

```

Services:
<class 'pandas.core.frame.DataFrame'>
Index: 2912 entries, 0 to 2911
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype  
--- 
 0   category        2912 non-null     object  
 1   service         2912 non-null     object  
 2   quantity        2912 non-null     float64 
 3   cost            2912 non-null     float64 
 4   sale_price      2912 non-null     float64 
 5   creation_date   2912 non-null     datetime64[ns]
dtypes: datetime64[ns](1), float64(3), object(2)
memory usage: 159.2+ KB
None

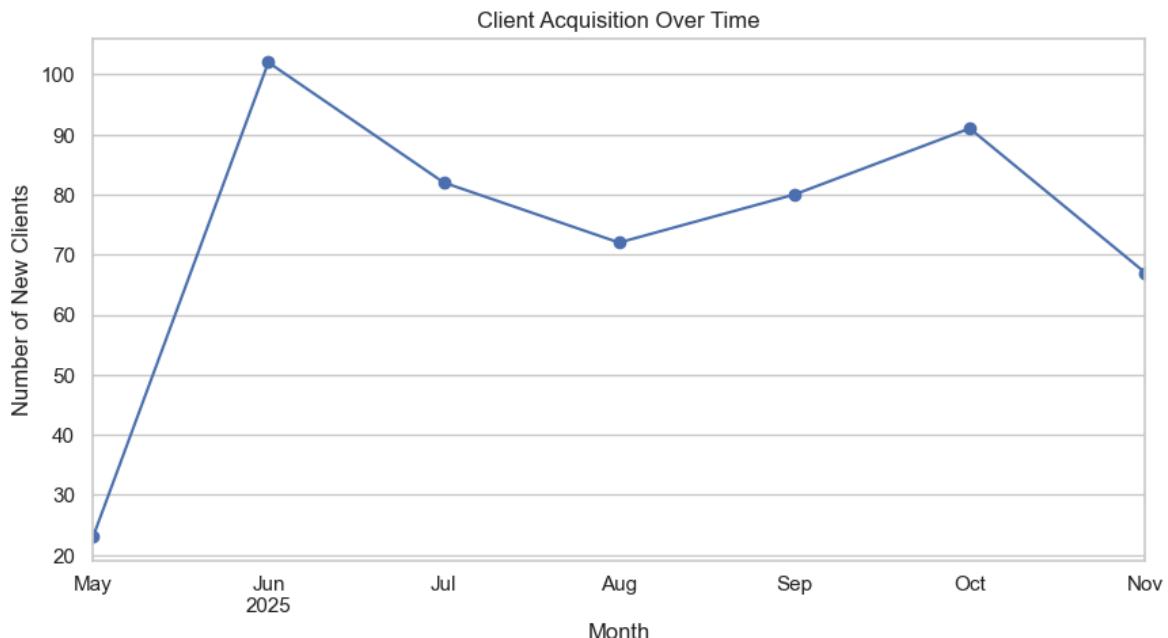
```

## 4. Visualizations

Analyzing operational and financial efficiency.

### 4.1 Operational Efficiency

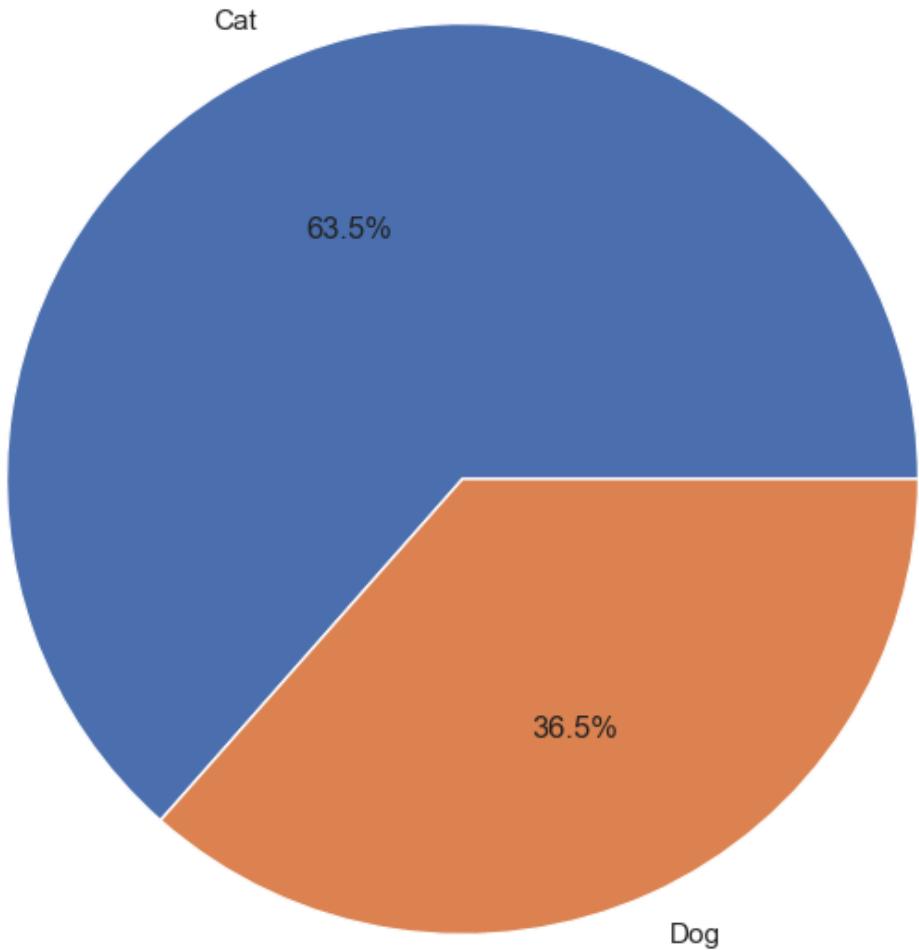
```
In [69]: # Client Acquisition over Time
clients_by_month = clients_df.groupby(clients_df["creation_date"].dt.to_p
plt.figure(figsize=(10, 5))
clients_by_month.plot(kind='line', marker='o')
plt.title("Client Acquisition Over Time")
plt.xlabel("Month")
plt.ylabel("Number of New Clients")
plt.grid(True)
plt.show()
```



```
In [70]: # Pets Distribution
plt.figure(figsize=(8, 8))
pets_df.groupby("type").size().plot(kind="pie", autopct="%1.1f%%" )
```

```
plt.title("Pets Distribution")
plt.show()
```

Pets Distribution

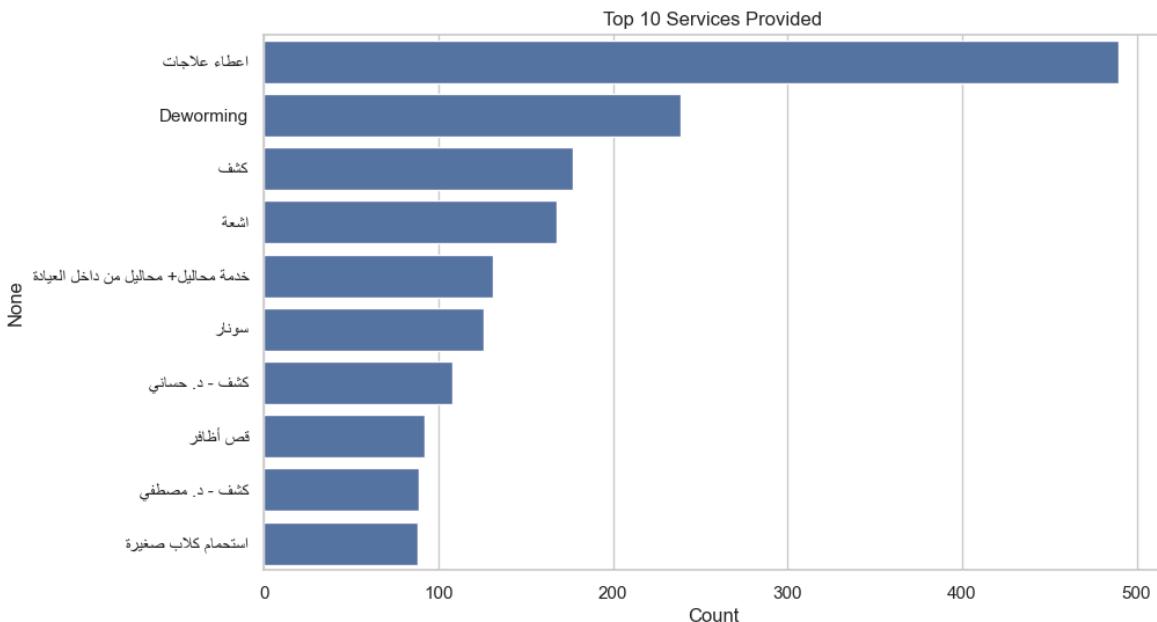


```
In [71]: def fix_arabic_text(text):
    if not isinstance(text, str):
        return str(text)
    return get_display(arabic_reshaper.reshape(text))

top_services = services_df['service'].value_counts().head(10)
# Fix Arabic text in index
top_services.index = [fix_arabic_text(label) for label in top_services.index]

plt.figure(figsize=(10, 6))

sns.barplot(x=top_services.values, y=top_services.index)
plt.title("Top 10 Services Provided")
plt.xlabel("Count")
plt.show()
```



## 4.2 Financial Efficiency

```
In [72]: # Calculate Total Revenue and Expenses by Month
# Revenue comes from Services and Inventory Sales

# Prepare Revenue Data
services_revenue = services_df.copy()
services_revenue['revenue'] = services_revenue['sale_price'] * services_revenue['quantity']
services_revenue = services_revenue.groupby(services_revenue['creation_date']).sum()

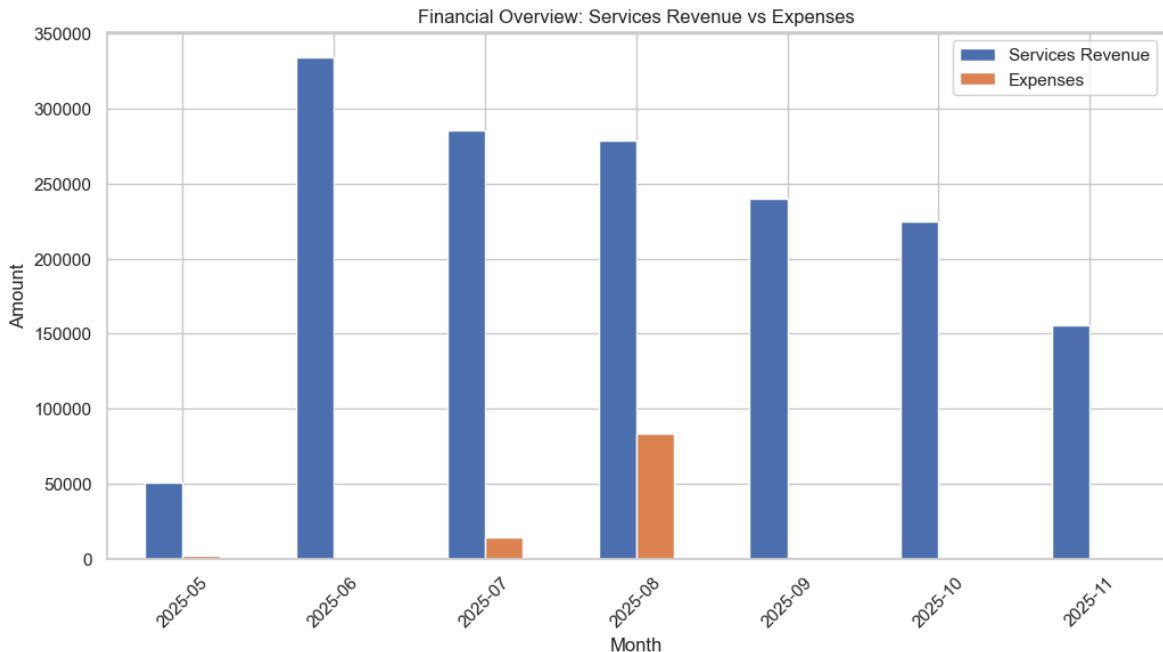
# Assuming inventory items sold also count as revenue (need to check if it's true)
# The prompt mentions "Revenues.xlsx" in the file list but I don't see it
# However, 'item.xlsx' has 'sale_price', suggesting sales.
# Let's assume 'item.xlsx' logs items sold.
inventory_revenue = inventory_df.copy()
# inventory_df has 'creation_date' as integer in original code output (1, 2, 3, 4, 5)
# Let's check the original output. Ah, in the original output for item.xlsx
# This suggests it might not be a real date in that file, or it needs parsing.
# Wait, the original code did: inventory_df.columns=["creation_date", ...]
# But the output showed 1, 2, 3, 4, 5.
# If 'creation_date' in item.xlsx is not a datetime, we can't group by month.
# Let's look at 'service.xlsx', it had real dates.
# Let's look at 'expenses.xlsx', it had real dates.

# For now, let's plot Expenses over time, and Services Revenue over time.

expenses_monthly = expenses_df.groupby(expenses_df['creation_date']).dt.to_period('M').sum()

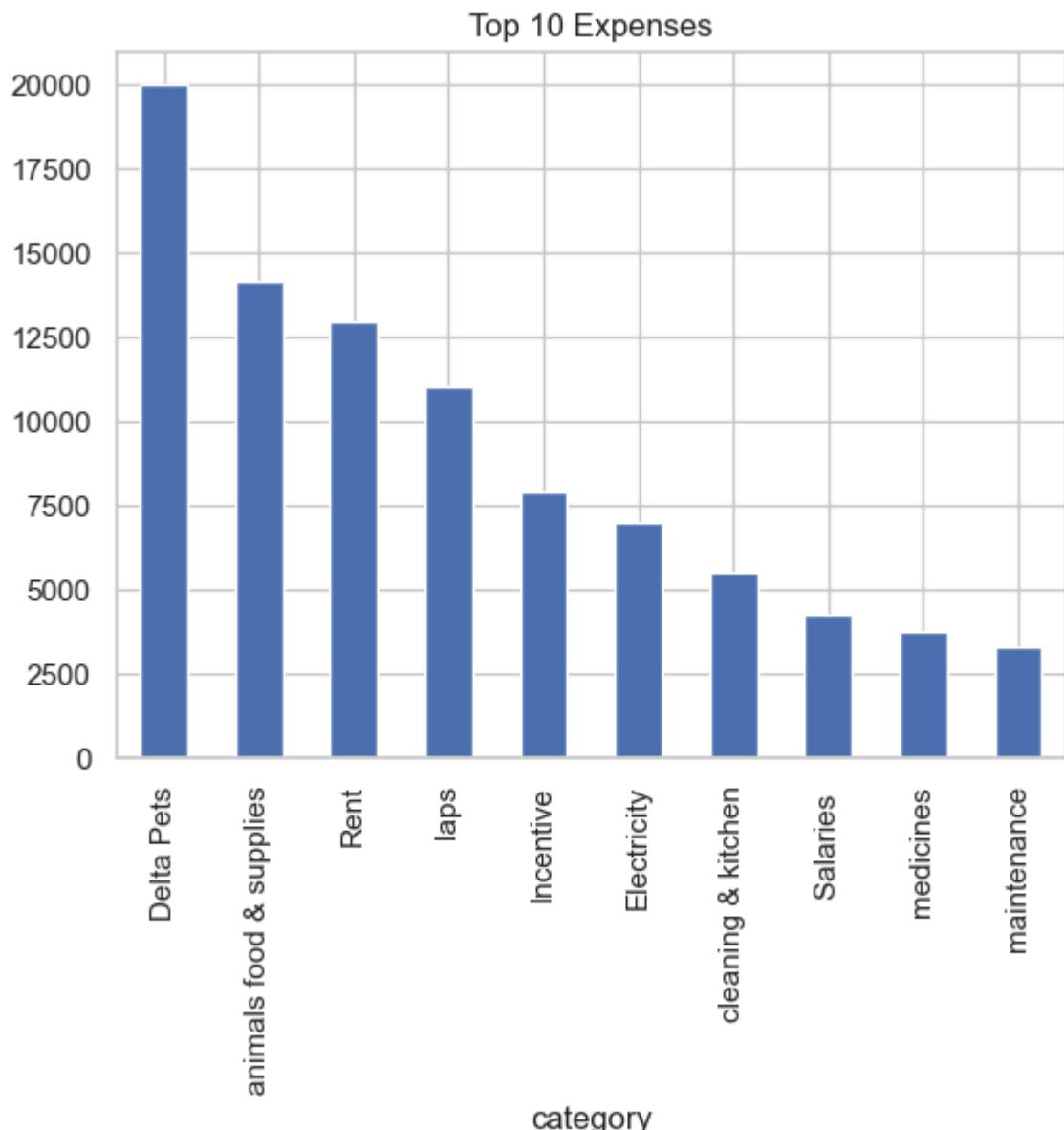
financial_df = pd.DataFrame({
    'Services Revenue': services_revenue,
    'Expenses': expenses_monthly
})

financial_df.plot(kind='bar', figsize=(12, 6))
plt.title("Financial Overview: Services Revenue vs Expenses")
plt.xlabel("Month")
plt.ylabel("Amount")
plt.xticks(rotation=45)
plt.show()
```



```
In [73]: expenses_df.groupby("category")["amount"].sum().sort_values(ascending=False)
```

```
Out[73]: <Axes: title={'center': 'Top 10 Expenses'}, xlabel='category'>
```



In [74]: `inventory_df.head()`

	creation_date	category	item	quantity	cost	sale_price
0	1	Vaccines	drontal cats - ديدان	1.0	103.0	250.0
1	2	N/A	Fipix Cat	1.0	120.0	200.0
2	3	Vaccines	drontal cats - ديدان	1.0	103.0	250.0
3	4	Vaccines	caninua	1.0	155.0	350.0
4	5	N/A	revelution dogs	1.0	400.0	750.0

In [75]: `clients_total=pd.merge(clients_df, revenue_df, left_on="phone", right_on="phone")  
clients_total=clients_total[["name","id","phone","amount","discount","paid"]]  
clients_stats=clients_total.groupby(["phone","name"])[["amount","paid","discount"]].sum()  
highest_debtors=clients_stats.sort_values(by="debit", ascending=False).head(10)`

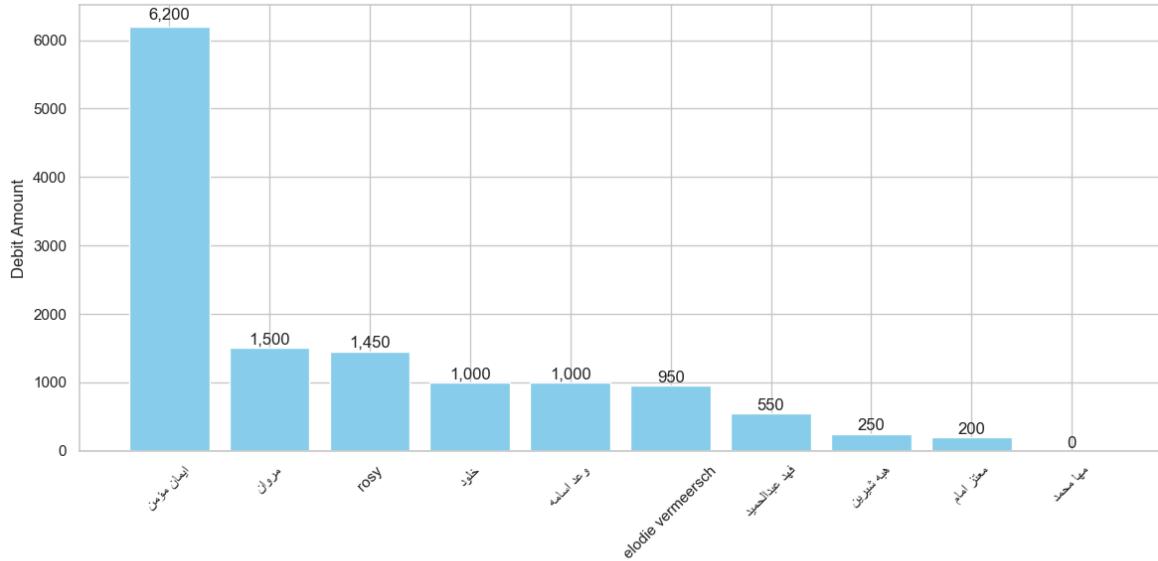
```
fig, axes = plt.subplots(3, 1, figsize=(12, 18))
# Plotting
ax = axes[0]
x_labels = top_debtors.index.get_level_values(1)
x_labels=x_labels.map(fix_arabic_text)
y_values = top_debtors['debit']
ax.bar(x_labels, y_values, color='skyblue')
ax.set_title('A. Top 10 Highest Debtors (Risk Management)')
ax.set_ylabel('Debit Amount')
ax.tick_params(axis='x', rotation=45)
for i, v in enumerate(y_values):
    ax.text(i, v + (v * 0.01), f'{v:.0f}', ha='center', va='bottom')

ax = axes[1]
x_labels = top_discount_receivers.index.get_level_values(1)
x_labels=x_labels.map(fix_arabic_text)
y_values = top_discount_receivers['discount']
ax.bar(x_labels, y_values, color='lightcoral')
ax.set_title('B. Top 10 Highest Discount Receivers (Leakage Analysis)')
ax.set_ylabel('Discount Amount')
ax.tick_params(axis='x', rotation=45)
for i, v in enumerate(y_values):
    ax.text(i, v + (v * 0.01), f'{v:.0f}', ha='center', va='bottom')

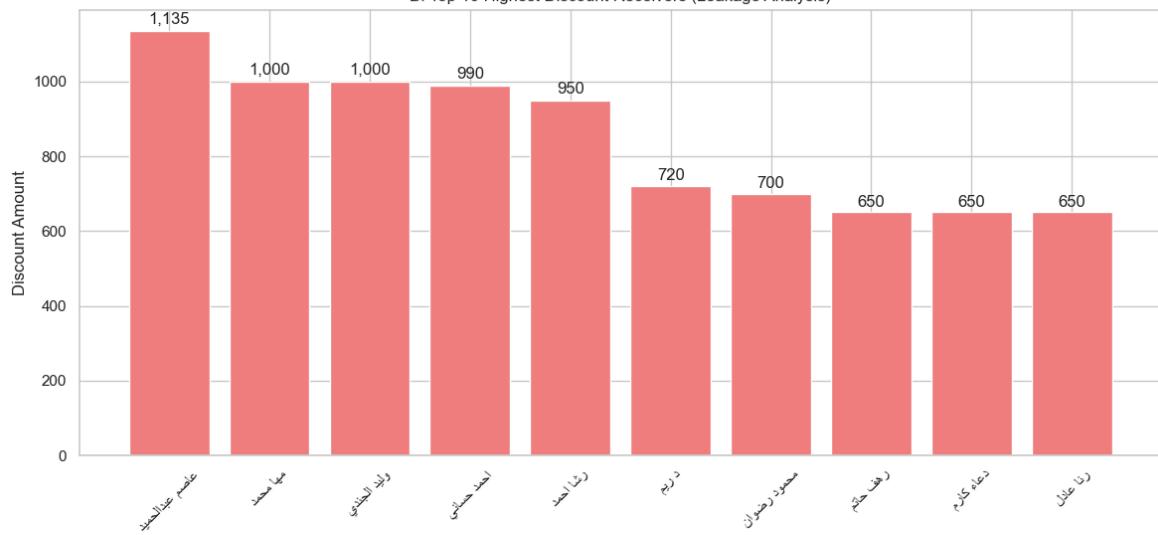
ax = axes[2]
x_labels = top_payees.index.get_level_values(1)
x_labels=x_labels.map(fix_arabic_text)
y_values = top_payees['paid']
ax.bar(x_labels, y_values, color='lightgreen')
ax.set_title('C. Top 10 Highest Payees / VIPs (Revenue Drivers)')
ax.set_ylabel('Paid Amount')
ax.tick_params(axis='x', rotation=45)
for i, v in enumerate(y_values):
    ax.text(i, v + (v * 0.01), f'{v:.0f}', ha='center', va='bottom')

plt.tight_layout()
plt.show()
```

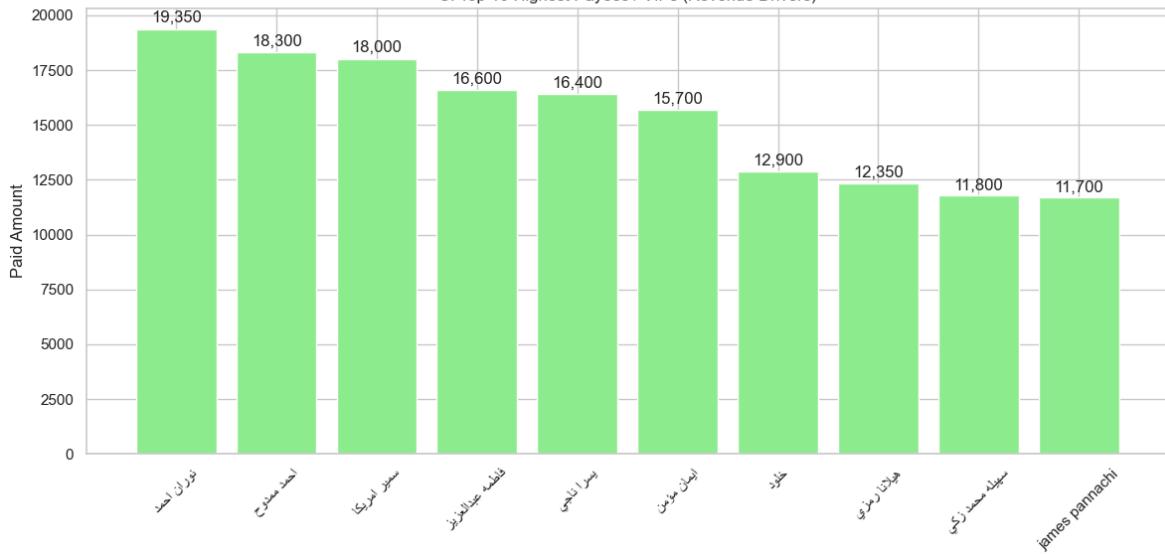
A. Top 10 Highest Debtors (Risk Management)



B. Top 10 Highest Discount Receivers (Leakage Analysis)



C. Top 10 Highest Payees / VIPs (Revenue Drivers)



In [ ]: