In [27]:

```
from pathlib import Path
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Paths
ROOT_DIR = Path().resolve().parent
DATA_DIR = ROOT_DIR / "data"

# Load data

claims = pd.read_csv(DATA_DIR / "fact_claims.csv")
members = pd.read_csv(DATA_DIR / "member_dimension.csv")
diagnoses = pd.read_csv(DATA_DIR / "diagnosis_dimension.csv")
providers = pd.read_csv(DATA_DIR / "provider_dimension.csv")
date=pd.read_csv(DATA_DIR/"date_dimension.csv")
# Quick look

df = claims.merge(members, on="member_id", how="left") \
           .merge(diagnoses, on="diagnosis_code", how="left") \
           .merge(providers, on="provider_id", how="left")\
            .merge(date,on="date_id",how= "left")
# Create age_group based on the age column
bins = [0, 25, 35, 45, 60, 120]
labels = ["18–25", "26–35", "36–45", "46–60", "60+"]

df['age_group'] = pd.cut(df['age'], bins=bins, labels=labels, right=False)

df.head()
```

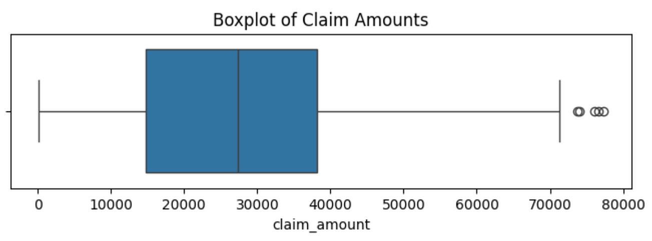
Out[27]:

	claim_id	member_id	provider_id	date_id	claim_amount	diagnosis_code	name	age	gender	Age Group	...	provider_name	speci
0	7	188	84	59	44405	D1	Joseph Williams	76	M	60+	...	Clinic 84	Neuro
1	9	543	86	146	12309	D1	Amanda Myers	48	M	46–60	...	Clinic 86	Neuro
2	14	725	37	44	41413	D1	Joseph Mahoney	25	M	18–25	...	Clinic 37	Neuro
3	20	342	43	163	18411	D1	Lindsey Glover	23	M	18–25	...	Clinic 43	Ger Pra
4	22	903	88	44	48453	D1	Mrs. Ashley Gilbert	60	M	46–60	...	Clinic 88	Psych

5 rows × 21 columns

In [2]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Detect outliers using Z-score
claims['z_score'] = (claims['claim_amount'] - claims['claim_amount'].mean()) / claims['claim_amount'].std()
outliers = claims[claims['z_score'].abs() > 3]

# Boxplot
plt.figure(figsize=(8, 2))
sns.boxplot(x=claims['claim_amount'])
plt.title("Boxplot of Claim Amounts")
plt.show()

print(f"Outliers detected: {len(outliers)}")
outliers[['member_id', 'claim_amount', 'z_score']].head()
```



```
Outliers detected: 3
```

Out[2]:

	member_id	claim_amount	z_score
361	655	77281	3.170308
570	802	76694	3.133429
1172	20	76153	3.099440

In [3]:

```
top_diag = (
    df.groupby(['age_group', 'diagnosis_description'], observed=True)['claim_id']
    .count()
    .reset_index()
    .sort_values(['age_group', 'claim_id'], ascending=[True, False])
    .groupby('age_group', observed=True)
    .head(1)
)

print("Top diagnosis per age group:")
top_diag
```

```
Top diagnosis per age group:
```

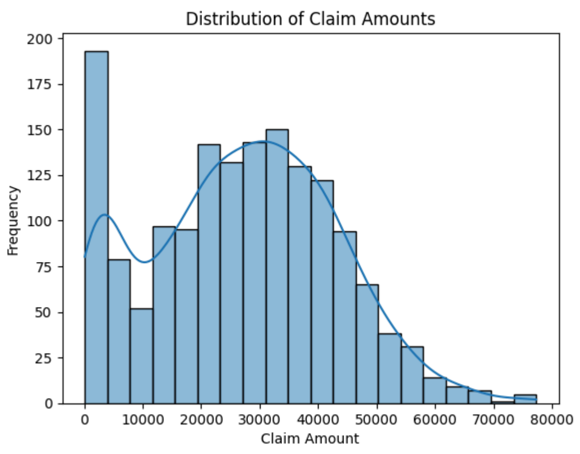
Out[3]:

	age_group	diagnosis_description	claim_id
2	18–25	Hypertension	37
7	26–35	Hypertension	58
10	36–45	Asthma	46
15	46–60	Asthma	79
22	60+	Hypertension	143

In [7]:

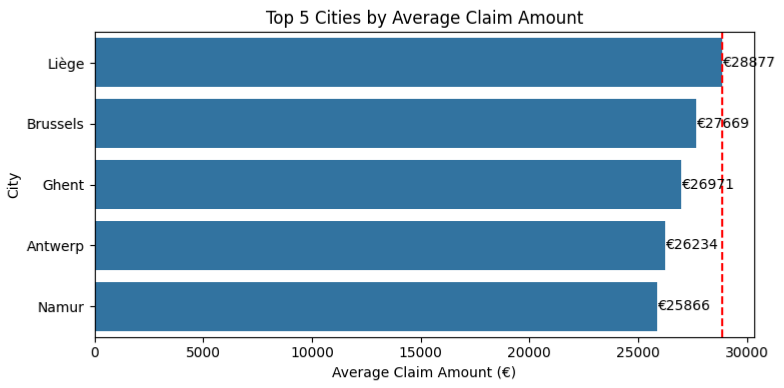
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.histplot(df['claim_amount'], kde=True)
```

```python
plt.title("Distribution of Claim Amounts")
plt.xlabel("Claim Amount")
plt.ylabel("Frequency")
plt.show()
```



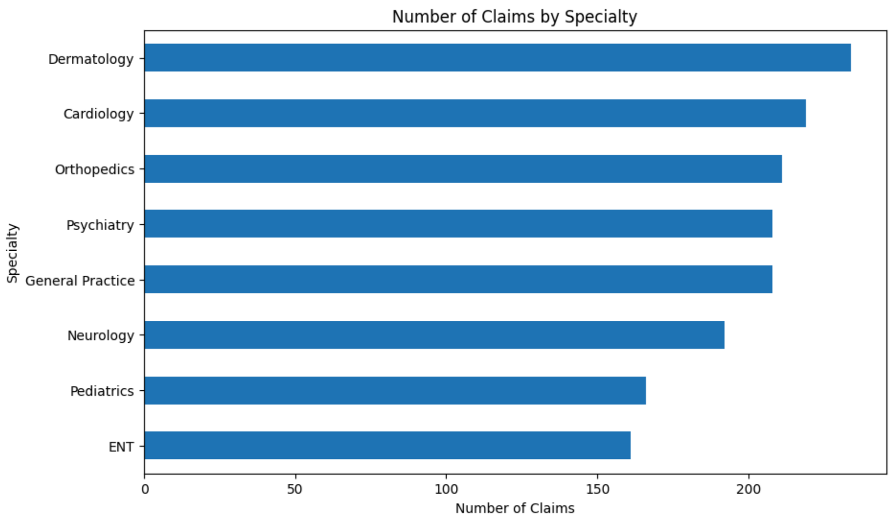
In [47]:

```python
# Calculate average claim amount per city
avg_claim_city = df.groupby('city')['claim_amount'].mean().sort_values(ascending=False).head(5)

# Prepare plot
plt.figure(figsize=(8, 4))
ax = sns.barplot(
    x=avg_claim_city.values,
    y=avg_claim_city.index
)


# Check if Liège is in the top 5 and plot the line
if 'Liège' in avg_claim_city.index:
    liege_value = avg_claim_city.loc['Liège']
    plt.axvline(liege_value, color='red', linestyle='--', label=f"Liège: €{liege_value:.2f}")

# Annotate each bar value
for i, (value, city) in enumerate(zip(avg_claim_city.values, avg_claim_city.index)):
    plt.text(value + 0.5, i, f"€{value:.0f}", va='center')

plt.title("Top 5 Cities by Average Claim Amount")
plt.xlabel("Average Claim Amount (€)")
plt.ylabel("City")
plt.tight_layout()
plt.show()
```

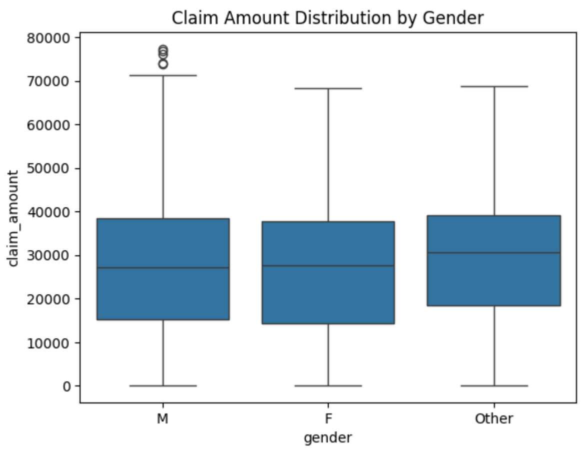


In [46]:

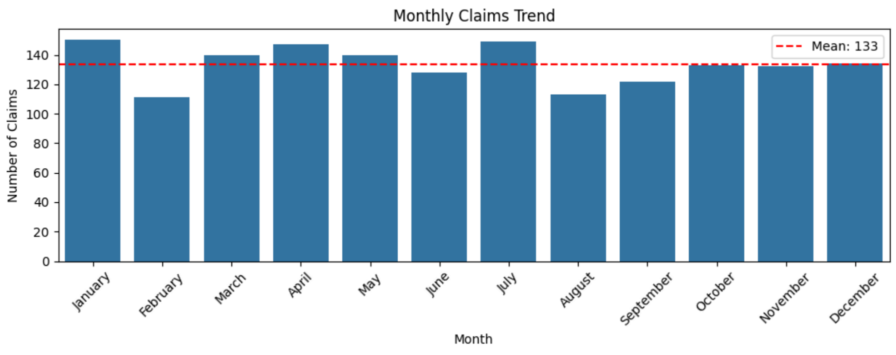
```python
claim_count_specialty = df['specialty'].value_counts().sort_values(ascending=True)

plt.figure(figsize=(10, 6))
claim_count_specialty.plot(kind='barh')
plt.title("Number of Claims by Specialty")
plt.xlabel("Number of Claims")
plt.ylabel("Specialty")
plt.show()
```



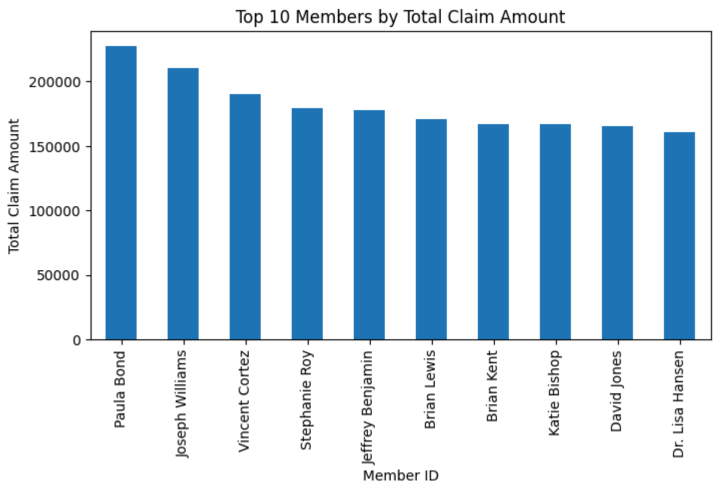
In [40]:
```
    sns.boxplot(x='gender', y='claim_amount', data=df)
    plt.title("Claim Amount Distribution by Gender")
    plt.show()
```



In [42]:
```
monthly_trend = df.groupby('month_name')['claim_id'].count().reindex(
    ['January', 'February', 'March', 'April', 'May', 'June',
     'July', 'August', 'September', 'October', 'November', 'December']
)

plt.figure(figsize=(10, 4))
sns.barplot(data=monthly_trend)

# Add horizontal mean line
plt.axhline(monthly_trend.mean(), color='red', linestyle='--', label=f"Mean: {monthly_trend.mean():.0f}")

# Add labels
plt.title("Monthly Claims Trend")
plt.xlabel("Month")
plt.ylabel("Number of Claims")
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()
```

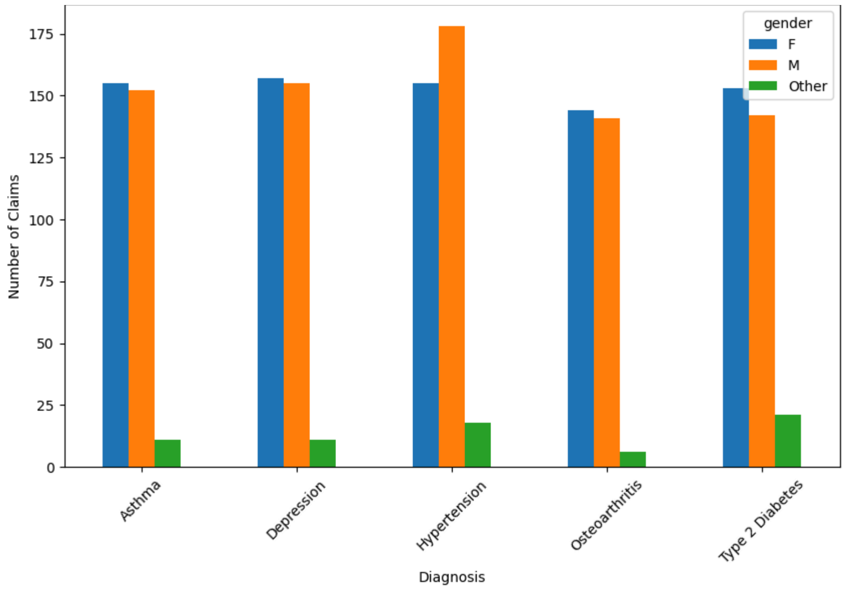


In [45]:
```
top_members = df.groupby('name')['claim_amount'].sum().sort_values(ascending=False).head(10)

top_members.plot(kind='bar', figsize=(8, 4))
plt.title("Top 10 Members by Total Claim Amount")
plt.ylabel("Total Claim Amount")
plt.xlabel("Member ID")
plt.show()
```



In [48]:
```
if 'gender' in df.columns:
    pd.crosstab(df['diagnosis_description'], df['gender']).plot(kind='bar', stacked=False, figsize=(10, 6))
    plt.title("Diagnosis Breakdown by Gender")
    plt.xlabel("Diagnosis")
    plt.ylabel("Number of Claims")
    plt.xticks(rotation=45)
    plt.show()
```

Diagnosis Breakdown by Gender


gender
F
M
Other
Number of Claims
175
150
125
100
75
50
25
0
Asthma
Depression
Hypertension
Osteoarthritis
Type 2 Diabetes
Diagnosis

```
In [ ]:
```