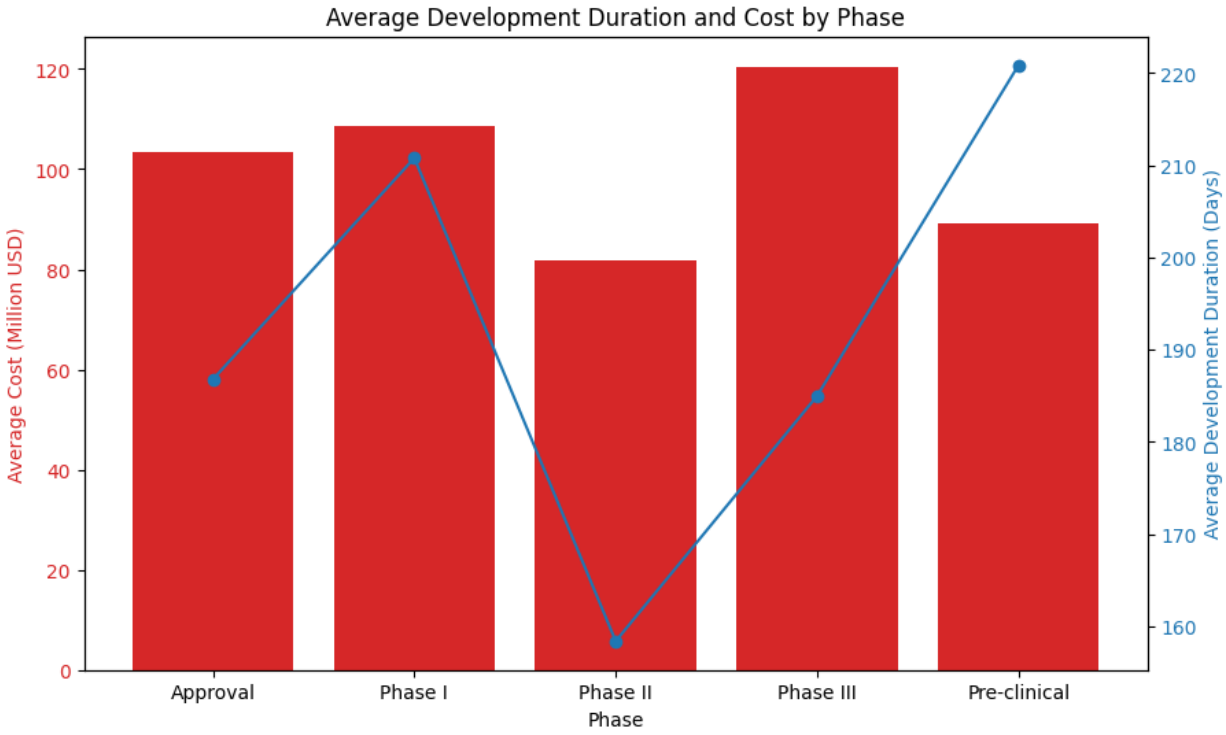
In [88]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x='Success_Rate', y='Therapeutic Area', data=success_rate_
plt.title('Success Rate by Therapeutic Area for High Rated Drugs')
plt.xlabel('Success Rate')
plt.ylabel('Therapeutic Area')
plt.show()
```



In [141]:
```
# Calculating the development duration in days
data['Development_Duration_Days'] = (data['End_Date'] - data['Start_Da

# Calculate average development time and cost by phase
phase_time_cost = data.groupby('Phase').agg({
    'Development_Duration_Days': 'mean',  # Average development durati
    'Cost_Million_USD': 'mean'  # Average cost
}).reset_index()

# Check the DataFrame to ensure calculations are correct
print(phase_time_cost)

# Create the visualization
fig, ax1 = plt.subplots(figsize=(10, 6))

# Plotting cost
color = 'tab:red'
ax1.set_xlabel('Phase')
ax1.set_ylabel('Average Cost (Million USD)', color=color)
ax1.bar(phase_time_cost['Phase'], phase_time_cost['Cost_Million_USD'],
ax1.tick_params(axis='y', labelcolor=color)
```

```
# Create a twin Axes sharing the x-axis for development duration
ax2 = ax1.twinx()
color = 'tab:blue'
ax2.set_ylabel('Average Development Duration (Days)', color=color)
ax2.plot(phase_time_cost['Phase'], phase_time_cost['Development_Durati
ax2.tick_params(axis='y', labelcolor=color)

plt.title('Average Development Duration and Cost by Phase')
plt.show()
```

	Phase	Development_Duration_Days	Cost_Million_USD
0	Approval	186.800000	103.542738
1	Phase I	210.800000	108.740599
2	Phase II	158.400000	81.820645
3	Phase III	185.000000	120.471740
4	Pre-clinical	220.857143	89.147004

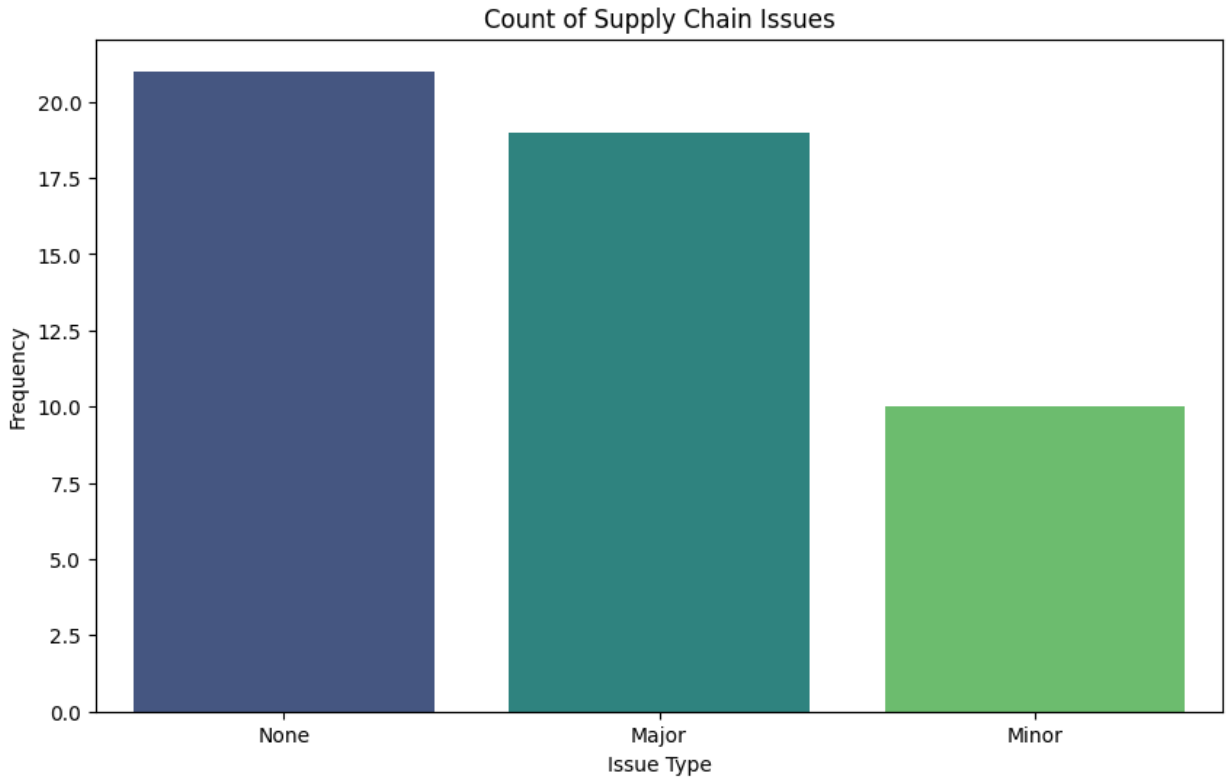


In [142]:
```
# This code summarizes the frequency of different supply chain issues

supply_chain_vulnerabilities = data['Supply_Chain_Issues'].value_count
supply_chain_vulnerabilities
```

Out[142]:

	index	Count
0	None	21
1	Major	19
2	Minor	10

In [91]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x='index', y='Count', data=supply_chain_vulnerabilities, p
plt.title('Count of Supply Chain Issues')
plt.xlabel('Issue Type')
plt.ylabel('Frequency')
plt.show()
```

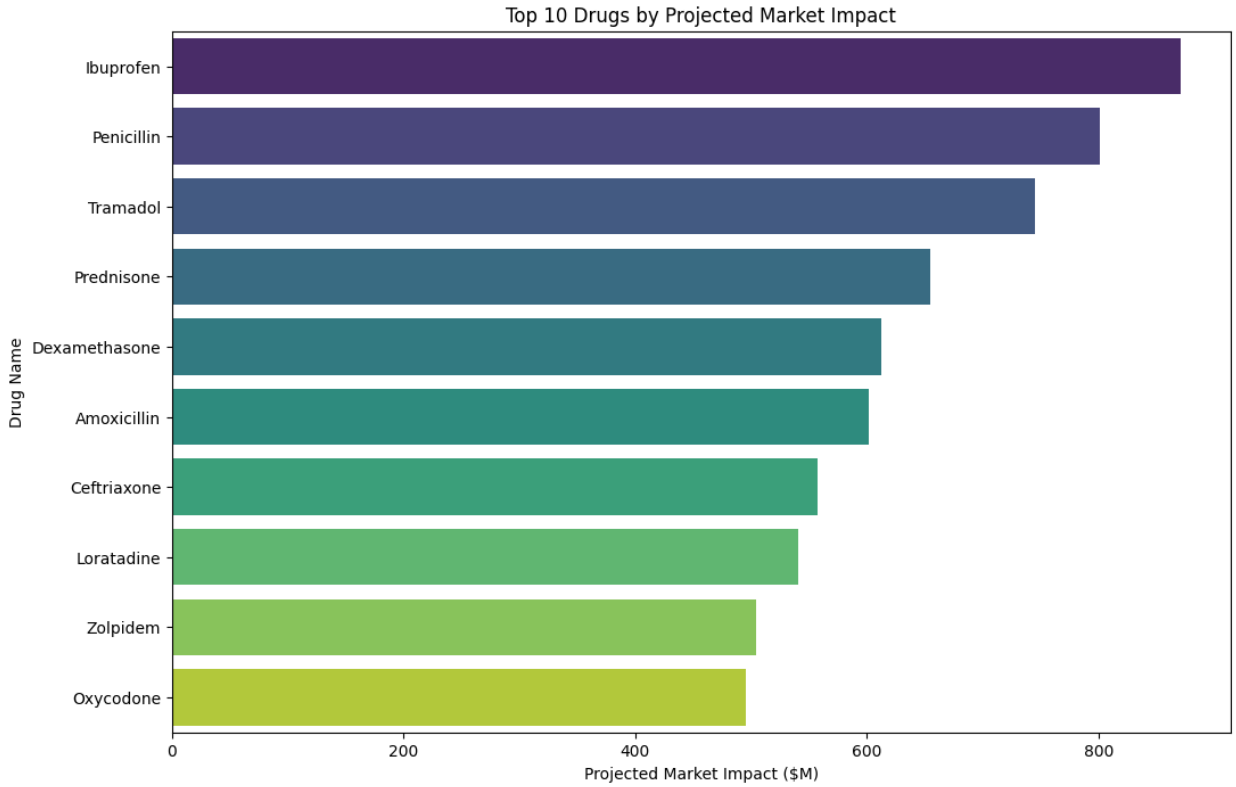


In [143]:
```
""" This code calculates the projected market impact of drugs by multi
with a success rate factor (0.1 for 'Low', 0.5 for 'Medium', and 0.9 f


data['Projected_Market_Impact'] = data['Market Potential $M'] * data['
market_impact = data[['Drug_Name', 'Projected_Market_Impact']].sort_va
market_impact
```

Out[143]:

	Drug_Name	Projected_Market_Impact
24	Ibuprofen	870.534
40	Penicillin	800.676
46	Tramadol	745.587
42	Prednisone	654.579
14	Dexamethasone	612.837
5	Amoxicillin	601.218
8	Ceftriaxone	557.289
31	Loratadine	541.008
50	Zolpidem	504.189
37	Oxycodone	495.525
23	Hydrochlorothiazide	484.995

In [93]:
```
plt.figure(figsize=(12, 8))
sns.barplot(x='Projected_Market_Impact', y='Drug_Name', data=market_im
plt.title('Top 10 Drugs by Projected Market Impact')
plt.xlabel('Projected Market Impact ($M)')
plt.ylabel('Drug Name')
plt.show()
```

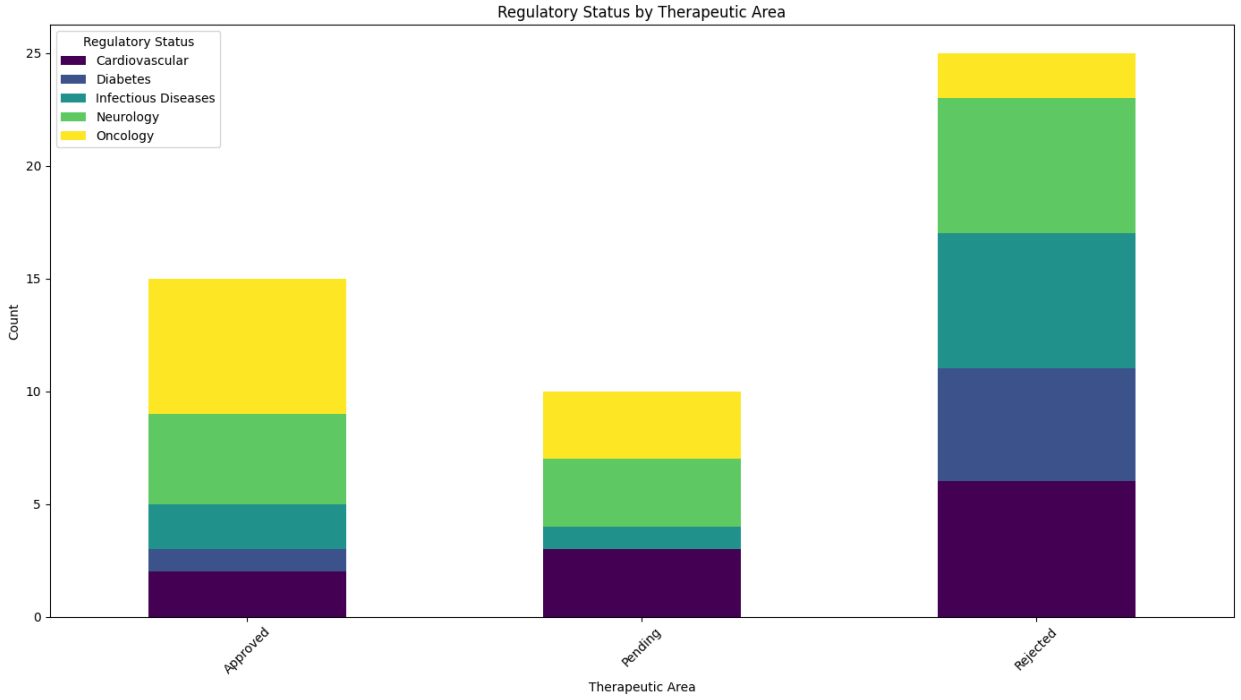


In [144]:
```
"""This code groups the data by both 'Regulatory Status' and 'Therapeu
for each combination"""

regulatory_analysis = data.groupby(['Regulatory Status', 'Therapeutic
regulatory_analysis
```

Out[144]:

Therapeutic Area	Cardiovascular	Diabetes	Infectious Diseases	Neurology	Oncology
Regulatory Status					
Approved	2	1	2	4	6
Pending	3	0	1	3	3
Rejected	6	5	6	6	2

In [95]:
```
regulatory_analysis.plot(kind='bar', stacked=True, figsize=(14, 8), co
plt.title('Regulatory Status by Therapeutic Area')
plt.xlabel('Therapeutic Area')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.legend(title='Regulatory Status')
plt.tight_layout()
plt.show()
```



In [145]:
```
""" This code aggregates the data by the results of clinical trials, c
potential in millions and counting the number of trials for each resul

clinical_outcome_impact = data.groupby('Clinical Trial Results').agg({
    'Market Potential $M': ['mean', 'count']
}).reset_index()

clinical_outcome_impact
```

Out[145]:

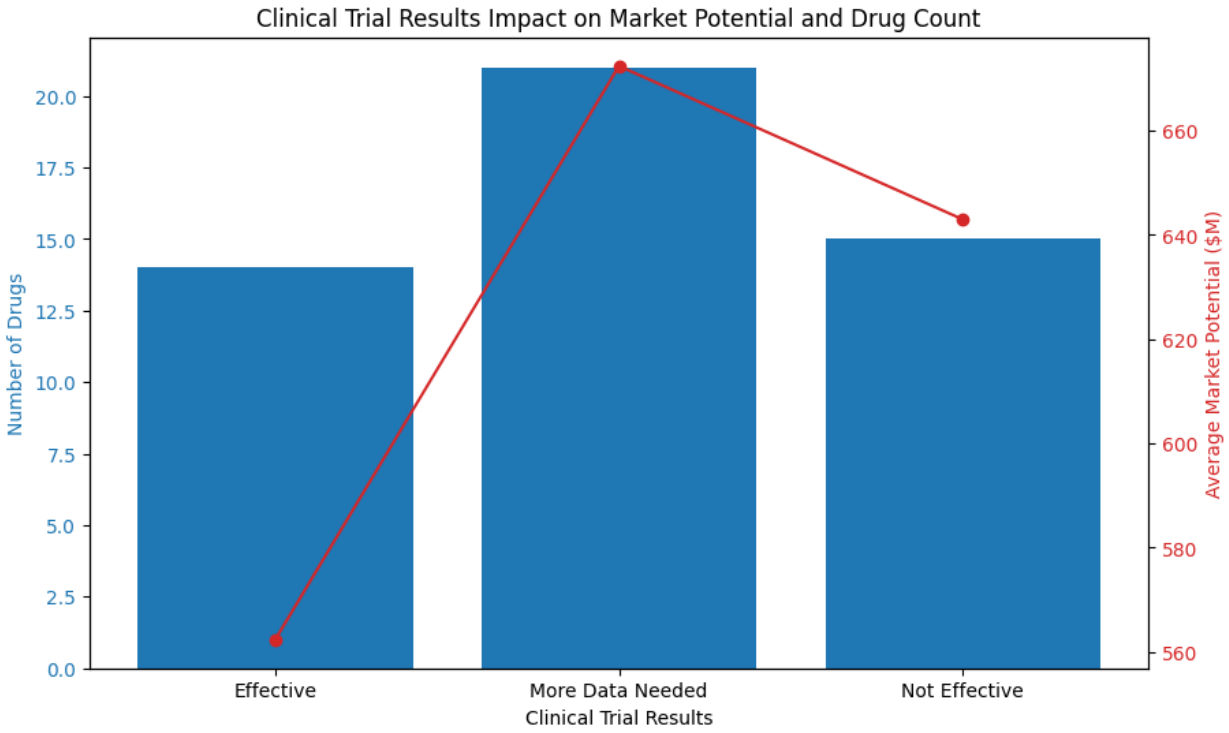
	Clinical Trial Results	Market Potential $M	
		mean	count
0	Effective	562.362143	14
1	More Data Needed	672.426667	21
2	Not Effective	642.996667	15

In [97]:
```
clinical_outcome_impact.columns = ['Clinical Trial Results', 'Average
clinical_outcome_impact
```

Out[97]:

	Clinical Trial Results	Average Market Potential	Number of Drugs
0	Effective	562.362143	14
1	More Data Needed	672.426667	21
2	Not Effective	642.996667	15

In [98]:
```
fig, ax1 = plt.subplots(figsize=(10, 6))

color = 'tab:blue'
ax1.set_xlabel('Clinical Trial Results')
ax1.set_ylabel('Number of Drugs', color=color)
ax1.bar(clinical_outcome_impact['Clinical Trial Results'], clinical_ou
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()
color = 'tab:red'
ax2.set_ylabel('Average Market Potential ($M)', color=color)
ax2.plot(clinical_outcome_impact['Clinical Trial Results'], clinical_c
ax2.tick_params(axis='y', labelcolor=color)

plt.title('Clinical Trial Results Impact on Market Potential and Drug
plt.show()
```



In [146]:
```
""" The code calculates the correlation between the development durati
providing a correlation matrix. This matrix helps to understand the re
is in development and its likelihood of success, quantitatively reflec
these factors"""

data['Success_Rate_Num'] = data['Success_Rate'].map({'Low': 1, 'Medium
correlation_info = data[['Development_Duration', 'Success_Rate_Num']].

print("Correlation between Development Duration and Success Rate:")
print(correlation_info)
```

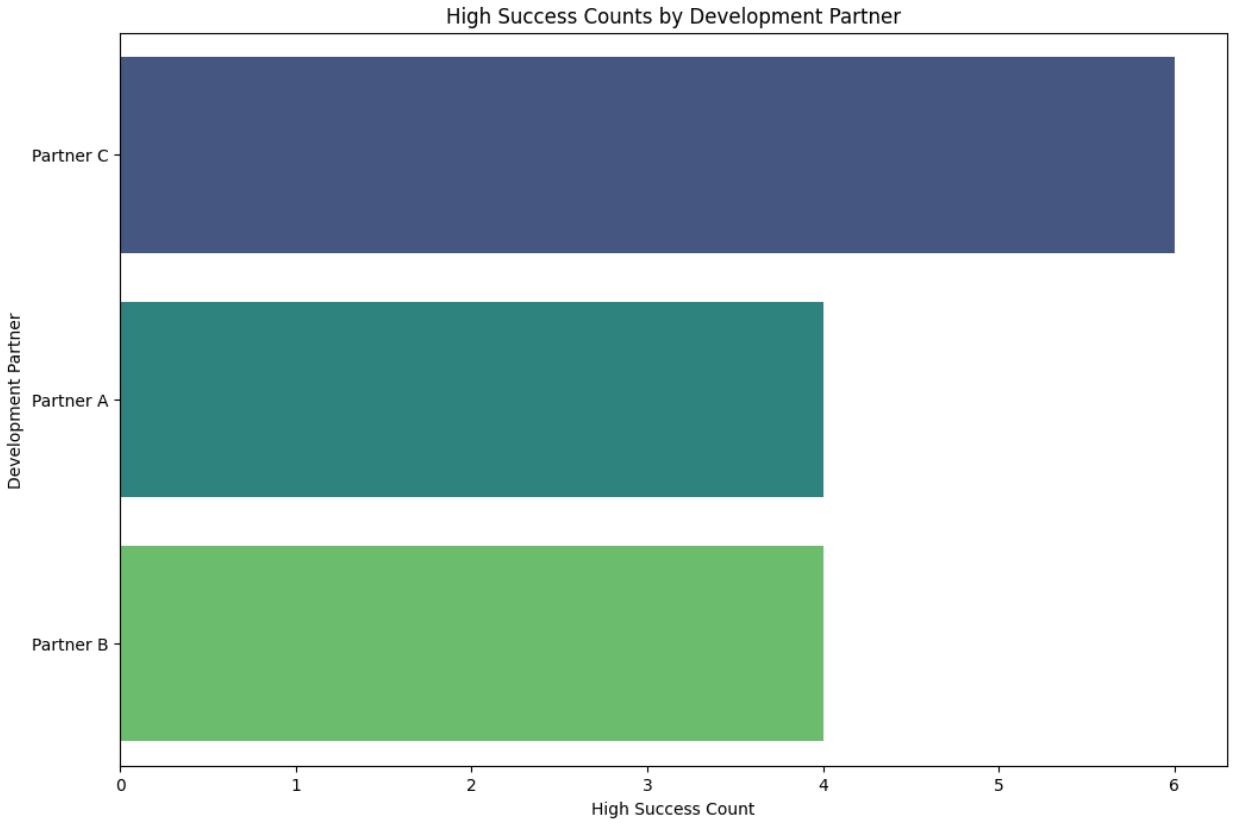
```
Correlation between Development Duration and Success Rate:
                      Development_Duration  Success_Rate_Num
Development_Duration              1.000000          0.148487
Success_Rate_Num                  0.148487          1.000000
```

In [147]:
```
""" This code identifies development partners involved in highly succe
drugs with a 'High' success rate and grouping by 'Development Partner'

active_partners = data[data['Success_Rate'] == 'High'].groupby('Develo
active_partners.sort_values('High_Success_Count', ascending=False, inp
active_partners
```

Out[147]:

	Development Partner	High_Success_Count
2	Partner C	6
0	Partner A	4
1	Partner B	4

In [101]:
```
plt.figure(figsize=(12, 8))
sns.barplot(x='High_Success_Count', y='Development Partner', data=acti
plt.title('High Success Counts by Development Partner')
plt.xlabel('High Success Count')
plt.ylabel('Development Partner')
plt.show()
```



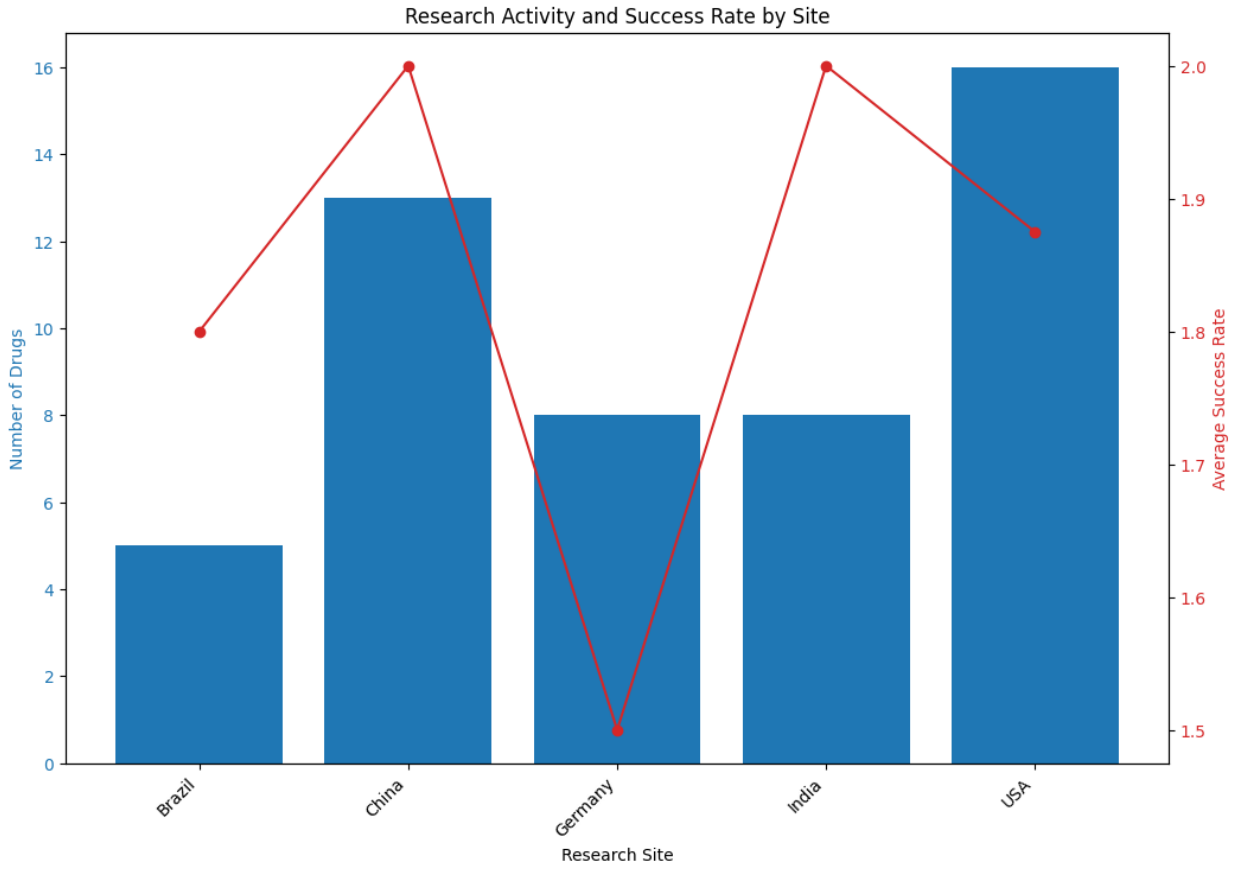
In [170]:
```
"""This code aggregates the number of drugs and average success rates
for clarity in the resulting DataFrame."""

geographic_analysis = data.groupby('Research Site').agg({
    'Drug_ID': 'count',
    'Success_Rate_Num': 'mean'
}).rename(columns={'Drug_ID': 'Number_of_Drugs', 'Success_Rate_Num': '

geographic_analysis
```

Out[170]:

Research Site	Number_of_Drugs	Average_Success_Rate
Brazil	5	1.800
China	13	2.000
Germany	8	1.500
India	8	2.000
USA	16	1.875

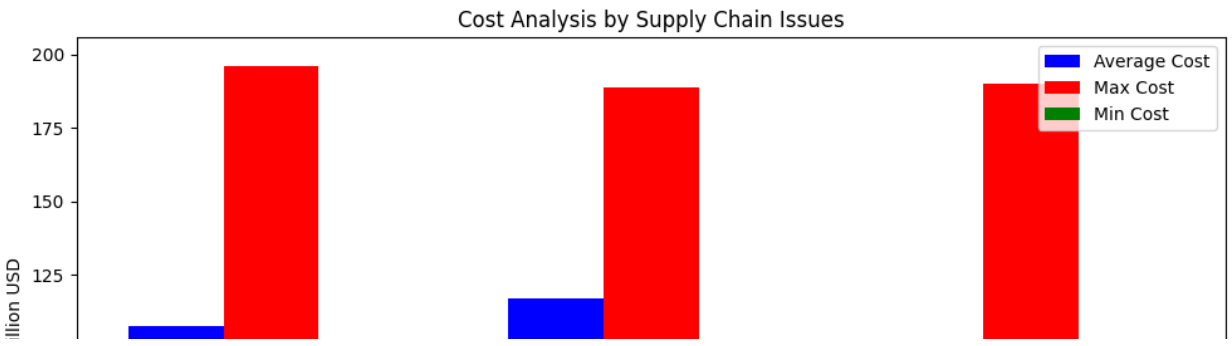
In [103]:
```
fig, ax1 = plt.subplots(figsize=(12, 8))

color = 'tab:blue'
ax1.set_xlabel('Research Site')
ax1.set_ylabel('Number of Drugs', color=color)
ax1.bar(geographic_analysis.index, geographic_analysis['Number_of_Drug
ax1.tick_params(axis='y', labelcolor=color)
ax1.set_xticklabels(geographic_analysis.index, rotation=45, ha="right"

ax2 = ax1.twinx()
color = 'tab:red'
ax2.set_ylabel('Average Success Rate', color=color)
ax2.plot(geographic_analysis.index, geographic_analysis['Average_Succe
ax2.tick_params(axis='y', labelcolor=color)

plt.title('Research Activity and Success Rate by Site')
plt.show()
```

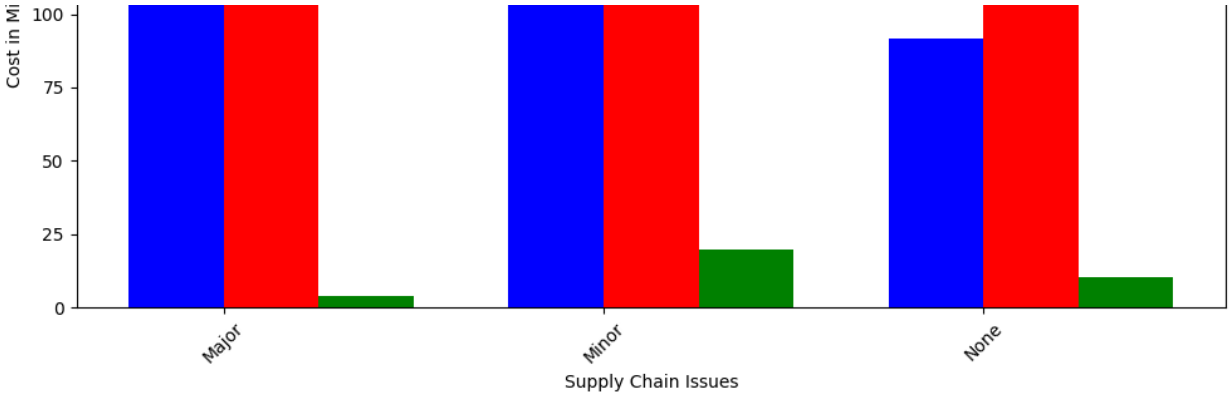
```
/var/folders/5q/mcr29lr105l0g_1p7s7y9ssr0000gn/T/ipykernel_7030/22269
07611.py:8: UserWarning: set_ticklabels() should only be used with a
fixed number of ticks, i.e. after set_ticks() or using a FixedLocato
r.
  ax1.set_xticklabels(geographic_analysis.index, rotation=45, ha="rig
ht")
```



In [171]:
```
""" This code calculates and formats the average, maximum, and minimum
different supply chain issues"""

import numpy as np
supply_chain_costs = data.groupby('Supply_Chain_Issues').agg({
    'Cost_Million_USD': ['mean', 'max', 'min']
}).reset_index()

supply_chain_costs.columns = [' '.join(col).strip() for col in supply_

supply_chain_costs.rename(columns={
    'Supply_Chain_Issues ': 'Supply_Chain_Issues',
    'Cost_Million_USD mean': 'Average_Cost',
    'Cost_Million_USD max': 'Max_Cost',
    'Cost_Million_USD min': 'Min_Cost'
}, inplace=True)

print(supply_chain_costs.columns)

fig, ax = plt.subplots(figsize=(10, 6))
index = np.arange(len(supply_chain_costs))
bar_width = 0.25

rects1 = ax.bar(index, supply_chain_costs['Average_Cost'], bar_width,
rects2 = ax.bar(index + bar_width, supply_chain_costs['Max_Cost'], bar
rects3 = ax.bar(index + 2 * bar_width, supply_chain_costs['Min_Cost'],

ax.set_xlabel('Supply Chain Issues')
ax.set_ylabel('Cost in Million USD')
ax.set_title('Cost Analysis by Supply Chain Issues')
ax.set_xticks(index + bar_width / 2)
ax.set_xticklabels(supply_chain_costs['Supply_Chain_Issues'])
ax.legend()

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
Index(['Supply_Chain_Issues', 'Average_Cost', 'Max_Cost', 'Min_Cos
t'], dtype='object')
```




Cost in Mi
100
75
50
25
0
Major
Minor
None
Supply Chain Issues

In [172]:
```
""" This code calculates the average development duration for drugs, g
providing insights into efficiency across different stages and areas o

efficiency_analysis = data.groupby(['Phase', 'Therapeutic Area']).agg(
    'Development_Duration': 'mean'
}).reset_index()

efficiency_analysis
```

Out[172]:

	Phase	Therapeutic Area	Development_Duration
0	Approval	Cardiovascular	169.333333
1	Approval	Diabetes	282.000000
2	Approval	Infectious Diseases	166.666667
3	Approval	Neurology	105.000000
4	Approval	Oncology	236.500000
5	Phase I	Cardiovascular	193.000000
6	Phase I	Diabetes	92.500000
7	Phase I	Infectious Diseases	341.000000
8	Phase I	Neurology	181.000000
9	Phase I	Oncology	278.000000
10	Phase II	Cardiovascular	151.000000
11	Phase II	Infectious Diseases	120.000000
12	Phase II	Neurology	139.000000
13	Phase II	Oncology	202.000000
14	Phase III	Cardiovascular	142.333333
15	Phase III	Diabetes	216.500000
16	Phase III	Infectious Diseases	189.000000
17	Phase III	Neurology	212.250000
18	Phase III	Oncology	159.000000
19	Pre-clinical	Cardiovascular	161.000000
20	Pre-clinical	Diabetes	220.000000
21	Pre-clinical	Infectious Diseases	194.500000
22	Pre-clinical	Neurology	299.500000
23	Pre-clinical	Oncology	177.000000

In [174]:
```
data['Year'] = data['Start_Date'].dt.year
```

In [175]:
```
longitudinal_study = data.groupby(['Year', 'Therapeutic Area']).size()
longitudinal_study
```

Out[175]:

Therapeutic Area	Cardiovascular	Diabetes	Infectious Diseases	Neurology	Oncology
Year					
2015	3	1	2	6	1
2016	5	0	3	3	1
2017	1	0	3	3	5
2018	2	5	1	1	3
2019	0	0	0	0	1

In [176]:
```
"""This code calculates the average development duration for drugs wit
name, providing insights into average launch delays"""

launch_delays = data[data['Supply_Chain_Issues'] != 'None'].groupby('D
    'Development_Duration': 'mean'
}).rename(columns={'Development_Duration': 'Average_Launch_Delay'})
launch_delays
```

Out[176]:

Drug_Name	Average_Launch_Delay
Acetaminophen	105.0
Amlodipine	103.0
Atorvastatin	259.0
Cetirizine	170.0
Clopidogrel	199.0
Dapagliflozin	151.0
Dexamethasone	223.0
Diphenhydramine	165.0
Esomeprazole	174.0
Fentanyl	356.0

In [111]:
```
## METRICS
```

In [169]:
```
""" This code maps qualitative success rates to numeric values and cal
rate for each therapeutic area."""

success_mapping = {'Low': 1, 'Medium': 2, 'High': 3}
data['Success_Rate_Numeric'] = data['Success_Rate'].map(success_mappin
average_success_by_area = data.groupby('Therapeutic Area')['Success_Ra
average_success_by_area
```

Out[169]:
```
Therapeutic Area
Cardiovascular         2.090909
Diabetes               2.166667
Infectious Diseases    1.777778
Neurology              1.615385
Oncology               1.818182
Name: Success_Rate_Numeric, dtype: float64
```

In [168]:
```
""" This code computes the percentage of approved drugs within each th

data['Is_Approved'] = (data['Regulatory Status'] == 'Approved').astype
approval_rates = data.groupby('Therapeutic Area')['Is_Approved'].mean(
approval_rates
```

Out[168]:
```
Therapeutic Area
Cardiovascular         18.181818
Diabetes               16.666667
Infectious Diseases    22.222222
Neurology              30.769231
Oncology               54.545455
Name: Is_Approved, dtype: float64
```

In [161]:
```
average_cost_by_phase = data.groupby('Phase')['Cost_Million_USD'].mean
average_cost_by_phase
```

Out[161]:
```
Phase
Approval        103.542738
Phase I         108.740599
Phase II         81.820645
Phase III       120.471740
Pre-clinical     89.147004
Name: Cost_Million_USD, dtype: float64
```

In [167]:
```
""" This code calculates projected revenue by adjusting market potenti
sums these values by therapeutic area """

data['Projected_Revenue'] = data['Market Potential $M'] * data['Succes
projected_revenue_by_area = data.groupby('Therapeutic Area')['Projecte
projected_revenue_by_area
```

Out[167]:
```
Therapeutic Area
Cardiovascular          4873.266667
Diabetes                3270.993333
Infectious Diseases     3542.973333
Neurology               3193.770000
Oncology                4175.476667
Name: Projected_Revenue, dtype: float64
```

In [163]:
```
""" This code analyzes supply chain issues by counting occurrences and
these metrics into a single DataFrame."""

supply_chain_impact = data.groupby('Supply_Chain_Issues').size()
success_rate_impact = data.groupby('Supply_Chain_Issues')['Success_Rat
supply_chain_analysis = pd.concat([supply_chain_impact, success_rate_i
supply_chain_analysis
```

Out[163]:

Supply_Chain_Issues	Issue_Count	Average_Success_Rate
Major	19	1.789474
Minor	10	1.700000
None	21	2.000000

In [164]:
```
""" This code calculates the percentage of drugs approved in each phas

data['Phase_Success'] = data['Regulatory Status'].apply(lambda x: 1 if
phase_transition_success_rate = data.groupby('Phase')['Phase_Success']
phase_transition_success_rate
```

Out[164]:
```
Phase
Approval        40.000000
Phase I         10.000000
Phase II        30.000000
Phase III       23.076923
Pre-clinical    57.142857
Name: Phase_Success, dtype: float64
```

In [165]:
```
""" This code maps risk levels to drugs based on their regulatory stat
these risks across therapeutic areas"""

risk_levels = {'Pending': 'Medium', 'Approved': 'Low', 'Rejected': 'Hi
data['Risk_Level'] = data['Regulatory Status'].map(risk_levels)

risk_profile = data.groupby('Therapeutic Area')['Risk_Level'].value_co
risk_profile
```

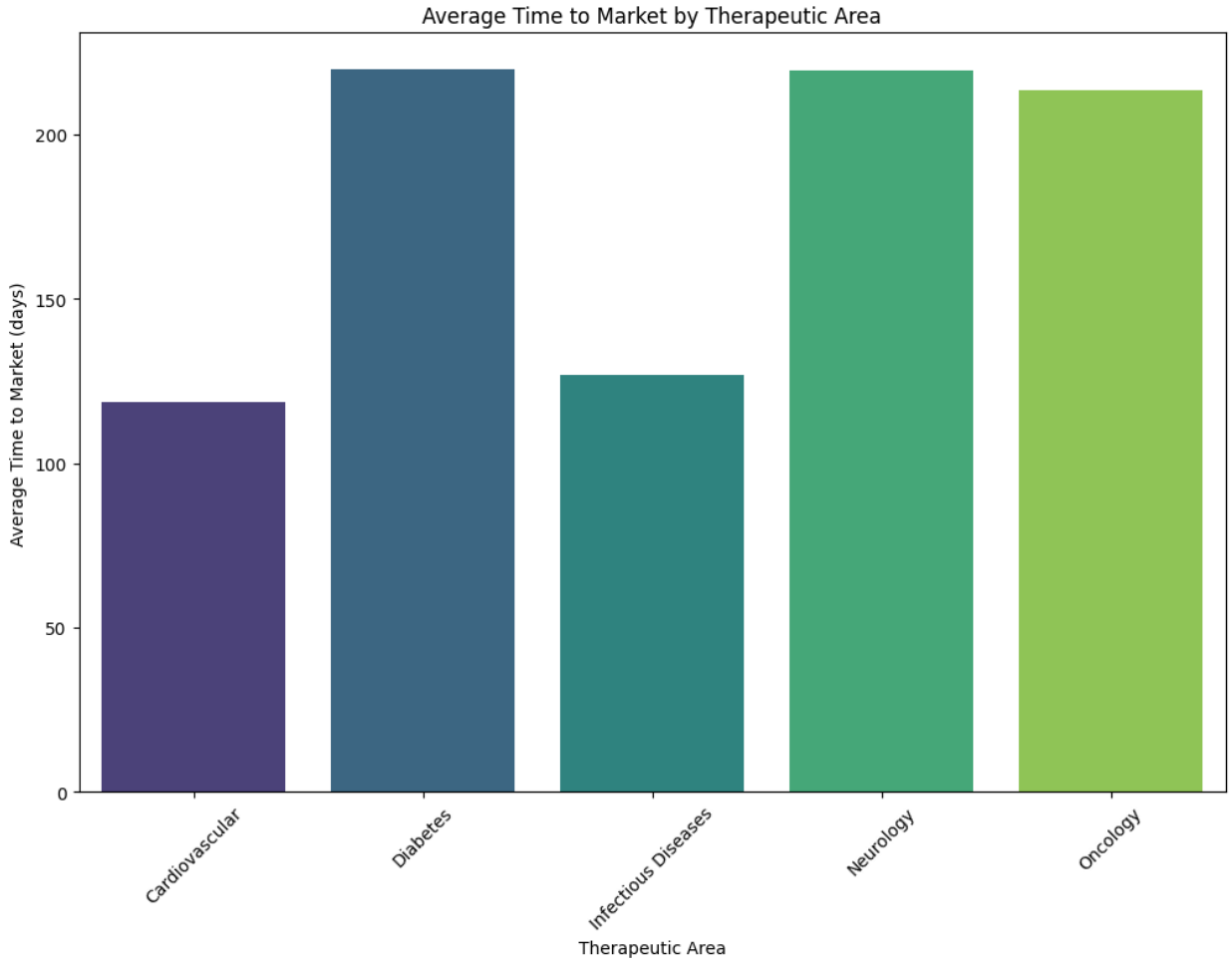
Out[165]:

Risk_Level	High	Low	Medium
Therapeutic Area			
Cardiovascular	6	2	3
Diabetes	5	1	0
Infectious Diseases	6	2	1
Neurology	6	4	3
Oncology	2	6	3

In [166]:
```
""" This code filters for drugs that have been approved, calculates th
and then computes the average time to market for approved drugs groupe

approved_drugs = data[data['Regulatory Status'] == 'Approved'].copy()

approved_drugs.loc[:, 'Time_To_Market'] = (approved_drugs['End_Date']

average_time_to_market = approved_drugs.groupby('Therapeutic Area')['T
average_time_to_market
```

Out[166]:
```
Therapeutic Area
Cardiovascular         118.500000
Diabetes               220.000000
Infectious Diseases    127.000000
Neurology              219.500000
Oncology               213.333333
Name: Time_To_Market, dtype: float64
```

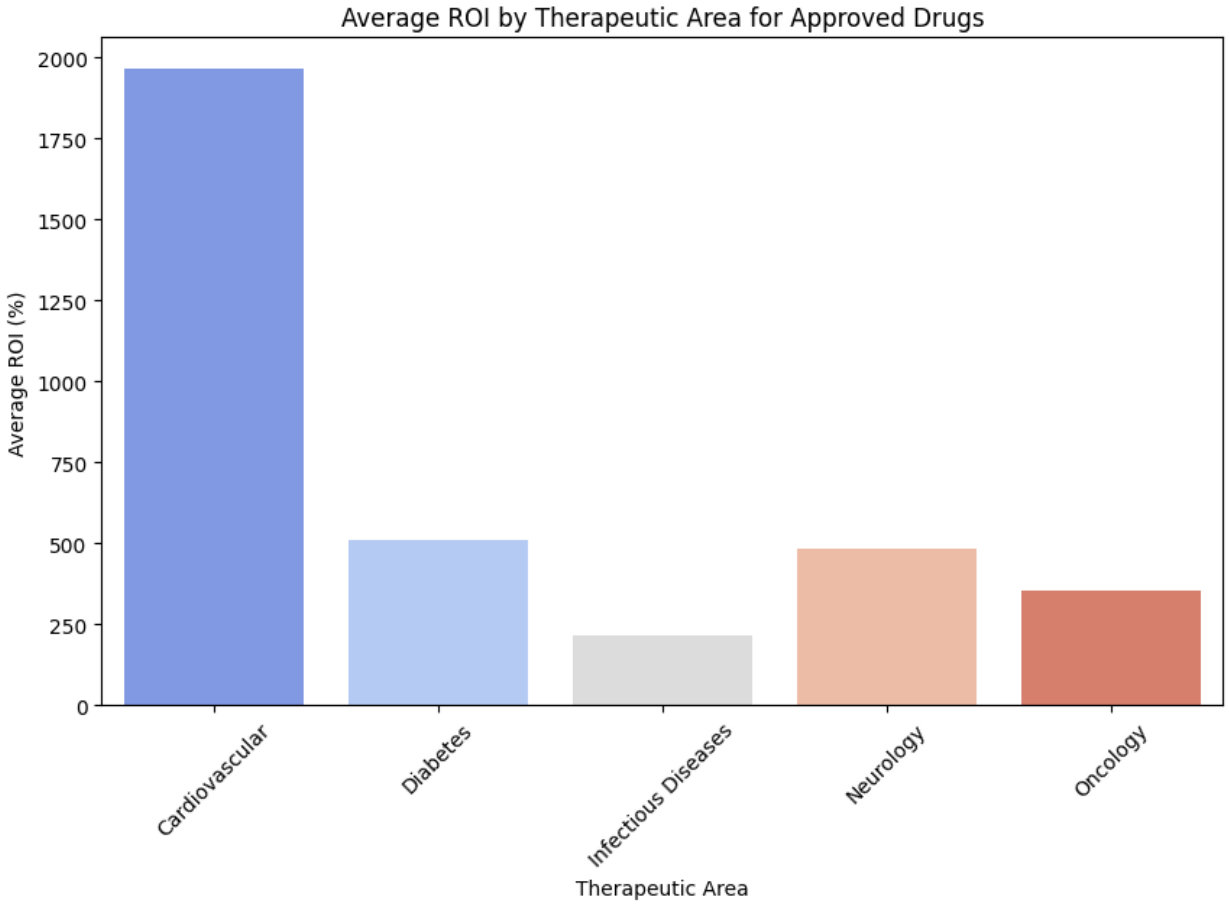
In [151]:
```
average_time_to_market_df = average_time_to_market.reset_index()

plt.figure(figsize=(12, 8))
sns.barplot(x='Therapeutic Area', y='Time_To_Market', data=average_tim
plt.title('Average Time to Market by Therapeutic Area')
plt.xlabel('Therapeutic Area')
plt.ylabel('Average Time to Market (days)')
plt.xticks(rotation=45)
plt.show()
```



In [149]:
```
""" This code calculates the ROI for each drug and averages it for app

data['ROI'] = ((data['Market Potential $M'] - data['Cost_Million_USD']
average_roi_approved = data[data['Regulatory Status'] == 'Approved'].g
average_roi_approved
```

Out[149]:
```
Therapeutic Area
Cardiovascular         1963.801475
Diabetes                510.232290
Infectious Diseases     215.993560
Neurology               484.692368
Oncology                353.151922
Name: ROI, dtype: float64
```

In [122]:
```
average_roi_approved_df = average_roi_approved.reset_index()

plt.figure(figsize=(10, 6))
sns.barplot(x='Therapeutic Area', y='ROI', data=average_roi_approved_d
plt.title('Average ROI by Therapeutic Area for Approved Drugs')
plt.xlabel('Therapeutic Area')
plt.ylabel('Average ROI (%)')
plt.xticks(rotation=45)
plt.show()
```



In [148]:
```
""" This code maps clinical trial results to numeric efficacy scores,
for each phase of drug development"""

trial_results_mapping = {'Not Effective': 0, 'More Data Needed': 1, 'E
data['Trial_Efficacy'] = data['Clinical Trial Results'].map(trial_resu

clinical_trial_efficacy = data.groupby('Phase')['Trial_Efficacy'].mean
clinical_trial_efficacy
```

Out[148]:
```
Phase
Approval        1.000000
Phase I         1.000000
Phase II        1.100000
Phase III       0.615385
Pre-clinical    1.428571
Name: Trial_Efficacy, dtype: float64
```

In [124]:
```
clinical_trial_efficacy_df = clinical_trial_efficacy.reset_index()

plt.figure(figsize=(10, 6))
sns.barplot(x='Phase', y='Trial_Efficacy', data=clinical_trial_efficac
plt.title('Average Clinical Trial Efficacy by Phase')
plt.xlabel('Phase')
plt.ylabel('Average Efficacy')
plt.xticks(rotation=45)
plt.show()
```