Kisumu-Nairobi_Preliminary_Data_Analysis

July 13, 2024

1 Assessing the effectiveness of trans-competent care education among healthcare workers on reducing stigma and discrimination and increasing uptake of HIV services and linkage to care

1.1 Kisumu Preliminary Data Analysis

Data collection in Kisumu started on 22nd March, 2024 and ended in 27th March, 2024. 24th, March 2024 was a Sunday and on advice of the trans* organizations in Kisumu data collection was not done. The collection was done using physical printed questionnaire. This was because the data tool was still under correction. On 27th, March 2024 the data collectors inputed the data using the odk data collection tool.

No respondent refused to join the study after mobilization. However, during mobilization one respondent refused to come to the study. She cited that she had a security incident before and was "staying away from the queer space for a while".

There were 63 respondents. There are 2 entries on the data base done during RA training on 21st March,2024.

[195]:

```
import pandas as pd
import sklearn as sk
import seaborn as sn
import matplotlib.pyplot as plt
import numpy as np
import re
import nltk

from nltk.sentiment.vader import SentimentIntensityAnalyzer
from datetime import datetime
from collections import Counter
from collections import defaultdict

nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /home/young/nltk_data…
[nltk_data]   Package vader_lexicon is already up-to-date!
```

[195]:

```
True
```

1.1.1 Reading Data from the excel sheet

We are going to read the data off the excel sheet and put it into a pandas object for further data analysis.

[196]:
```
excel_file = 'Nairobi-Kisumu.xlsx'

# Load the Excel file into a Pandas DataFrame
df = pd.read_excel(excel_file)
df.head(5)
len(df)
```

[196]:
```
165
```

We are then removing all entries not made on the data entry date for Kisumu(27th March, 2024) and respondents who are not eligible. We will then store the final pandas data frame object in a python variable that we will call **kisimu_respondents**.

All data collection entries for participants were made on 27th March, 2024. The rest were made during RA training and pretest. Therefore the total number of participants engaged with was

[130]:
```
len(df[df['today'] == '2024-03-27'])
```

[130]:
```
63
```

Of the 63 participants. Let us only get the number that was eligible:

[131]:
```
len(df[(df['today'] == '2024-03-27') & (df['Final Outcome'] != 0)])
```

[131]:
```
60
```

Therefore 65 entries were made using the odk data collection tool. Of the 65 only 63 were made during the data entry day for Kisumu which was 27th March, 2024. 2 entries were made during RA training and pretest & tool piloting. Of the 63 entries made during the data entry day only 60 participants were eligible.

We will proceed to store this data in the variable **kisumu_respondents**.

[132]:
```
kisumu_respondents = df[(df['today'] == '2024-03-27') & (df['Final Outcome'] !=␣
  ↪0)]
```

1.1.2 Demographics

Demographics Profile

[62]:
```
# Extracting relevant columns for socio-demographics
socio_demographics = [
                      "1. Do you know your date of birth?",
                      "1a. If yes, What is your date of birth?",
                      "2.Do you know how old you were at your last birthday?",
                      "2a.If yes, how old were you at your last birthday?",
                      "3. What is the highest level of school you attended?",
                      "4. What is your marital status?",
                      "5. What is your current religious affiliation?",
                      "Specify other",
                      "6. What is your occupation",

                       ]
kisumu_demographics = kisumu_respondents[socio_demographics]
kd = kisumu_demographics
```

[63]:

```
kd.columns =['If_Know_BD','Known_BD', 'Last_BD_Date', 'Last_BD_Age',␣
  ↪'Highest_Education', 'Marital_Status', 'Religios Affliation',␣
  ↪'Other_Religon','Occupation']
```

All the 60 respondents answered the question of whether they knew their birth year. 45 Knew their birth dates; 12 respondents didn't know; 3 didn't know if they knew their birth dates.

32 respondents didn't know their exact birthday's (who didn't know the month and year of their birth days) and we imputed their birthday to " June 15th" of the year they reported to be born.

Surprising all respondents reported to know their age in their last birthday.

[64]:

```
#Known_BD_counts = kd['Known_BD'].value_counts()
Known_BD = pd.to_datetime(kd['Known_BD'], errors='coerce')
today = datetime.today()


Age = Known_BD.apply(lambda x: today.year - x.year - ((today.month, today.day)␣
  ↪< (x.month, x.day)) if pd.notnull(x) else None)
Imputed_BD = kd[Known_BD.dt.month == 6][Known_BD.dt.day == 15]['Known_BD']

#Age.isna().sum() - number of respondents who don't know their age -15
#len(Imputed_BD) - number of respondents who didn't know the month and year of␣
  ↪their birth days
```

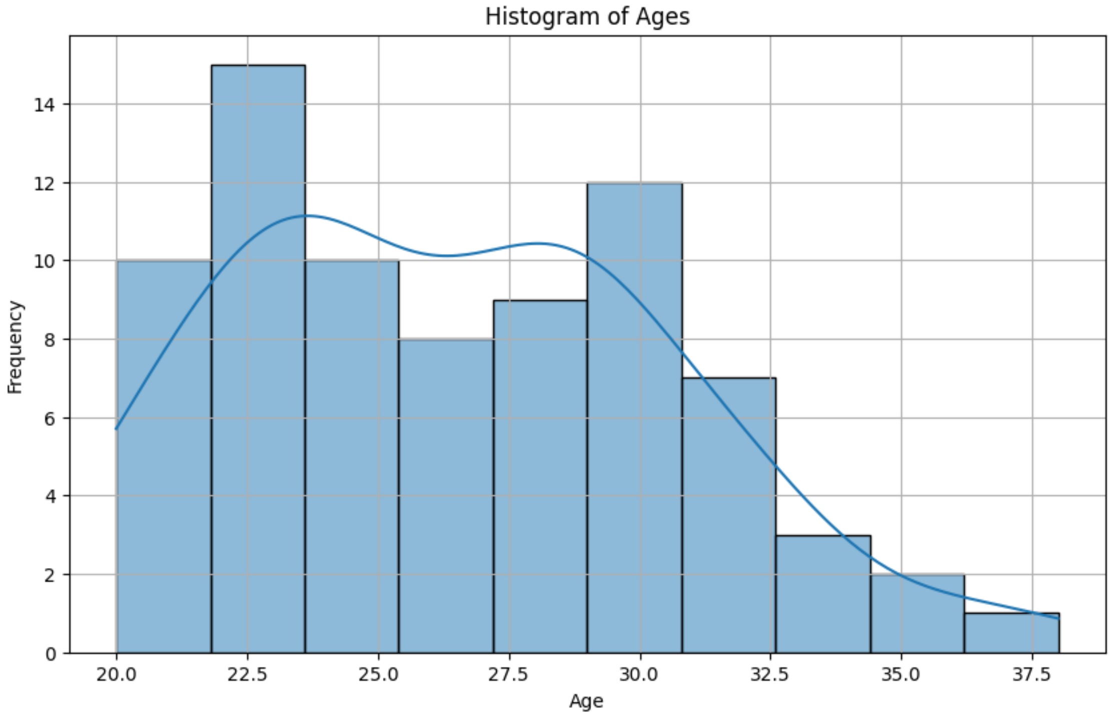
```
/tmp/ipykernel_8037/3398843229.py:7: UserWarning: Boolean Series key will be
reindexed to match DataFrame index.
  Imputed_BD = kd[Known_BD.dt.month == 6][Known_BD.dt.day == 15]['Known_BD']
```

[65]:

```
kd_ages = Age.append(Imputed_BD.apply(lambda x: today.year - x.year if pd.
  ↪notnull(x) else None))

# Remove NaN values
kd_ages = kd_ages.dropna()

# Plotting
plt.figure(figsize=(10, 6))
sn.histplot(kd_ages, bins=10, kde=True)
```

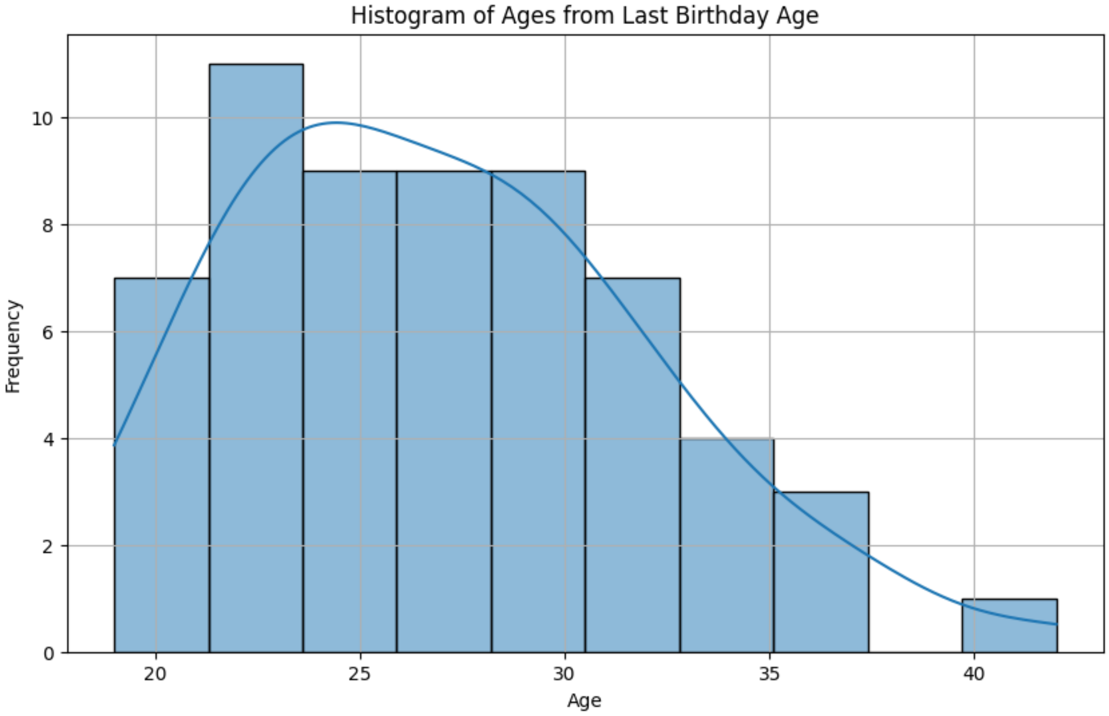
```python
plt.title('Histogram of Ages')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

```
/tmp/ipykernel_8037/2595973642.py:1: FutureWarning: The series.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  kd_ages = Age.append(Imputed_BD.apply(lambda x: today.year - x.year if
pd.notnull(x) else None))
```



[66]:

```python
# Number of people who said they knew their age in the last birthday
#len(kd[kd['Last_BD_Date'] == 'Yes'])

Age_lBD = kd['Last_BD_Age']

# Plotting
plt.figure(figsize=(10, 6))
sn.histplot(Age_lBD, bins=10, kde=True)
plt.title('Histogram of Ages from Last Birthday Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



Summary Statistics for Age

[67]:
```
print("The mean age for Kisumu respondents is:",kd_ages.mean(),"\n")
print("The median age for Kisumu respondents is:", kd_ages.median(), "\n")
print("The mode age for Kisumu respondents is:",kd_ages.mode(),"\n")

print("The mean age for Kisumu respondents based on their reported age on last␣
 ↪birthday is:",Age_lBD.mean(),"\n")
print("The median age for Kisumu respondents based on their reported age on␣
 ↪last birthday is:", Age_lBD.median(), "\n")
print("The mode age for Kisumu respondents based on their reported age on last␣
 ↪birthday is:",Age_lBD.mode(),"\n")
```

```
The mean age for Kisumu respondents is: 26.441558441558442

The median age for Kisumu respondents is: 26.0

The mode age for Kisumu respondents is: 0    23.0
Name: Known_BD, dtype: float64

The mean age for Kisumu respondents based on their reported age on last birthday
is: 27.05
```

```
The median age for Kisumu respondents based on their reported age on last
birthday is: 26.5

The mode age for Kisumu respondents based on their reported age on last birthday
is: 0    29.0
Name: Last_BD_Age, dtype: float64
```

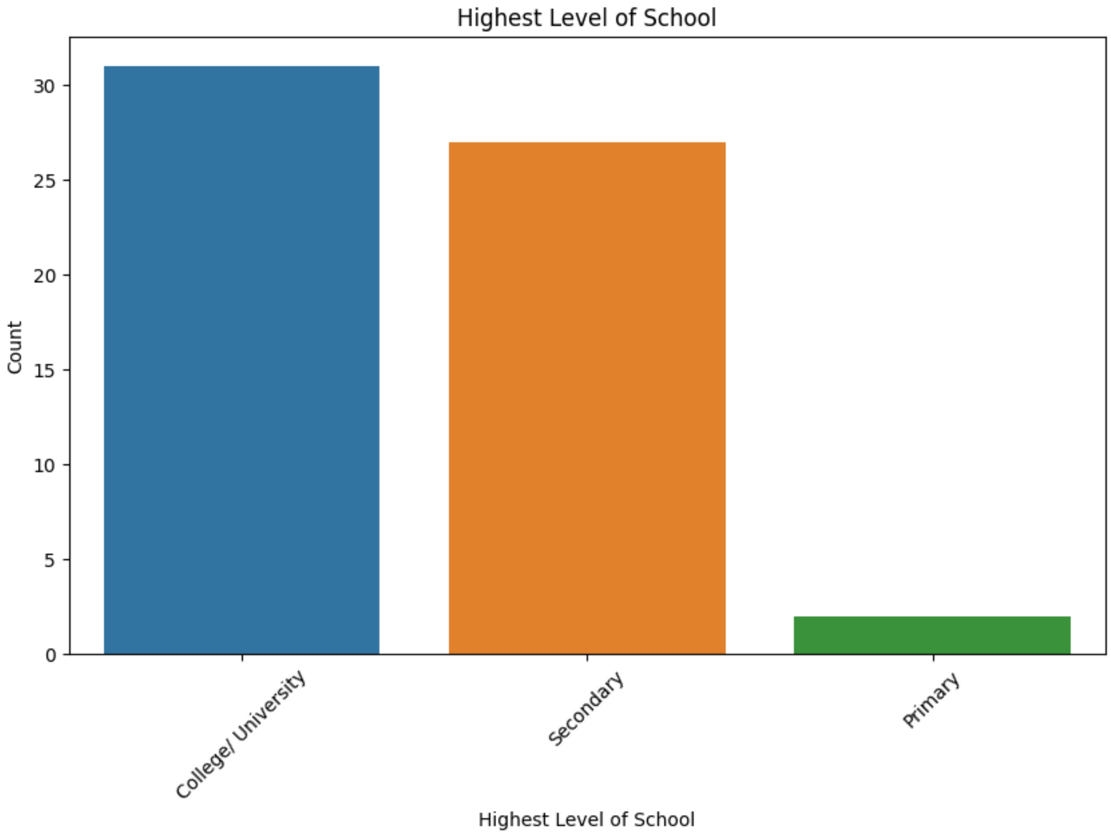
Level of Education 31 of the respondents reported to have attended University or College; 27 to have attended secondary school; and 2 to have only attended Primary School.

This means 51 1/3 % attended Tertiary Institution. 45% attended Secondary School and 3% attended Primary School.

[68]:

```
# # Valid categories
# valid_categories = ['Secondary', 'College/University', 'Primary']
# kd = kd[kd['Highest_Education'].isin(valid_categories)]
# This was truncating out Colleges and Universities
#A dist plot was also truncating Colleges and university even with the format␣
 ↪below

education = kisumu_respondents['3. What is the highest level of school you␣
 ↪attended?']
education_counts = kisumu_respondents['3. What is the highest level of school␣
 ↪you attended?'].value_counts()
education.value_counts()
education.value_counts(normalize=True)


print(education.value_counts().to_string(index=True))
plt.figure(figsize=(10, 6))
sn.countplot(data=kisumu_respondents, x='3. What is the highest level of school␣
 ↪you attended?', order=education_counts.index)
plt.title('Highest Level of School')
plt.xlabel('Highest Level of School')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
College/ University    31
Secondary              27
Primary                 2
```

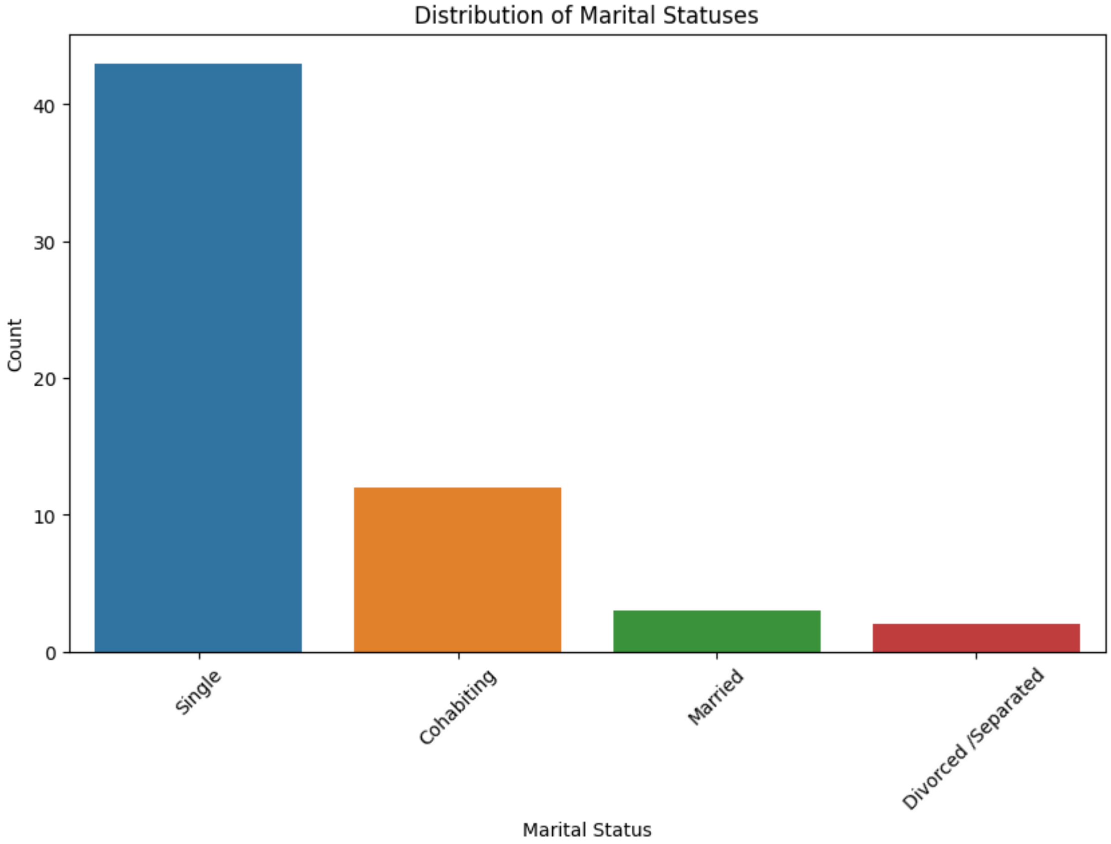


Marital Status 43 respondents reported their marital status as being single; 12 reported as cohabiting; 3 as married and 2 as separated.

[69]:
```
marital_status_counts = kisumu_respondents['4. What is your marital status?'].
  ↪value_counts()

print(marital_status_counts.to_string(index=True))

plt.figure(figsize=(10, 6))
sn.countplot(data=kisumu_respondents, x='4. What is your marital status?',␣
  ↪order=marital_status_counts.index)
plt.title('Distribution of Marital Statuses')
plt.xlabel('Marital Status')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
Single                  43
Cohabiting              12
Married                  3
```

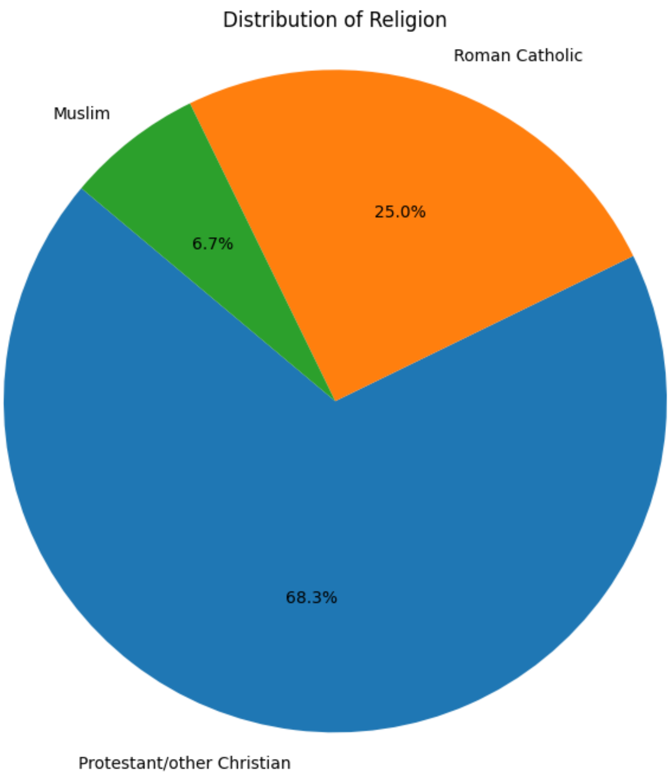
```
Divorced /Separated     2
```



[70]:
```
#### Religion
```

[71]:
```
religion = kisumu_respondents['5. What is your current religious affiliation?']
religion_counts = religion.value_counts()
print(religion_counts.to_string(index=True))

# Plotting the data
plt.figure(figsize=(12, 8))
plt.pie(religion_counts.values, labels=religion_counts.index, autopct='%1.
  ↪1f%%', startangle=140)
plt.title('Distribution of Religion')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

```
Protestant/other Christian    41
Roman Catholic                15
Muslim                         4
```



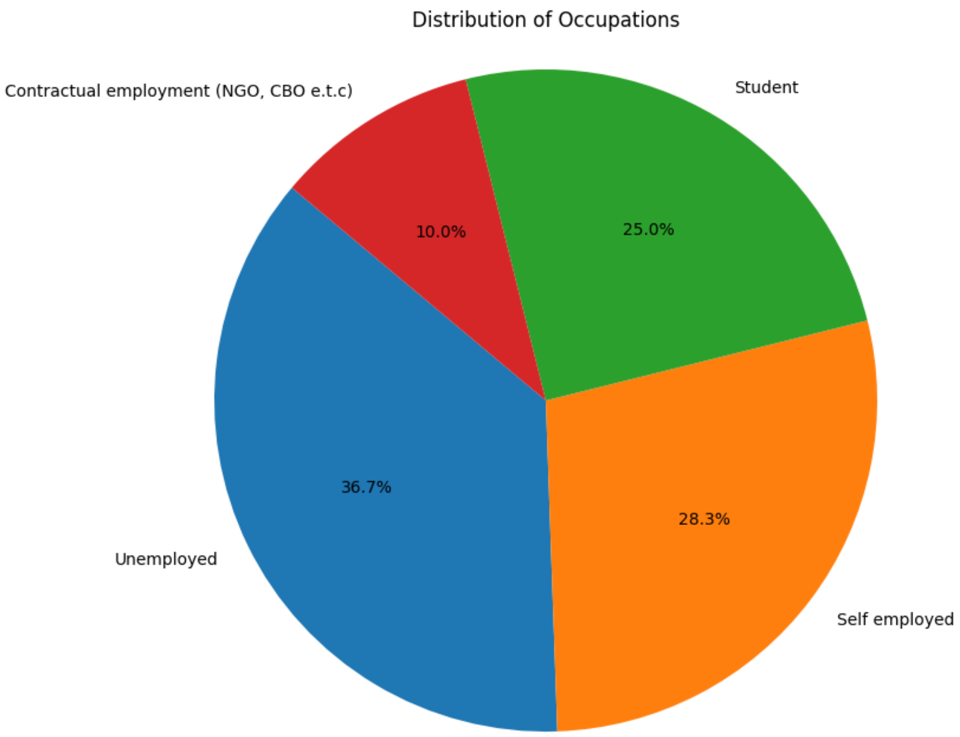
[72]:

```
#### Occupation
```

[73]:

```
occupation= kisumu_respondents['6. What is your occupation']
occupation_counts = occupation.value_counts()
print(occupation_counts.to_string(index=True))


# Plotting the data
plt.figure(figsize=(12, 8))
plt.pie(occupation_counts.values, labels=occupation_counts.index, autopct='%1.
  ↪1f%%', startangle=140)
plt.title('Distribution of Occupations')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

```
Unemployed                                22
Self employed                             17
Student                                   15
Contractual employment (NGO, CBO e.t.c)    6
```



1.1.3 Summary Statistics for Demographics

[ ]:

1.1.4 Sexual Behavior

Each and every one of 60 Kisumu respondents reported to have had sex.

[74]:

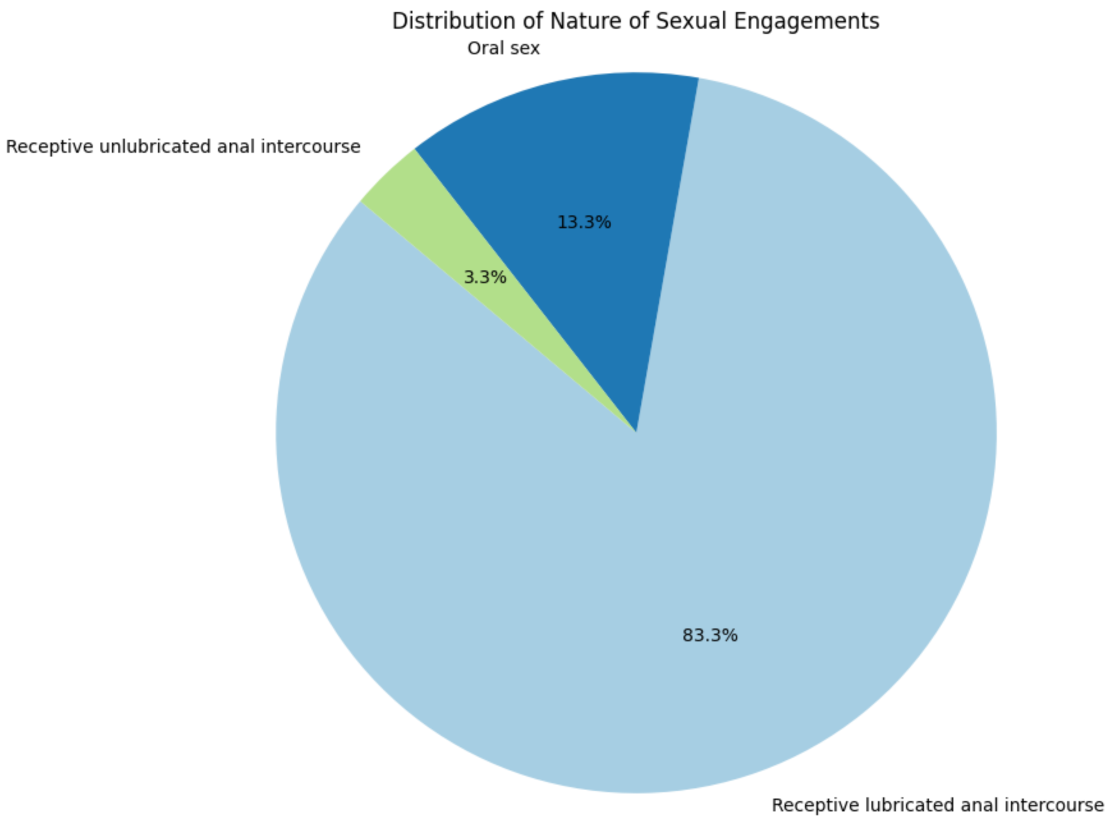
```
sexual_intercourse = kisumu_respondents['7. In the past 12 months, have you ever had sexual intercourse?']

# Counting the occurrences of each response
sexual_intercourse_counts = sexual_intercourse.value_counts()
print(sexual_intercourse_counts.to_string(index=True))
print('\nPercentages\n',sexual_intercourse.value_counts(normalize=True).to_string(index=True))
```

```
Yes    60

Percentages
 Yes    1.0
```

[75]:

```
nature_of_sex =  kisumu_respondents['8. In the past 12 months you had sex, what␣
  ↪was the nature of the sexual engagement?']
nature_of_sex_counts = nature_of_sex.value_counts()

print(nature_of_sex_counts.to_string(index=True))
print('\nPercentages\n',nature_of_sex.value_counts(normalize=True).
  ↪to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(nature_of_sex_counts, labels=nature_of_sex_counts.index, autopct='%1.
  ↪1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Distribution of Nature of Sexual Engagements')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

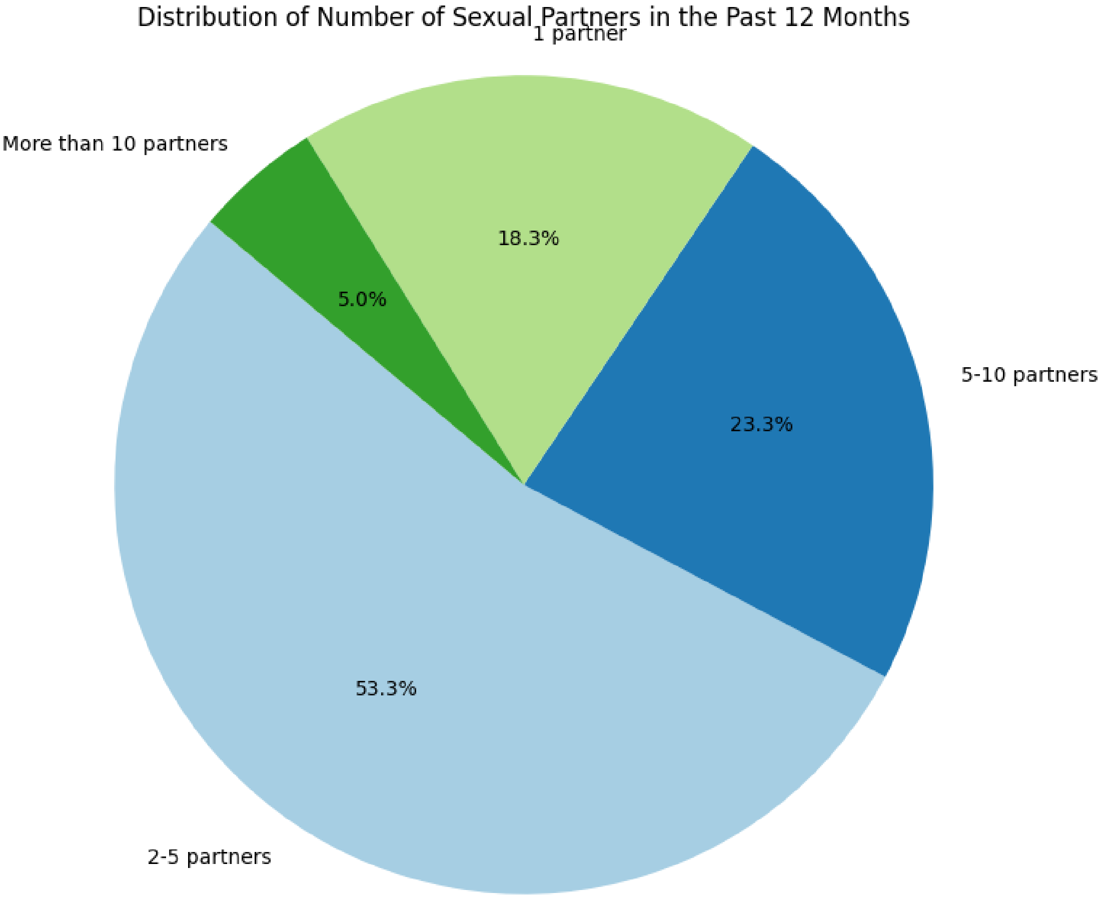
```
Receptive lubricated anal intercourse      50
Oral sex                                    8
Receptive unlubricated anal intercourse     2

Percentages
 Receptive lubricated anal intercourse      0.833333
Oral sex                                   0.133333
Receptive unlubricated anal intercourse    0.033333
```



```
[76]: partners_no =  kisumu_respondents['9. How many partners did you have sex with␣
        ↪in the past 12 months?']
      partners_no_counts = partners_no.value_counts()

      print(partners_no_counts.to_string(index=True))
      print('\nPercentages\n',partners_no.value_counts(normalize=True).
        ↪to_string(index=True))

      # Plotting the counts as a pie chart
      plt.figure(figsize=(8, 8))
      plt.pie(partners_no_counts, labels=partners_no_counts.index, autopct='%1.1f%%',␣
        ↪startangle=140, colors=plt.cm.Paired.colors)
      plt.title('Distribution of Number of Sexual Partners in the Past 12 Months')
      plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

      # Display the plot
      plt.show()
```

```
2-5 partners            32
5-10 partners           14
1 partner               11
```

```
More than 10 partners     3

Percentages
 2-5 partners              0.533333
5-10 partners             0.233333
1 partner                 0.183333
More than 10 partners     0.050000
```

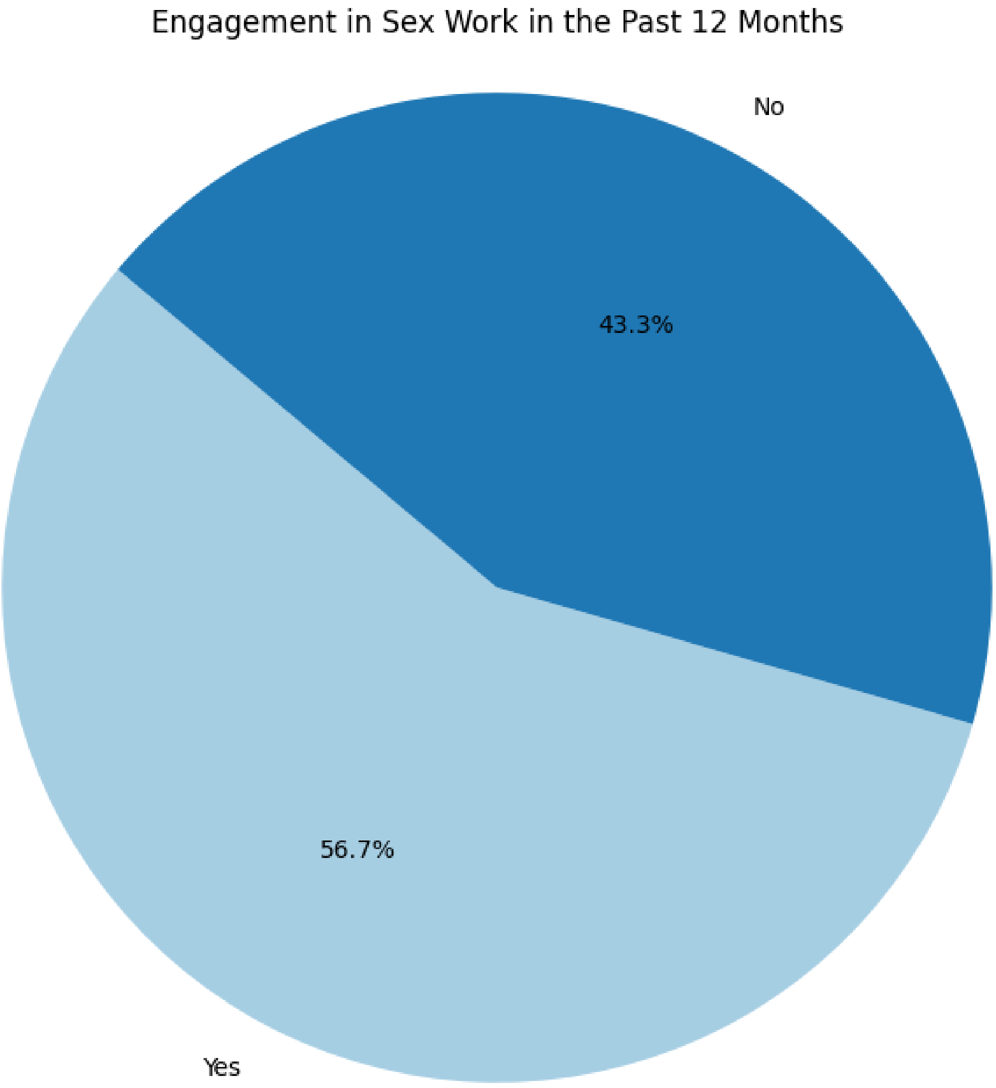


[77]:

```
sex_work = kisumu_respondents['10. In the past 12 months, did you have sex in␣
 ↪exchange for money, favours or goods?']

sex_work_counts = sex_work.value_counts()

print(sex_work_counts.to_string(index=True))

plt.figure(figsize=(8, 8))
plt.pie(sex_work_counts, labels=sex_work_counts.index, autopct='%1.1f%%',␣
 ↪startangle=140, colors=plt.cm.Paired.colors)
```

```
plt.title('Engagement in Sex Work in the Past 12 Months')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

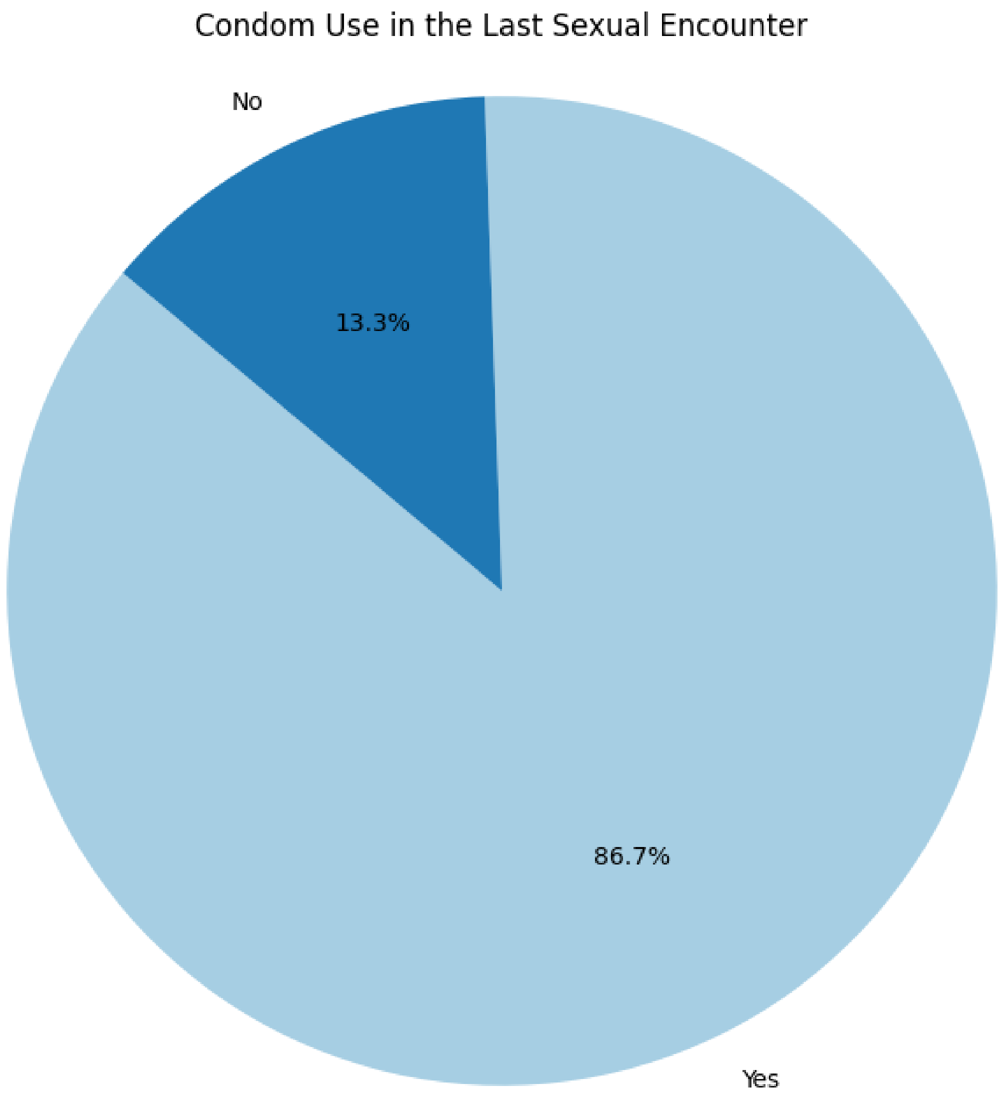
```
Yes    34
No     26
```



[78]:
```
# Extracting the relevant column
condom_use = kisumu_respondents['11. Did you use a condom the last time you had␣
↪sex with in the past 12 months?']
```

```
# Counting the occurrences of each response
condom_use_counts = condom_use.value_counts()

# Print percentages
print('\nPercentages\n', condom_use_counts.value_counts(normalize=True).
 ↪to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(condom_use_counts, labels=condom_use_counts.index, autopct='%1.1f%%',␣
 ↪startangle=140, colors=plt.cm.Paired.colors)
plt.title('Condom Use in the Last Sexual Encounter')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

```
Percentages
 52    0.5
8     0.5
```



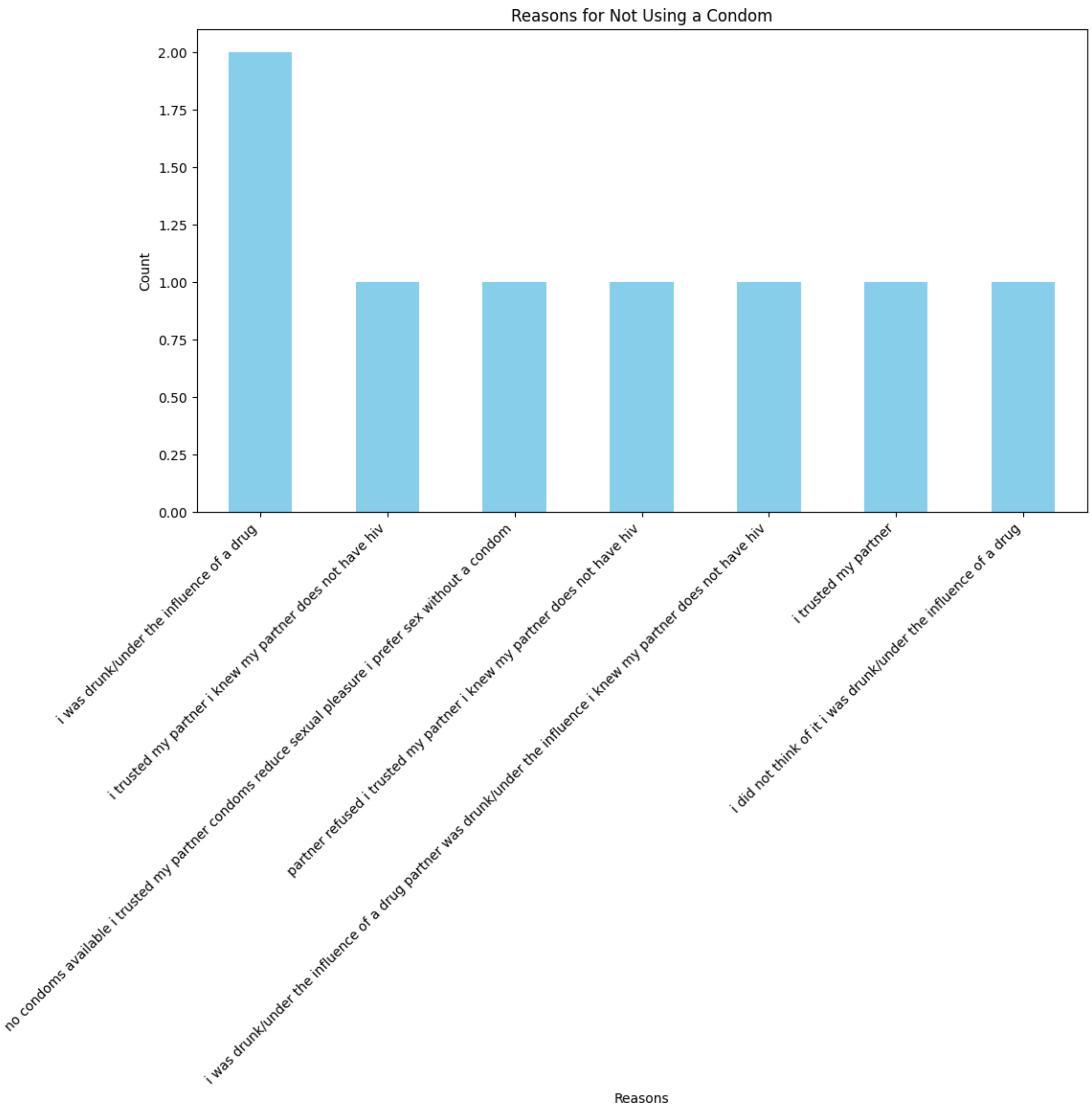
[79]:

```
# Extracting the relevant column
reasons_for_no_condom = kisumu_respondents['12.why  did you not use a condom?']

# Cleaning and normalizing text (optional depending on data cleanliness)
reasons_for_no_condom = reasons_for_no_condom.str.strip().str.lower()

# Counting the occurrences of each reason
reason_counts = reasons_for_no_condom.value_counts()



print('Reasons', reason_counts.to_string(index=True))
print('Reason count sum', reason_counts.sum() )
```

```
# Print percentages if needed
print('\nPercentages\n', reasons_for_no_condom.value_counts(normalize=True).
 ↪to_string(index=True))

# Plotting the counts as a bar chart
plt.figure(figsize=(12, 8))  # Adjust the figure size as needed
reason_counts.plot(kind='bar', color='skyblue')
plt.title('Reasons for Not Using a Condom')
plt.xlabel('Reasons')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')

# Adjusting margins if necessary
plt.subplots_adjust(bottom=0.25)  # Example adjustment, increase if needed

# Display the plot
plt.show()
```

```
Reasons i was drunk/under the influence of a drug
2
i trusted my partner i knew my partner does not have hiv
1
no condoms available i trusted my partner condoms reduce sexual pleasure i
prefer sex without a condom                 1
partner refused i trusted my partner i knew my partner does not have hiv
1
i was drunk/under the influence of a drug partner was drunk/under the influence
i knew my partner does not have hiv    1
i trusted my partner
1
i did not think of it i was drunk/under the influence of a drug
1
Reason count sum 8

Percentages
 i was drunk/under the influence of a drug
0.250
i trusted my partner i knew my partner does not have hiv
0.125
no condoms available i trusted my partner condoms reduce sexual pleasure i
prefer sex without a condom                 0.125
partner refused i trusted my partner i knew my partner does not have hiv
0.125
i was drunk/under the influence of a drug partner was drunk/under the influence
i knew my partner does not have hiv    0.125
i trusted my partner
0.125
```

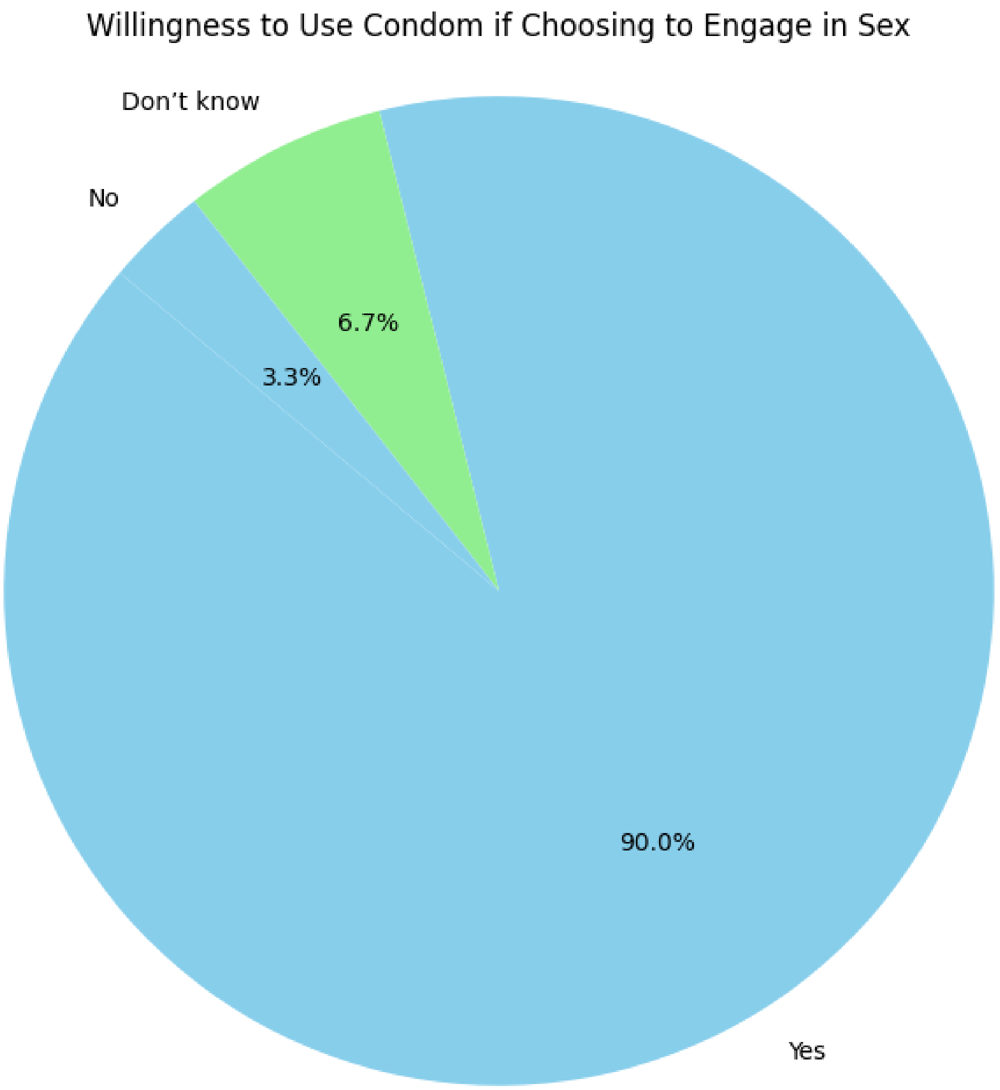
```
i did not think of it i was drunk/under the influence of a drug
0.125
```



[80]:
```
willing_to_use_condom = kisumu_respondents['13. Should you choose to engage in␣
 ↪sex, would you be willing to use condom?']

# Counting the occurrences of each response
willing_to_use_condom_counts = willing_to_use_condom.value_counts()


print(willing_to_use_condom_counts.to_string(index=True))
print('\nPercentages\n', willing_to_use_condom.value_counts(normalize=True).
 ↪to_string(index=True))
```

```python


# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(willing_to_use_condom_counts, labels=willing_to_use_condom_counts.
 ↪index, autopct='%1.1f%%', startangle=140, colors=['skyblue', 'lightgreen'])
plt.title('Willingness to Use Condom if Choosing to Engage in Sex')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

```
Yes             54
Don’t know       4
No               2

Percentages
 Yes             0.900000
Don’t know    0.066667
No            0.033333
```



[81]:
```
last_sex_date = kisumu_respondents['14. If No when was the last time you had␣
 ↪sex?']

# Convert to datetime if not already in datetime format
#last_sex_date = pd.to_datetime(last_sex_date, errors='coerce')

# Counting the occurrences of each date
last_sex_date_counts = last_sex_date.value_counts()
len(last_sex_date_counts)
#For Kisumu last sex date if no was 0

# # Convert to datetime if not already in datetime format
# last_sex_date = pd.to_datetime(last_sex_date, errors='coerce')

# # Counting the occurrences of each date
# last_sex_date_counts = last_sex_date.value_counts().sort_index()

# # Plotting the counts as a line chart
# plt.figure(figsize=(10, 6))
# last_sex_date_counts.plot(marker='o')
# plt.title('Last Time Respondents Had Sex')
# plt.xlabel('Date')
# plt.ylabel('Count')
# plt.grid(True)
# plt.xticks(rotation=45)

# # Display the plot
# plt.tight_layout()
# plt.show()


##This forces us to skip to HIV Perception questions, from No. 21
```

[81]: 0

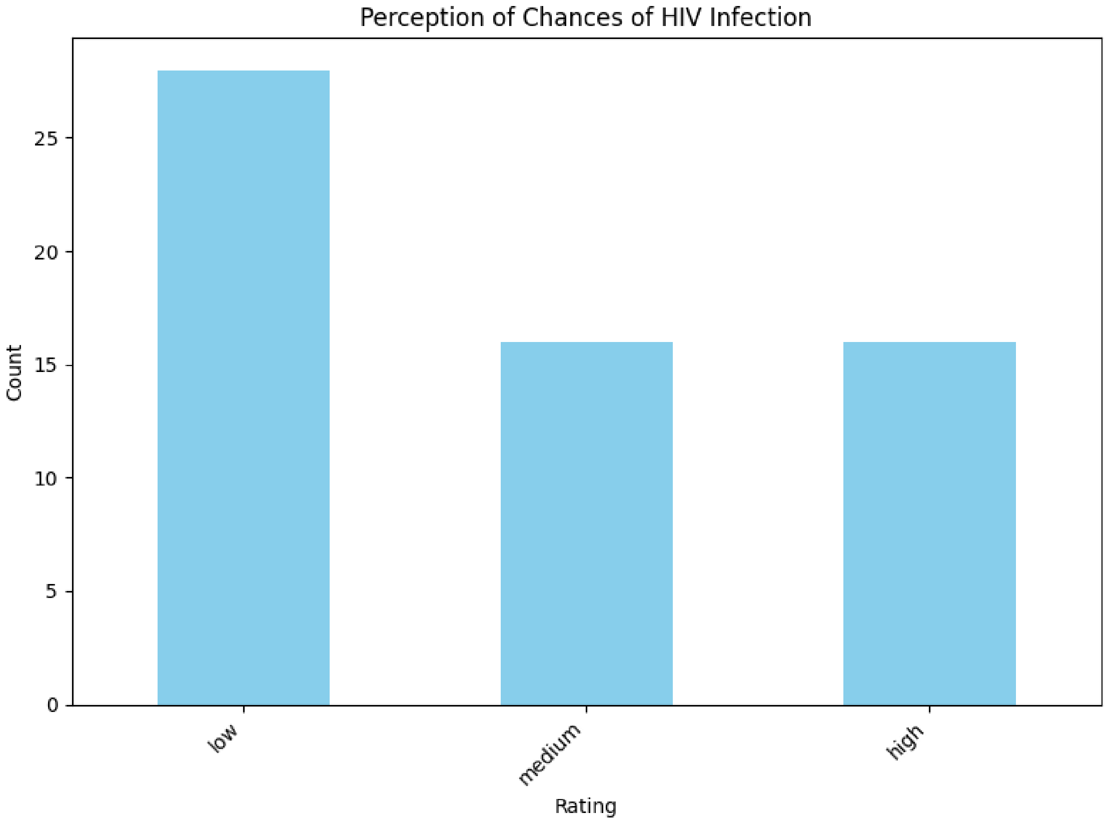
1.1.5 HIV Risk Perception

[82]:

```
# Extracting the relevant column
hiv_infection_chances = kisumu_respondents['21. How do you rate your chances of␣
 ↪HIV infection?']

# Cleaning and normalizing text (optional depending on data cleanliness)
hiv_infection_chances = hiv_infection_chances.str.strip().str.lower()

# Counting the occurrences of each category
chances_counts = hiv_infection_chances.value_counts()

print(chances_counts.to_string(index=True))
print('\nPercentages\n', hiv_infection_chances.value_counts(normalize=True).
 ↪to_string(index=True))


# Plotting the counts as a bar chart
plt.figure(figsize=(8, 6))
chances_counts.plot(kind='bar', color='skyblue')
plt.title('Perception of Chances of HIV Infection')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
```

```
# Display the plot
plt.tight_layout()
plt.show()
```

```
low       28
medium    16
high      16

Percentages
 low        0.466667
medium    0.266667
high      0.266667
```

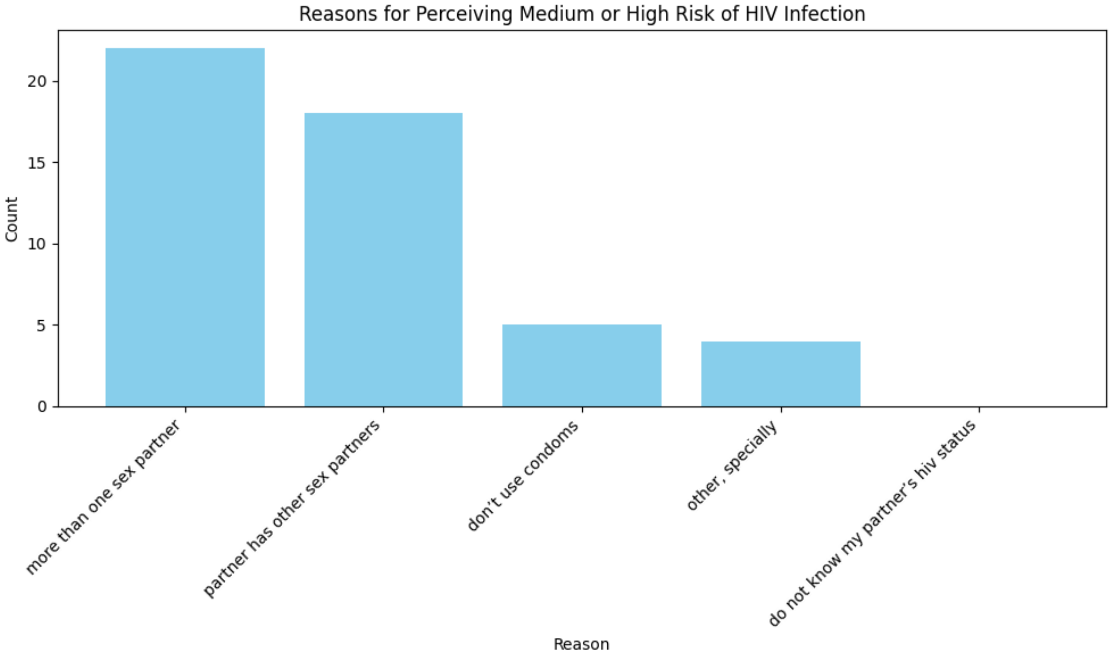


[83]:
```
risk_perception_reasons = kisumu_respondents['22. Why do you think you have a␣
 ↪medium or high risk of getting HIV?']

# Cleaning and normalizing text (optional depending on data cleanliness)
risk_perception_reasons = risk_perception_reasons.str.strip().str.lower()

# Defining key phrases for analysis
```

```python
key_phrases = [
    'more than one sex partner',
    'don't use condoms',
    'do not know my partner's hiv status',
    'partner has other sex partners',
    'other, specially'
]

# Counting occurrences of key phrases
phrase_counts = {}
for phrase in key_phrases:
    phrase_counts[phrase] = risk_perception_reasons.str.contains(phrase).sum()

# Print the counts of each key phrase
print("Counts:\n", pd.Series(phrase_counts).to_string(index=True))

# Calculate and print the percentages of each key phrase
percentages = pd.Series(phrase_counts).value_counts(normalize=True) * 100
print("\nPercentages:\n", percentages.to_string(index=True))

# Sorting the results by frequency
sorted_counts = {k: v for k, v in sorted(phrase_counts.items(), key=lambda item:
  ↪ item[1], reverse=True)}

# Plotting the counts as a bar chart
plt.figure(figsize=(10, 6))
plt.bar(sorted_counts.keys(), sorted_counts.values(), color='skyblue')
plt.title('Reasons for Perceiving Medium or High Risk of HIV Infection')
plt.xlabel('Reason')
plt.ylabel('Count')

# Display the plot
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
Counts:
 more than one sex partner              22
don't use condoms                       5
do not know my partner's hiv status     0
partner has other sex partners         18
other, specially                        4

Percentages:
 22    20.0
5     20.0
0     20.0
```

```
18     20.0
4      20.0
```



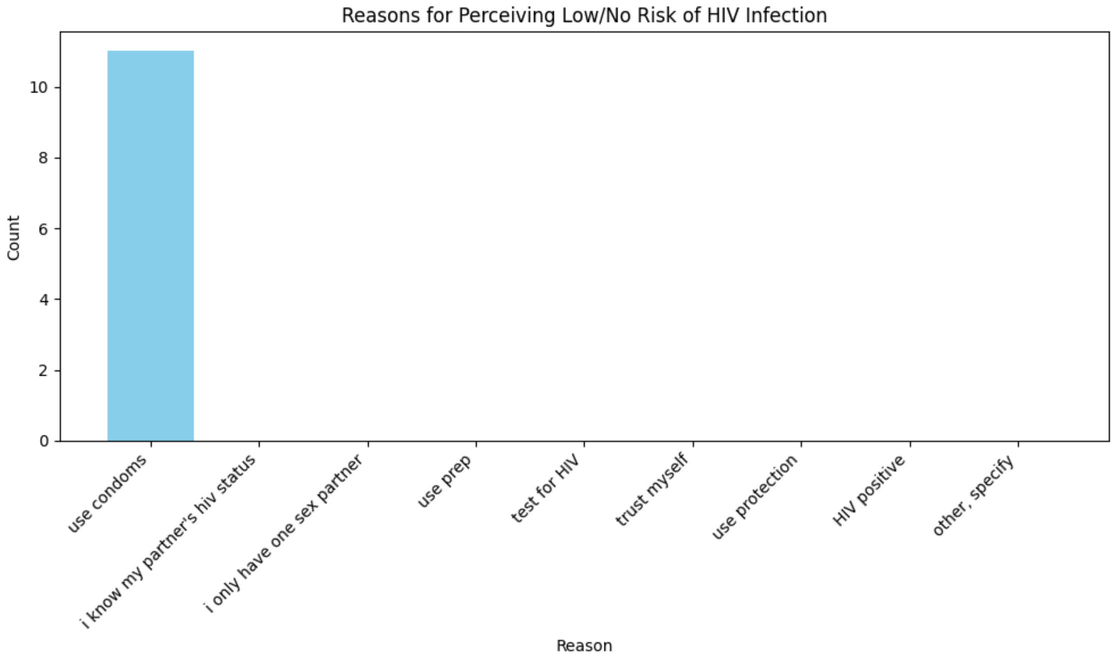
[84]:

```
risk_perception_low_no = kisumu_respondents['23. If low or no,Why do you think␣
↪you have a low/ no risk chance of getting HIV?']

# Cleaning and normalizing text for 'Others Specify'
def clean_other_specify(text):
    if pd.isna(text):
        return ''
    text = text.strip().lower()  # Remove leading/trailing spaces and convert␣
↪to lowercase
    text = re.sub(r'\s+', ' ', text)  # Remove extra spaces
    # Normalize specific responses
    text = re.sub(r'\buse of condoms\b', 'use condoms', text)
    text = re.sub(r'\buse condom\b', 'use condoms', text)
    text = re.sub(r'\btest for hiv before sex\b', 'test for HIV', text)
    text = re.sub(r'\bgo for testing and use condom\b', 'test for HIV', text)
    text = re.sub(r'\btest for hiv\b', 'test for HIV', text)
    text = re.sub(r'\btest before sex\b', 'test for HIV', text)
    text = re.sub(r'\bi trust myself\b', 'trust myself', text)
    text = re.sub(r'\buse prep\b', 'use PrEP', text)
    text = re.sub(r'\buse prep and condom\b', 'use PrEP and condom', text)
    text = re.sub(r'\buse protection\b', 'use protection', text)
    text = re.sub(r'\buse of protection everytime engaging in sex\b', 'use␣
↪protection', text)
```

```
    text = re.sub(r'\bparticipate uses prep\b', 'use PrEP', text)
    text = re.sub(r'\buse protection and aware of sex partner status\b', 'use protection and know partner HIV status', text)
    text = re.sub(r'\bi use protection and have low sexual partners\b', 'use protection and low number of sexual partners', text)
    text = re.sub(r'\bvery few sexual partners\b', 'use protection and low number of sexual partners', text)
    text = re.sub(r'\bthe last hiv test was negative\b', 'test for HIV', text)
    text = re.sub(r'\brespondent is hiv\b', 'HIV positive', text)
    text = re.sub(r'\bthe participant uses prep/date prep\b', 'HIV positive', text)
    return text

# Apply the cleaning function to the column
risk_perception_low_no = risk_perception_low_no.apply(clean_other_specify)

# Define key phrases for analysis
key_phrases = [
    'use condoms',
    'i know my partner\'s hiv status',
    'i only have one sex partner',
    'use prep',
    'test for HIV',
    'trust myself',
    'use protection',
    'HIV positive',
    'other, specify'
]

# Initialize dictionary to store counts
phrase_counts = {phrase: 0 for phrase in key_phrases}

# Count occurrences of key phrases
for phrase in key_phrases:
    phrase_counts[phrase] = risk_perception_low_no.str.contains(phrase).sum()

# Print the counts of each key phrase
print("Counts:\n", pd.Series(phrase_counts).to_string())

# Optionally, calculate and print the percentages of each key phrase
percentages = pd.Series(phrase_counts).div(len(risk_perception_low_no)) * 100
print("\nPercentages:\n", percentages.to_string())

# Optionally, plot the counts as a bar chart
plt.figure(figsize=(10, 6))
plt.bar(phrase_counts.keys(), phrase_counts.values(), color='skyblue')
plt.title('Reasons for Perceiving Low/No Risk of HIV Infection')
```

```python
plt.xlabel('Reason')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
Counts:
 use condoms                         11
i know my partner's hiv status      0
i only have one sex partner         0
use prep                            0
test for HIV                        0
trust myself                        0
use protection                      0
HIV positive                        0
other, specify                      0

Percentages:
 use condoms                         18.333333
i know my partner's hiv status      0.000000
i only have one sex partner         0.000000
use prep                            0.000000
test for HIV                        0.000000
trust myself                        0.000000
use protection                      0.000000
HIV positive                        0.000000
other, specify                      0.000000
```



[85]:

```
sid = SentimentIntensityAnalyzer()

# Function to clean text
def clean_text(text):
    # Remove special characters and numbers using regex
    text = re.sub(r'[^a-zA-Z\s]', '', text)

    # Convert to lowercase
    text = text.lower()

    # Remove extra whitespace
    text = ' '.join(text.split())

    return text
```

[86]:

```
cleaned_responses = risk_perception_low_no.dropna().apply(clean_text)
sentiments = cleaned_responses.apply(lambda x: sid.polarity_scores(x))



# Combine sentiments with responses for sorting
sentiments_df = pd.DataFrame(list(sentiments))
sentiments_df['Response'] = cleaned_responses.values

# Sort by compound sentiment score
sorted_responses = sentiments_df.sort_values(by='compound',␣
 ↪ascending=False)['Response']

# Print sorted responses
for i, response in enumerate(sorted_responses):
    print(f"{i+1}. {response}")
```

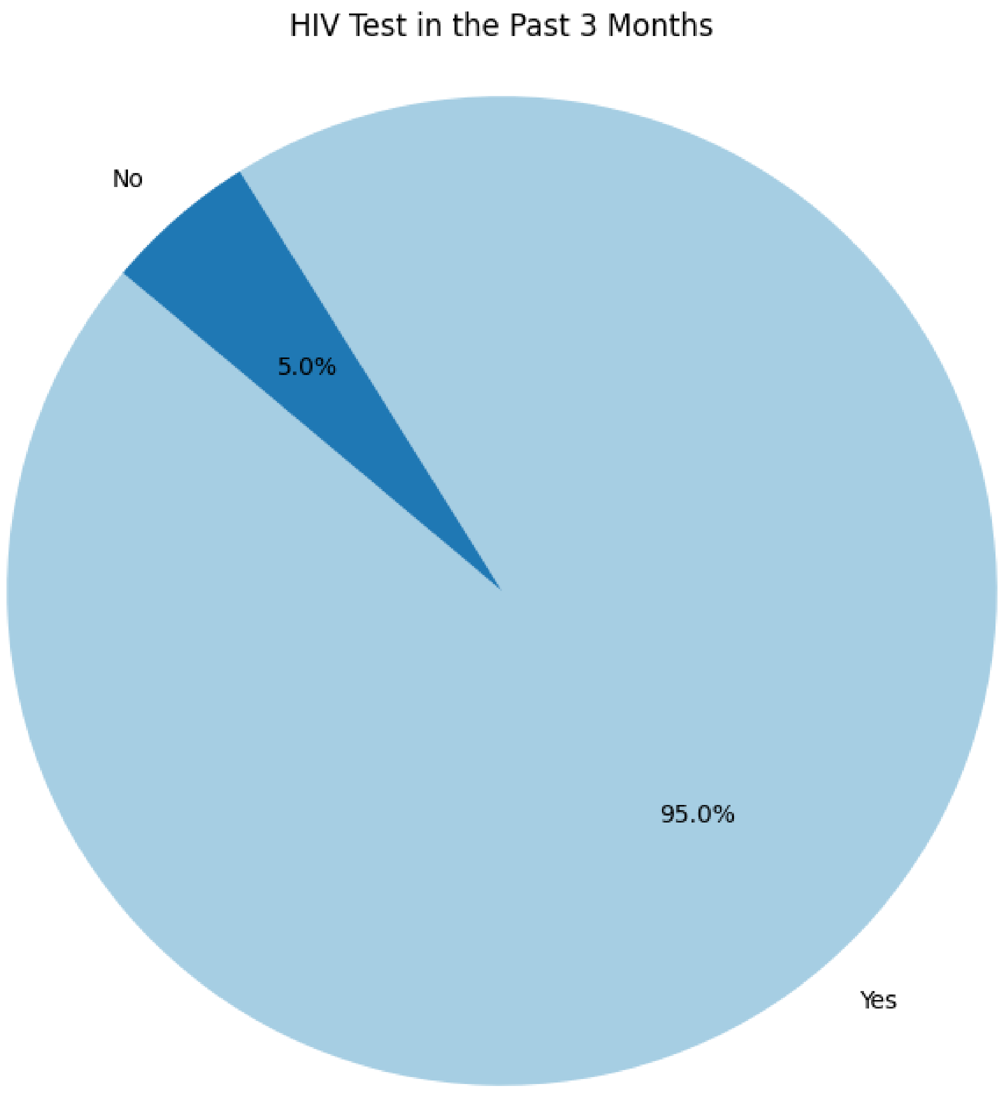
```
1.
2. other specially
3. other specially
4. other specially
5. other specially
6. other specially
7.
8.
9.
10.
11.
12.
13.
14.
```

```
15. my partner has other sex partners
16.
17. other specially
18.
19. i do not know my partners hiv status
20.
21. other specially
22.
23. i dont use condoms
24.
25. i dont use condoms
26. other specially
27.
28.
29. i dont use condoms
30.
31.
32. i dont use condoms i do not know my partners hiv status
33. i dont use condoms
34. other specially
35.
36.
37.
38. other specially
39. other specially
40. other specially
41. other specially
42.
43.
44. i dont use condoms
45. i dont use condoms
46. i dont use condoms
47. i dont use condoms
48.
49. i have more than one sex partner
50. i dont use condoms
51.
52.
53.
54.
55.
56.
57.
58.
59. i do not know my partners hiv status
60. i dont use condoms
```

HIV Testing

```
[87]: hiv_test_responses = kisumu_respondents['24. In the past 3 months, have you␣
        ↪ever gone for a HIV test?']

      # Counting the occurrences of each response
      hiv_test_counts = hiv_test_responses.value_counts()



      # Print the counts
      print(hiv_test_counts.to_string(index=True))
      print('\nPercentages\n', hiv_test_responses.value_counts(normalize=True).
        ↪to_string(index=True))



      # Plotting the counts as a pie chart
      plt.figure(figsize=(8, 8))
      plt.pie(hiv_test_counts, labels=hiv_test_counts.index, autopct='%1.1f%%',␣
        ↪startangle=140, colors=plt.cm.Paired.colors)
      plt.title('HIV Test in the Past 3 Months')
      plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
      plt.show()
```

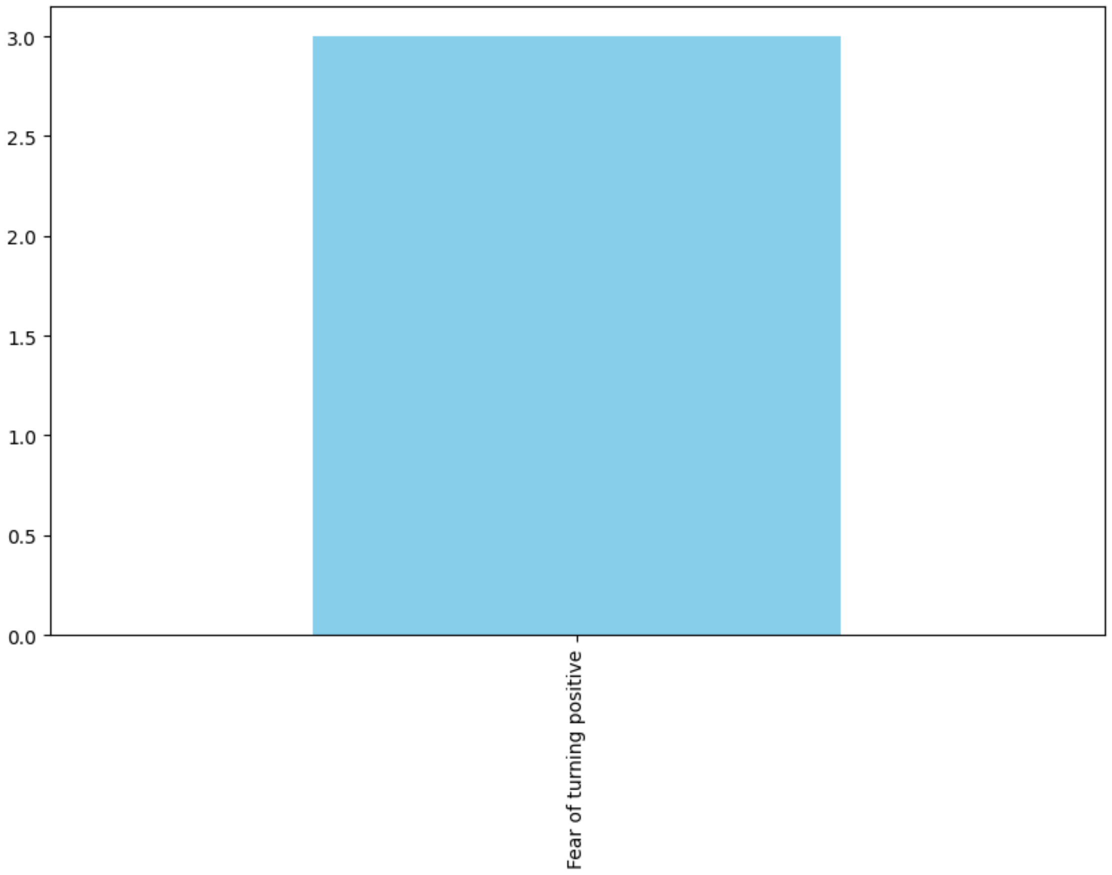
```
Yes    57
No      3

Percentages
 Yes    0.95
No     0.05
```



```
[88]: no_hiv_test_reasons = kisumu_respondents['25. If No, why have you not gone for␣
        ↪a HIV test?']

      # Counting the occurrences of each reason
      reason_counts = no_hiv_test_reasons.value_counts()

      # Print the counts
      print(reason_counts.to_string(index=True))

      # Plotting the counts as a bar chart
      plt.figure(figsize=(10, 6))
      reason_counts.plot(kind='bar', color='skyblue')
```

```
Fear of turning positive    3
```

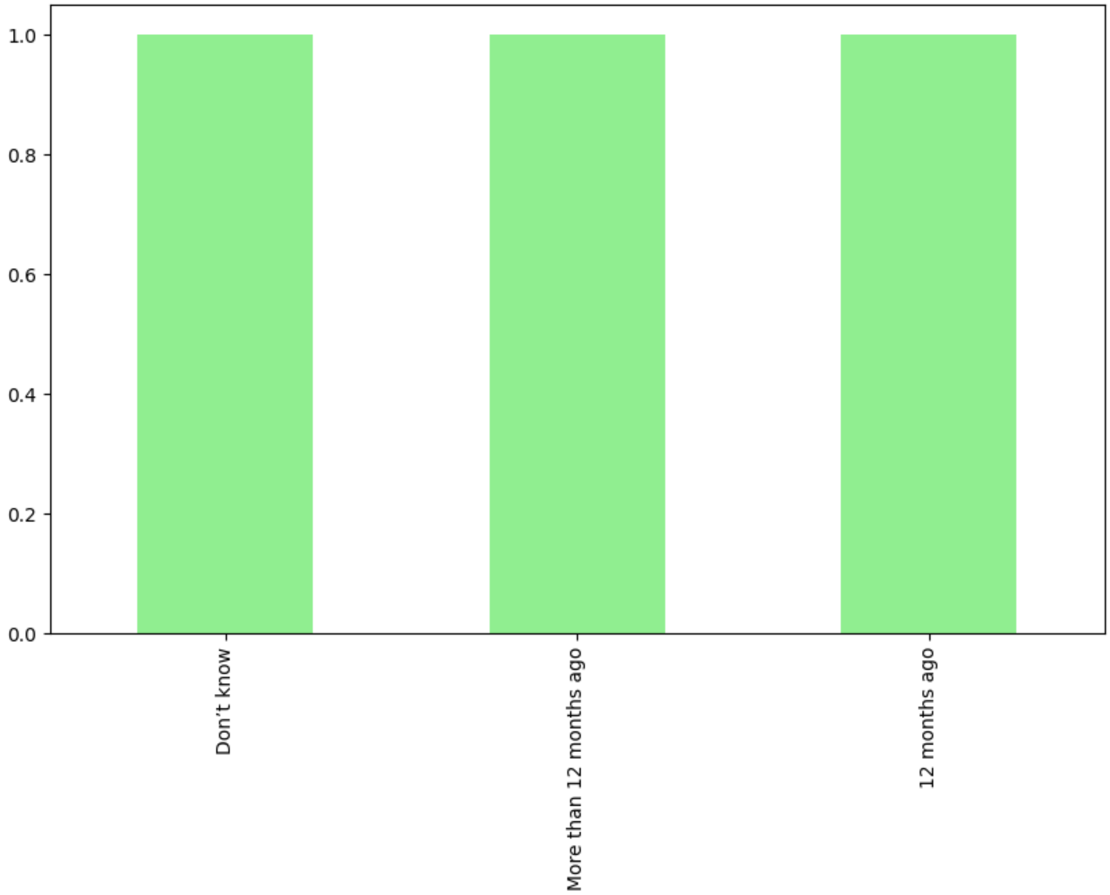
[88]: <AxesSubplot: >



[89]:
```
last_testing_times = kisumu_respondents['26. If No, when was the last time you␣
↪went for testing?']

# Counting the occurrences of each response
time_counts = last_testing_times.value_counts()

# Print the counts
print(time_counts.to_string(index=True))

# Plotting the counts as a bar chart
plt.figure(figsize=(10, 6))
time_counts.plot(kind='bar', color='lightgreen')
```

```
Don't know                 1
More than 12 months ago    1
12 months ago              1
```

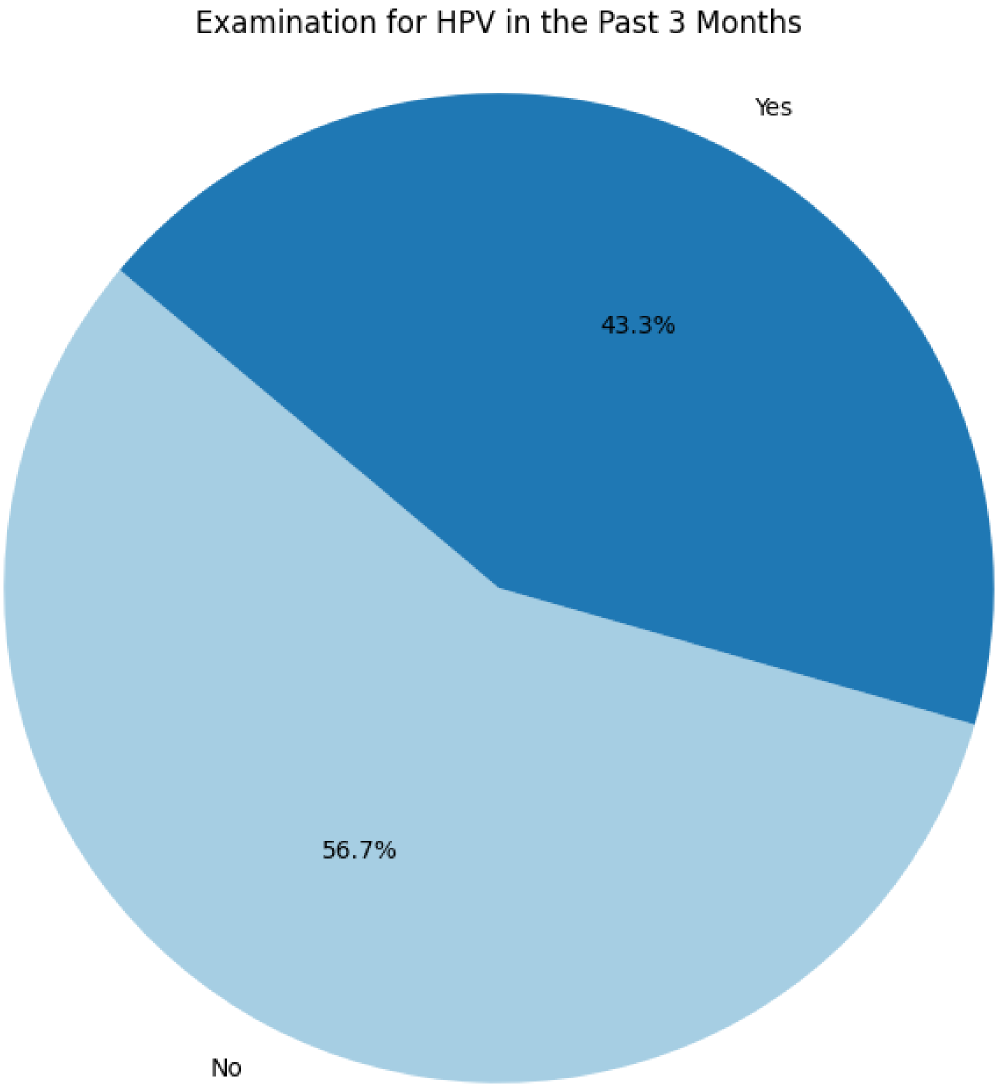
```
[89]: <AxesSubplot: >
```



HPV Examination

```
[90]: hpv_examination = kisumu_respondents['27. In the past 3 months, have you ever␣
       ↪gone for an examination for HPV?']

      # Counting the occurrences of each response
      hpv_counts = hpv_examination.value_counts()

      # Print the counts
      print(hpv_counts.to_string(index=True))

      # Plotting the counts as a pie chart
      plt.figure(figsize=(8, 8))
      plt.pie(hpv_counts, labels=hpv_counts.index, autopct='%1.1f%%', startangle=140,␣
       ↪colors=plt.cm.Paired.colors)
      plt.title('Examination for HPV in the Past 3 Months')
      plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

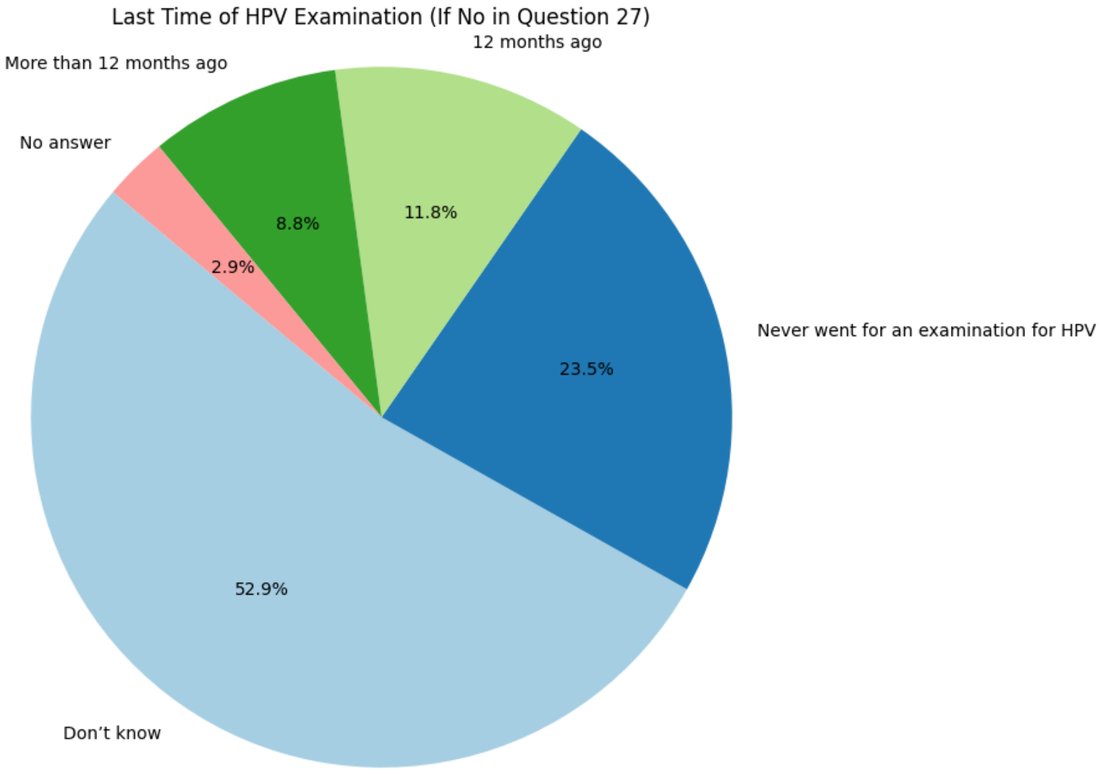
```
No     34
Yes    26
```



[91]:
```
hpv_examination_last_time = kisumu_respondents['28. If No, when was the last time you went for an examination for HPV?']

# Counting the occurrences of each response
hpv_last_time_counts = hpv_examination_last_time.value_counts()

# Print the counts
```

```
print(hpv_last_time_counts.to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(hpv_last_time_counts, labels=hpv_last_time_counts.index, autopct='%1.
  ↪1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Last Time of HPV Examination (If No in Question 27)')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

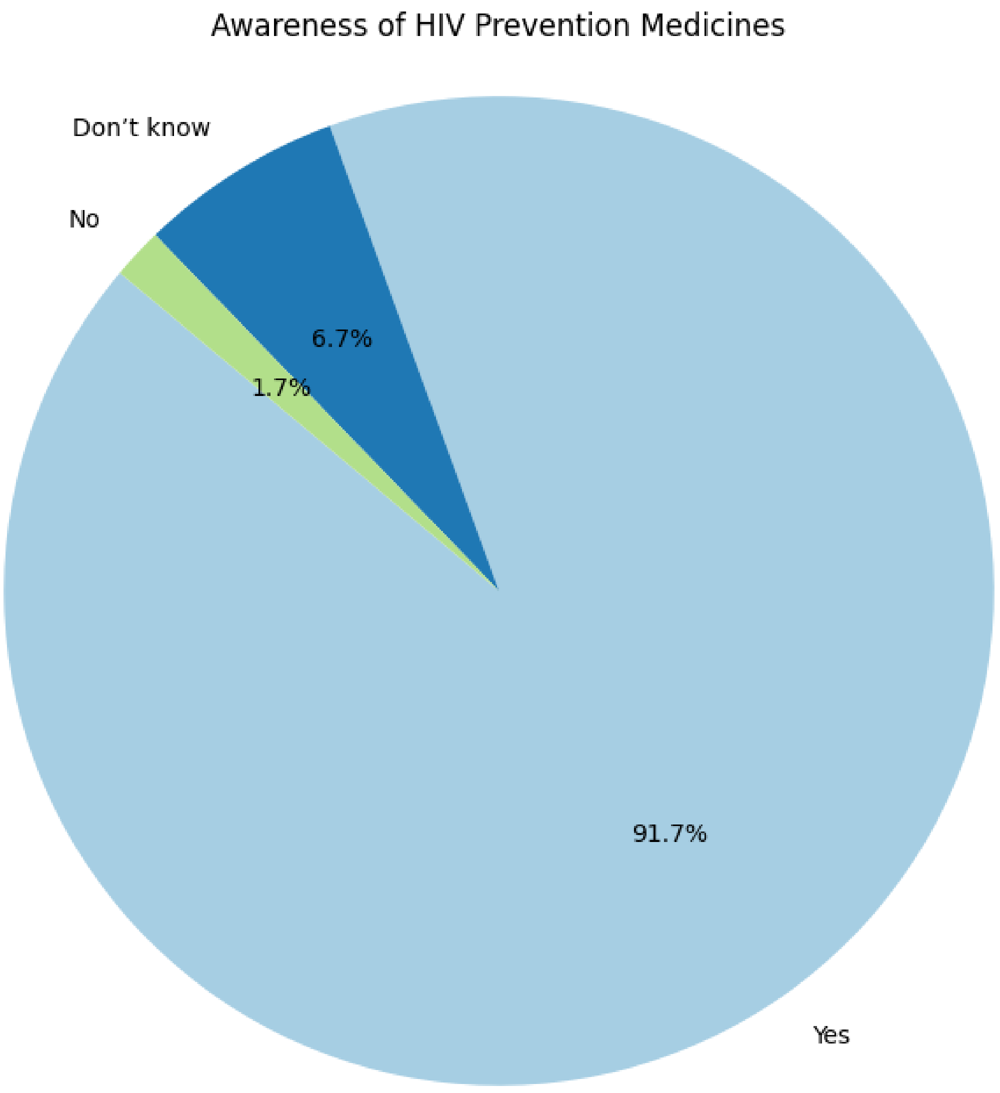
```
Don’t know                               18
Never went for an examination for HPV     8
12 months ago                             4
More than 12 months ago                   3
No answer                                 1
```



[92]:
```
hiv_medicines = kisumu_respondents['29. Are there medicines that a person who␣
  ↪is exposed to HIV can take to prevent HIV infection?']

# Counting the occurrences of each response
hiv_medicines_counts = hiv_medicines.value_counts()
```

```
# Print the counts
print(hiv_medicines_counts.to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(hiv_medicines_counts, labels=hiv_medicines_counts.index, autopct='%1.
 ↪1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Awareness of HIV Prevention Medicines')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

```
Yes            55
Don’t know      4
No              1
```



[93]:

```
medicine_responses = kisumu_respondents['30. If yes, Can you name the medicine?
 ↪']

# Cleaning up the responses
medicine_responses = medicine_responses.str.strip().str.lower()

# Counting the occurrences of each response
medicine_counts = medicine_responses.value_counts()

# Print the counts
print(medicine_counts.to_string(index=True))
```

```
pep                              24
prep                             20
prep, pep                         4
pep and prep                      2
prep and prep                     1
prep and pep                      1
preps                             1
she could not recall the name     1
preps and condom                  1
```

[94]:

```python
medicines_for_partner = kisumu_respondents['31. Are there medicines that a person who has a sexually active HIV positive partner can take to prevent infection?']

# Clean up the responses if necessary (e.g., standardize spellings)
medicines_for_partner = medicines_for_partner.str.strip().str.lower()

# Counting the occurrences of each response
medicines_counts = medicines_for_partner.value_counts()

# Print the counts
print(medicines_counts.to_string(index=True))
```

```
yes           53
don't know     2
no             2
```

[95]:

```python
medicine_names = kisumu_respondents['32. If yes, Can you name the medicine?']

# Clean up the responses if necessary (e.g., standardize spellings)
medicine_names = medicine_names.str.strip().str.lower()

# Counting the occurrences of each response
medicine_counts = medicine_names.value_counts()

# Print the counts
print(medicine_counts.to_string(index=True))
```

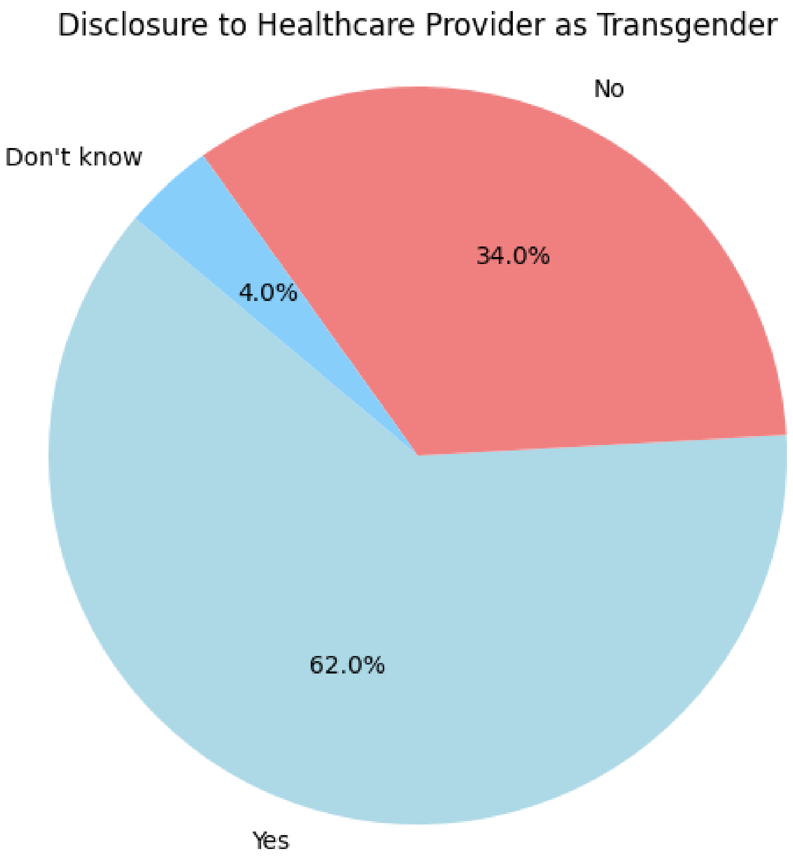
```
prep
39
pep
8
prep, pep
1
pep, prep
1
preps
1
```

```
she doesn't remember the name
1
she states that if she had not taken prep then she will take pep immediately
1
prep and pep
1
```

1.1.6 Health Care Access and Utilization

```
[96]: disclosure_responses = kisumu_respondents['33. Since you discovered you are a␣
       ↪transgender have you ever disclosed  to a healthcare provider that you are a␣
       ↪transgender person?']

      # Clean up the responses if necessary (standardize capitalization, strip spaces)
      disclosure_responses = disclosure_responses.str.strip().str.lower()

      # Counting the occurrences of each response
      disclosure_counts = disclosure_responses.value_counts()

      # Print the counts
      print(disclosure_counts.to_string(index=True))



      labels = ['Yes', 'No', "Don't know"]
      sizes = [62, 34, 4]  # Replace with actual counts from your analysis

      # Colors for each section of the pie chart
      colors = ['lightblue', 'lightcoral', 'lightskyblue']

      # Plotting the pie chart
      plt.figure(figsize=(8, 6))
      plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140)
      plt.title('Disclosure to Healthcare Provider as Transgender')
      plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
      plt.show()
```

```
no     34
yes    26
```

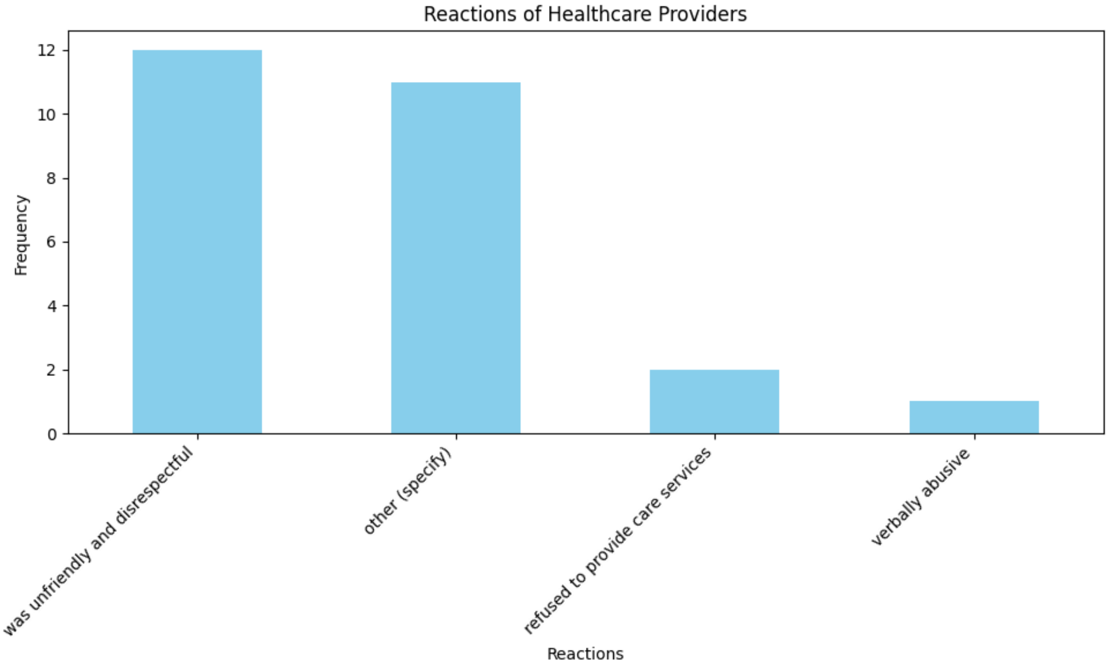


[97]:

```python
responses = kisumu_respondents['34. If Yes, what was the reactions of the health care provider?'].str.strip().str.lower()

# Counting the occurrences of each response
response_counts = responses.value_counts()

# Print the counts
print(response_counts.to_string(index=True))

# Plotting the frequency of each response type
plt.figure(figsize=(10, 6))
response_counts.plot(kind='bar', color='skyblue')
plt.title('Reactions of Healthcare Providers')
plt.xlabel('Reactions')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
was unfriendly and disrespectful    12
```

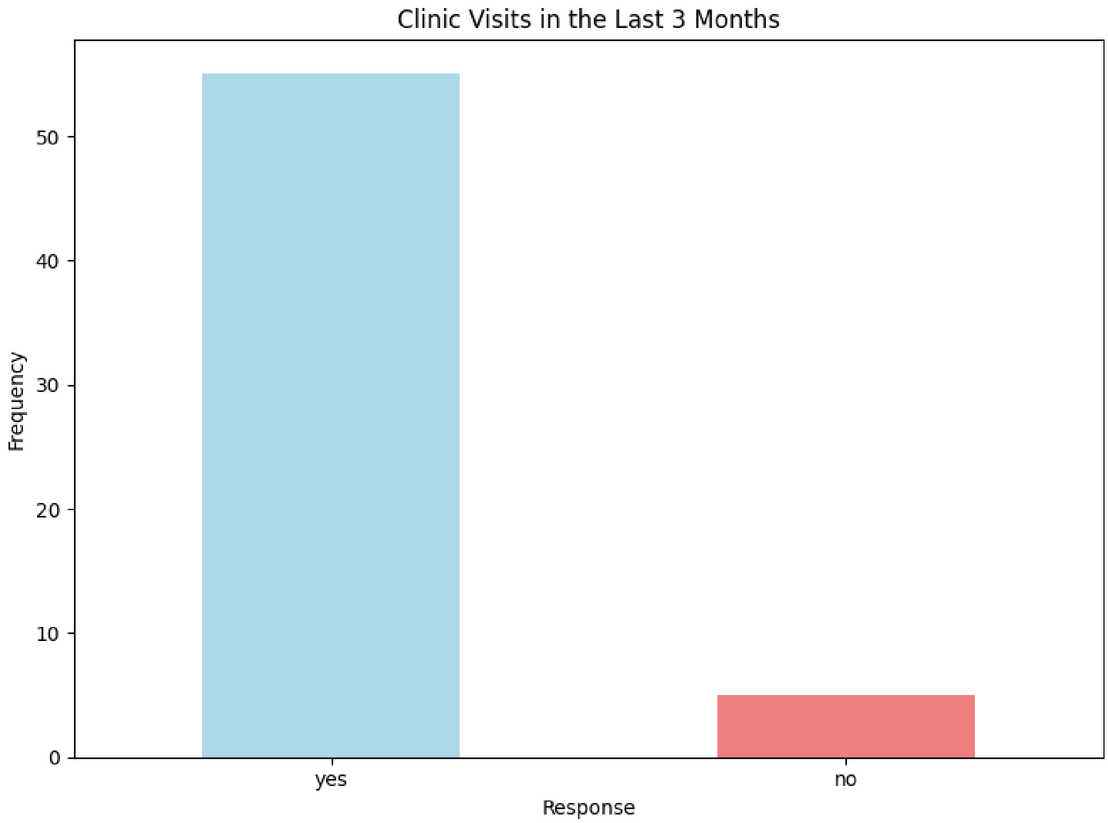
```
other (specify)                   11
refused to provide care services   2
verbally abusive                   1
```



[98]:

```
# Clean up the responses (standardize capitalization, strip spaces)
clinic_visits = kisumu_respondents['35. In the last 3 months, have you been to␣
 ↪a clinic for any healthcare service?'].str.strip().str.lower()

# Counting the occurrences of each response
clinic_visit_counts = clinic_visits.value_counts()

# Print the counts
print(clinic_visit_counts.to_string(index=True))

# Plotting the frequency of each response type
plt.figure(figsize=(8, 6))
clinic_visit_counts.plot(kind='bar', color=['lightblue', 'lightcoral'])
plt.title('Clinic Visits in the Last 3 Months')
plt.xlabel('Response')
plt.ylabel('Frequency')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

```
yes    55
no      5
```

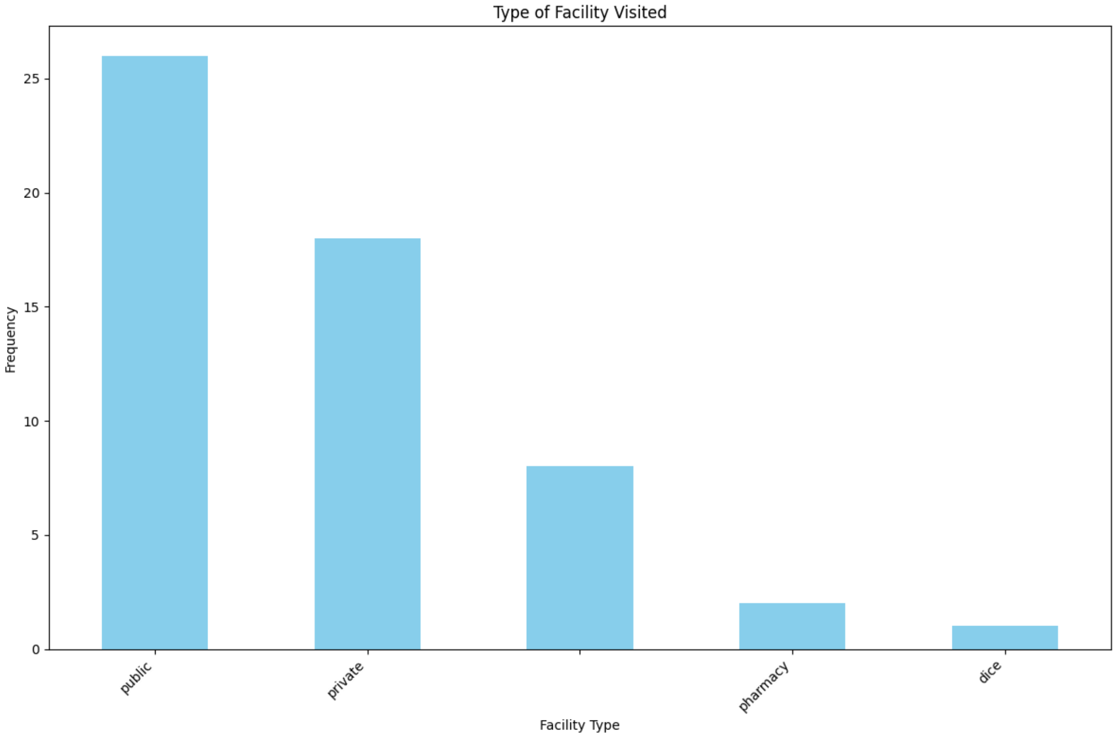


[99]:

```
# Manually input the 'specify other' data
other_specify_data = [
    "DICE", "", "", "", "", "DICE", "", "", "", "", "", "", "", "", "", "", "",␣
 ↪"DICE", "", "", "", "", "", "", "",
    "", "", "", "DICE", "", "", "", "", "", "DICE", "", "", "", "", "", "", "",␣
 ↪"", "Dice", "", "", "", "", "", "Dice",
    "", "", "DICE", "", "", "", "", "", "Community Dice", "", "", "", "", "",␣
 ↪"", "", "", "", "", "", "", "Dice", "",
    "", "Transcare", "", "", "", "", "Public, Private and Pharmacy", "", "",␣
 ↪"Ngo", "Ngo", "Ngo", "Ngo", "", "", "",
    "", "", "", "", "", "", "", "", "", "", "", "", "", "", "Hoymas Kenya", "",␣
 ↪"", "", "", "", "", "", "", "", "",
    "", "", "", "", "", "NGO hoymas queer facility", "", "", "", "", "Transform␣
 ↪,CBD", "Transform", "", "", "", "",
    "", "", "", "", "", "", "", "", "", "Machakos dice", "", "", "", "", "",␣
 ↪"", "Transform"
]

# Convert to pandas Series
other_specify = pd.Series(other_specify_data).str.strip().str.lower()
```

```
# Clean up the main facility responses (standardize capitalization, strip␣
 ↪spaces)
facility_visits = kisumu_respondents['36. If Yes, What was the type of facility␣
 ↪you visited'].str.strip().str.lower()

# Replace "other specify" in the main facility responses with actual specified␣
 ↪responses
facility_visits = facility_visits.mask(facility_visits == 'other specify',␣
 ↪other_specify)

# Replace any additional 'other specify' entries with their corresponding values
facility_visits = facility_visits.replace({
    'public, private and pharmacy': 'public, private, pharmacy',
    'ngo': 'ngo',
    'ngo hoymas queer facility': 'ngo',
    'transform ,cbd': 'transform',
    'transform': 'transform',
    'machakos dice': 'dice'
})

# Counting the occurrences of each response
facility_visit_counts = facility_visits.value_counts()

# Print the counts
print(facility_visit_counts.to_string(index=True))

# Plotting the frequency of each response type
plt.figure(figsize=(12, 8))
facility_visit_counts.plot(kind='bar', color='skyblue')
plt.title('Type of Facility Visited')
plt.xlabel('Facility Type')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
public      26
private     18
             8
pharmacy     2
dice         1
```

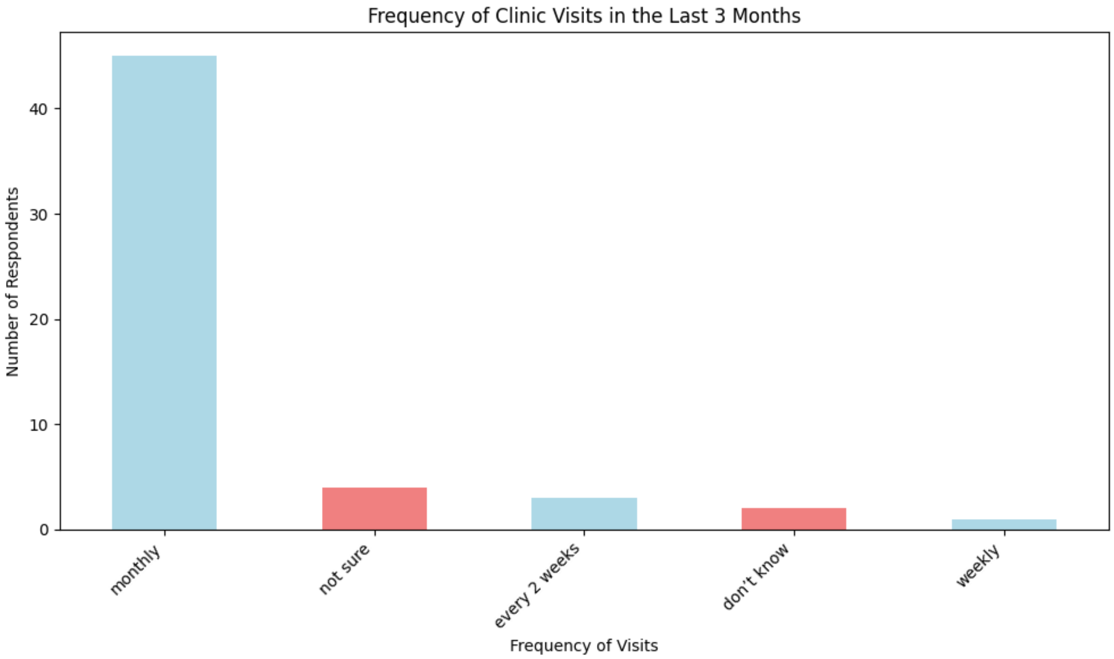


In Facility type we had mentions like: Hoymas Care, Transform, Transcare and NGOs. The following array enumerates the mentioned facility types: ‘public, private and pharmacy’: ‘public, private, pharmacy’, ‘ngo’: ‘ngo’, ‘ngo hoymas queer facility’: ‘ngo’, ‘transform ,cbd’: ‘transform’, ‘transform’: ‘transform’, ‘machakos dice’: ‘dice’]

[100]:

```python
clinic_visits = kisumu_respondents['37. If Yes, how frequently did you visit the facility in the last 3 months?'].str.strip().str.lower()

# Counting the occurrences of each response
clinic_visit_counts = clinic_visits.value_counts()

# Print the counts
print(clinic_visit_counts.to_string(index=True))

# Plotting the frequency of each response type
plt.figure(figsize=(10, 6))
clinic_visit_counts.plot(kind='bar', color=['lightblue', 'lightcoral'])
plt.title('Frequency of Clinic Visits in the Last 3 Months')
plt.xlabel('Frequency of Visits')
plt.ylabel('Number of Respondents')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

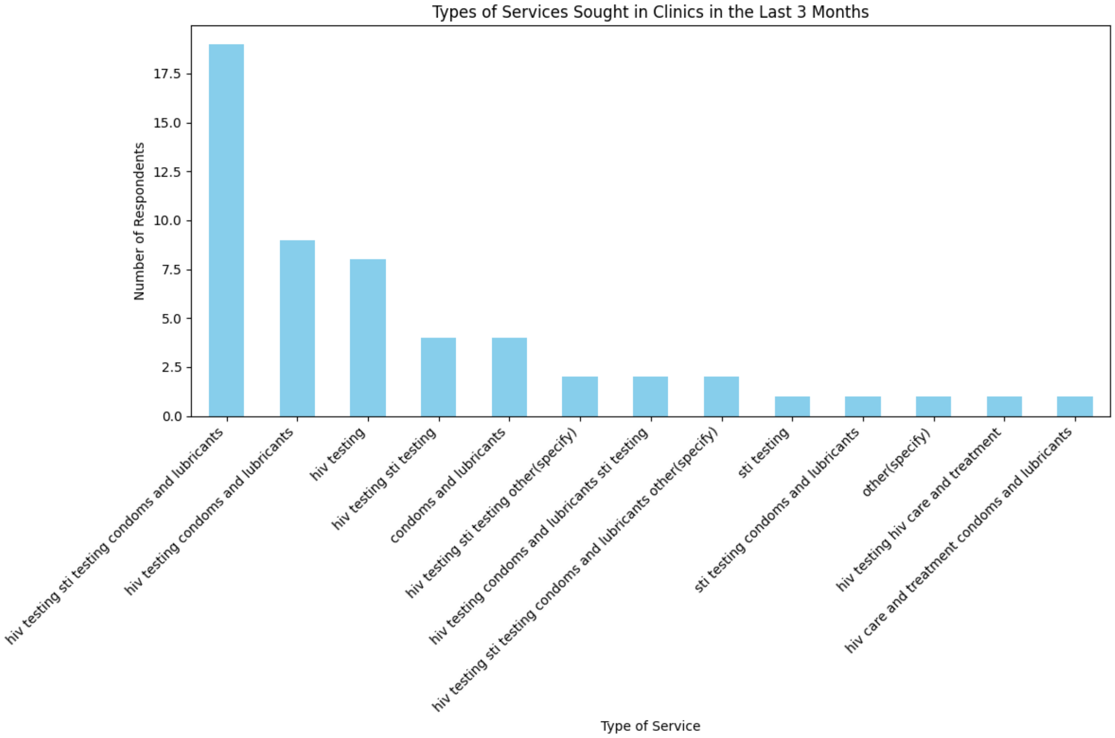
```
monthly          45
not sure          4
every 2 weeks     3
don't know        2
weekly            1
```



[101]:

```
service_types = kisumu_respondents['38. If Yes, what type of service did you␣
↪seek?'].str.strip().str.lower()

# Counting the occurrences of each response
service_type_counts = service_types.value_counts()

# Print the counts
print(service_type_counts.to_string(index=True))

# Plotting the frequency of each service type
plt.figure(figsize=(12, 8))
service_type_counts.plot(kind='bar', color='skyblue')
plt.title('Types of Services Sought in Clinics in the Last 3 Months')
plt.xlabel('Type of Service')
plt.ylabel('Number of Respondents')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

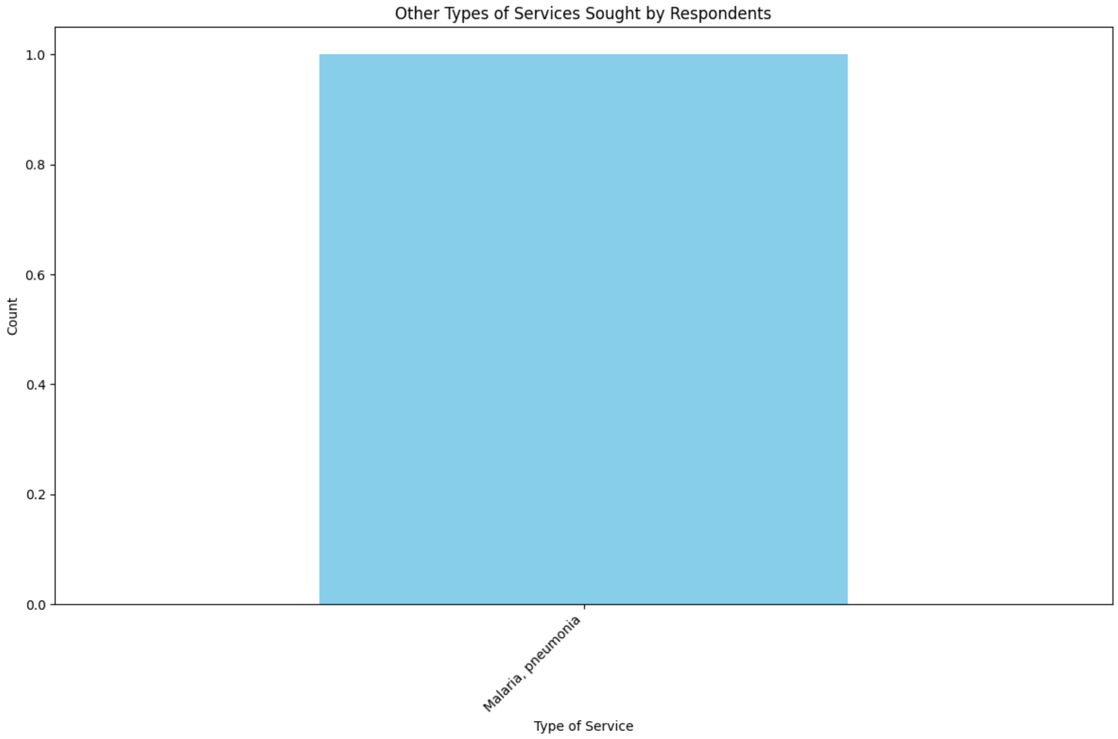
```
hiv testing sti testing condoms and lubricants                 19
hiv testing condoms and lubricants                                9
hiv testing                                                       8
hiv testing sti testing                                           4
condoms and lubricants                                            4
hiv testing sti testing other(specify)                            2
hiv testing condoms and lubricants sti testing                    2
hiv testing sti testing condoms and lubricants other(specify)     2
sti testing                                                       1
sti testing condoms and lubricants                                1
other(specify)                                                    1
hiv testing hiv care and treatment                                1
hiv care and treatment condoms and lubricants                     1
```



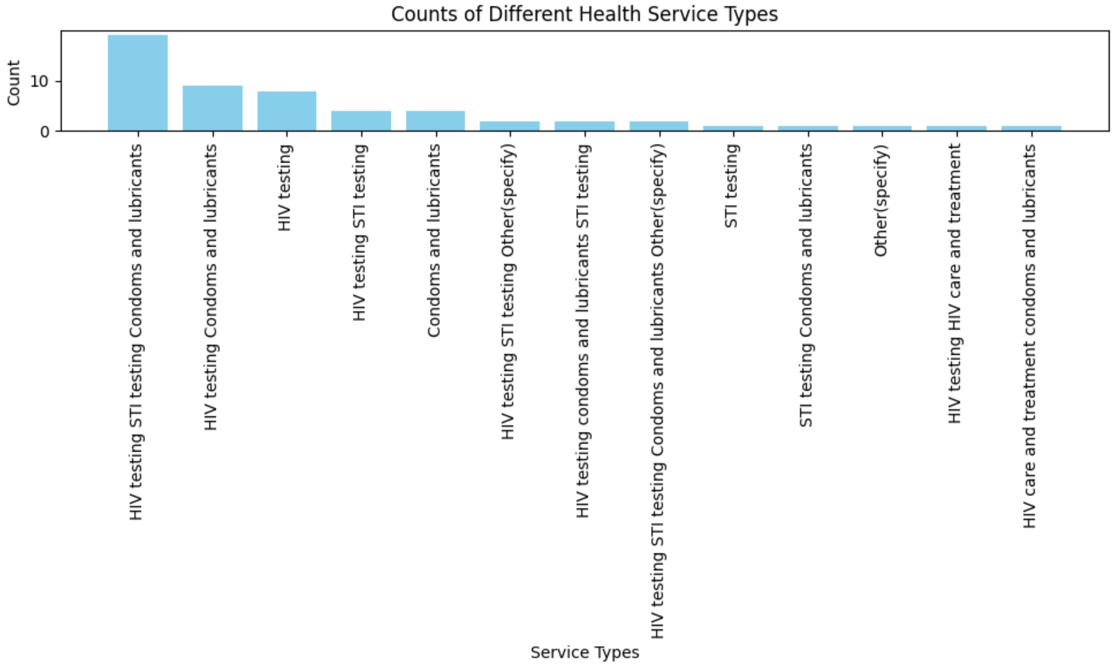
```
[102]: # Assuming the column name is '38b. Specify other type of service' in␣
        ↪kisumu_respondents

       # Clean up the responses if necessary (strip spaces)
       other_services = kisumu_respondents['38b. Specify other type of service'].str.
        ↪strip()

       # Count the occurrences of each service
       service_counts = other_services.value_counts()
```

```
# Print the counts
print(service_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(12, 8))
service_counts.plot(kind='bar', color='skyblue')
plt.title('Other Types of Services Sought by Respondents')
plt.xlabel('Type of Service')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
Malaria, pneumonia    1
```



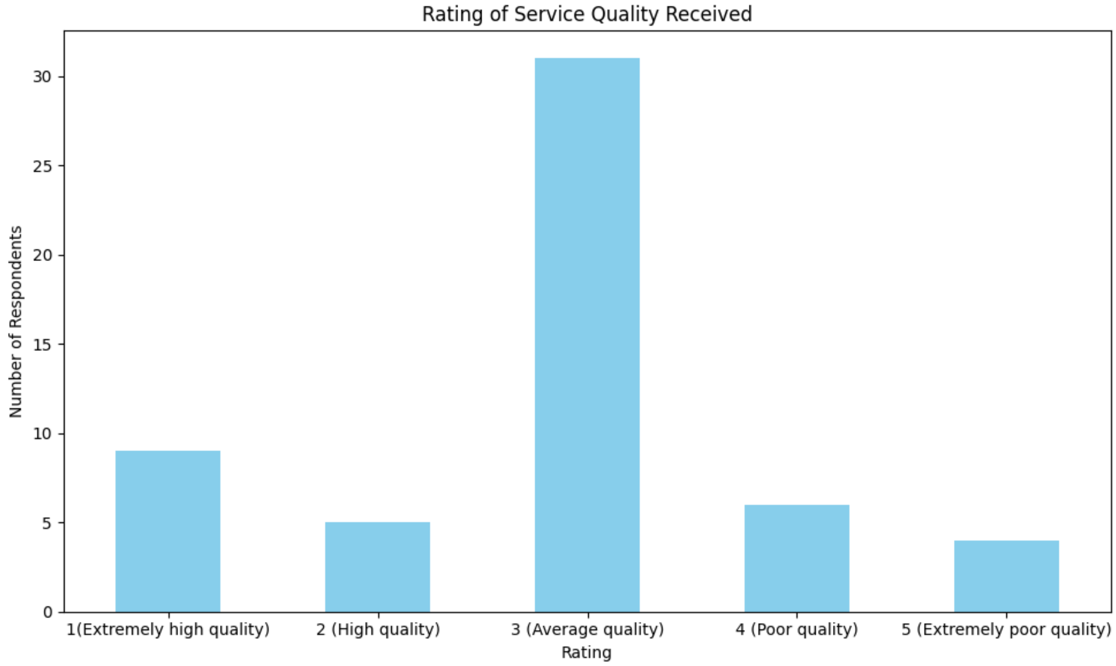
[103]:
```
data = {
    'Category': ['HIV testing STI testing Condoms and lubricants', 'HIV testing␣
 ↪Condoms and lubricants',
                 'HIV testing', 'HIV testing STI testing', 'Condoms and␣
 ↪lubricants',
                 'HIV testing STI testing Other(specify)', 'HIV testing condoms␣
 ↪and lubricants STI testing',
                 'HIV testing STI testing Condoms and lubricants␣
 ↪Other(specify)', 'STI testing',
                 'STI testing Condoms and lubricants', 'Other(specify)', 'HIV␣
 ↪testing HIV care and treatment',
                 'HIV care and treatment condoms and lubricants'],
    'Count': [19, 9, 8, 4, 4, 2, 2, 2, 1, 1, 1, 1, 1]
}

# Convert data to a pandas DataFrame
import pandas as pd
df = pd.DataFrame(data)

# Plotting the bar plot
plt.figure(figsize=(10, 6))
plt.bar(df['Category'], df['Count'], color='skyblue')
plt.title('Counts of Different Health Service Types')
plt.xlabel('Service Types')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



```
[104]: ratings = kisumu_respondents['39. If Yes, How would you rate the quality of␣
        ↪service you received?']
```

```
# Counting the occurrences of each rating
rating_counts = ratings.value_counts().sort_index()

# Print the counts
print(rating_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 6))
rating_counts.plot(kind='bar', color='skyblue')
plt.title('Rating of Service Quality Received')
plt.xlabel('Rating')
plt.ylabel('Number of Respondents')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

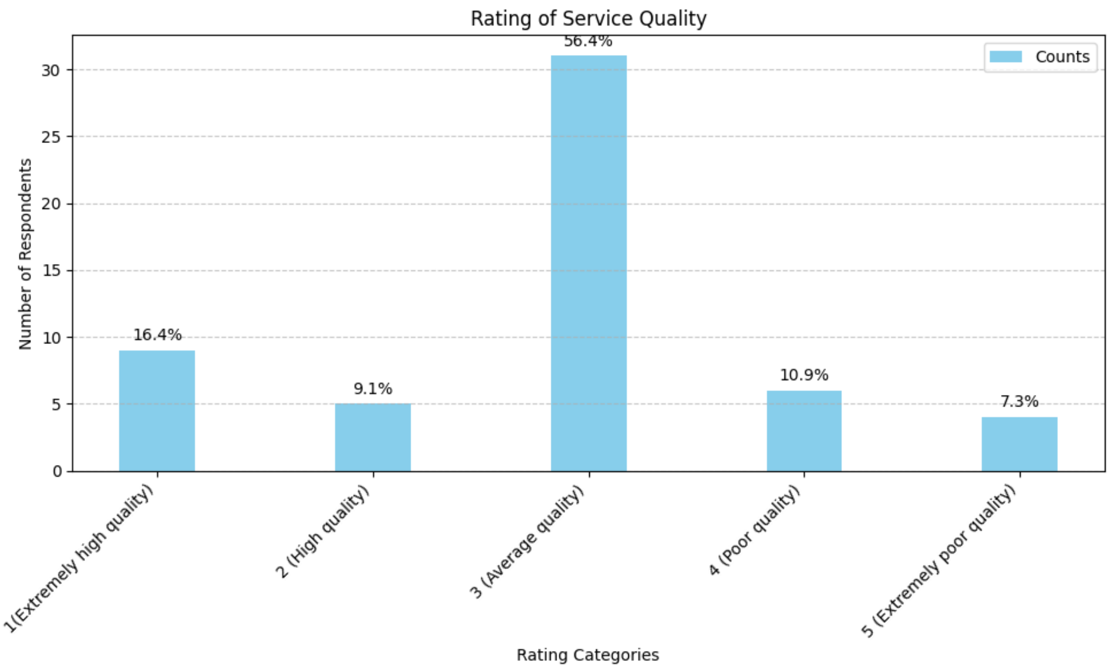
```
1(Extremely high quality)      9
2 (High quality)               5
3 (Average quality)           31
4 (Poor quality)               6
5 (Extremely poor quality)     4
```



[105]:

```
# Count the occurrences of each rating
rating_counts = ratings.value_counts().sort_index()
```

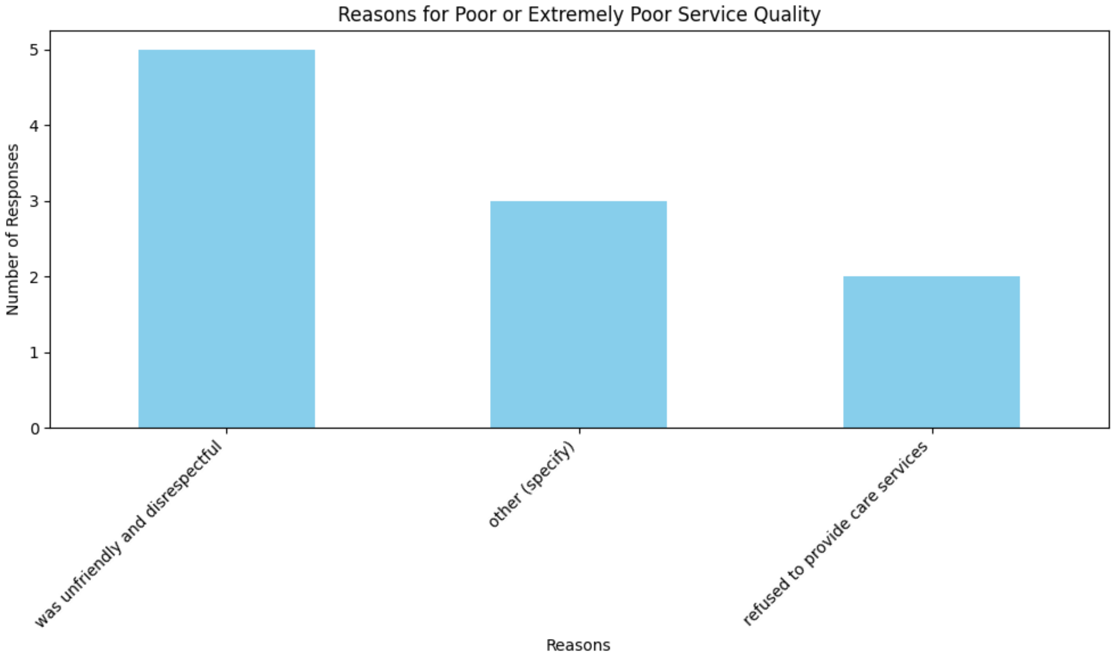
```python
# Define the categories and their counts
categories = ['1(Extremely high quality)', '2 (High quality)', '3 (Average quality)', '4 (Poor quality)', '5 (Extremely poor quality)']
counts = [rating_counts.get(cat, 0) for cat in categories]

# Calculate the total number of respondents
total_respondents = sum(counts)

# Calculate percentages for each category
percentages = [(count / total_respondents) * 100 for count in counts]

# Create a grouped bar chart
plt.figure(figsize=(10, 6))

# Bar positions and width
bar_width = 0.35
index = np.arange(len(categories))

# Plotting the bars
bars = plt.bar(index, counts, bar_width, label='Counts', color='skyblue')

# Adding labels, title, and grid
plt.xlabel('Rating Categories')
plt.ylabel('Number of Respondents')
plt.title('Rating of Service Quality')
plt.xticks(index, categories, rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Adding percentages above each bar
for i, bar in enumerate(bars):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 0.5,
             f'{percentages[i]:.1f}%', ha='center', va='bottom', fontsize=10)

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



[106]:

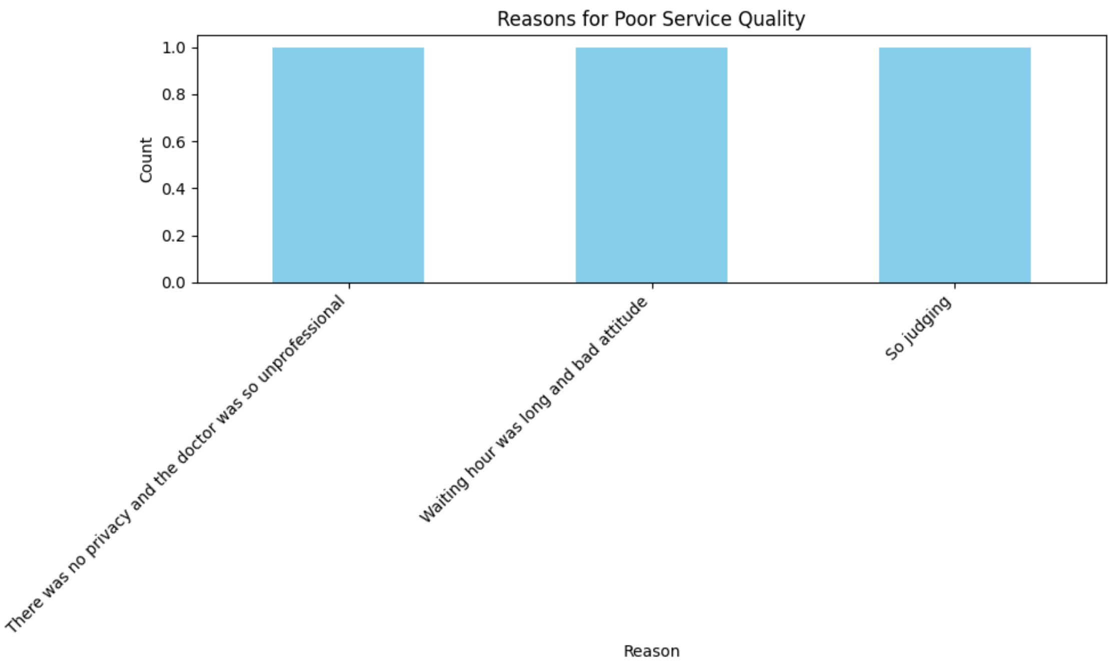
```python
responses = kisumu_respondents['40. If poor or extremely poor, why?']

# Clean up responses (strip spaces and convert to lowercase)
clean_responses = responses.str.strip().str.lower()

# Count occurrences of each reason
reason_counts = clean_responses.value_counts()

# Print the counts
print(reason_counts.to_string())

# Plotting the reasons for poor service quality
plt.figure(figsize=(10, 6))
reason_counts.plot(kind='bar', color='skyblue')
plt.title('Reasons for Poor or Extremely Poor Service Quality')
plt.xlabel('Reasons')
plt.ylabel('Number of Responses')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
was unfriendly and disrespectful    5
other (specify)                     3
refused to provide care services    2
```



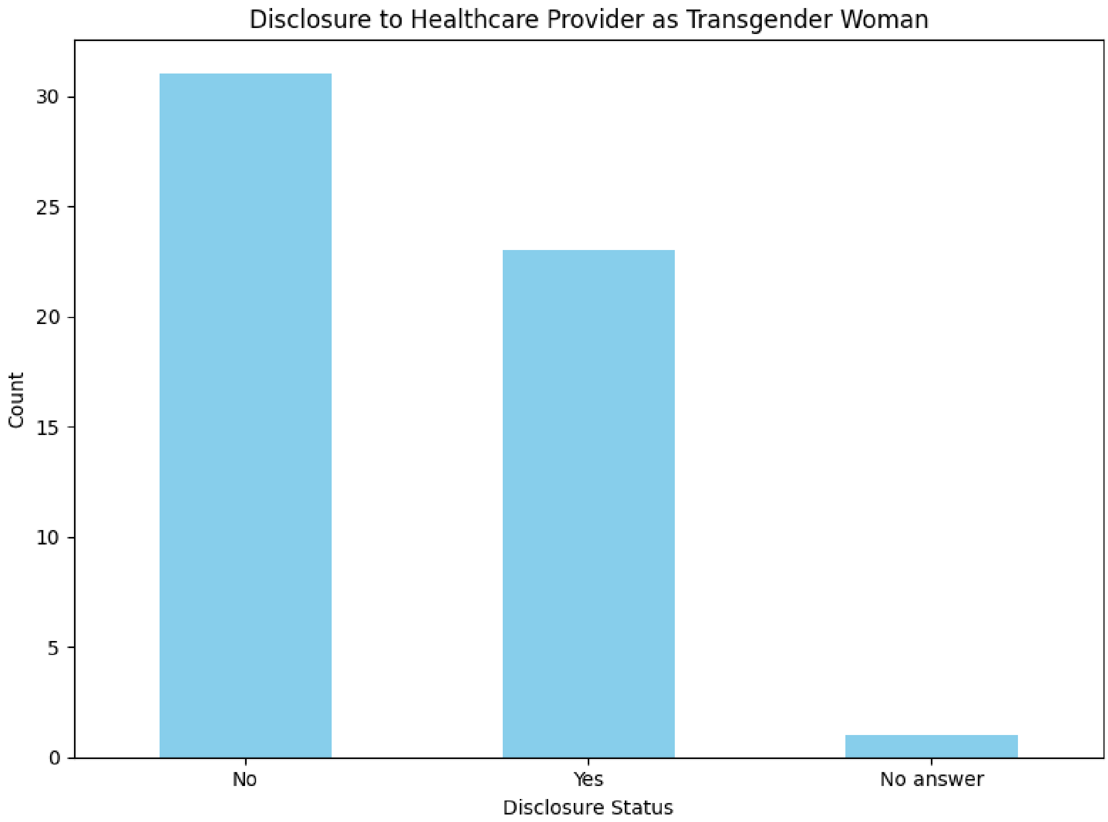
```
[107]: other_reasons = kisumu_respondents['Others- Poor quality of service'].str.
        ↪strip()

       # Count the occurrences of each reason
       reason_counts = other_reasons.value_counts()

       # Print the counts
       print(reason_counts.to_string(index=True))

       # Plotting the bar chart
       plt.figure(figsize=(10, 6))
       reason_counts.plot(kind='bar', color='skyblue')
       plt.title('Reasons for Poor Service Quality')
       plt.xlabel('Reason')
       plt.ylabel('Count')
       plt.xticks(rotation=45, ha='right')
       plt.tight_layout()
       plt.show()
```

```
There was no privacy and the doctor was so unprofessional    1
Waiting hour was long and bad attitude                       1
So judging                                                   1
```



```
[108]: disclosure_responses = kisumu_respondents['41.If Yes, did you disclose to the␣
       ↪health care provider that you are a transgender woman?'].str.strip()

       # Count the occurrences of each response
       disclosure_counts = disclosure_responses.value_counts()

       # Print the counts
       print(disclosure_counts.to_string(index=True))

       # Plotting the bar chart
       plt.figure(figsize=(8, 6))
       disclosure_counts.plot(kind='bar', color='skyblue')
       plt.title('Disclosure to Healthcare Provider as Transgender Woman')
       plt.xlabel('Disclosure Status')
       plt.ylabel('Count')
       plt.xticks(rotation=0)
       plt.tight_layout()
       plt.show()
```

```
No           31
Yes          23
No answer     1
```

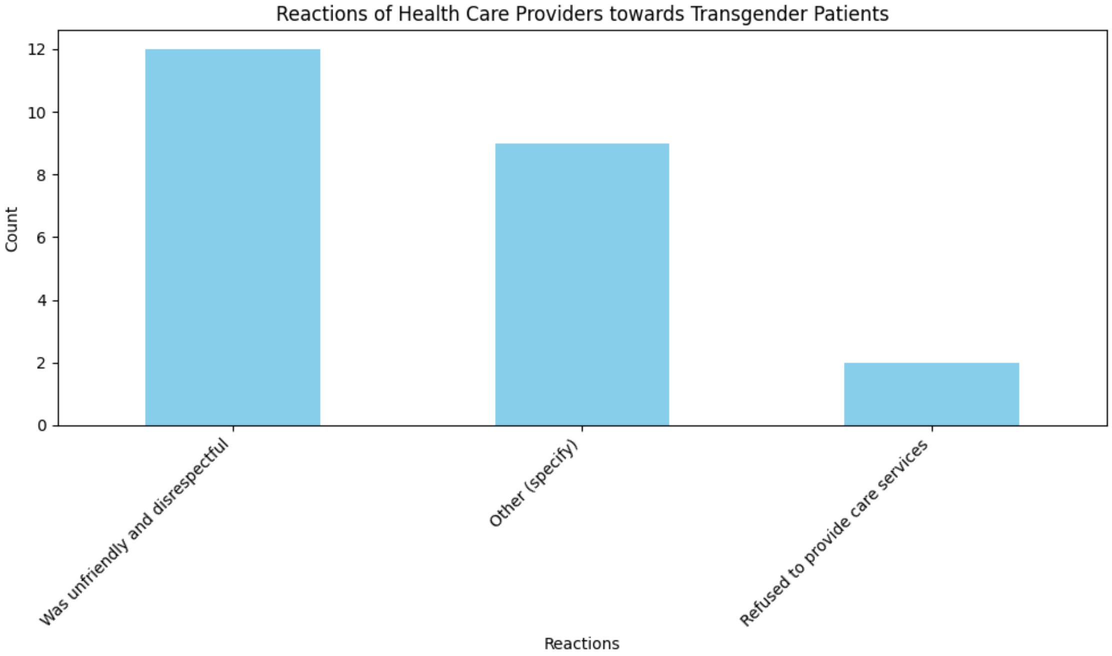


[109]:

```
reactions = kisumu_respondents['42. If Yes, what was the reactions of the␣
  ↪health care provider?'].str.strip()

# Count the occurrences of each response
reaction_counts = reactions.value_counts()

# Print the counts
print(reaction_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 6))
reaction_counts.plot(kind='bar', color='skyblue')
plt.title('Reactions of Health Care Providers towards Transgender Patients')
plt.xlabel('Reactions')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

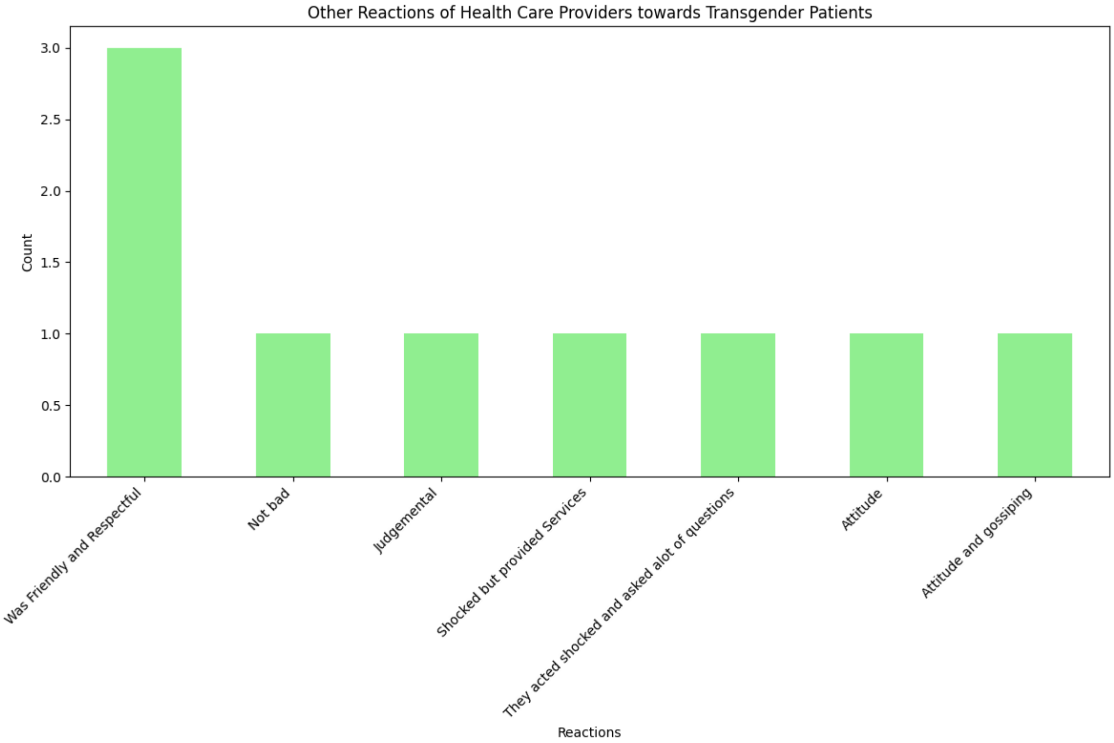
```
Was unfriendly and disrespectful    12
Other (specify)                      9
```

```
Refused to provide care services     2
```



```
[110]: import pandas as pd
       import matplotlib.pyplot as plt

       # Assuming the column name is 'Specify other reaction' in kisumu_respondents

       # Clean up the responses if necessary (strip spaces)
       other_reactions = kisumu_respondents['Specify other reaction'].str.strip()

       # Count the occurrences of each response
       other_reaction_counts = other_reactions.value_counts()

       # Print the counts
       print(other_reaction_counts.to_string(index=True))

       # Plotting the bar chart
       plt.figure(figsize=(12, 8))
       other_reaction_counts.plot(kind='bar', color='lightgreen')
       plt.title('Other Reactions of Health Care Providers towards Transgender␣
        ↪Patients')
       plt.xlabel('Reactions')
       plt.ylabel('Count')
       plt.xticks(rotation=45, ha='right')
       plt.tight_layout()
       plt.show()
```

```
Was Friendly and Respectful                        3
Not bad                                            1
Judgemental                                        1
Shocked but provided Services                      1
They acted shocked and asked alot of questions     1
Attitude                                           1
Attitude and gossiping                             1
```

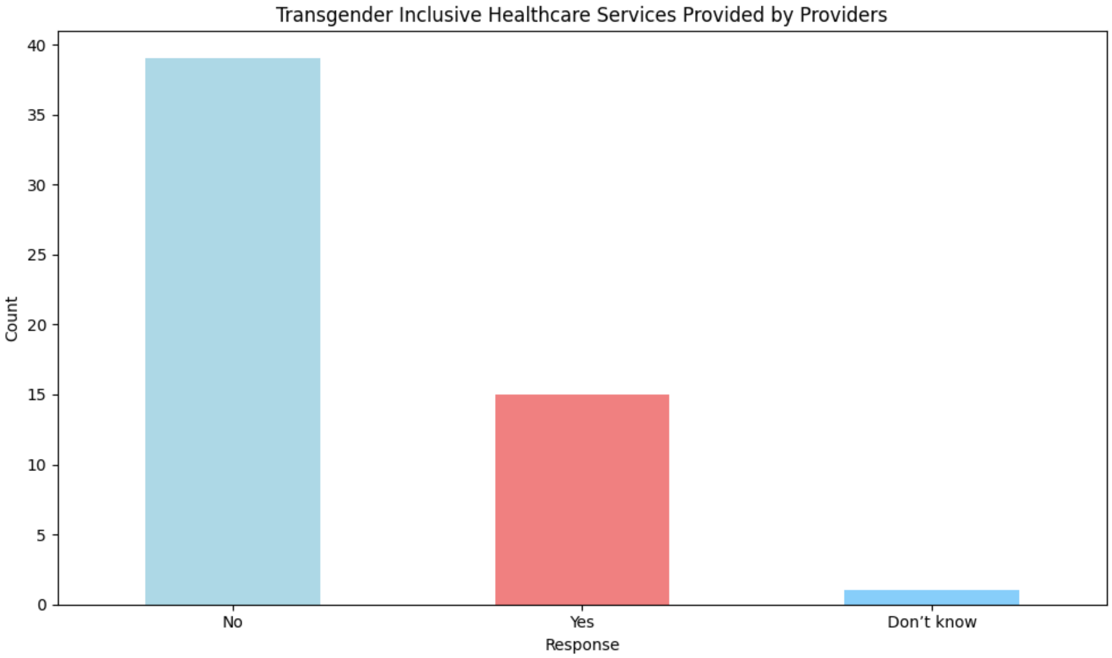


[111]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Assuming the column name is '43. Did the health care provider provide you␣
 ↪with transgender inclusive healthcare services?'

# Clean up the responses if necessary (strip spaces)
transgender_healthcare = kisumu_respondents['43. Did the health care provider␣
 ↪provide you with  transgender inclusive healthcare services?'].str.strip()

# Count the occurrences of each response
transgender_healthcare_counts = transgender_healthcare.value_counts()

# Print the counts
print(transgender_healthcare_counts.to_string(index=True))
```

```
# Plotting the grouped bar chart
plt.figure(figsize=(10, 6))
transgender_healthcare_counts.plot(kind='bar', color=['lightblue',␣
 ↪'lightcoral', 'lightskyblue'])
plt.title('Transgender Inclusive Healthcare Services Provided by Providers')
plt.xlabel('Response')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

```
No            39
Yes           15
Don’t know     1
```

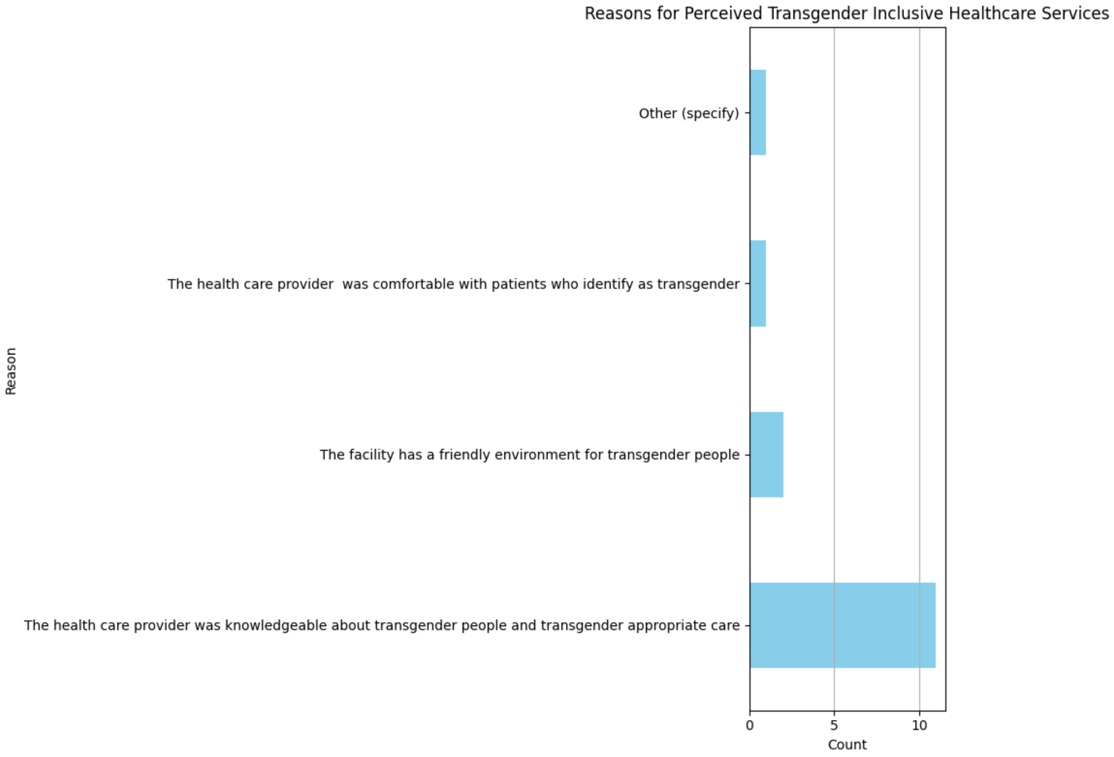


[112]:

```
transgender_inclusive_reasons = kisumu_respondents['44. If Yes, why do you␣
 ↪think that the service(s) provided were transgender inclusive?'].str.strip()

# Count the occurrences of each response
transgender_inclusive_reasons_counts = transgender_inclusive_reasons.
 ↪value_counts()

# Print the counts
print(transgender_inclusive_reasons_counts.to_string(index=True))
```

```
# Plotting the bar chart
plt.figure(figsize=(10, 8))
transgender_inclusive_reasons_counts.plot(kind='barh', color='skyblue')
plt.title('Reasons for Perceived Transgender Inclusive Healthcare Services')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

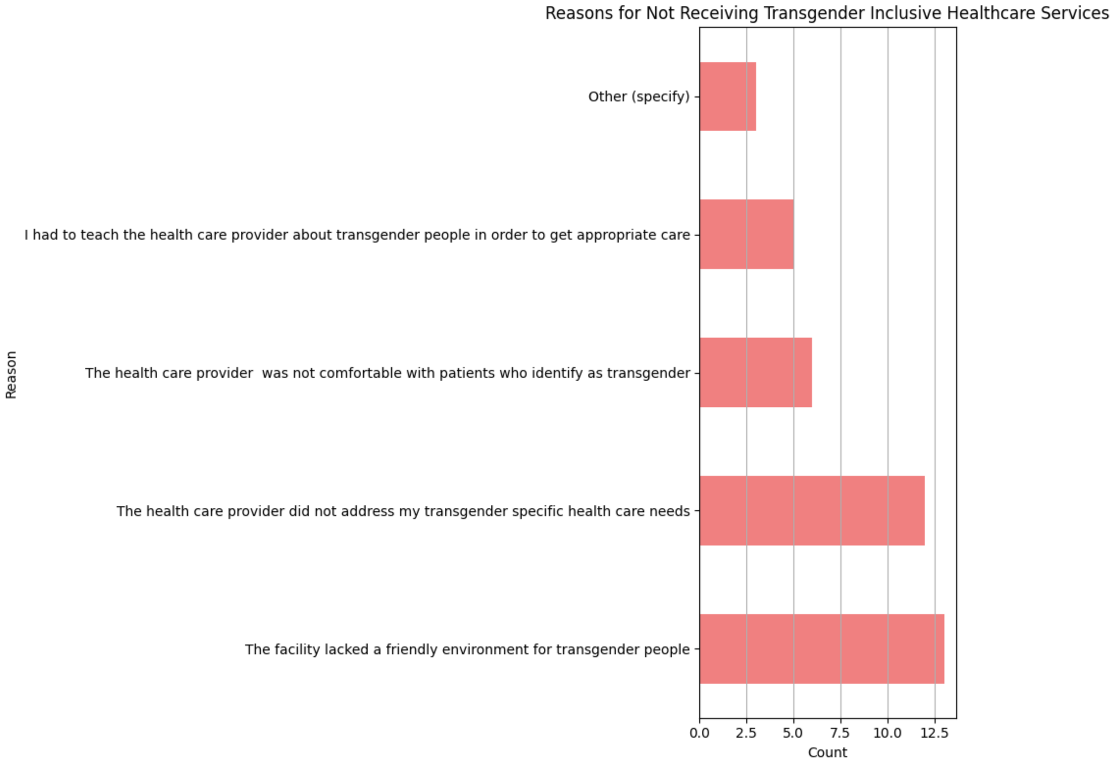
```
The health care provider was knowledgeable about transgender people and
transgender appropriate care    11
The facility has a friendly environment for transgender people
2
The health care provider  was comfortable with patients who identify as
transgender                      1
Other (specify)
1
```



```
[113]: import pandas as pd
import matplotlib.pyplot as plt
```

```
# Assuming the column name is '45. If No, why do you think the health care␣
 ↪provider did not provide you with transgender inclusive health care?'

# Clean up the responses if necessary (strip spaces)
not_transgender_inclusive_reasons = kisumu_respondents['45. If No, why do you␣
 ↪think the health care provider did not provide you with transgender␣
 ↪inclusive health care?'].str.strip()

# Count the occurrences of each response
not_transgender_inclusive_reasons_counts = not_transgender_inclusive_reasons.
 ↪value_counts()

# Print the counts
print(not_transgender_inclusive_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
not_transgender_inclusive_reasons_counts.plot(kind='barh', color='lightcoral')
plt.title('Reasons for Not Receiving Transgender Inclusive Healthcare Services')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

```
The facility lacked a friendly environment for transgender people
13
The health care provider did not address my transgender specific health care
needs                        12
The health care provider  was not comfortable with patients who identify as
transgender                   6
I had to teach the health care provider about transgender people in order to get
appropriate care      5
Other (specify)
3
```

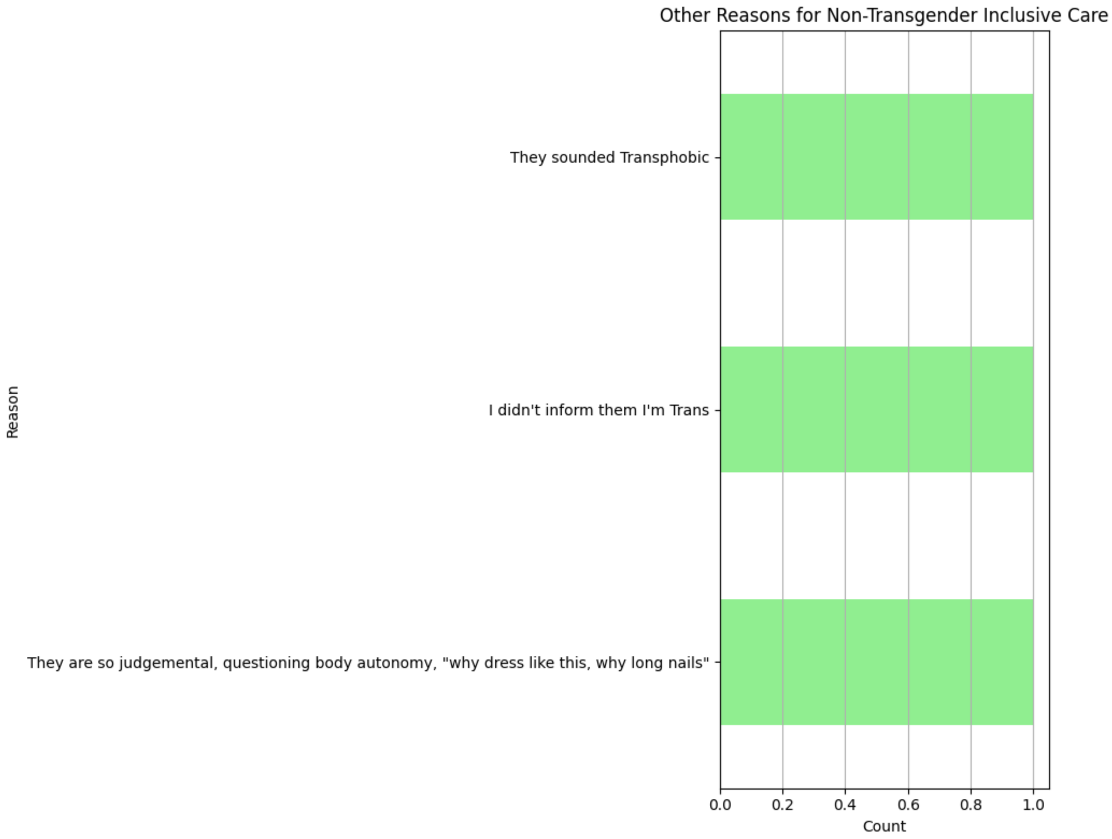


[114]:

```
other_trans_inclusive_reasons = kisumu_respondents['Specify other - NP␣
↪TransInclusive'].str.strip()

# Count the occurrences of each response
other_trans_inclusive_reasons_counts = other_trans_inclusive_reasons.
↪value_counts()

# Print the counts
print(other_trans_inclusive_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
other_trans_inclusive_reasons_counts.plot(kind='barh', color='lightgreen')
plt.title('Other Reasons for Non-Transgender Inclusive Care')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

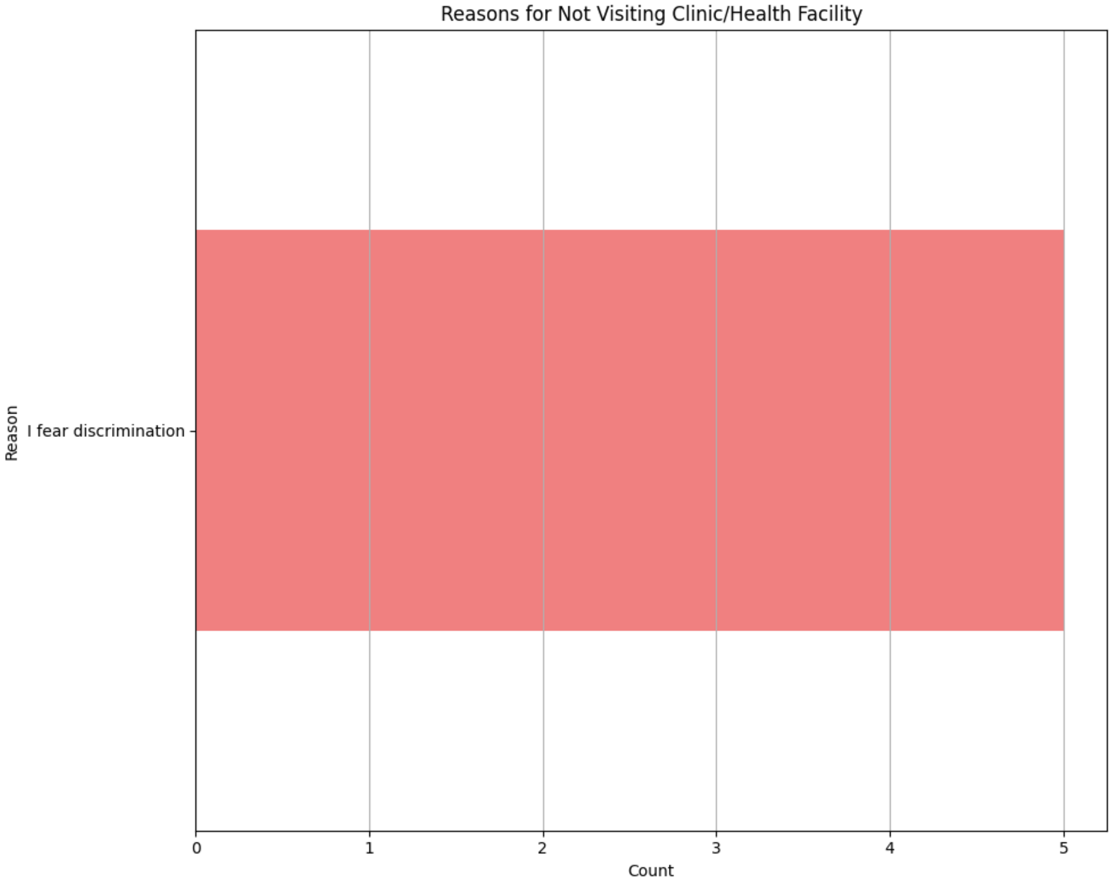
```
They are so judgemental, questioning body autonomy, "why dress like this, why
long nails"    1
I didn't inform them I'm Trans
```

```
1
They sounded Transphobic
1
```



[115]:
```
no_visit_reasons = kisumu_respondents['46. If No , why have you not gone to a␣
↪clinic/ health facility?'].str.strip()

# Count the occurrences of each response
no_visit_reasons_counts = no_visit_reasons.value_counts()

# Print the counts
print(no_visit_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
no_visit_reasons_counts.plot(kind='barh', color='lightcoral')
plt.title('Reasons for Not Visiting Clinic/Health Facility')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

```
I fear discrimination    5
```



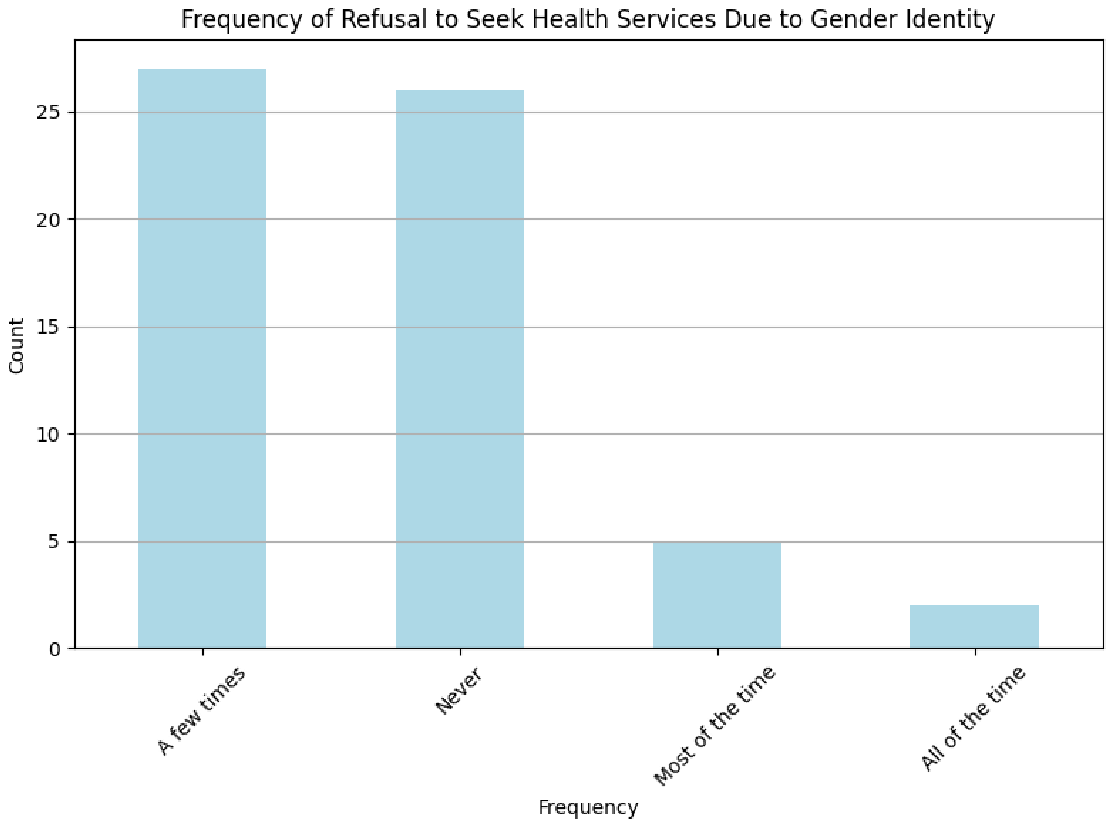
[116]:
```
other_reasons = kisumu_respondents['Specify other reason- Not going to health␣
↪Facility'].str.strip()

# Count the occurrences of each response
other_reasons_counts = other_reasons.value_counts()

# Print the counts
print(other_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
other_reasons_counts.plot(kind='barh', color='lightblue')
plt.title('Other Reasons for Not Going to Health Facility')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~/.local/lib/python3.8/site-packages/pandas/core/indexes/base.py:3803, in␣
 ↪Index.get_loc(self, key, method, tolerance)
   3802 try:
-> 3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:

File ~/.local/lib/python3.8/site-packages/pandas/_libs/index.pyx:138, in pandas
 ↪_libs.index.IndexEngine.get_loc()

File ~/.local/lib/python3.8/site-packages/pandas/_libs/index.pyx:165, in pandas
 ↪_libs.index.IndexEngine.get_loc()

File pandas/_libs/hashtable_class_helper.pxi:5745, in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

File pandas/_libs/hashtable_class_helper.pxi:5753, in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

KeyError: 'Specify other reason- Not going to health Facility'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In [116], line 1
----> 1 other_reasons =␣
 ↪kisumu_respondents['Specify other reason- Not going to health Facility'].str.
 ↪strip()
      3 # Count the occurrences of each response
      4 other_reasons_counts = other_reasons.value_counts()

File ~/.local/lib/python3.8/site-packages/pandas/core/frame.py:3805, in␣
 ↪DataFrame.__getitem__(self, key)
   3803 if self.columns.nlevels > 1:
   3804     return self._getitem_multilevel(key)
-> 3805 indexer = self.columns.get_loc(key)
   3806 if is_integer(indexer):
   3807     indexer = [indexer]

File ~/.local/lib/python3.8/site-packages/pandas/core/indexes/base.py:3805, in␣
 ↪Index.get_loc(self, key, method, tolerance)
   3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:
-> 3805     raise KeyError(key) from err
   3806 except TypeError:
   3807     # If we have a listlike key, _check_indexing_error will raise
   3808     #  InvalidIndexError. Otherwise we fall through and re-raise
   3809     #  the TypeError.
   3810     self._check_indexing_error(key)

KeyError: 'Specify other reason- Not going to health Facility'
```

[117]:

```python
refusal_responses = kisumu_respondents['47. In the last 3 months, how often did you refuse to seek health services because of your gender identity?'].str.strip()

# Count the occurrences of each response
refusal_counts = refusal_responses.value_counts()

# Print the counts
print(refusal_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
refusal_counts.plot(kind='bar', color='lightblue')
plt.title('Frequency of Refusal to Seek Health Services Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
A few times         27
Never               26
Most of the time     5
All of the time      2
```

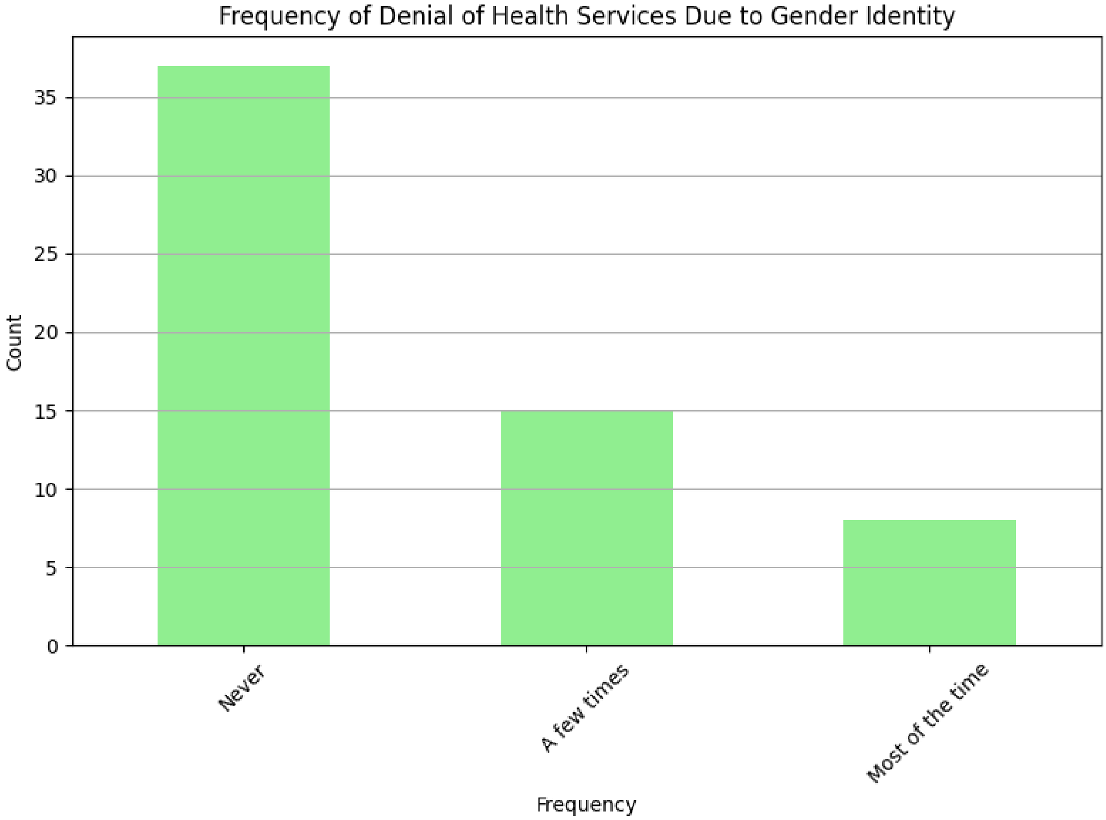


[118]:

```python
denial_responses = kisumu_respondents['48. In the last 3months, how often were you denied health services because of your gender identity?'].str.strip()

# Count the occurrences of each response
denial_counts = denial_responses.value_counts()

# Print the counts
print(denial_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
denial_counts.plot(kind='bar', color='lightgreen')
plt.title('Frequency of Denial of Health Services Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
Never                    37
```

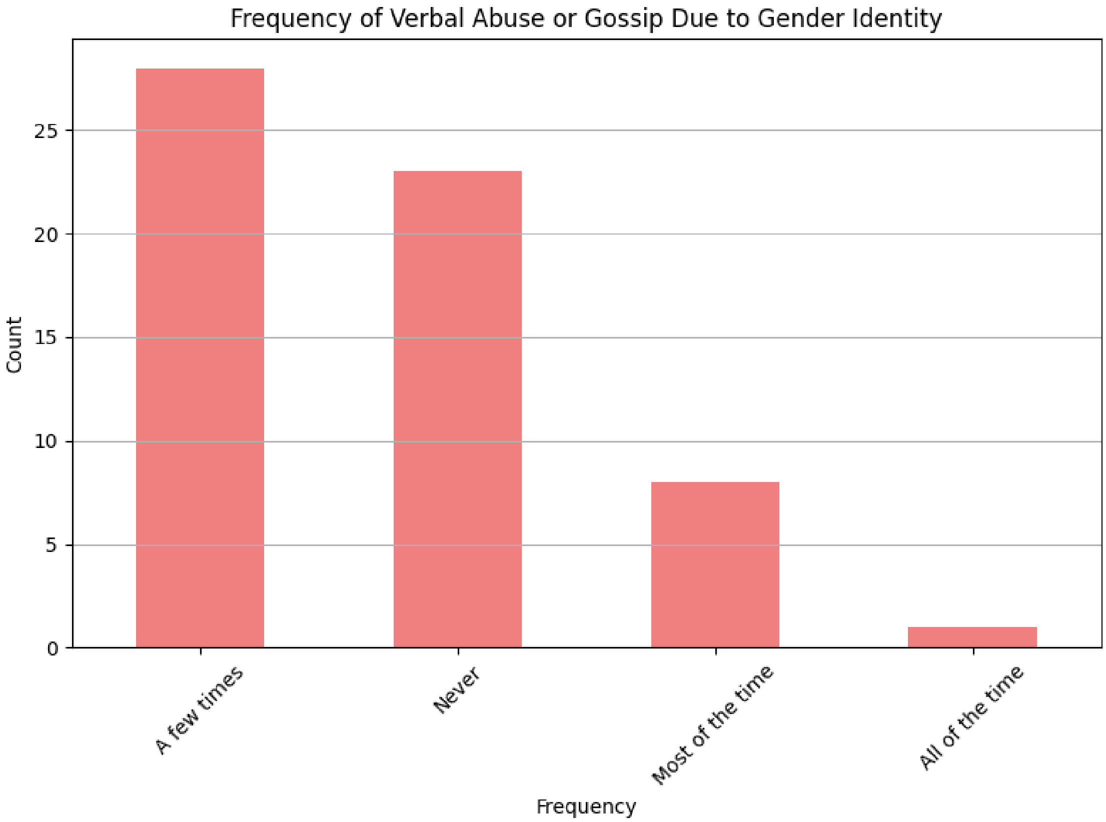
```
A few times         15
Most of the time     8
```



[119]:

```
abuse_responses = kisumu_respondents['49. In the last 3 months, how often were␣
↪you gossiped or verbally abused at a health facility because of your gender␣
↪identity?'].str.strip()

# Count the occurrences of each response
abuse_counts = abuse_responses.value_counts()

# Print the counts
print(abuse_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
abuse_counts.plot(kind='bar', color='lightcoral')
plt.title('Frequency of Verbal Abuse or Gossip Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

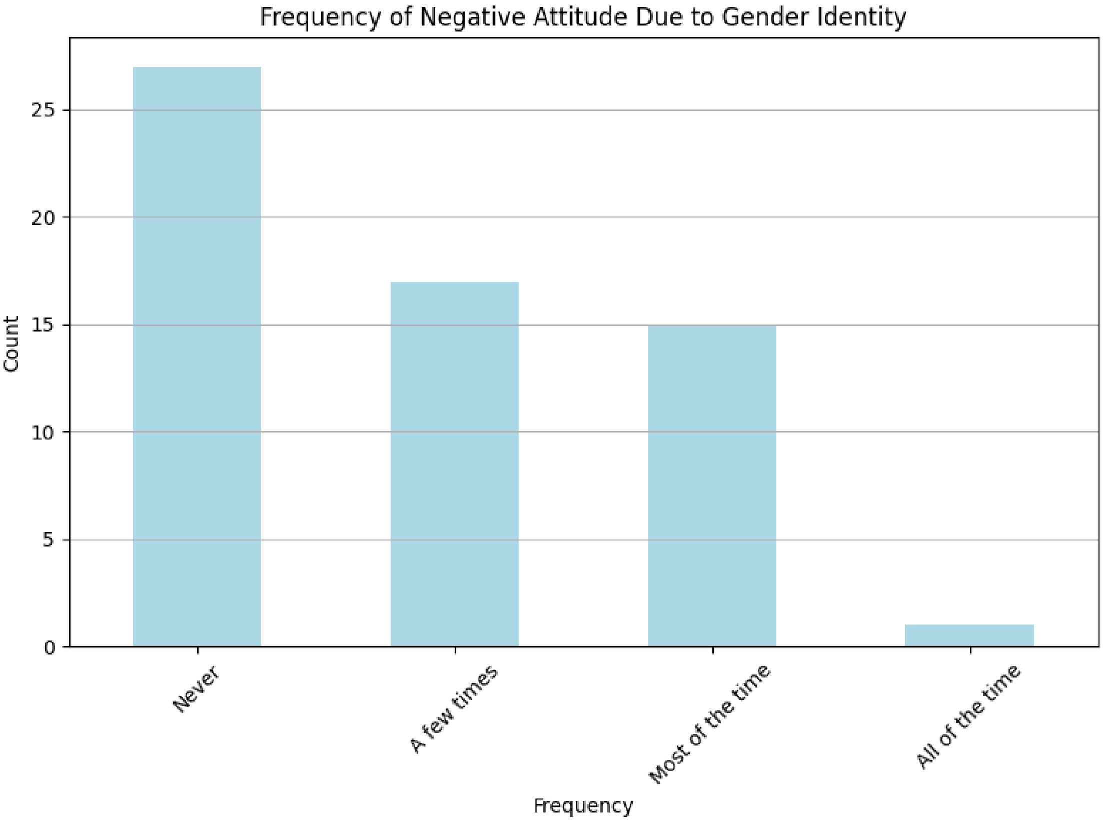
```
A few times          28
Never                23
Most of the time      8
All of the time       1
```



[120]:
```
attitude_responses = kisumu_respondents['50. In the last 3 months, how often␣
↪did you feel that a health care worker was having  negative attitude towards␣
↪you because of your gender identity?'].str.strip()

# Count the occurrences of each response
attitude_counts = attitude_responses.value_counts()

# Print the counts
print(attitude_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
attitude_counts.plot(kind='bar', color='lightblue')
```

```
plt.title('Frequency of Negative Attitude Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
Never              27
A few times        17
Most of the time   15
All of the time     1
```

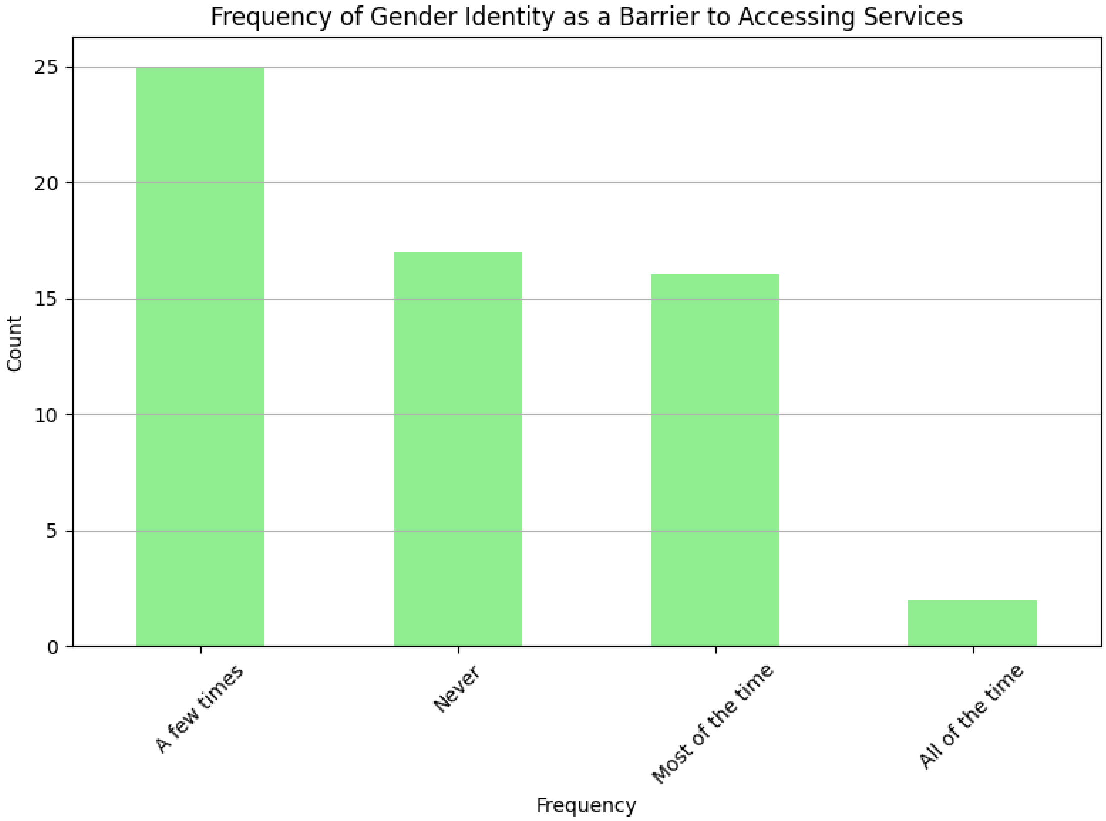


[121]:

```
barrier_responses = kisumu_respondents['51. In the last 3 months, how often did␣
  ↪you feel your gender identity  was  a barrier to accessing services at a␣
  ↪health facility?'].str.strip()

# Count the occurrences of each response
barrier_counts = barrier_responses.value_counts()

# Print the counts
```

```
print(barrier_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
barrier_counts.plot(kind='bar', color='lightgreen')
plt.title('Frequency of Gender Identity as a Barrier to Accessing Services')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
A few times        25
Never              17
Most of the time   16
All of the time     2
```

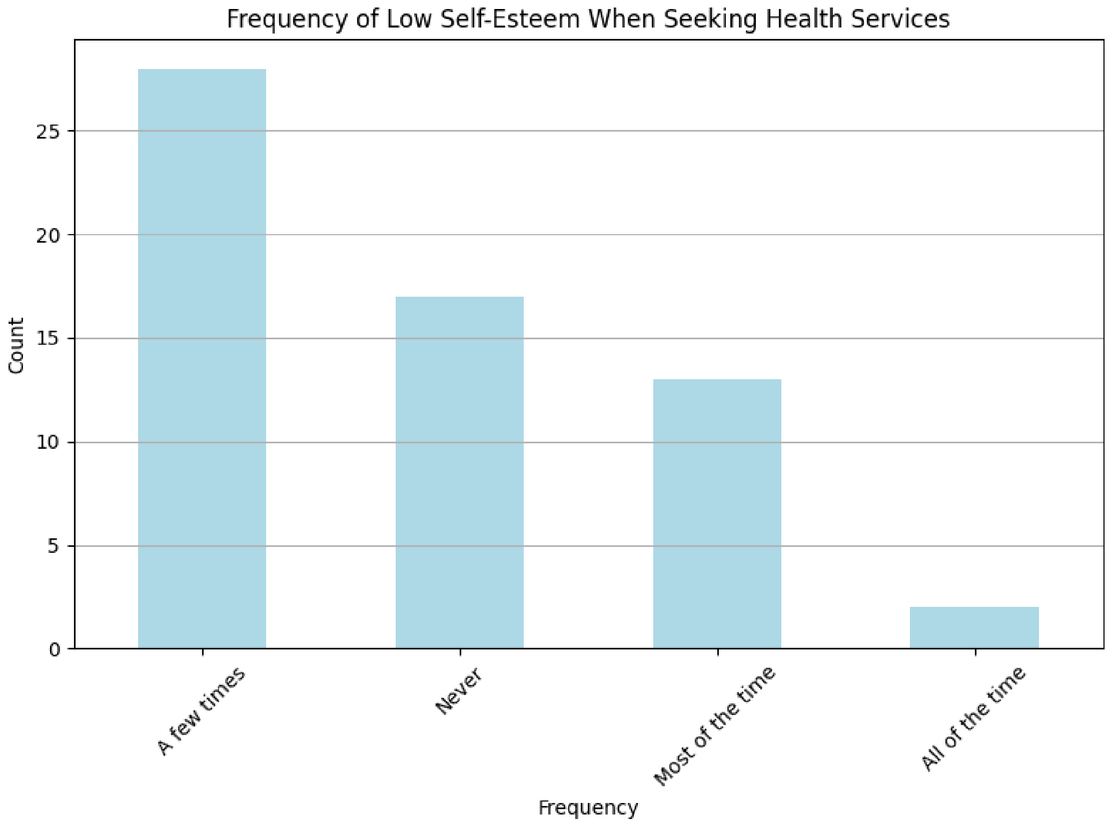


[122]:

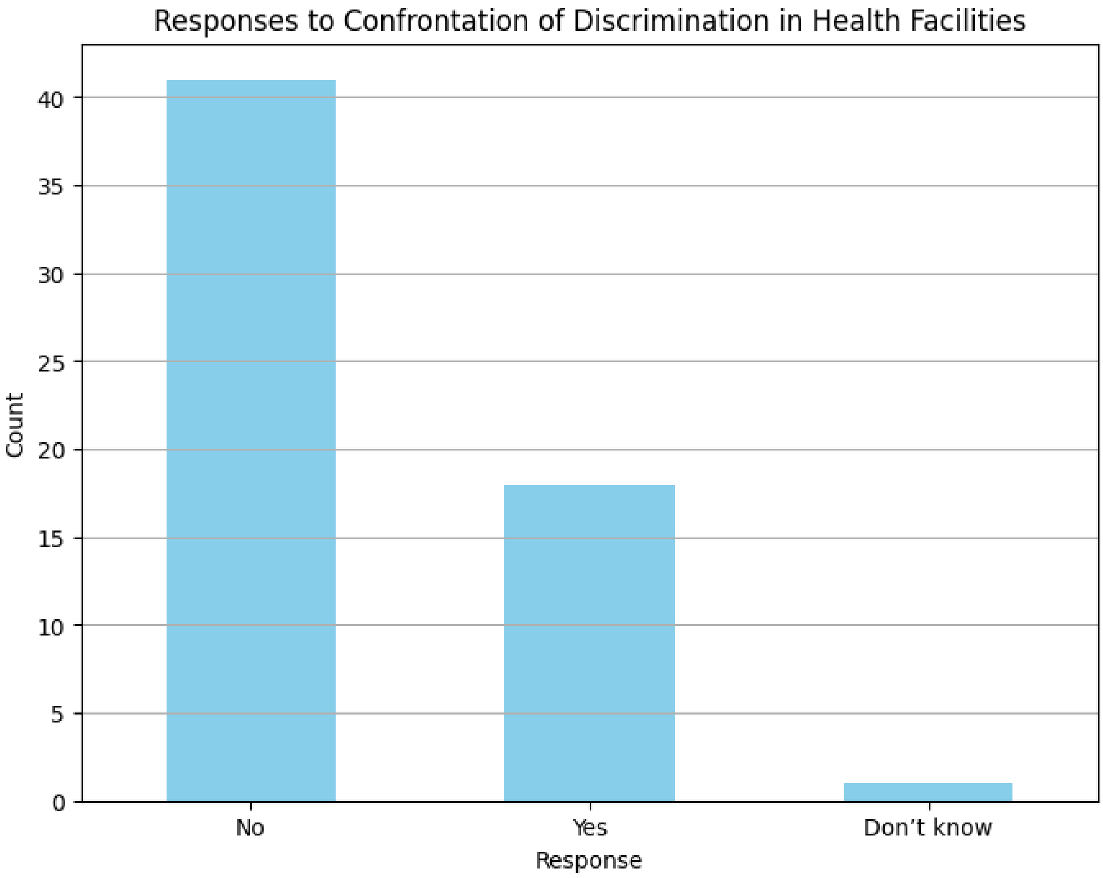
```
self_esteem_responses = kisumu_respondents['53. In the last 3, how often did␣
↪you experience low self-esteem when seeking services at a health facility?'].
↪str.strip()
```

```python
# Count the occurrences of each response
self_esteem_counts = self_esteem_responses.value_counts()

# Print the counts
print(self_esteem_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
self_esteem_counts.plot(kind='bar', color='lightblue')
plt.title('Frequency of Low Self-Esteem When Seeking Health Services')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
A few times         28
Never               17
Most of the time    13
All of the time      2
```

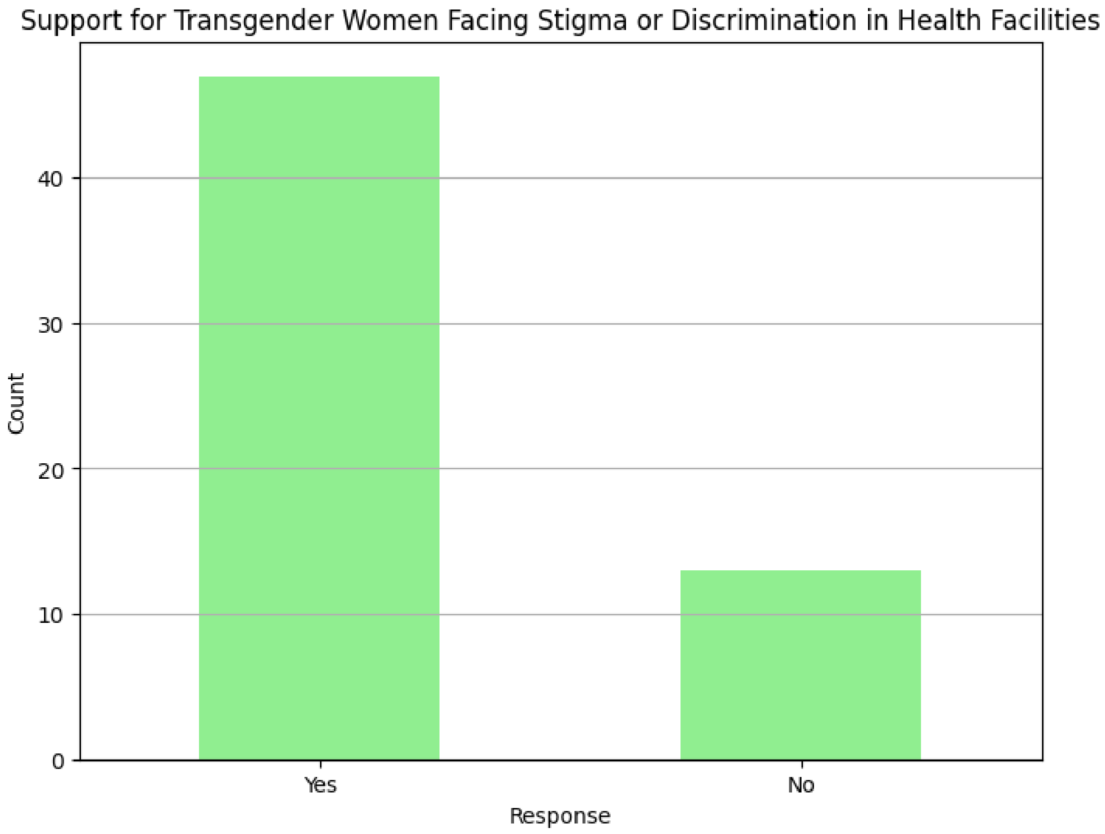


[123]:
```
response_counts = kisumu_respondents['54. In the last 3 months, did you ever␣
 ↪confront, challenge or educate a health care worker in a facility who was␣
 ↪Stigmatising and/or discriminating against you or other transgender women␣
 ↪within the facility?'].value_counts()

# Plotting the responses
plt.figure(figsize=(8, 6))
response_counts.plot(kind='bar', color='skyblue')
plt.title('Responses to Confrontation of Discrimination in Health Facilities')
plt.xlabel('Response')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.grid(axis='y')

plt.show()
```



[124]:

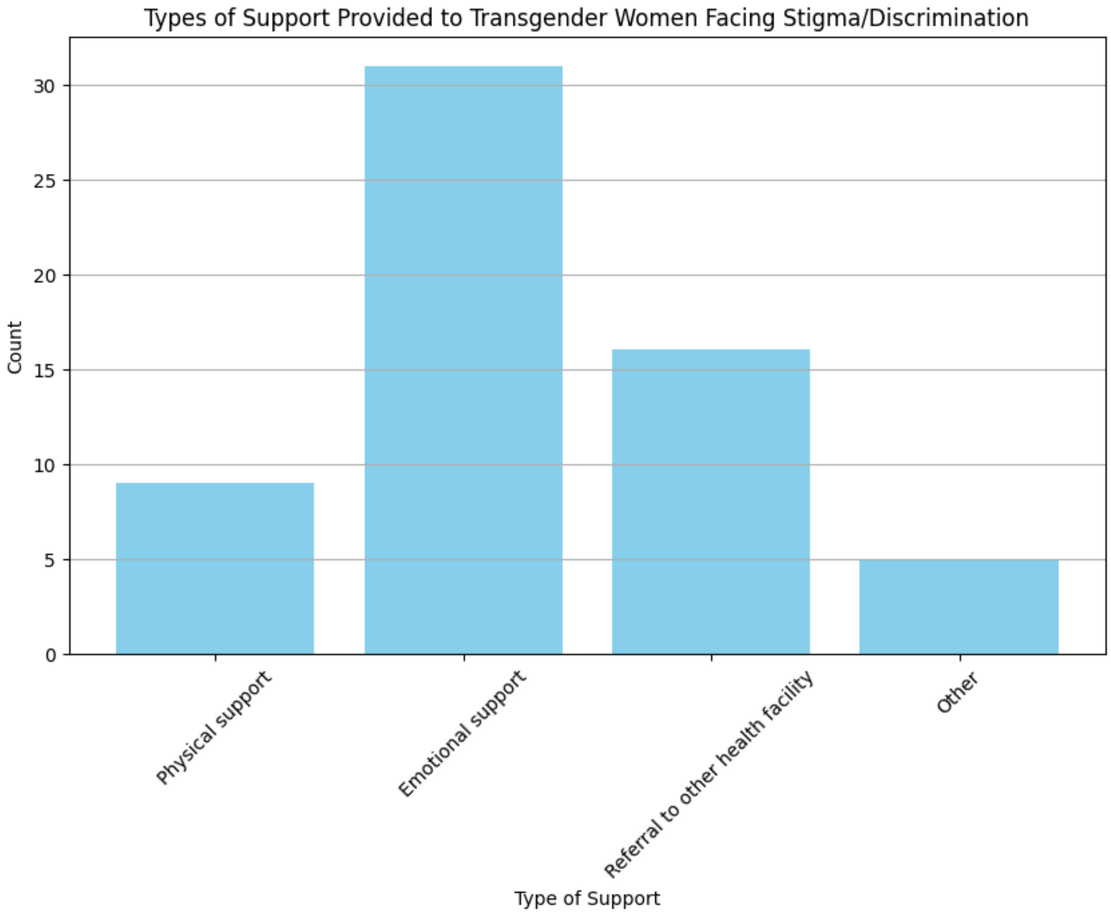
```
response_counts_advocacy = kisumu_respondents['55. In the last 3  months,did␣
 ↪you support any transgender woman/ women who has/have been stigmatised and/
 ↪or discriminated against in a health care facility?'].value_counts()

# Plotting the responses
plt.figure(figsize=(8, 6))
response_counts_advocacy.plot(kind='bar', color='lightgreen')
plt.title('Support for Transgender Women Facing Stigma or Discrimination in␣
 ↪Health Facilities')
plt.xlabel('Response')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.grid(axis='y')

plt.show()
```



[125]:

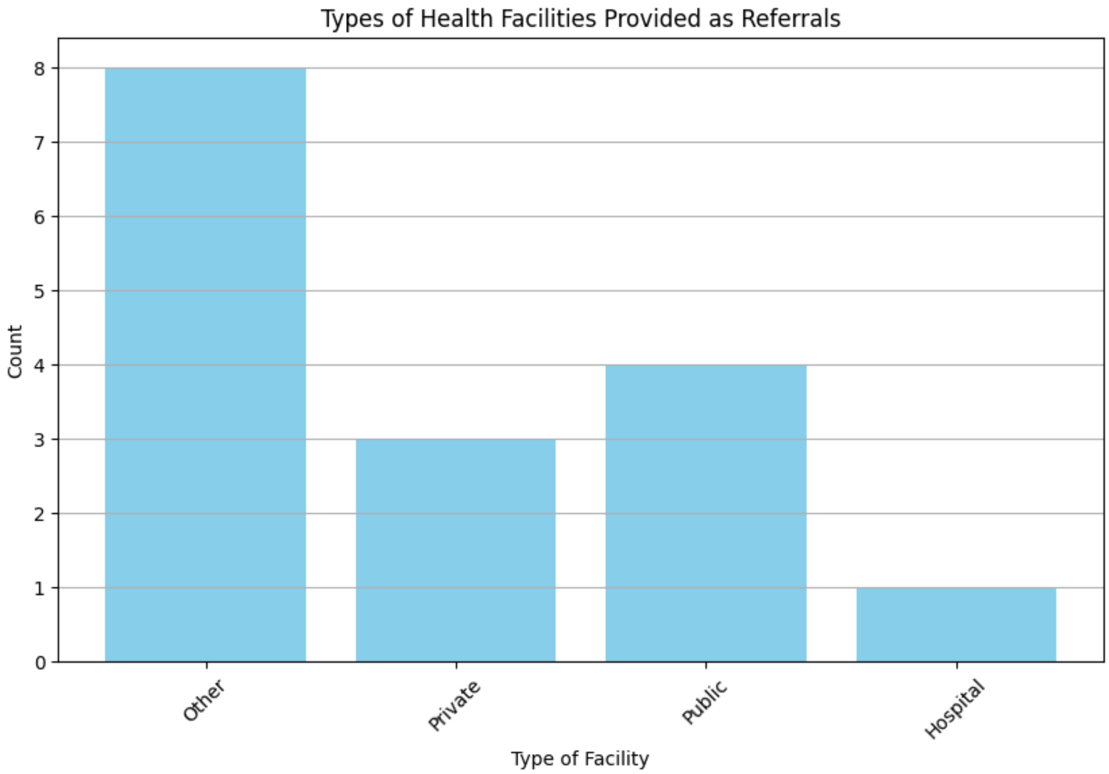
```
support_responses = kisumu_respondents['56. If Yes, what types of support did␣
 ↪you provide?'].dropna().str.strip()

# Splitting the responses by their content
```

```
types_of_support = {
    'Physical support': 0,
    'Emotional support': 0,
    'Referral to other health facility': 0,
    'Other': 0
}

# Counting each type of support
for response in support_responses:
    if 'Physical support' in response:
        types_of_support['Physical support'] += 1
    if 'Emotional support' in response:
        types_of_support['Emotional support'] += 1
    if 'Referral to other health facility' in response:
        types_of_support['Referral to other health facility'] += 1
    if 'other(' in response.lower():
        types_of_support['Other'] += 1

# Plotting the results
plt.figure(figsize=(10, 6))
plt.bar(types_of_support.keys(), types_of_support.values(), color='skyblue')
plt.title('Types of Support Provided to Transgender Women Facing Stigma/
 ↪Discrimination')
plt.xlabel('Type of Support')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')

plt.show()
```



[126]:

```
facility_responses = kisumu_respondents['Specify type of facility'].dropna().
 ↪str.strip()

# Cleaning and standardizing facility types
facility_counts = defaultdict(int)

for response in facility_responses:
    # Standardize variations and categorize
    if 'public' in response.lower():
        facility_counts['Public'] += 1
    elif 'private' in response.lower():
        facility_counts['Private'] += 1
    elif 'ngo' in response.lower():
        facility_counts['NGO'] += 1
    elif 'clinic' in response.lower():
        facility_counts['Clinic'] += 1
    elif 'hospital' in response.lower():
        facility_counts['Hospital'] += 1
    else:
        facility_counts['Other'] += 1

# Plotting the results
plt.figure(figsize=(10, 6))
plt.bar(facility_counts.keys(), facility_counts.values(), color='skyblue')
plt.title('Types of Health Facilities Provided as Referrals')
plt.xlabel('Type of Facility')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')

plt.show()
```



The list of facilities mentioned by the respondents include:

```
MAYGO Milimani & Migosi DICE
MAAYGO Milimani DICE
Private
Private hospital
Public hospital
Russia District Hospital
Anza Mapema
```

```
Dice
Maygo
NGO
Private facility
Clinic
Ngo
Public facility
Westlands health facility
KASARANI HEALTH CENTER
Public
Amref related facility at Juja
WESTLANDS HEALTH FACILITY
Roysambu at G complex a ladies private facility
Transform CBD
Westland facility
Ishtar rice, kyumbi dice
A private hospital in Westlands
Transform facility, Jinsiangu at Westlands
Kitui general
```

These facilities include a mix of public hospitals, private hospitals, clinics, NGO facilities, and specific named health centers and hospitals.

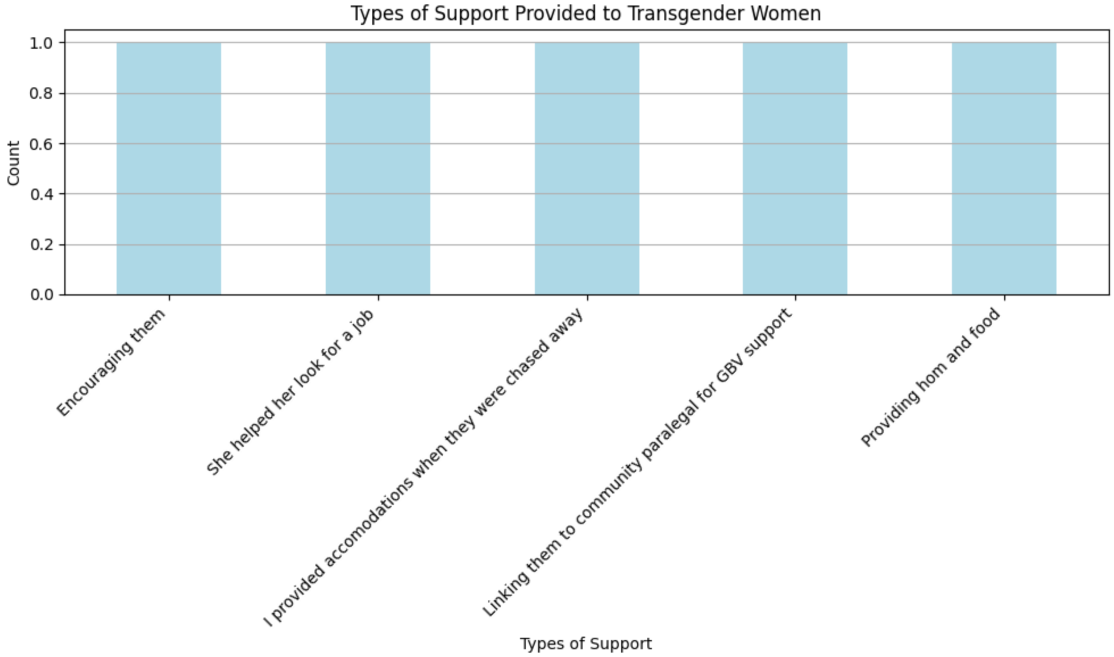
Other support given by the respondents tothe transgender people include: Encouraging them She helped her look for a job I provided accommodations when they were chased away Linking them to community paralegal for GBV support Providing home and food

These responses indicate various forms of support given to transgender women who have experienced stigma or discrimination in healthcare facilities. The support includes emotional encouragement, practical assistance in finding employment, providing accommodation when needed, connecting with legal support for gender-based violence (GBV), and offering basic necessities like housing and food

[127]:

```python
# Extracting and cleaning the responses
other_support_responses = kisumu_respondents['Specify other type of support'].
  ↪str.strip().dropna()

# Counting the occurrences of each response
other_support_counts = other_support_responses.value_counts()

# Printing the counts
print(other_support_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(10, 6))
other_support_counts.plot(kind='bar', color='lightblue')
plt.title('Types of Support Provided to Transgender Women')
plt.xlabel('Types of Support')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
Encouraging them                                    1
She helped her look for a job                       1
I provided accomodations when they were chased away    1
Linking them to community paralegal for GBV support    1
Providing hom and food                              1
```



[ ]:
```

```

[ ]:
```

```

[ ]:
```

```

1.2 Nairobi Preliminary Data Analysis

Data Collection in Nairobi started on 18th July, 2022 through to 22nd July 2022. Some of the respondents mobilized to join the study refused saying they had meetings and others that the 1000 shillings reimbursement was insufficient as they had to take bolt to cabs to come to the meeting (About 8).

A transgender refugee woman was identified to be homeless and her details were forwarded to Jinsiangu the hosting organization.

The total number of respondents who completed the screening tool are:

```
[133]: Nairobi_data_collection = df[(df['today'] >= '2024-06-18') & (df['today'] <=␣
        ↪'2024-06-22') ]

       len(Nairobi_data_collection)
```

```
[133]: 95
```

The total number of respondents who were eligible for the research and proceeded to be administered to the questionnaire are:

```
[134]: nairobi_respondents = df[(df['today'] >= '2024-06-18') & (df['today'] <=␣
        ↪'2024-06-22')& (df['Final Outcome'] != 0) ]
       len(nairobi_respondents)
```

```
[134]: 85
```

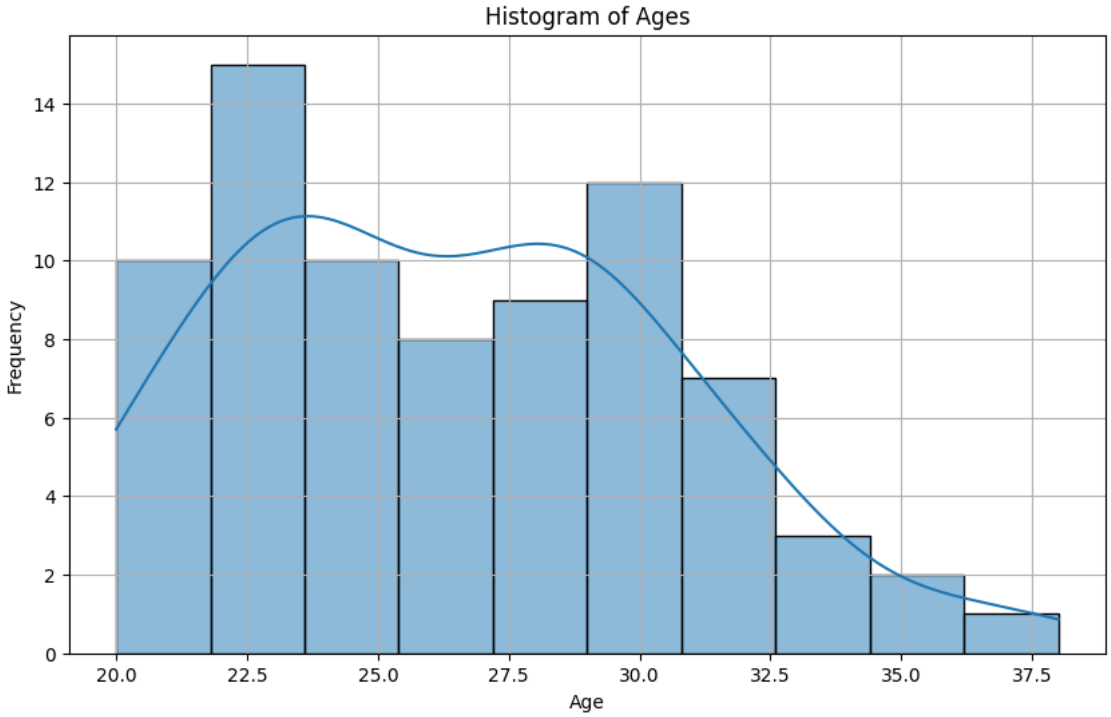
```
[135]: # Extracting relevant columns for socio-demographics
       socio_demographics = [
                             "1. Do you know your date of birth?",
                             "1a. If yes, What is your date of birth?",
                             "2.Do you know how old you were at your last birthday?",
                             "2a.If yes, how old were you at your last birthday?",
                             "3. What is the highest level of school you attended?",
                             "4. What is your marital status?",
                             "5. What is your current religious affiliation?",
                             "Specify other",
                             "6. What is your occupation",

                              ]
       nairobi_demographics = nairobi_respondents[socio_demographics]
       nrb = nairobi_demographics
```

```
[136]: nrb.columns =['If_Know_BD','Known_BD', 'Last_BD_Date', 'Last_BD_Age',␣
        ↪'Highest_Education', 'Marital_Status', 'Religios Affliation',␣
        ↪'Other_Religon','Occupation']
```

```
[137]: nrb_ages = Age.append(Imputed_BD.apply(lambda x: today.year - x.year if pd.
        ↪notnull(x) else None))

       # Remove NaN values
       nrb_ages = kd_ages.dropna()

       # Plotting
       plt.figure(figsize=(10, 6))
       sn.histplot(nrb_ages, bins=10, kde=True)
       plt.title('Histogram of Ages')
```

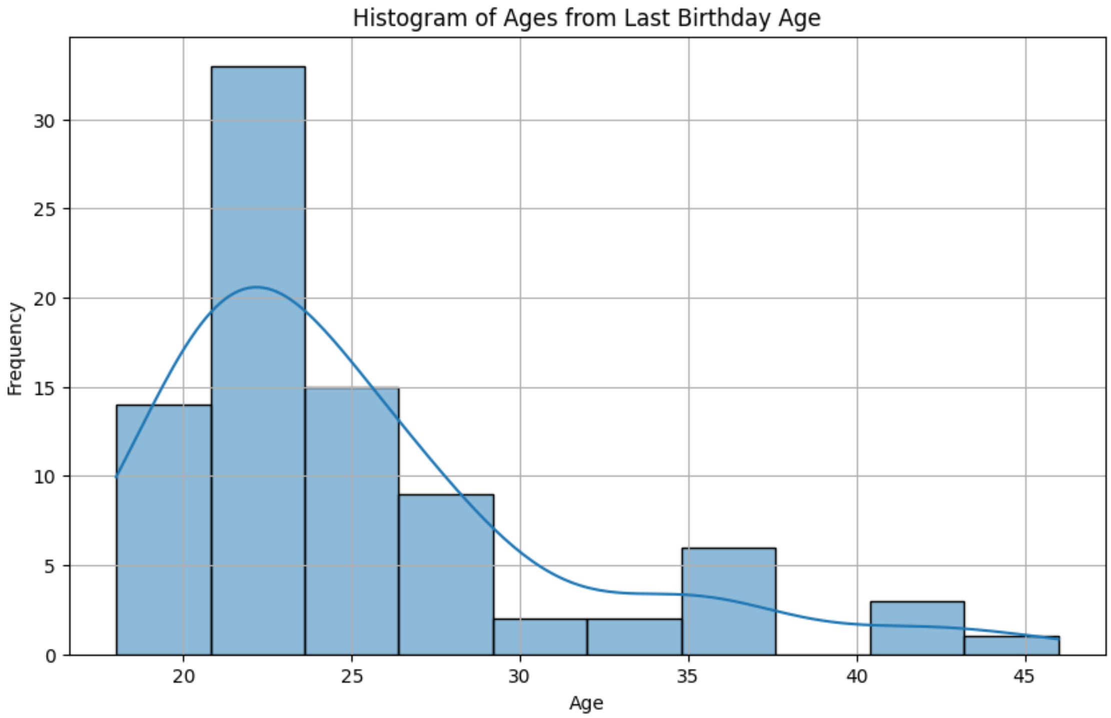
```python
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

```
/tmp/ipykernel_8037/1902265193.py:1: FutureWarning: The series.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  nrb_ages = Age.append(Imputed_BD.apply(lambda x: today.year - x.year if
pd.notnull(x) else None))
```



[138]:

```python
# Number of people who said they knew their age in the last birthday
#len(kd[kd['Last_BD_Date'] == 'Yes'])

Age_lBD = nrb['Last_BD_Age']

# Plotting
plt.figure(figsize=(10, 6))
sn.histplot(Age_lBD, bins=10, kde=True)
plt.title('Histogram of Ages from Last Birthday Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



Summary Statistics for Age

```
[139]: print("The mean age for Nairobi respondents is:", nrb_ages.mean(),"\n")
print("The median age for Nairobi respondents is:", nrb_ages.median(), "\n")
print("The mode age for Nairobi respondents is:",nrb_ages.mode(),"\n")

print("The mean age for Nairobi respondents based on their reported age on last␣
  ↪birthday is:",Age_lBD.mean(),"\n")
print("The median age for Nairobi respondents based on their reported age on␣
  ↪last birthday is:", Age_lBD.median(), "\n")
print("The mode age for Nairobi respondents based on their reported age on last␣
  ↪birthday is:",Age_lBD.mode(),"\n")
```

```
The mean age for Nairobi respondents is: 26.441558441558442

The median age for Nairobi respondents is: 26.0

The mode age for Nairobi respondents is: 0    23.0
Name: Known_BD, dtype: float64

The mean age for Nairobi respondents based on their reported age on last
birthday is: 25.129411764705882

The median age for Nairobi respondents based on their reported age on last
```

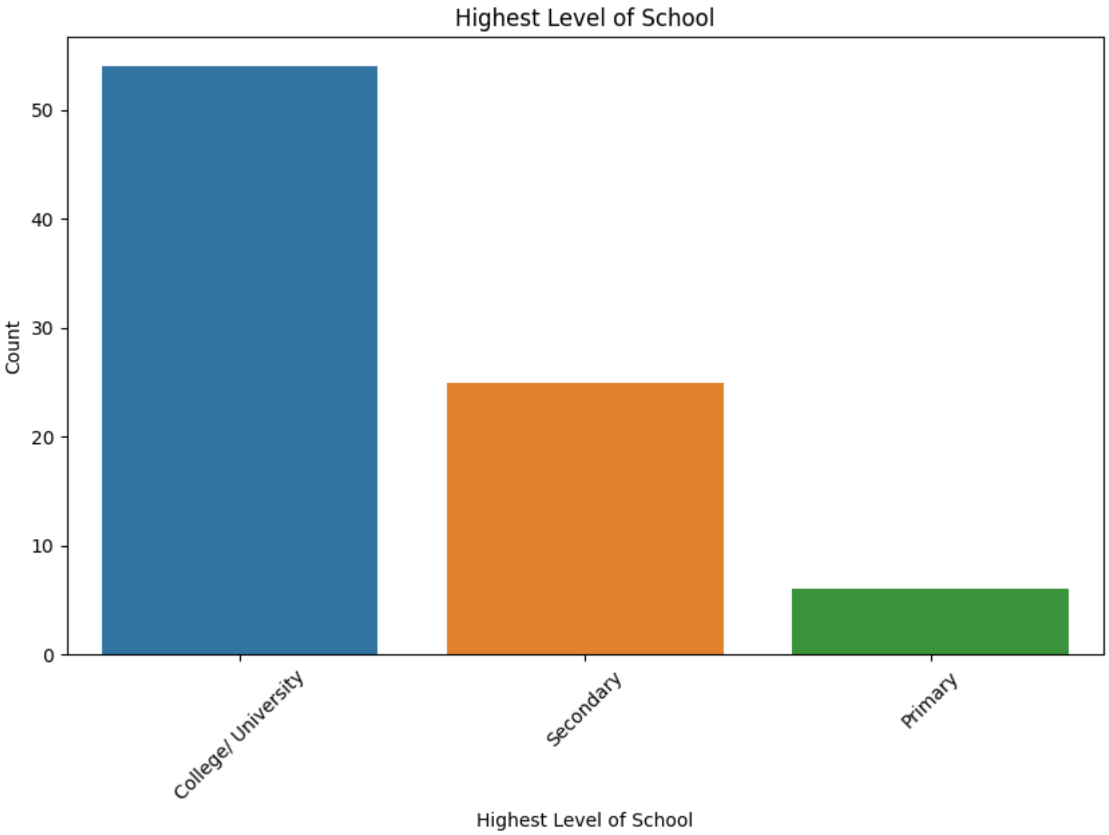
```
birthday is: 23.0

The mode age for Nairobi respondents based on their reported age on last
birthday is: 0    22.0
Name: Last_BD_Age, dtype: float64
```

Level of Education

[140]:

```
# # Valid categories
# valid_categories = ['Secondary', 'College/University', 'Primary']
# kd = kd[kd['Highest_Education'].isin(valid_categories)]
# This was truncating out Colleges and Universities
#A dist plot was also truncating Colleges and university even with the format␣
 ↪below

education = nairobi_respondents['3. What is the highest level of school you␣
 ↪attended?']
education_counts = nairobi_respondents['3. What is the highest level of school␣
 ↪you attended?'].value_counts()
education.value_counts()
education.value_counts(normalize=True)


print(education.value_counts().to_string(index=True))
plt.figure(figsize=(10, 6))
sn.countplot(data=nairobi_respondents, x='3. What is the highest level of␣
 ↪school you attended?', order=education_counts.index)
plt.title('Highest Level of School')
plt.xlabel('Highest Level of School')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
College/ University    54
Secondary              25
Primary                 6
```

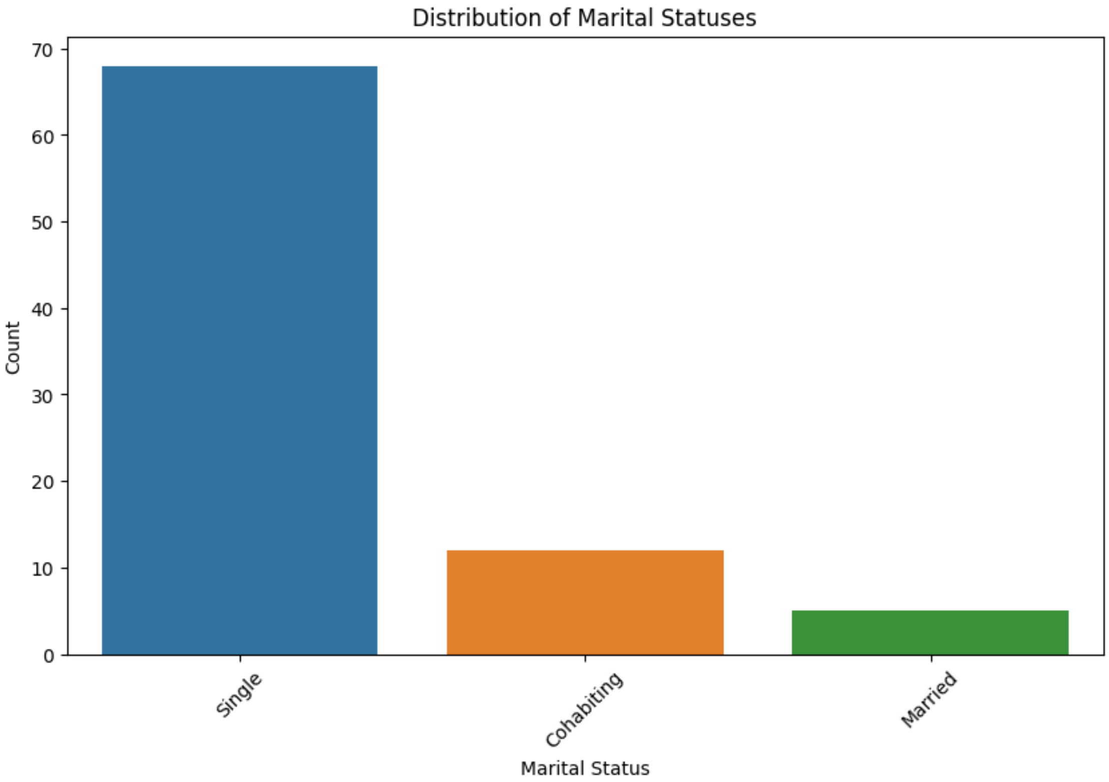


```
[ ]:
```

Marital Status

```
[141]: marital_status_counts = nairobi_respondents['4. What is your marital status?'].
        ↪value_counts()

       print(marital_status_counts.to_string(index=True))

       plt.figure(figsize=(10, 6))
       sn.countplot(data=nairobi_respondents, x='4. What is your marital status?',␣
        ↪order=marital_status_counts.index)
       plt.title('Distribution of Marital Statuses')
       plt.xlabel('Marital Status')
       plt.ylabel('Count')
       plt.xticks(rotation=45)
       plt.show()
```

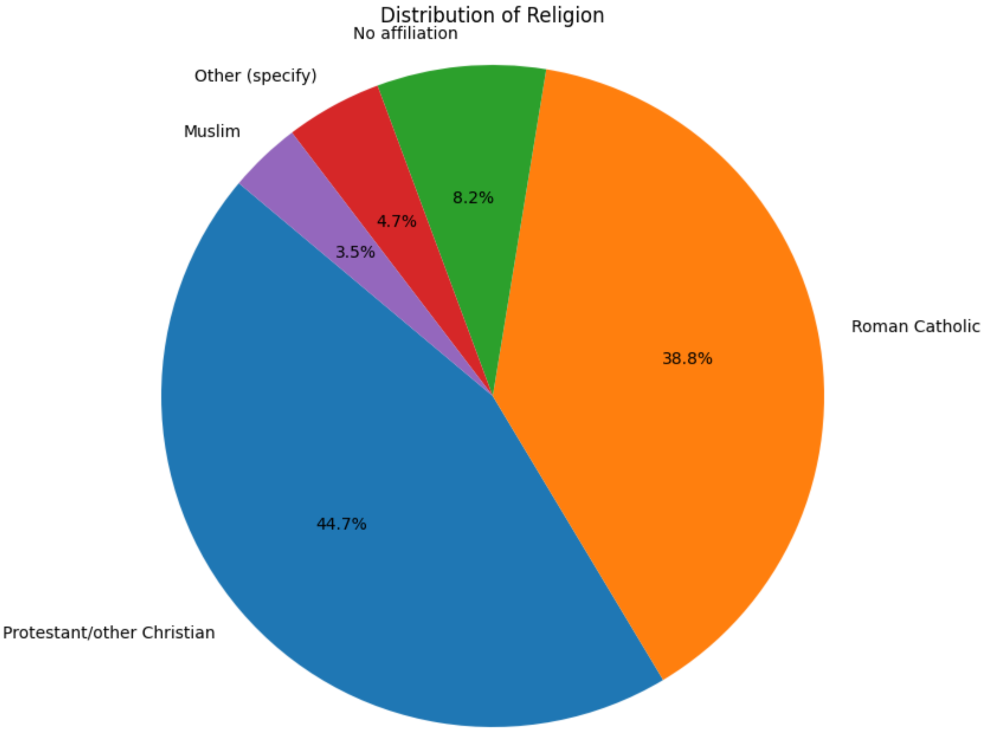
```
Single        68
Cohabiting    12
Married        5
```



[142]:
```
#### Religion
```

[143]:
```
religion = nairobi_respondents['5. What is your current religious affiliation?']
religion_counts = religion.value_counts()
print(religion_counts.to_string(index=True))

# Plotting the data
plt.figure(figsize=(12, 8))
plt.pie(religion_counts.values, labels=religion_counts.index, autopct='%1.
 ↪1f%%', startangle=140)
plt.title('Distribution of Religion')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

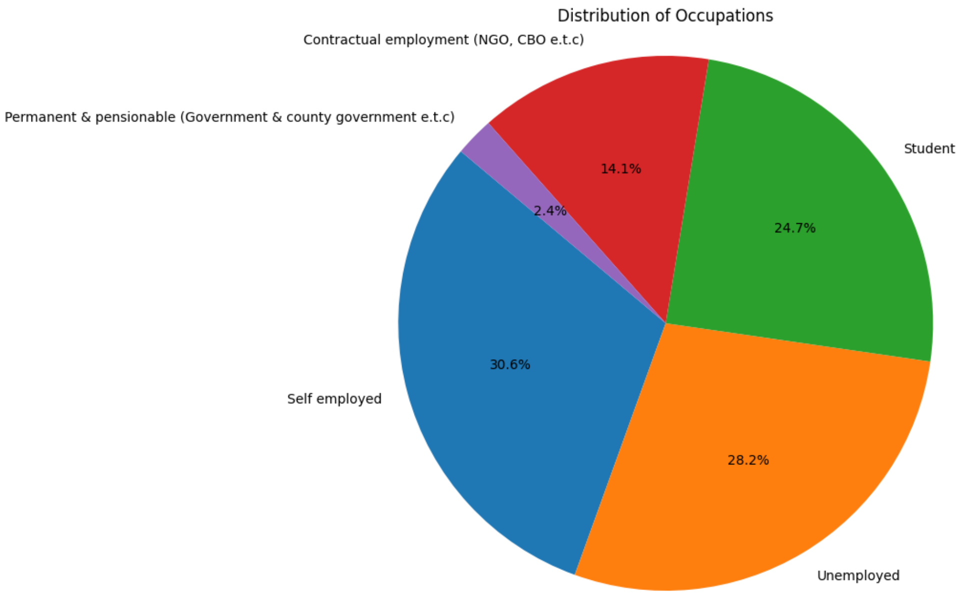
```
Protestant/other Christian    38
Roman Catholic                33
No affiliation                 7
Other (specify)                4
Muslim                         3
```



[144]:
```
#### Occupation
```

[145]:
```
occupation= nairobi_respondents['6. What is your occupation']
occupation_counts = occupation.value_counts()
print(occupation_counts.to_string(index=True))


# Plotting the data
plt.figure(figsize=(12, 8))
plt.pie(occupation_counts.values, labels=occupation_counts.index, autopct='%1.
  ↪1f%%', startangle=140)
plt.title('Distribution of Occupations')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

```
Self employed                                                   26
Unemployed                                                      24
Student                                                         21
Contractual employment (NGO, CBO e.t.c)                         12
Permanent & pensionable (Government & county government e.t.c)   2
```



1.2.1 Summary Statistics for Demographics

[ ]:

```
```

1.2.2 Sexual Behavior

[146]:

```
sexual_intercourse = nairobi_respondents['7. In the past 12 months, have you␣
↪ever had sexual intercourse?']

# Counting the occurrences of each response
sexual_intercourse_counts = sexual_intercourse.value_counts()
print(sexual_intercourse_counts.to_string(index=True))
print('\nPercentages\n',sexual_intercourse.value_counts(normalize=True).
↪to_string(index=True))
```

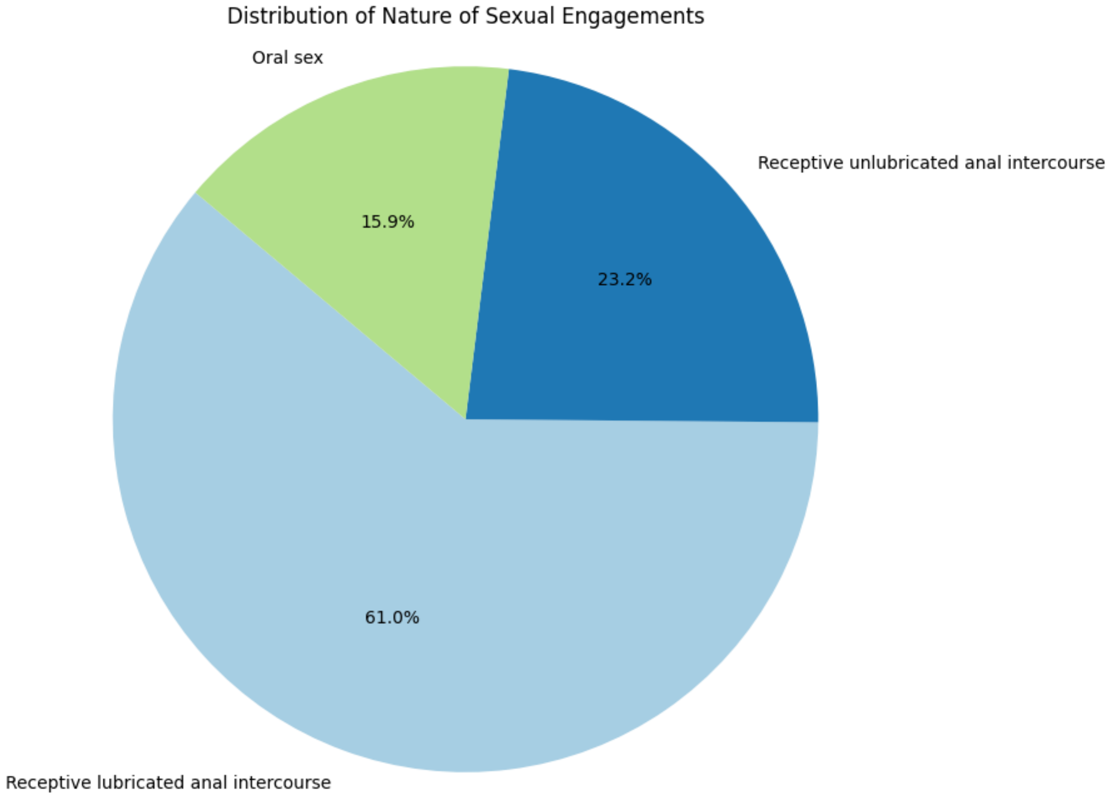
```
Yes          82
No            2
No answer     1

Percentages
 Yes          0.964706
No           0.023529
No answer    0.011765
```

[147]:

```
nature_of_sex =  nairobi_respondents['8. In the past 12 months you had sex,␣
↪what was the nature of the sexual engagement?']
```

```
nature_of_sex_counts = nature_of_sex.value_counts()

print(nature_of_sex_counts.to_string(index=True))
print('\nPercentages\n',nature_of_sex.value_counts(normalize=True).
 ↪to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(nature_of_sex_counts, labels=nature_of_sex_counts.index, autopct='%1.
 ↪1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Distribution of Nature of Sexual Engagements')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

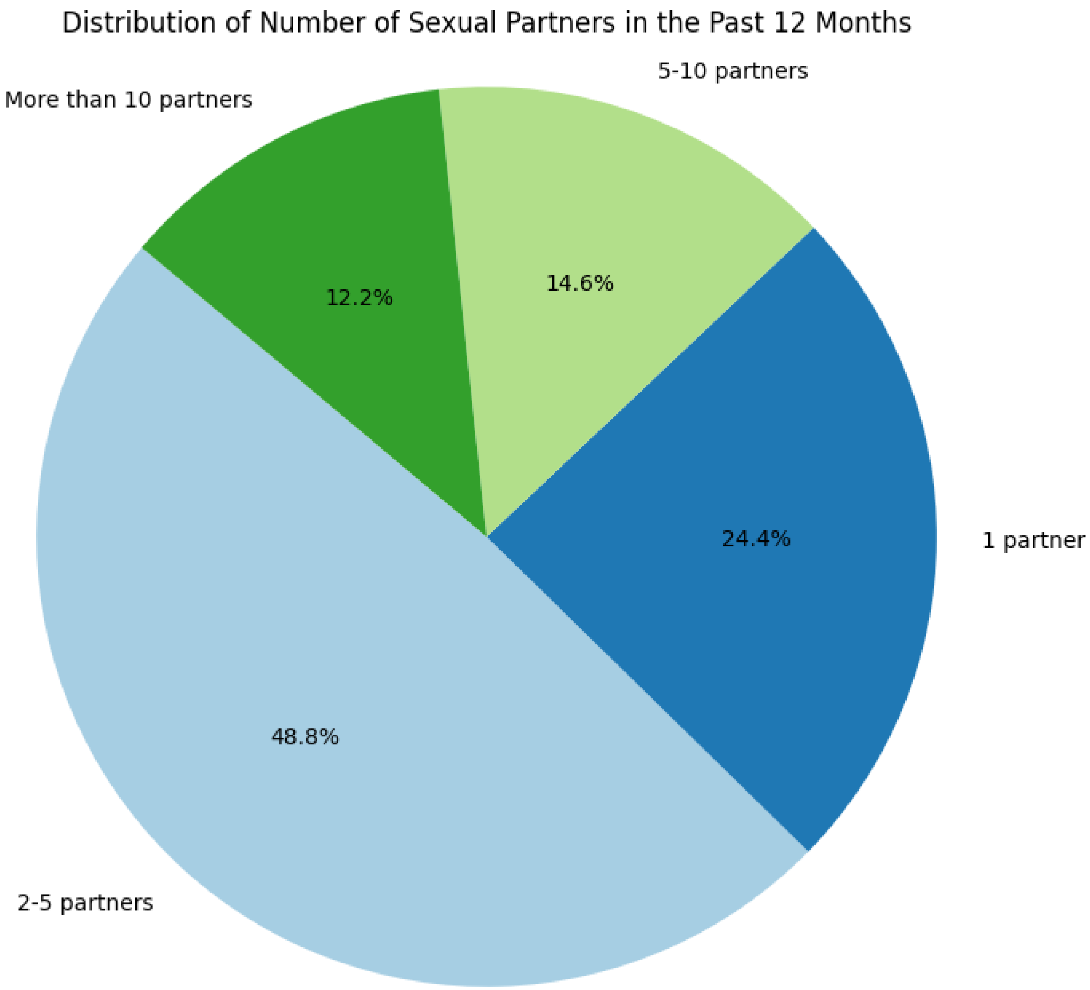
```
Receptive lubricated anal intercourse      50
Receptive unlubricated anal intercourse    19
Oral sex                                   13

Percentages
 Receptive lubricated anal intercourse      0.609756
Receptive unlubricated anal intercourse    0.231707
Oral sex                                   0.158537
```



[148]:

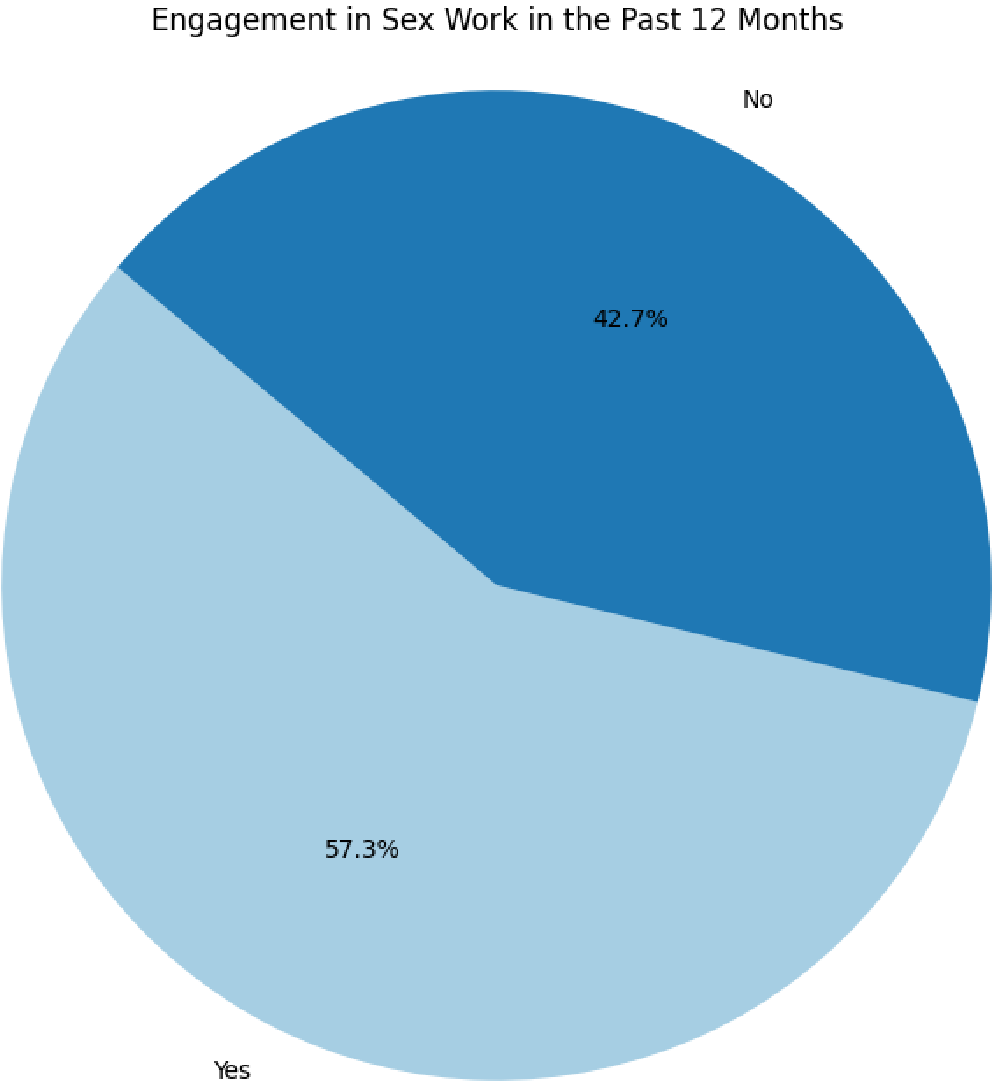
```
partners_no =  nairobi_respondents['9. How many partners did you have sex with␣
  ↪in the past 12 months?']
partners_no_counts = partners_no.value_counts()

print(partners_no_counts.to_string(index=True))
print('\nPercentages\n',partners_no.value_counts(normalize=True).
  ↪to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(partners_no_counts, labels=partners_no_counts.index, autopct='%1.1f%%',␣
  ↪startangle=140, colors=plt.cm.Paired.colors)
plt.title('Distribution of Number of Sexual Partners in the Past 12 Months')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

```
2-5 partners             40
1 partner                20
5-10 partners            12
More than 10 partners    10

Percentages
 2-5 partners             0.487805
1 partner                0.243902
5-10 partners            0.146341
More than 10 partners    0.121951
```



```
[149]: sex_work = nairobi_respondents['10. In the past 12 months, did you have sex in␣
         ↪exchange for money, favours or goods?']

       sex_work_counts = sex_work.value_counts()

       print(sex_work_counts.to_string(index=True))

       plt.figure(figsize=(8, 8))
       plt.pie(sex_work_counts, labels=sex_work_counts.index, autopct='%1.1f%%',␣
         ↪startangle=140, colors=plt.cm.Paired.colors)
       plt.title('Engagement in Sex Work in the Past 12 Months')
       plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

       # Display the plot
       plt.show()
```

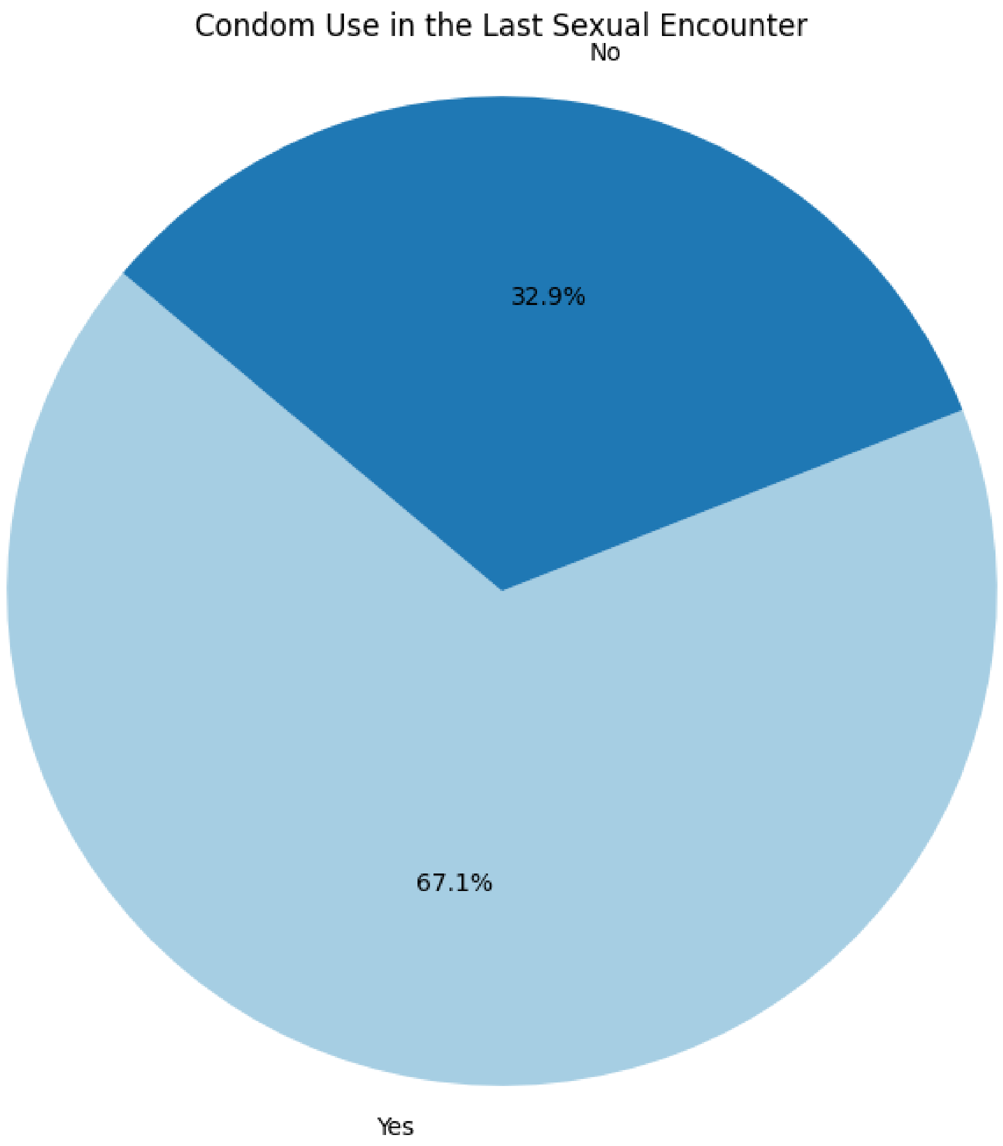
```
Yes     47
No      35
```



```
[150]: # Extracting the relevant column
       condom_use = nairobi_respondents['11. Did you use a condom the last time you␣
        ↪had sex with in the past 12 months?']

       # Counting the occurrences of each response
       condom_use_counts = condom_use.value_counts()

       # Print percentages
```

```python
print('\nPercentages\n', condom_use_counts.value_counts(normalize=True).to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(condom_use_counts, labels=condom_use_counts.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Condom Use in the Last Sexual Encounter')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

```
Percentages
 55    0.5
27    0.5
```

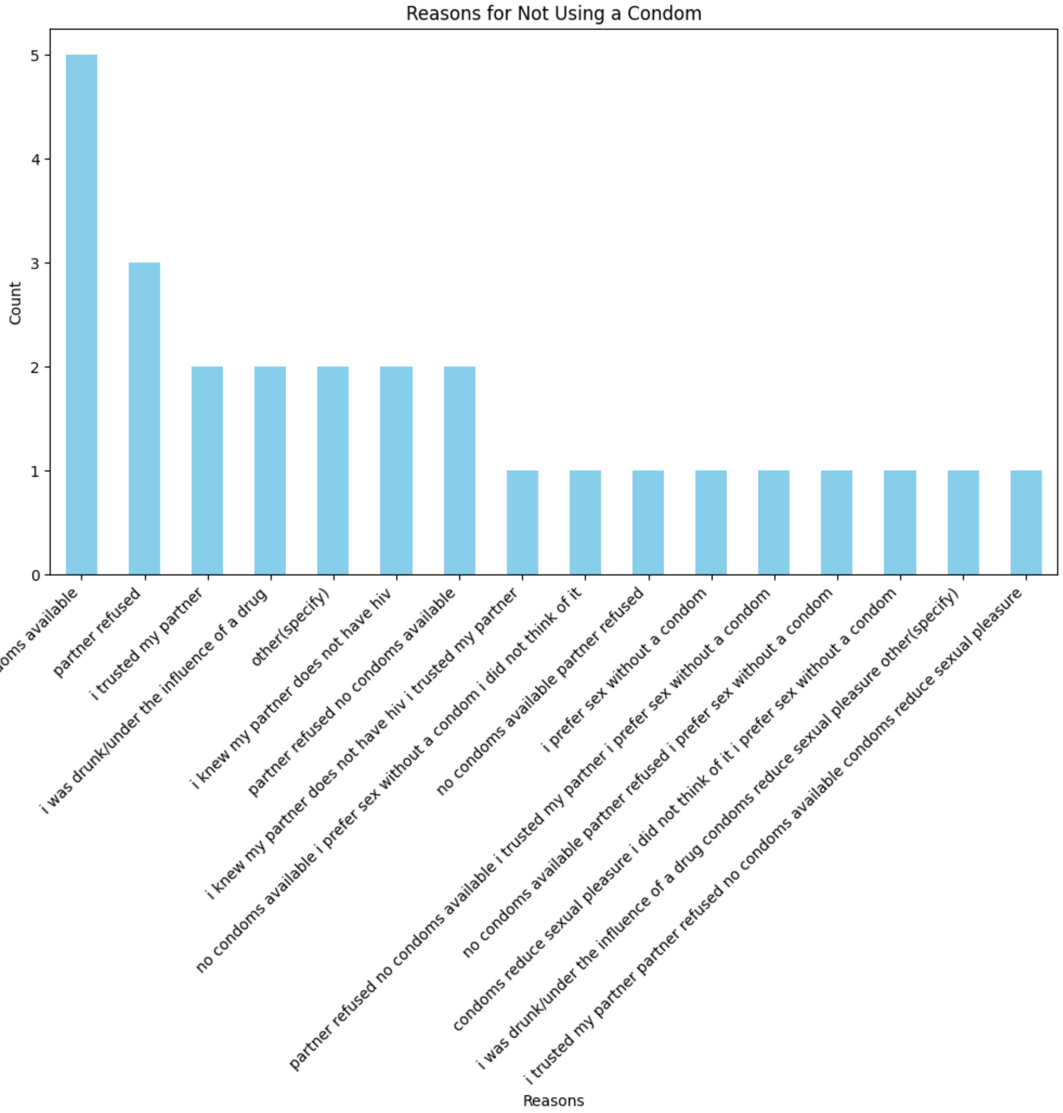


[151]:
```python
# Extracting the relevant column
reasons_for_no_condom = nairobi_respondents['12.why  did you not use a condom?']

# Cleaning and normalizing text (optional depending on data cleanliness)
reasons_for_no_condom = reasons_for_no_condom.str.strip().str.lower()

# Counting the occurrences of each reason
reason_counts = reasons_for_no_condom.value_counts()


print('Reasons', reason_counts.to_string(index=True))
print('Reason count sum', reason_counts.sum() )
```

```
# Print percentages if needed
print('\nPercentages\n', reasons_for_no_condom.value_counts(normalize=True).
 ↪to_string(index=True))

# Plotting the counts as a bar chart
plt.figure(figsize=(12, 8))  # Adjust the figure size as needed
reason_counts.plot(kind='bar', color='skyblue')
plt.title('Reasons for Not Using a Condom')
plt.xlabel('Reasons')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')

# Adjusting margins if necessary
plt.subplots_adjust(bottom=0.25)  # Example adjustment, increase if needed

# Display the plot
plt.show()
```

```
Reasons no condoms available
5
partner refused
3
i trusted my partner
2
i was drunk/under the influence of a drug
2
other(specify)
2
i knew my partner does not have hiv
2
partner refused no condoms available
2
i knew my partner does not have hiv i trusted my partner
1
no condoms available i prefer sex without a condom i did not think of it
1
no condoms available partner refused
1
i prefer sex without a condom
1
partner refused no condoms available i trusted my partner i prefer sex without a
condom     1
no condoms available partner refused i prefer sex without a condom
1
condoms reduce sexual pleasure i did not think of it i prefer sex without a
condom          1
i was drunk/under the influence of a drug condoms reduce sexual pleasure
```

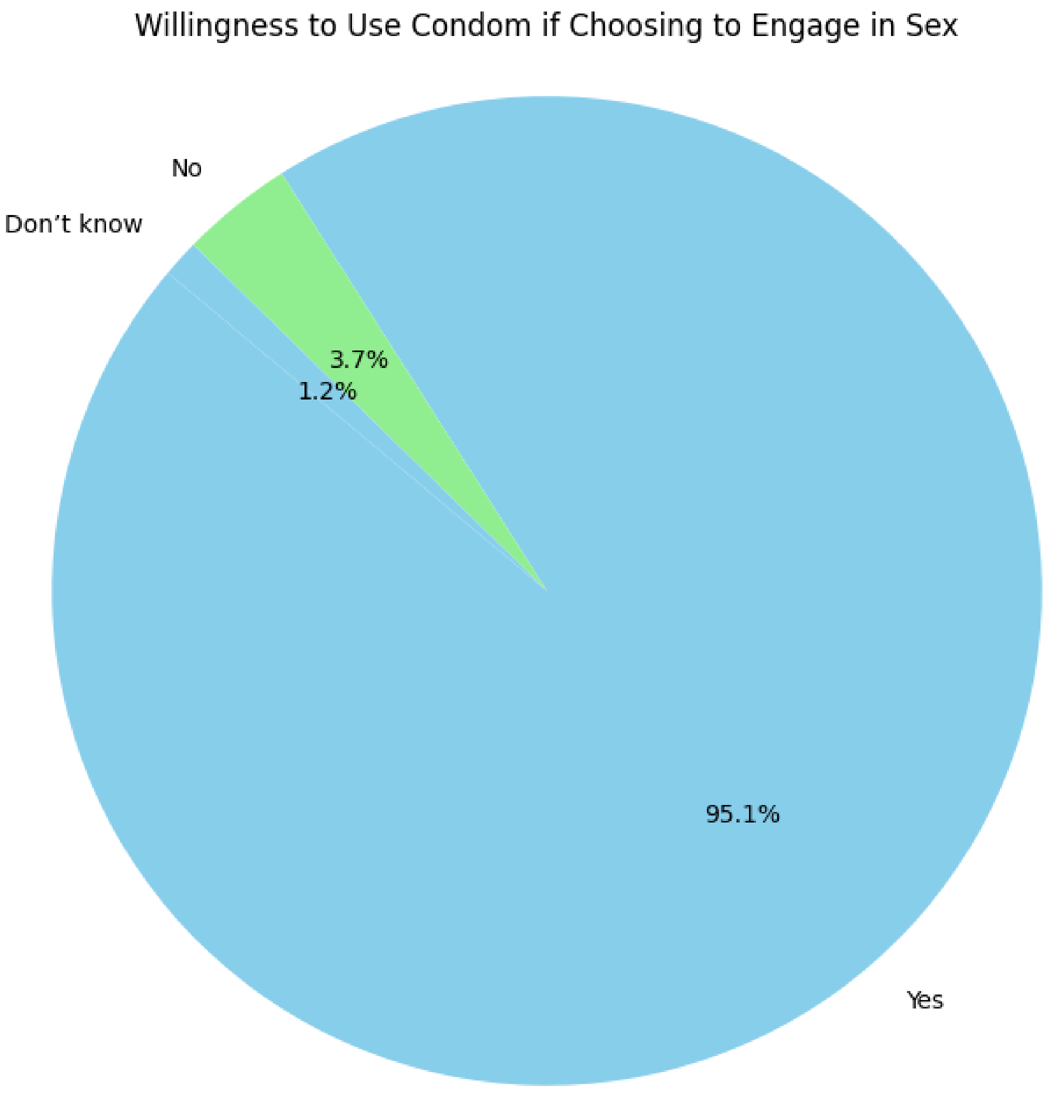
```
other(specify)     1
i trusted my partner partner refused no condoms available condoms reduce sexual
pleasure    1
Reason count sum 27

Percentages
 no condoms available
0.185185
partner refused
0.111111
i trusted my partner
0.074074
i was drunk/under the influence of a drug
0.074074
other(specify)
0.074074
i knew my partner does not have hiv
0.074074
partner refused no condoms available
0.074074
i knew my partner does not have hiv i trusted my partner
0.037037
no condoms available i prefer sex without a condom i did not think of it
0.037037
no condoms available partner refused
0.037037
i prefer sex without a condom
0.037037
partner refused no condoms available i trusted my partner i prefer sex without a
condom     0.037037
no condoms available partner refused i prefer sex without a condom
0.037037
condoms reduce sexual pleasure i did not think of it i prefer sex without a
condom          0.037037
i was drunk/under the influence of a drug condoms reduce sexual pleasure
other(specify)     0.037037
i trusted my partner partner refused no condoms available condoms reduce sexual
pleasure    0.037037
```



[152]:
```
willing_to_use_condom = nairobi_respondents['13. Should you choose to engage in␣
 ↪sex, would you be willing to use condom?']

# Counting the occurrences of each response
willing_to_use_condom_counts = willing_to_use_condom.value_counts()


print(willing_to_use_condom_counts.to_string(index=True))
print('\nPercentages\n', willing_to_use_condom.value_counts(normalize=True).
 ↪to_string(index=True))


# Plotting the counts as a pie chart
```

```
plt.figure(figsize=(8, 8))
plt.pie(willing_to_use_condom_counts, labels=willing_to_use_condom_counts.
 ↪index, autopct='%1.1f%%', startangle=140, colors=['skyblue', 'lightgreen'])
plt.title('Willingness to Use Condom if Choosing to Engage in Sex')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# Display the plot
plt.show()
```

```
Yes            78
No              3
Don’t know      1

Percentages
 Yes            0.951220
No             0.036585
Don’t know     0.012195
```



[153]:

```
last_sex_date = nairobi_respondents['14. If No when was the last time you had␣
 ↪sex?']

# Convert to datetime if not already in datetime format
#last_sex_date = pd.to_datetime(last_sex_date, errors='coerce')

# Counting the occurrences of each date
last_sex_date_counts = last_sex_date.value_counts()
len(last_sex_date_counts)
#For Kisumu last sex date if no was 0

# # Convert to datetime if not already in datetime format
# last_sex_date = pd.to_datetime(last_sex_date, errors='coerce')

# # Counting the occurrences of each date
# last_sex_date_counts = last_sex_date.value_counts().sort_index()

# # Plotting the counts as a line chart
# plt.figure(figsize=(10, 6))
# last_sex_date_counts.plot(marker='o')
# plt.title('Last Time Respondents Had Sex')
# plt.xlabel('Date')
# plt.ylabel('Count')
# plt.grid(True)
# plt.xticks(rotation=45)

# # Display the plot
# plt.tight_layout()
# plt.show()


##This forces us to skip to HIV Perception questions, from No. 21
```

[153]: 2

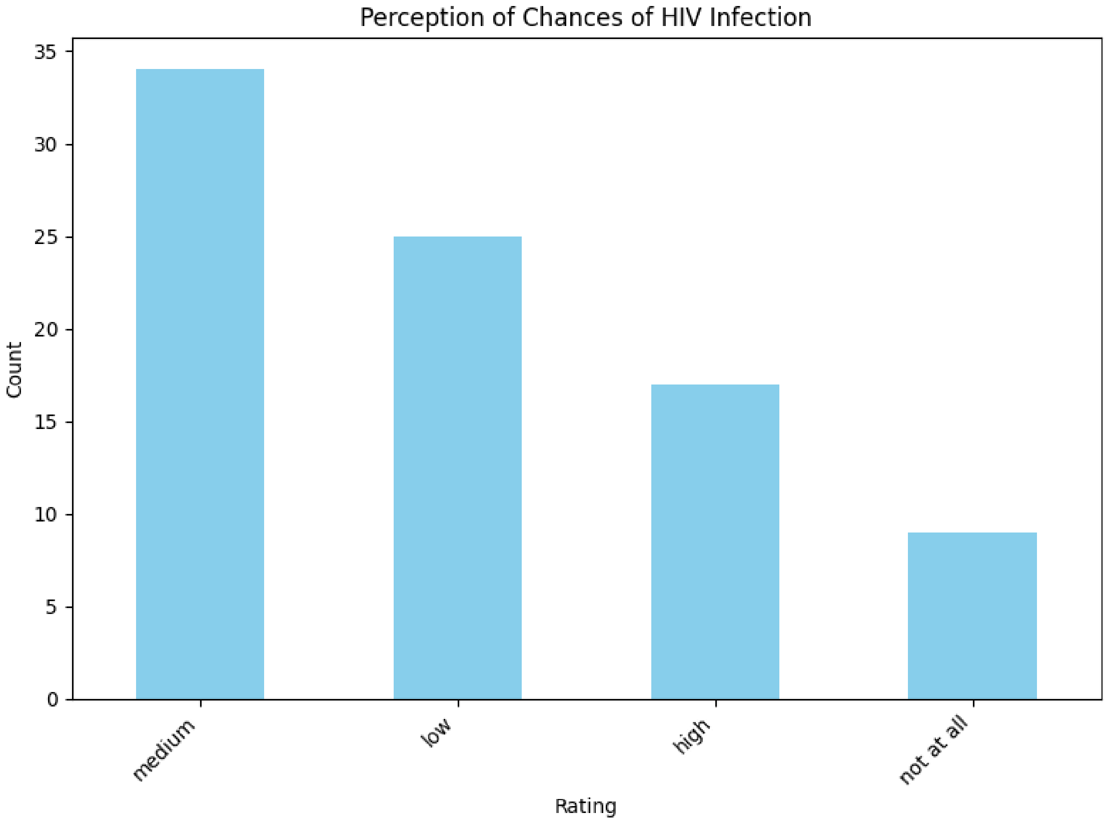
1.2.3 HIV Risk Perception

[154]:

```
# Extracting the relevant column
hiv_infection_chances = nairobi_respondents['21. How do you rate your chances␣
 ↪of HIV infection?']

# Cleaning and normalizing text (optional depending on data cleanliness)
hiv_infection_chances = hiv_infection_chances.str.strip().str.lower()

# Counting the occurrences of each category
chances_counts = hiv_infection_chances.value_counts()

print(chances_counts.to_string(index=True))
print('\nPercentages\n', hiv_infection_chances.value_counts(normalize=True).
 ↪to_string(index=True))


# Plotting the counts as a bar chart
plt.figure(figsize=(8, 6))
chances_counts.plot(kind='bar', color='skyblue')
plt.title('Perception of Chances of HIV Infection')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
```

```
# Display the plot
plt.tight_layout()
plt.show()
```

```
medium        34
low           25
high          17
not at all     9

Percentages
 medium        0.400000
low           0.294118
high          0.200000
not at all    0.105882
```

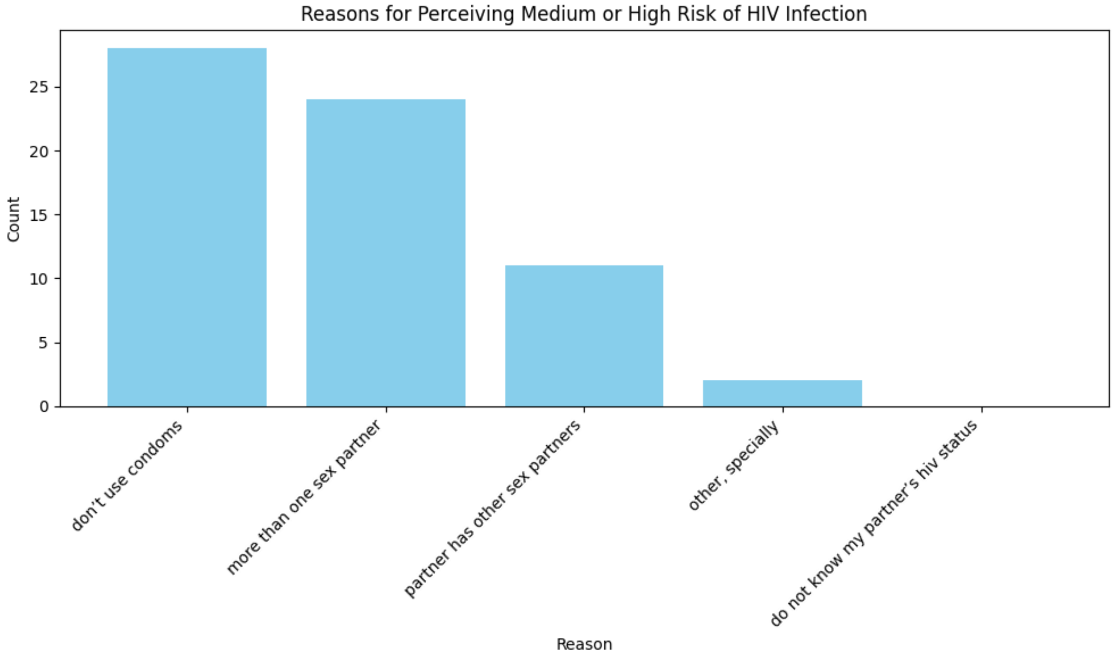


```
[155]: risk_perception_reasons = nairobi_respondents['22. Why do you think you have a␣
         ↪medium or high risk of getting HIV?']

       # Cleaning and normalizing text (optional depending on data cleanliness)
       risk_perception_reasons = risk_perception_reasons.str.strip().str.lower()
```

```python
# Defining key phrases for analysis
key_phrases = [
    'more than one sex partner',
    'don’t use condoms',
    'do not know my partner’s hiv status',
    'partner has other sex partners',
    'other, specially'
]

# Counting occurrences of key phrases
phrase_counts = {}
for phrase in key_phrases:
    phrase_counts[phrase] = risk_perception_reasons.str.contains(phrase).sum()

# Print the counts of each key phrase
print("Counts:\n", pd.Series(phrase_counts).to_string(index=True))

# Calculate and print the percentages of each key phrase
percentages = pd.Series(phrase_counts).value_counts(normalize=True) * 100
print("\nPercentages:\n", percentages.to_string(index=True))

# Sorting the results by frequency
sorted_counts = {k: v for k, v in sorted(phrase_counts.items(), key=lambda item:
 ↪ item[1], reverse=True)}

# Plotting the counts as a bar chart
plt.figure(figsize=(10, 6))
plt.bar(sorted_counts.keys(), sorted_counts.values(), color='skyblue')
plt.title('Reasons for Perceiving Medium or High Risk of HIV Infection')
plt.xlabel('Reason')
plt.ylabel('Count')

# Display the plot
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
Counts:
 more than one sex partner              24
don’t use condoms                      28
do not know my partner’s hiv status     0
partner has other sex partners         11
other, specially                        2

Percentages:
 24    20.0
```

```
28     20.0
0      20.0
11     20.0
2      20.0
```

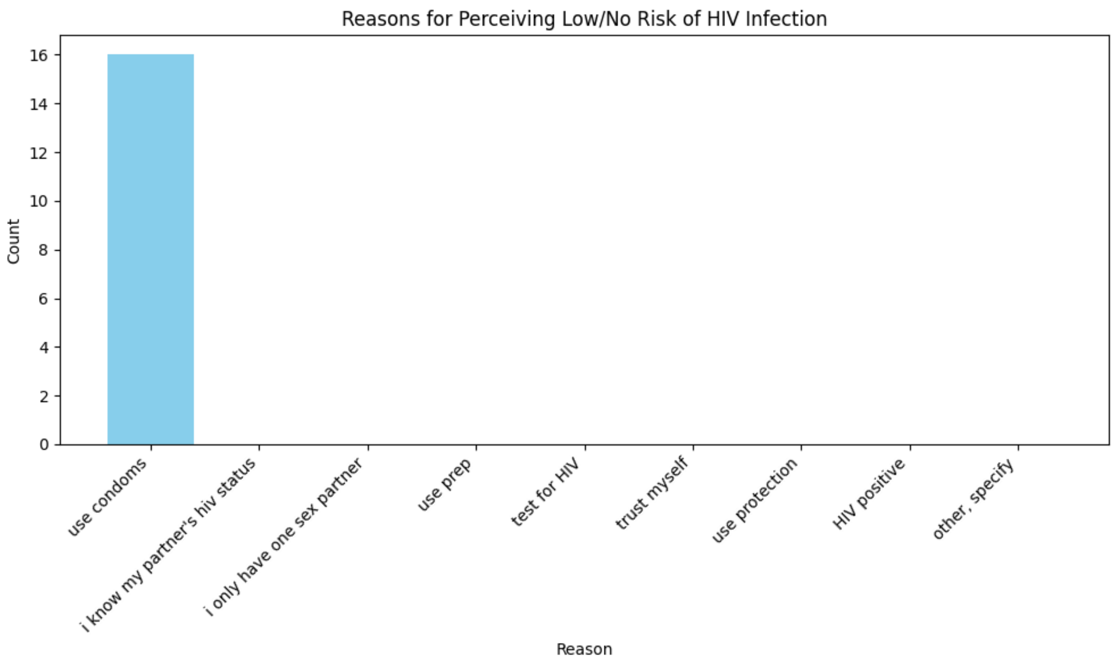


```
[156]: risk_perception_low_no = nairobi_respondents['23. If low or no,Why do you think␣
         ↪you have a low/ no risk chance of getting HIV?']

        # Cleaning and normalizing text for 'Others Specify'
        def clean_other_specify(text):
            if pd.isna(text):
                return ''
            text = text.strip().lower()  # Remove leading/trailing spaces and convert␣
         ↪to lowercase
            text = re.sub(r'\s+', ' ', text)  # Remove extra spaces
            # Normalize specific responses
            text = re.sub(r'\buse of condoms\b', 'use condoms', text)
            text = re.sub(r'\buse condom\b', 'use condoms', text)
            text = re.sub(r'\btest for hiv before sex\b', 'test for HIV', text)
            text = re.sub(r'\bgo for testing and use condom\b', 'test for HIV', text)
            text = re.sub(r'\btest for hiv\b', 'test for HIV', text)
            text = re.sub(r'\btest before sex\b', 'test for HIV', text)
            text = re.sub(r'\bi trust myself\b', 'trust myself', text)
            text = re.sub(r'\buse prep\b', 'use PrEP', text)
            text = re.sub(r'\buse prep and condom\b', 'use PrEP and condom', text)
            text = re.sub(r'\buse protection\b', 'use protection', text)
```

```
    text = re.sub(r'\buse of protection everytime engaging in sex\b', 'use␣
  ↪protection', text)
    text = re.sub(r'\bparticipate uses prep\b', 'use PrEP', text)
    text = re.sub(r'\buse protection and aware of sex partner status\b', 'use␣
  ↪protection and know partner HIV status', text)
    text = re.sub(r'\bi use protection and have low sexual partners\b', 'use␣
  ↪protection and low number of sexual partners', text)
    text = re.sub(r'\bvery few sexual partners\b', 'use protection and low␣
  ↪number of sexual partners', text)
    text = re.sub(r'\bthe last hiv test was negative\b', 'test for HIV', text)
    text = re.sub(r'\brespondent is hiv\b', 'HIV positive', text)
    text = re.sub(r'\bthe participant uses prep/date prep\b', 'HIV positive',␣
  ↪text)
    return text

# Apply the cleaning function to the column
risk_perception_low_no = risk_perception_low_no.apply(clean_other_specify)

# Define key phrases for analysis
key_phrases = [
    'use condoms',
    'i know my partner\'s hiv status',
    'i only have one sex partner',
    'use prep',
    'test for HIV',
    'trust myself',
    'use protection',
    'HIV positive',
    'other, specify'
]

# Initialize dictionary to store counts
phrase_counts = {phrase: 0 for phrase in key_phrases}

# Count occurrences of key phrases
for phrase in key_phrases:
    phrase_counts[phrase] = risk_perception_low_no.str.contains(phrase).sum()

# Print the counts of each key phrase
print("Counts:\n", pd.Series(phrase_counts).to_string())

# Optionally, calculate and print the percentages of each key phrase
percentages = pd.Series(phrase_counts).div(len(risk_perception_low_no)) * 100
print("\nPercentages:\n", percentages.to_string())

# Optionally, plot the counts as a bar chart
plt.figure(figsize=(10, 6))
```

```python
plt.bar(phrase_counts.keys(), phrase_counts.values(), color='skyblue')
plt.title('Reasons for Perceiving Low/No Risk of HIV Infection')
plt.xlabel('Reason')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
Counts:
 use condoms                       16
i know my partner's hiv status     0
i only have one sex partner        0
use prep                           0
test for HIV                       0
trust myself                       0
use protection                     0
HIV positive                       0
other, specify                     0

Percentages:
 use condoms                       18.823529
i know my partner's hiv status     0.000000
i only have one sex partner        0.000000
use prep                           0.000000
test for HIV                       0.000000
trust myself                       0.000000
use protection                     0.000000
HIV positive                       0.000000
other, specify                     0.000000
```



[157]:

```
cleaned_responses = risk_perception_low_no.dropna().apply(clean_text)
sentiments = cleaned_responses.apply(lambda x: sid.polarity_scores(x))


# Combine sentiments with responses for sorting
sentiments_df = pd.DataFrame(list(sentiments))
sentiments_df['Response'] = cleaned_responses.values

# Sort by compound sentiment score
sorted_responses = sentiments_df.sort_values(by='compound',␣
 ↪ascending=False)['Response']

# Print sorted responses
for i, response in enumerate(sorted_responses):
    print(f"{i+1}. {response}")
```

```
1.
2.
3.
4.
5.
6. i dont use condoms
7.
8. i dont use condoms i have more than one sex partner my partner has other sex
partners
9.
```

```
10.
11. i dont use condoms i have more than one sex partner
12.
13. i have more than one sex partner
14. i dont use condoms
15.
16.
17.
18.
19.
20.
21. i dont use condoms
22.
23. i do not know my partners hiv status i have more than one sex partner
24.
25.
26. other specially
27. i do not know my partners hiv status
28.
29. other specially
30. i dont use condoms i have more than one sex partner
31. i dont use condoms
32. i dont use condoms
33. i dont use condoms
34. i dont use condoms
35. i dont use condoms i have more than one sex partner
36.
37. other specially
38. other specially
39. other specially
40.
41. other specially
42.
43.
44.
45.
46.
47.
48.
49.
50. other specially
51. other specially
52.
53.
54. other specially
55. other specially
56.
57. other specially
```

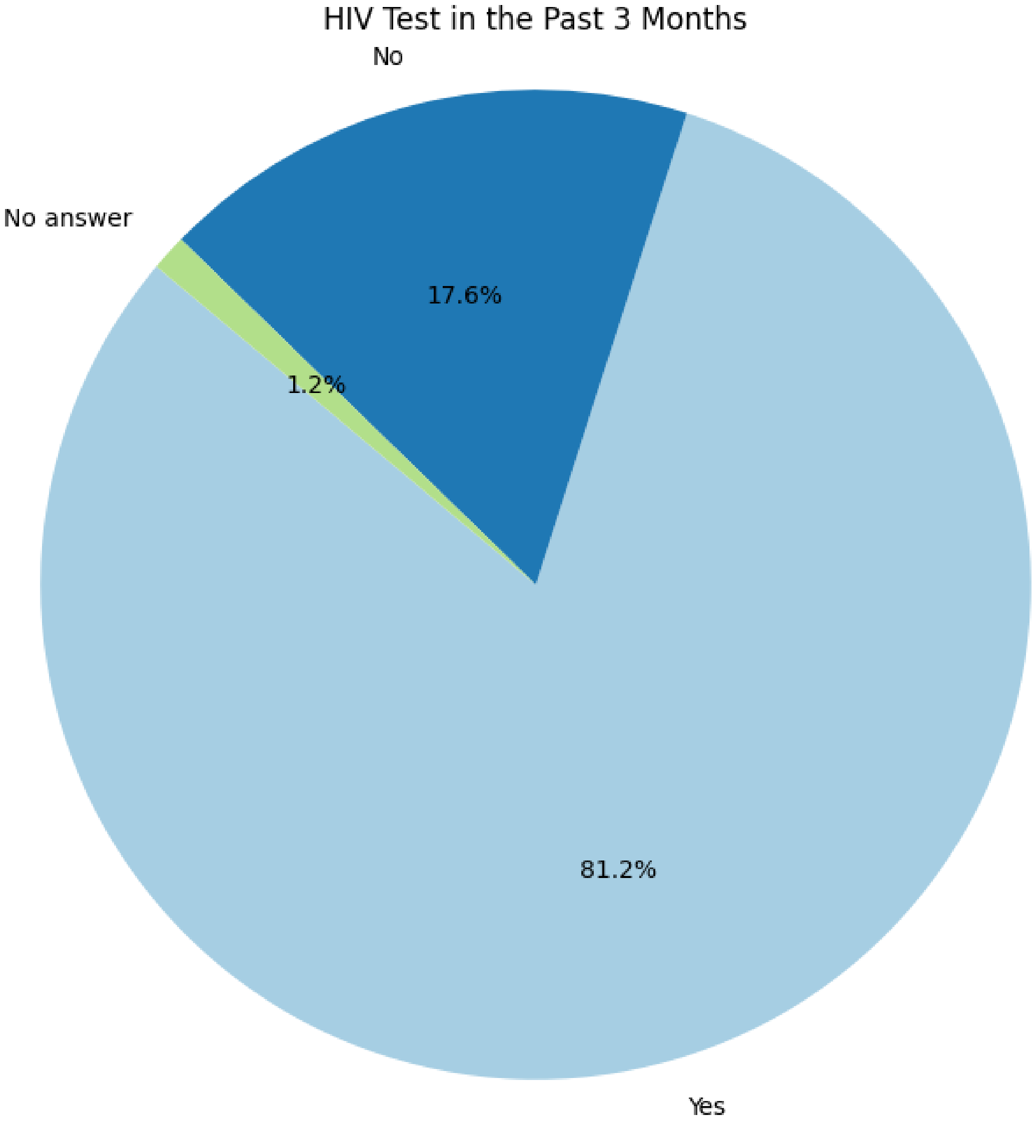
```
58.
59. other specially
60.
61.
62.
63.
64.
65. other specially
66.
67. other specially
68.
69.
70. i dont use condoms
71. i dont use condoms i have more than one sex partner
72.
73. i dont use condoms i have more than one sex partner i do not know my
partners hiv status
74. i dont use condoms
75.
76.
77.
78.
79.
80. other specially
81.
82.
83.
84.
85. i dont use condoms
```

HIV Testing

[158]:

```
hiv_test_responses = nairobi_respondents['24. In the past 3 months, have you␣
  ↪ever gone for a HIV test?']

# Counting the occurrences of each response
hiv_test_counts = hiv_test_responses.value_counts()


# Print the counts
print(hiv_test_counts.to_string(index=True))
print('\nPercentages\n', hiv_test_responses.value_counts(normalize=True).
  ↪to_string(index=True))


# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(hiv_test_counts, labels=hiv_test_counts.index, autopct='%1.1f%%',␣
    ↪startangle=140, colors=plt.cm.Paired.colors)
plt.title('HIV Test in the Past 3 Months')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

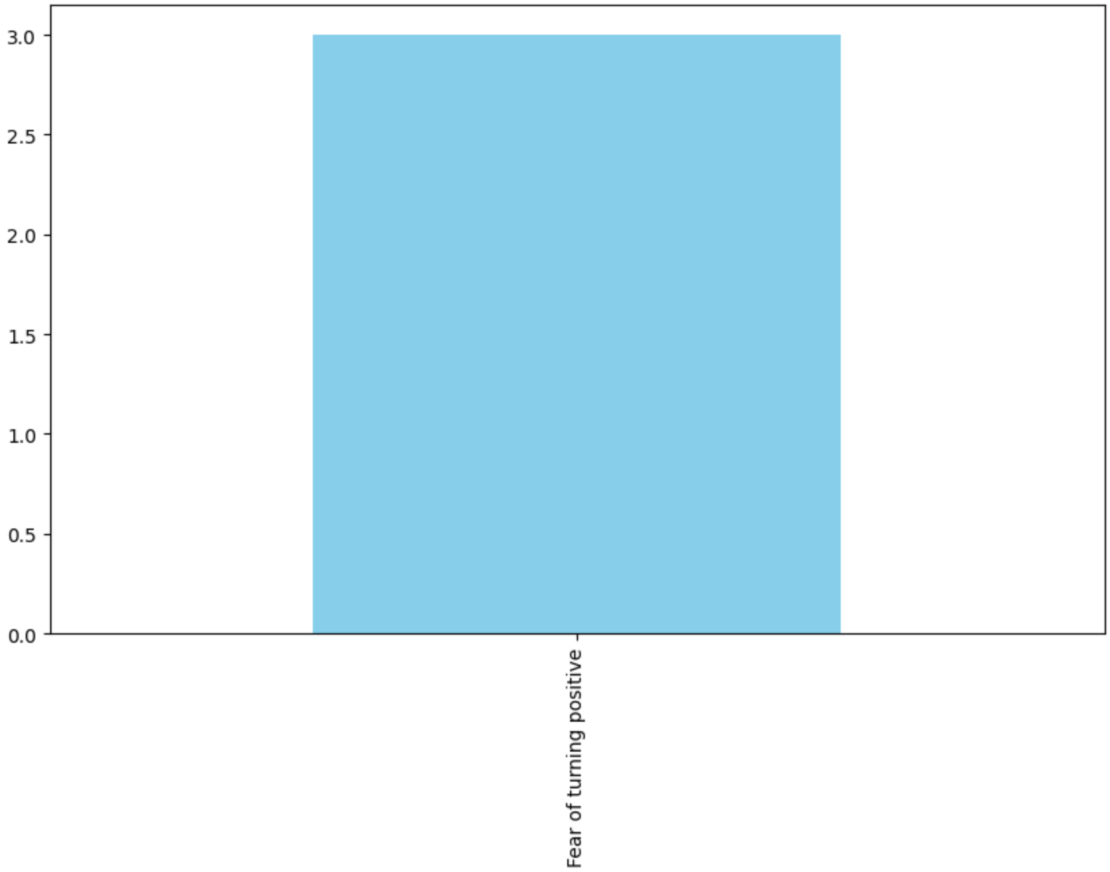
```
Yes          69
No           15
No answer     1

Percentages
 Yes          0.811765
No           0.176471
No answer    0.011765
```



[159]:
```
no_hiv_test_reasons = kisumu_respondents['25. If No, why have you not gone for␣
 ↪a HIV test?']

# Counting the occurrences of each reason
reason_counts = no_hiv_test_reasons.value_counts()

# Print the counts
print(reason_counts.to_string(index=True))

# Plotting the counts as a bar chart
plt.figure(figsize=(10, 6))
reason_counts.plot(kind='bar', color='skyblue')
```

```
Fear of turning positive    3
```

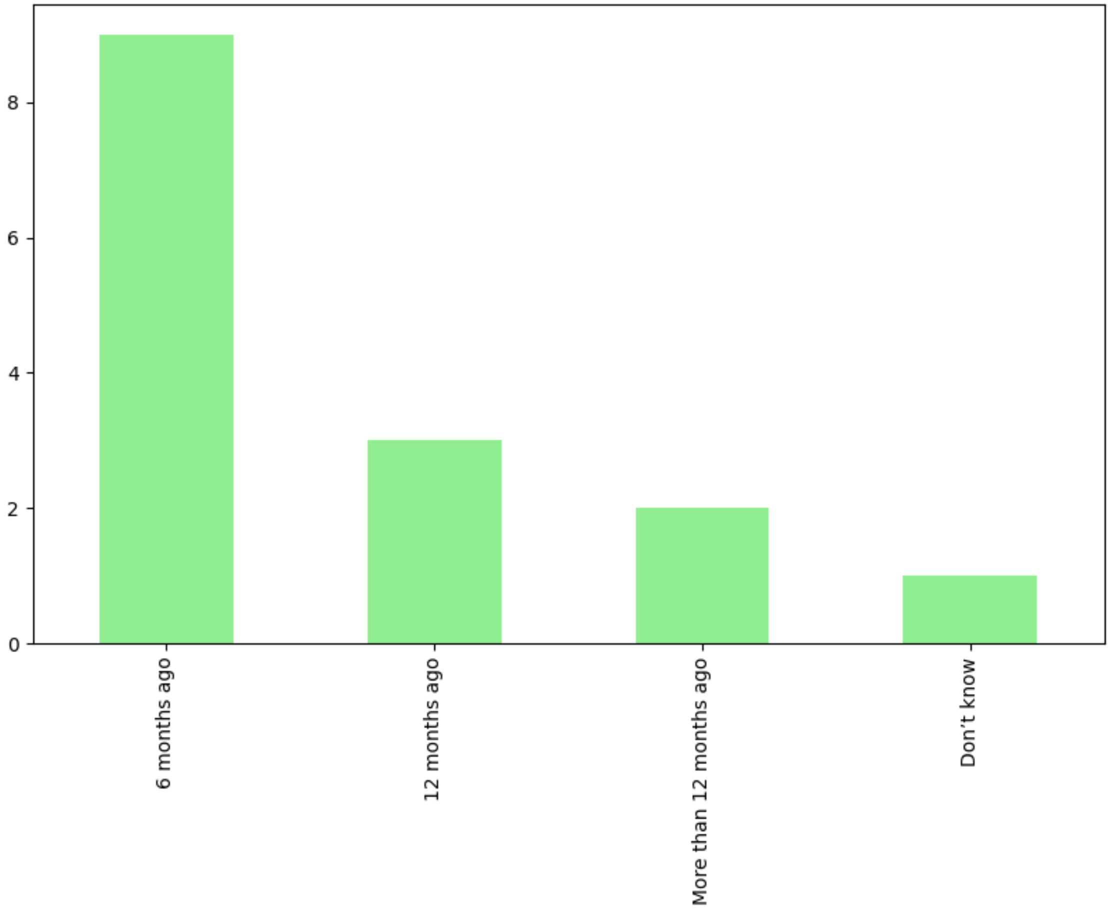
[159]:
```
<AxesSubplot: >
```



[160]:
```
last_testing_times = nairobi_respondents['26. If No, when was the last time you␣
↪went for testing?']

# Counting the occurrences of each response
time_counts = last_testing_times.value_counts()

# Print the counts
print(time_counts.to_string(index=True))

# Plotting the counts as a bar chart
plt.figure(figsize=(10, 6))
time_counts.plot(kind='bar', color='lightgreen')
```

```
6 months ago              9
12 months ago             3
More than 12 months ago   2
Don’t know                1
```

[160]:
```
<AxesSubplot: >
```

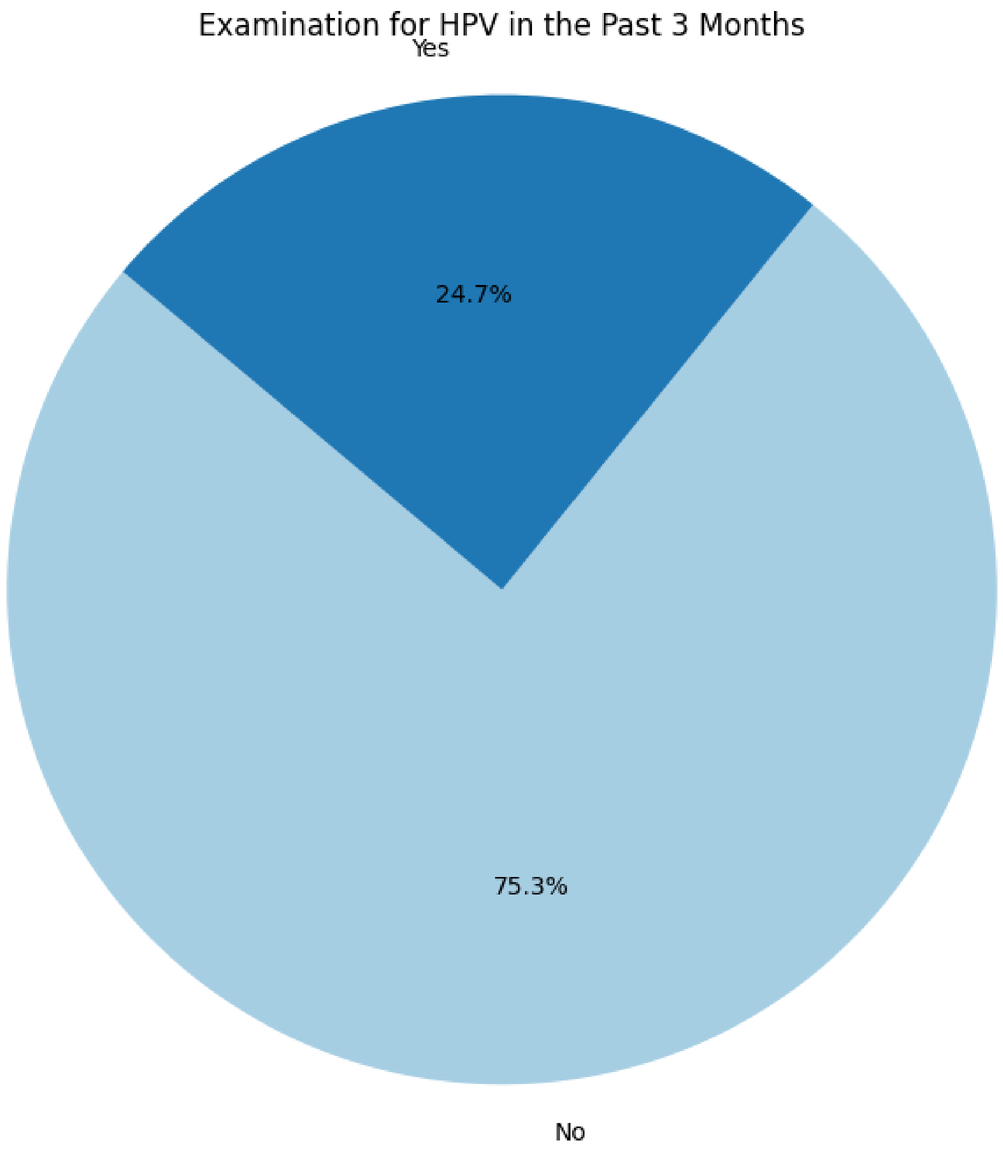


HPV Examination

[161]:

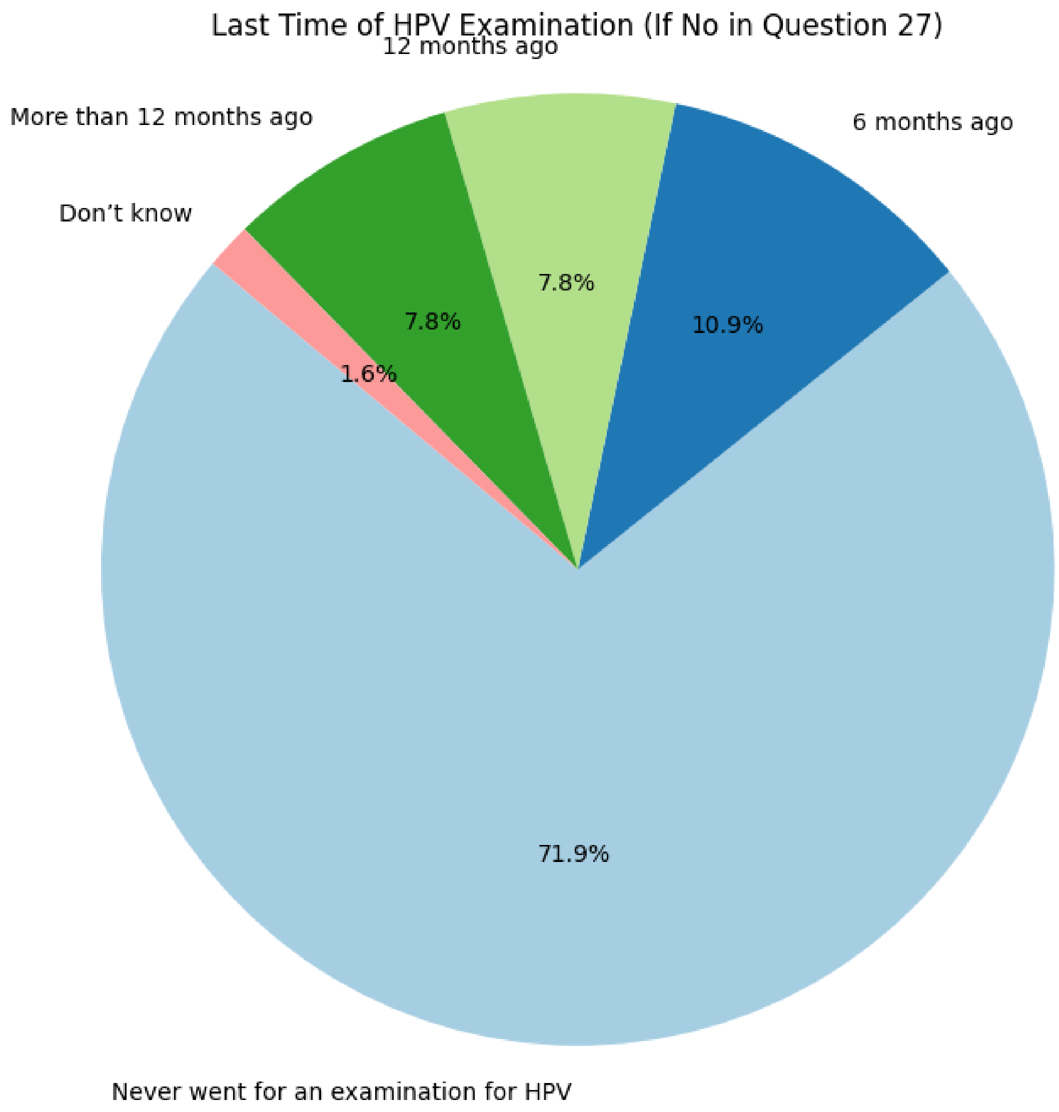
```python
hpv_examination = nairobi_respondents['27. In the past 3 months, have you ever gone for an examination for HPV?']

# Counting the occurrences of each response
hpv_counts = hpv_examination.value_counts()

# Print the counts
print(hpv_counts.to_string(index=True))

# Plotting the counts as a pie chart
plt.figure(figsize=(8, 8))
plt.pie(hpv_counts, labels=hpv_counts.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Examination for HPV in the Past 3 Months')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

```
No     64
Yes    21
```



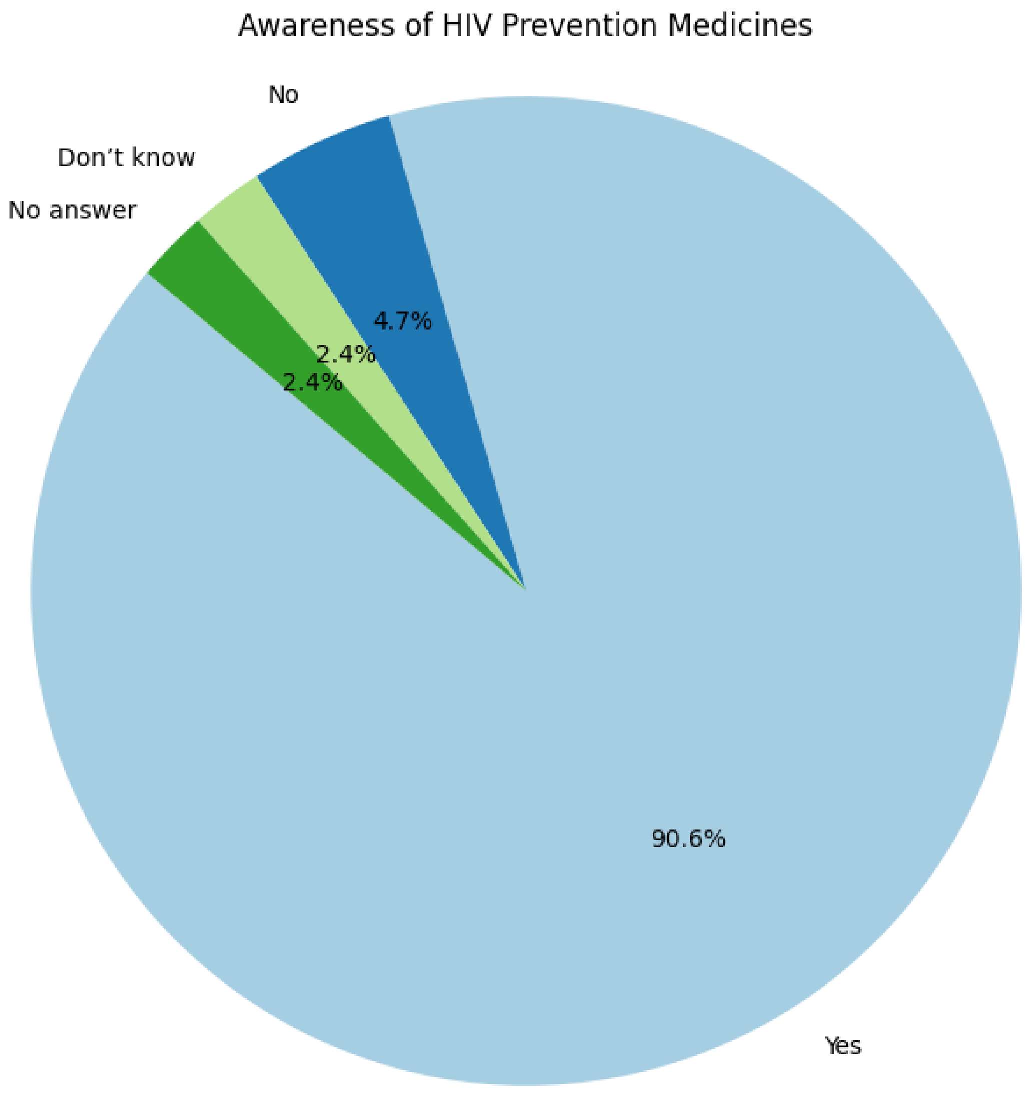
```
[162]: hpv_examination_last_time = nairobi_respondents['28. If No, when was the last␣
        ↪time you went for an examination for HPV?']

       # Counting the occurrences of each response
       hpv_last_time_counts = hpv_examination_last_time.value_counts()

       # Print the counts
       print(hpv_last_time_counts.to_string(index=True))

       # Plotting the counts as a pie chart
       plt.figure(figsize=(8, 8))
       plt.pie(hpv_last_time_counts, labels=hpv_last_time_counts.index, autopct='%1.
        ↪1f%%', startangle=140, colors=plt.cm.Paired.colors)
       plt.title('Last Time of HPV Examination (If No in Question 27)')
       plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
       plt.show()
```

```
Never went for an examination for HPV    46
6 months ago                              7
12 months ago                             5
More than 12 months ago                   5
Don’t know                                1
```



```
[163]: hiv_medicines = nairobi_respondents['29. Are there medicines that a person who␣
        ↪is exposed to HIV can take to prevent HIV infection?']

       # Counting the occurrences of each response
       hiv_medicines_counts = hiv_medicines.value_counts()

       # Print the counts
       print(hiv_medicines_counts.to_string(index=True))

       # Plotting the counts as a pie chart
       plt.figure(figsize=(8, 8))
       plt.pie(hiv_medicines_counts, labels=hiv_medicines_counts.index, autopct='%1.
        ↪1f%%', startangle=140, colors=plt.cm.Paired.colors)
       plt.title('Awareness of HIV Prevention Medicines')
       plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
       plt.show()
```

```
Yes           77
No             4
Don't know     2
No answer      2
```



[164]:

```
medicine_responses = nairobi_respondents['30. If yes, Can you name the medicine?
  ↪']

# Cleaning up the responses
medicine_responses = medicine_responses.str.strip().str.lower()

# Counting the occurrences of each response
medicine_counts = medicine_responses.value_counts()

# Print the counts
print(medicine_counts.to_string(index=True))
```

```
prep              35
pep               11
pep,prep           6
prep and pep       4
prep/pep           3
arvs               3
pep and prep       2
prep & pep         2
preps              2
pep/prep           2
prep, pep          1
prep pep           1
pep prep           1
rt                 1
arvs,prep,pep      1
prep,rep art       1
prep , pep         1
```

[165]:

```python
medicines_for_partner = nairobi_respondents['31. Are there medicines that a person who has a sexually active HIV positive partner can take to prevent infection?']

# Clean up the responses if necessary (e.g., standardize spellings)
medicines_for_partner = medicines_for_partner.str.strip().str.lower()

# Counting the occurrences of each response
medicines_counts = medicines_for_partner.value_counts()

# Print the counts
print(medicines_counts.to_string(index=True))
```

```
yes           66
don’t know     2
no             1
```

[166]:

```python
medicine_names = nairobi_respondents['32. If yes, Can you name the medicine?']

# Clean up the responses if necessary (e.g., standardize spellings)
medicine_names = medicine_names.str.strip().str.lower()

# Counting the occurrences of each response
medicine_counts = medicine_names.value_counts()

# Print the counts
print(medicine_counts.to_string(index=True))
```

```
prep                      38
pep                       10
```

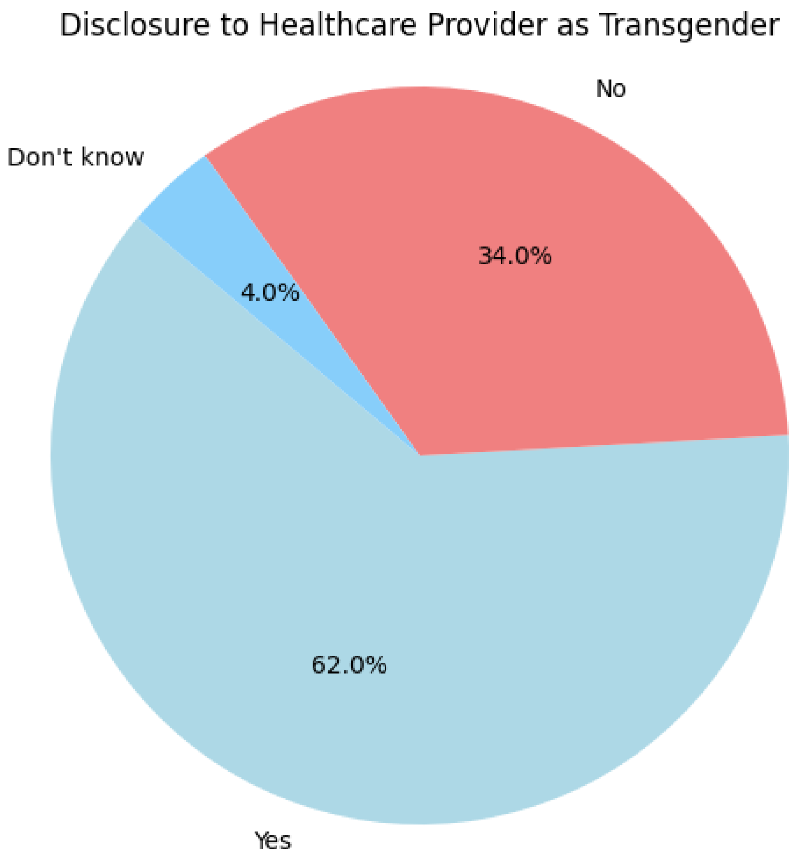
```
prep & pep              4
pep,prep                3
don't know the name     1
prep, condom            1
arv                     1
arvs,prep,peps          1
pep and prep            1
condom , prep and pep   1
pep , prep              1
prep ,pep               1
peps                    1
prep and pep            1
arvs, prep              1
```

1.2.4 Health Care Access and Utilization

[167]:

```
disclosure_responses = nairobi_respondents['33. Since you discovered you are a␣
 ↪transgender have you ever disclosed  to a healthcare provider that you are a␣
 ↪transgender person?']

# Clean up the responses if necessary (standardize capitalization, strip spaces)
disclosure_responses = disclosure_responses.str.strip().str.lower()

# Counting the occurrences of each response
disclosure_counts = disclosure_responses.value_counts()

# Print the counts
print(disclosure_counts.to_string(index=True))


labels = ['Yes', 'No', "Don't know"]
sizes = [62, 34, 4]  # Replace with actual counts from your analysis

# Colors for each section of the pie chart
colors = ['lightblue', 'lightcoral', 'lightskyblue']

# Plotting the pie chart
plt.figure(figsize=(8, 6))
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140)
plt.title('Disclosure to Healthcare Provider as Transgender')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

```
yes           62
no            22
don’t know     1
```

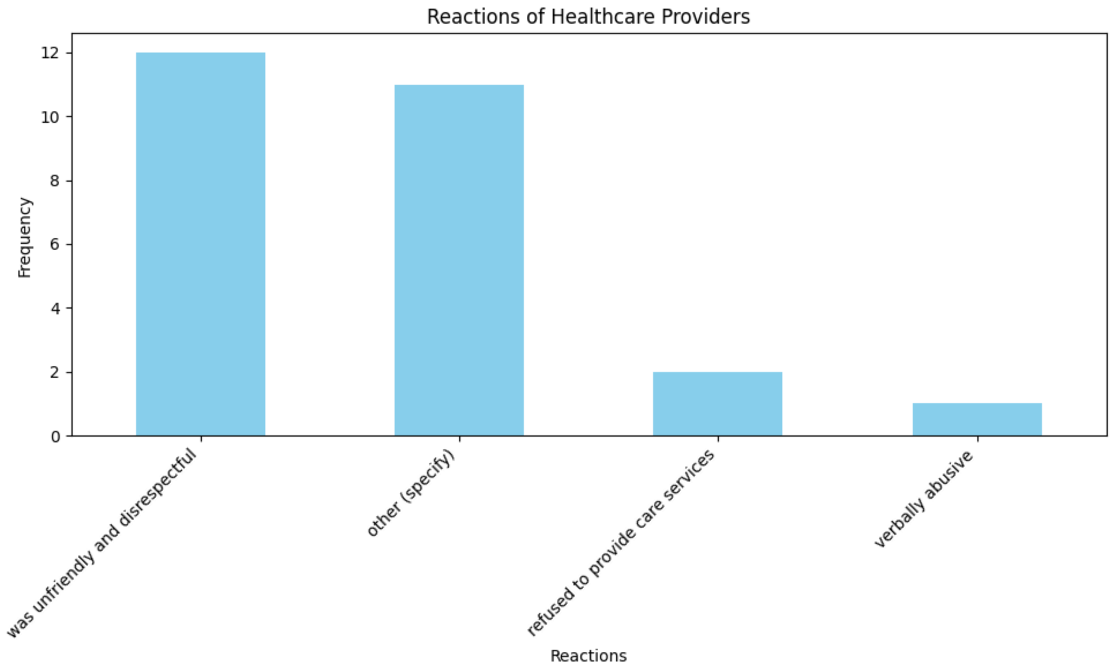


[168]:

```
responses = kisumu_respondents['34. If Yes, what was the reactions of the␣
  ↪health care provider?'].str.strip().str.lower()

# Counting the occurrences of each response
response_counts = responses.value_counts()

# Print the counts
print(response_counts.to_string(index=True))

# Plotting the frequency of each response type
plt.figure(figsize=(10, 6))
response_counts.plot(kind='bar', color='skyblue')
plt.title('Reactions of Healthcare Providers')
plt.xlabel('Reactions')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

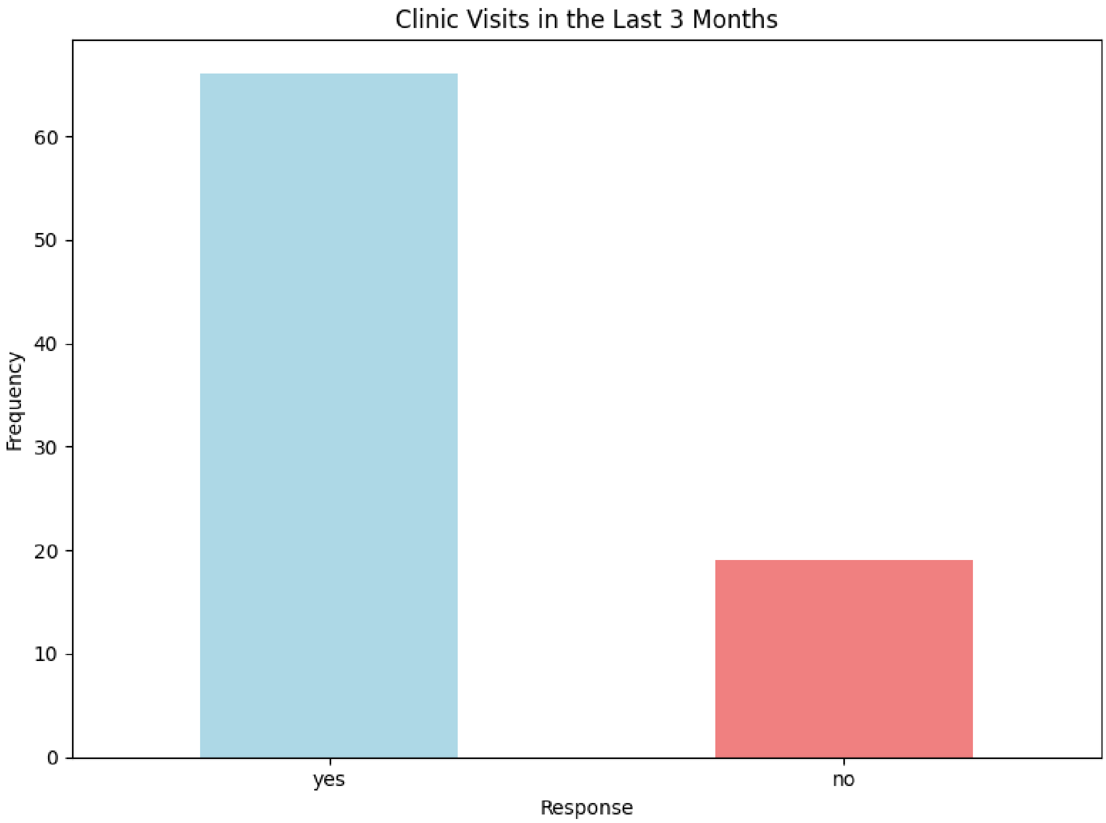
```
was unfriendly and disrespectful    12
```

```
other (specify)                    11
refused to provide care services    2
verbally abusive                    1
```



[169]:
```python
# Clean up the responses (standardize capitalization, strip spaces)
clinic_visits = nairobi_respondents['35. In the last 3 months, have you been to␣
↪a clinic for any healthcare service?'].str.strip().str.lower()

# Counting the occurrences of each response
clinic_visit_counts = clinic_visits.value_counts()

# Print the counts
print(clinic_visit_counts.to_string(index=True))

# Plotting the frequency of each response type
plt.figure(figsize=(8, 6))
clinic_visit_counts.plot(kind='bar', color=['lightblue', 'lightcoral'])
plt.title('Clinic Visits in the Last 3 Months')
plt.xlabel('Response')
plt.ylabel('Frequency')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

```
yes    66
no     19
```



[170]:

```
# # Manually input the 'specify other' data


# # Convert to pandas Series
# other_specify = pd.Series(other_specify_data).str.strip().str.lower()

# # Clean up the main facility responses (standardize capitalization, strip␣
 ↪spaces)
# facility_visits = nairobi_respondents['36. If Yes, What was the type of␣
 ↪facility you visited'].str.strip().str.lower()

# # Replace "other specify" in the main facility responses with actual␣
 ↪specified responses
# facility_visits = facility_visits.mask(facility_visits == 'other specify',␣
 ↪other_specify)

# # Replace any additional 'other specify' entries with their corresponding␣
 ↪values
# facility_visits = facility_visits.replace({
#     'public, private and pharmacy': 'public, private, pharmacy',
```

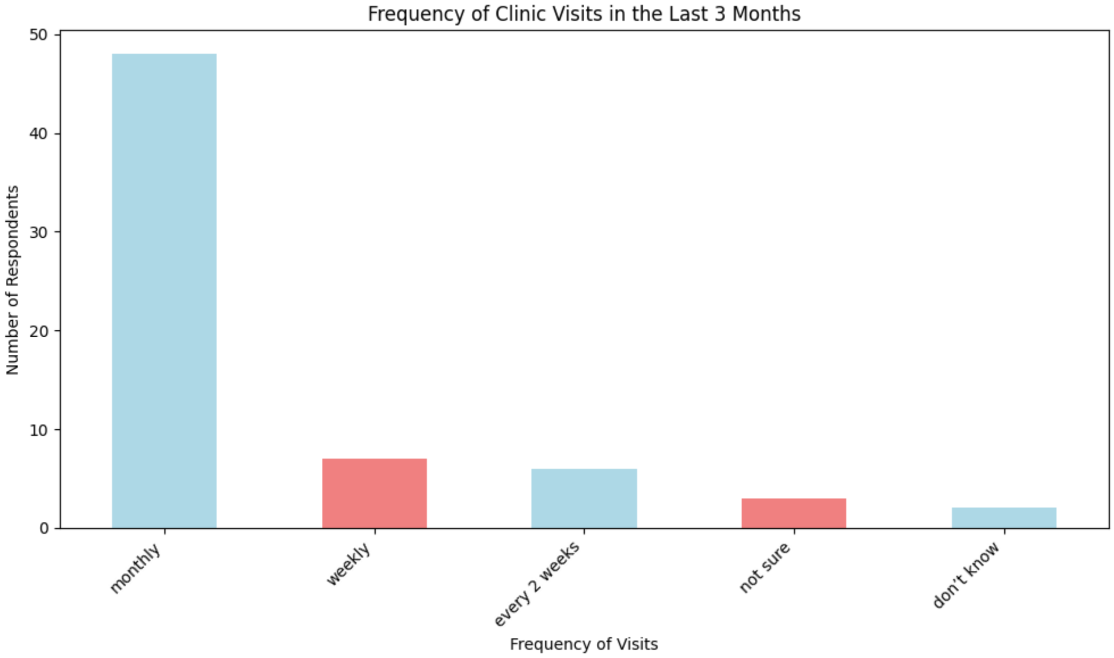
```python
#     'ngo': 'ngo',
#     'ngo hoymas queer facility': 'ngo',
#     'transform ,cbd': 'transform',
#     'transform': 'transform',
#     'machakos dice': 'dice'
# })

# # Counting the occurrences of each response
# facility_visit_counts = facility_visits.value_counts()

# # Print the counts
# print(facility_visit_counts.to_string(index=True))

# # Plotting the frequency of each response type
# plt.figure(figsize=(12, 8))
# facility_visit_counts.plot(kind='bar', color='skyblue')
# plt.title('Type of Facility Visited')
# plt.xlabel('Facility Type')
# plt.ylabel('Frequency')
# plt.xticks(rotation=45, ha='right')
# plt.tight_layout()
# plt.show()
```

In Facility type we had mentions like: Hoymas Care, Transform, Transcare and NGOs. The following array enumerates the mentioned facility types: ‘public, private and pharmacy’: ‘public, private, pharmacy’, ‘ngo’: ‘ngo’, ‘ngo hoymas queer facility’: ‘ngo’, ‘transform ,cbd’: ‘transform’, ‘transform’: ‘transform’, ‘machakos dice’: ‘dice’]

```python
[171]: clinic_visits = nairobi_respondents['37. If Yes, how frequently did you visit␣
        ↪the facility in the last 3 months?'].str.strip().str.lower()

       # Counting the occurrences of each response
       clinic_visit_counts = clinic_visits.value_counts()

       # Print the counts
       print(clinic_visit_counts.to_string(index=True))

       # Plotting the frequency of each response type
       plt.figure(figsize=(10, 6))
       clinic_visit_counts.plot(kind='bar', color=['lightblue', 'lightcoral'])
       plt.title('Frequency of Clinic Visits in the Last 3 Months')
       plt.xlabel('Frequency of Visits')
       plt.ylabel('Number of Respondents')
       plt.xticks(rotation=45, ha='right')
       plt.tight_layout()
       plt.show()
```

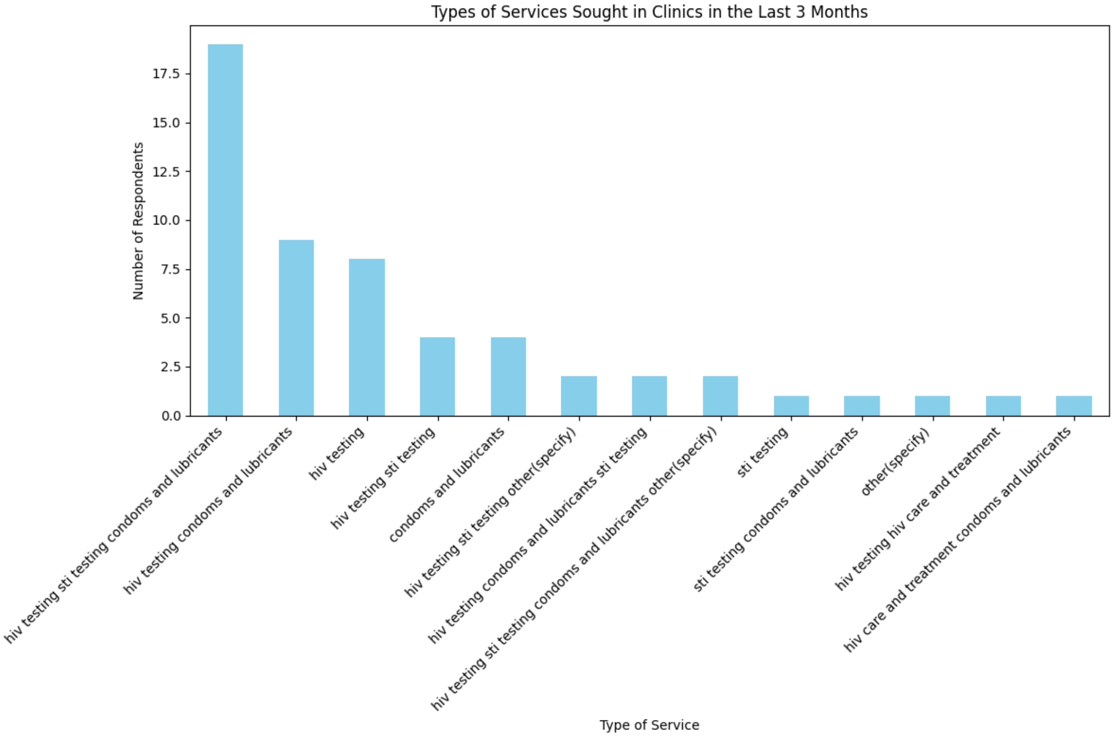
```
monthly          48
```

```
weekly            7
every 2 weeks     6
not sure          3
don't know        2
```



[172]:
```
service_types = kisumu_respondents['38. If Yes, what type of service did you␣
↪seek?'].str.strip().str.lower()

# Counting the occurrences of each response
service_type_counts = service_types.value_counts()

# Print the counts
print(service_type_counts.to_string(index=True))

# Plotting the frequency of each service type
plt.figure(figsize=(12, 8))
service_type_counts.plot(kind='bar', color='skyblue')
plt.title('Types of Services Sought in Clinics in the Last 3 Months')
plt.xlabel('Type of Service')
plt.ylabel('Number of Respondents')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

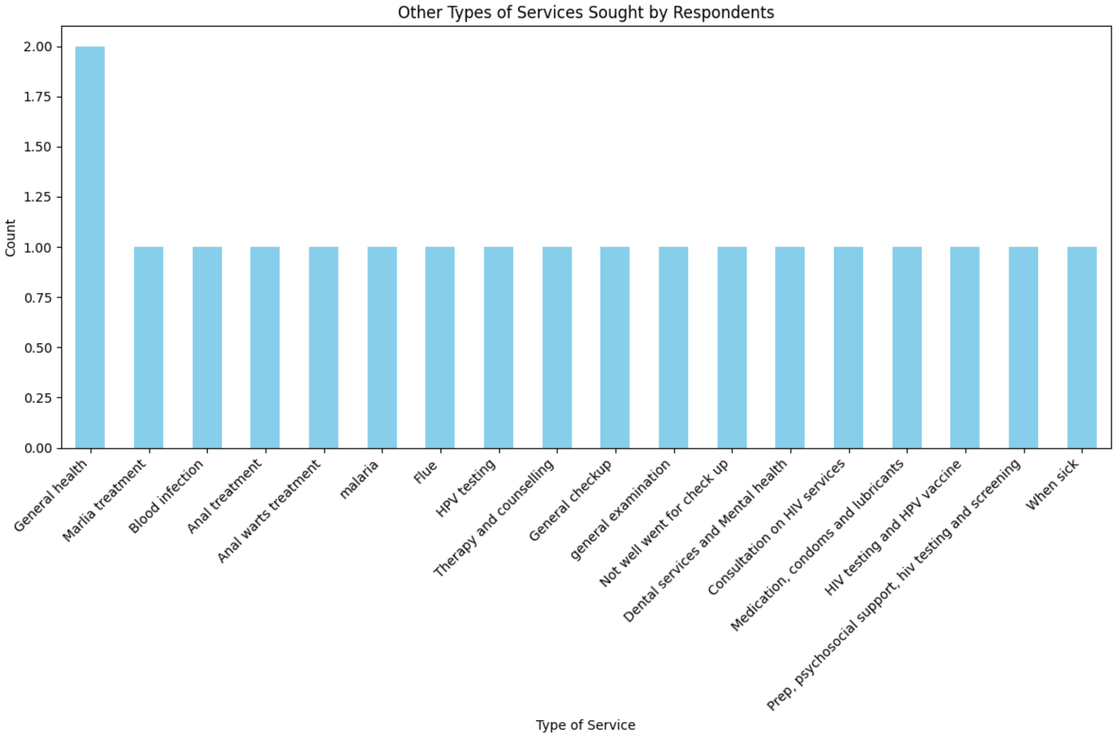
```
hiv testing sti testing condoms and lubricants                   19
hiv testing condoms and lubricants                                9
hiv testing                                                     8
hiv testing sti testing                                         4
condoms and lubricants                                          4
hiv testing sti testing other(specify)                          2
hiv testing condoms and lubricants sti testing                  2
hiv testing sti testing condoms and lubricants other(specify)    2
sti testing                                                     1
sti testing condoms and lubricants                              1
other(specify)                                                  1
hiv testing hiv care and treatment                              1
hiv care and treatment condoms and lubricants                   1
```



[173]:
```
# Assuming the column name is '38b. Specify other type of service' in␣
 ↪kisumu_respondents

# Clean up the responses if necessary (strip spaces)
other_services = nairobi_respondents['38b. Specify other type of service'].str.
 ↪strip()

# Count the occurrences of each service
service_counts = other_services.value_counts()

# Print the counts
```

```python
print(service_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(12, 8))
service_counts.plot(kind='bar', color='skyblue')
plt.title('Other Types of Services Sought by Respondents')
plt.xlabel('Type of Service')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
General health                                           2
Marlia treatment                                         1
Blood infection                                          1
Anal treatment                                           1
Anal warts treatment                                     1
malaria                                                  1
Flue                                                     1
HPV testing                                              1
Therapy and counselling                                  1
General checkup                                          1
general examination                                      1
Not well went for check up                               1
Dental services and Mental health                        1
Consultation on HIV services                             1
Medication, condoms and lubricants                       1
HIV testing and HPV vaccine                              1
Prep, psychosocial support, hiv testing and screening    1
When sick                                                1
```

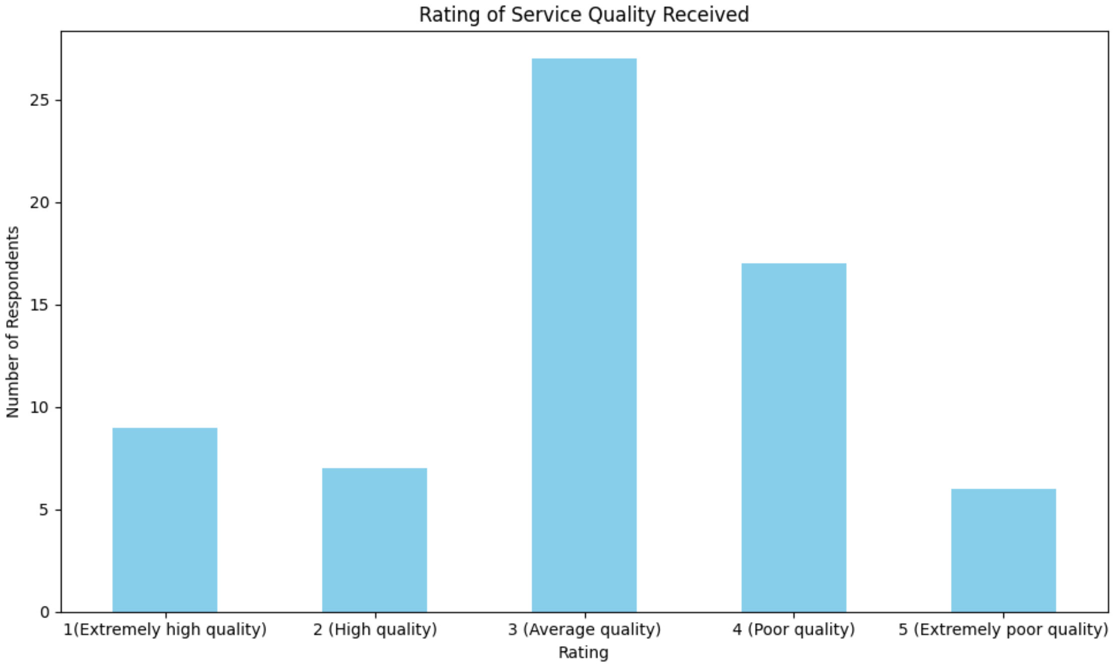


```
[174]: # data = {
       #     'Category': ['HIV testing STI testing Condoms and lubricants', 'HIV␣
        ↪testing Condoms and lubricants',
       #                  'HIV testing', 'HIV testing STI testing', 'Condoms and␣
        ↪lubricants',
       #                  'HIV testing STI testing Other(specify)', 'HIV testing␣
        ↪condoms and lubricants STI testing',
       #                  'HIV testing STI testing Condoms and lubricants␣
        ↪Other(specify)', 'STI testing',
       #                  'STI testing Condoms and lubricants', 'Other(specify)', 'HIV␣
        ↪testing HIV care and treatment',
       #                  'HIV care and treatment condoms and lubricants'],
       #     'Count': [19, 9, 8, 4, 4, 2, 2, 2, 1, 1, 1, 1, 1]
       # }

       # # Convert data to a pandas DataFrame
       # import pandas as pd
       # df = pd.DataFrame(data)

       # # Plotting the bar plot
       # plt.figure(figsize=(10, 6))
       # plt.bar(df['Category'], df['Count'], color='skyblue')
       # plt.title('Counts of Different Health Service Types')
       # plt.xlabel('Service Types')
       # plt.ylabel('Count')
       # plt.xticks(rotation=90)
       # plt.tight_layout()
       # plt.show()
```

[175]:

```
ratings = nairobi_respondents['39. If Yes, How would you rate the quality of␣
 ↪service you received?']

# Counting the occurrences of each rating
rating_counts = ratings.value_counts().sort_index()

# Print the counts
print(rating_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 6))
rating_counts.plot(kind='bar', color='skyblue')
plt.title('Rating of Service Quality Received')
plt.xlabel('Rating')
plt.ylabel('Number of Respondents')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

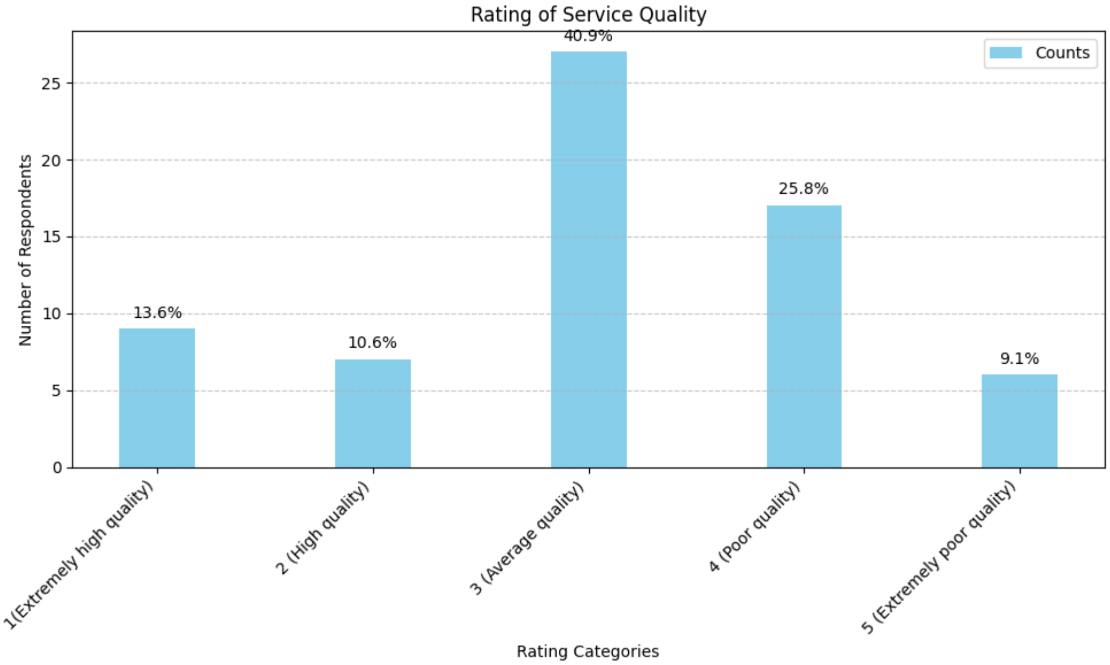
```
1(Extremely high quality)      9
2 (High quality)               7
3 (Average quality)           27
4 (Poor quality)              17
5 (Extremely poor quality)     6
```



[176]:

```
# Count the occurrences of each rating
rating_counts = ratings.value_counts().sort_index()

# Define the categories and their counts
categories = ['1(Extremely high quality)', '2 (High quality)', '3 (Average␣
 ↪quality)', '4 (Poor quality)', '5 (Extremely poor quality)']
counts = [rating_counts.get(cat, 0) for cat in categories]

# Calculate the total number of respondents
total_respondents = sum(counts)

# Calculate percentages for each category
percentages = [(count / total_respondents) * 100 for count in counts]

# Create a grouped bar chart
plt.figure(figsize=(10, 6))

# Bar positions and width
bar_width = 0.35
index = np.arange(len(categories))

# Plotting the bars
bars = plt.bar(index, counts, bar_width, label='Counts', color='skyblue')

# Adding labels, title, and grid
```

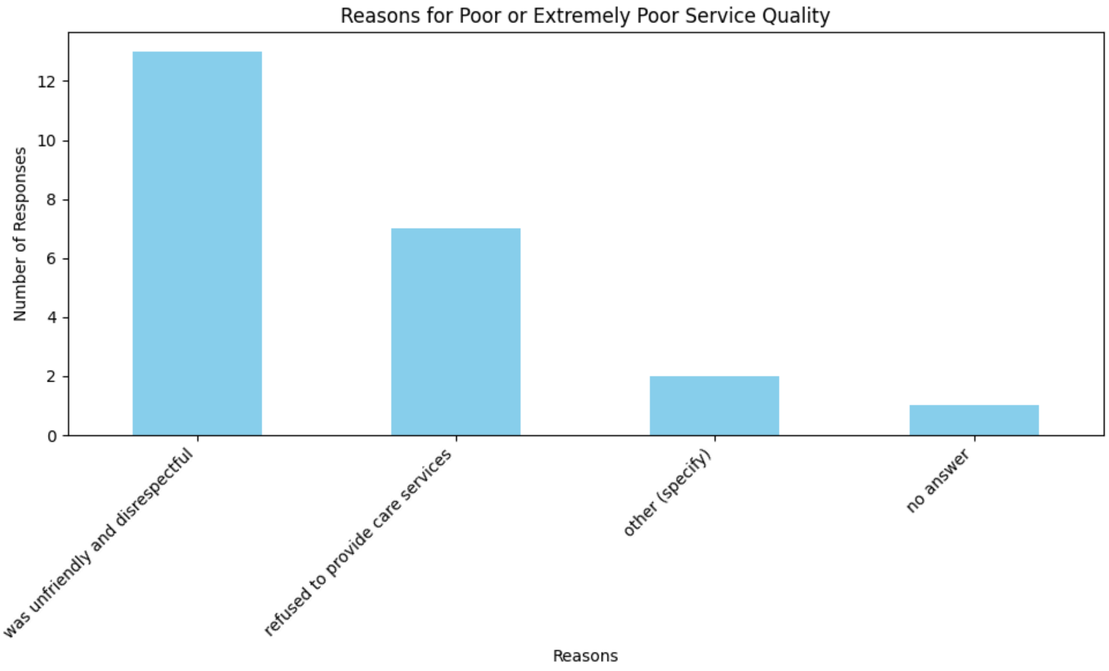
```
plt.xlabel('Rating Categories')
plt.ylabel('Number of Respondents')
plt.title('Rating of Service Quality')
plt.xticks(index, categories, rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Adding percentages above each bar
for i, bar in enumerate(bars):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 0.5,
             f'{percentages[i]:.1f}%', ha='center', va='bottom', fontsize=10)

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



[177]:
```
responses = nairobi_respondents['40. If poor or extremely poor, why?']

# Clean up responses (strip spaces and convert to lowercase)
clean_responses = responses.str.strip().str.lower()

# Count occurrences of each reason
reason_counts = clean_responses.value_counts()

# Print the counts
print(reason_counts.to_string())
```

```
# Plotting the reasons for poor service quality
plt.figure(figsize=(10, 6))
reason_counts.plot(kind='bar', color='skyblue')
plt.title('Reasons for Poor or Extremely Poor Service Quality')
plt.xlabel('Reasons')
plt.ylabel('Number of Responses')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

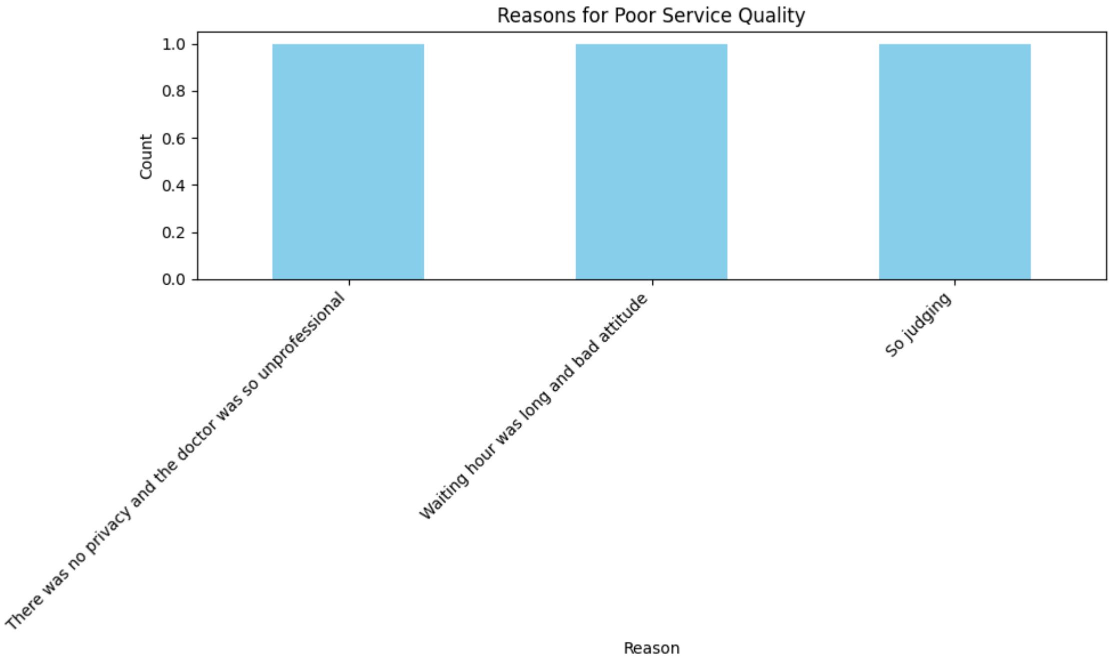
```
was unfriendly and disrespectful    13
refused to provide care services     7
other (specify)                      2
no answer                            1
```



```
[178]: other_reasons = kisumu_respondents['Others- Poor quality of service'].str.
        ↪strip()

       # Count the occurrences of each reason
       reason_counts = other_reasons.value_counts()

       # Print the counts
       print(reason_counts.to_string(index=True))
```

```
# Plotting the bar chart
plt.figure(figsize=(10, 6))
reason_counts.plot(kind='bar', color='skyblue')
plt.title('Reasons for Poor Service Quality')
plt.xlabel('Reason')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

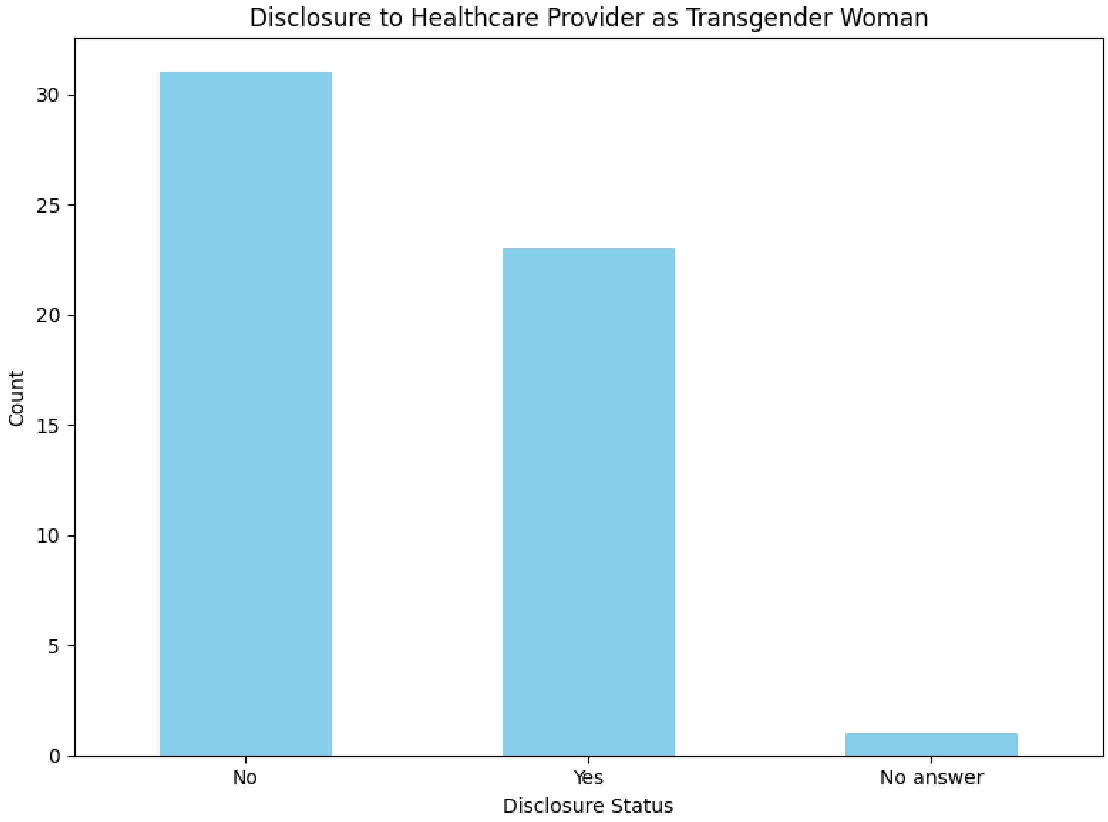
```
There was no privacy and the doctor was so unprofessional    1
Waiting hour was long and bad attitude                       1
So judging                                                   1
```



[179]:
```
disclosure_responses = kisumu_respondents['41.If Yes, did you disclose to the␣
↪health care provider that you are a transgender woman?'].str.strip()

# Count the occurrences of each response
disclosure_counts = disclosure_responses.value_counts()

# Print the counts
print(disclosure_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(8, 6))
```

```
disclosure_counts.plot(kind='bar', color='skyblue')
plt.title('Disclosure to Healthcare Provider as Transgender Woman')
plt.xlabel('Disclosure Status')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

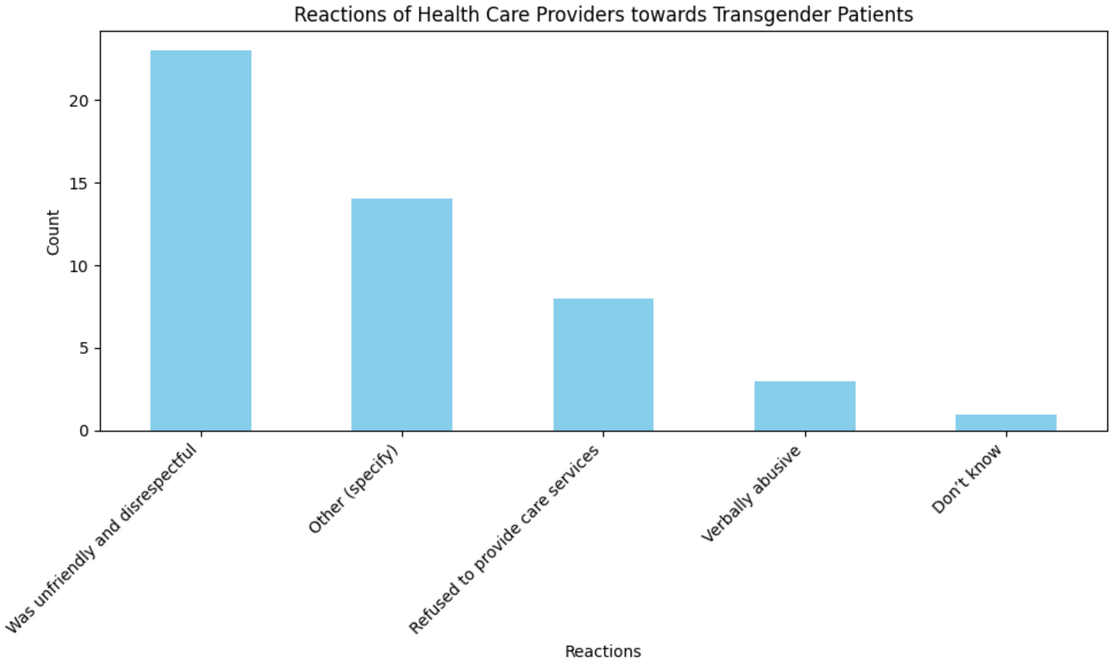
```
No           31
Yes          23
No answer     1
```



```
[180]: reactions = nairobi_respondents['42. If Yes, what was the reactions of the␣
       ↪health care provider?'].str.strip()

       # Count the occurrences of each response
       reaction_counts = reactions.value_counts()

       # Print the counts
       print(reaction_counts.to_string(index=True))
```

```
# Plotting the bar chart
plt.figure(figsize=(10, 6))
reaction_counts.plot(kind='bar', color='skyblue')
plt.title('Reactions of Health Care Providers towards Transgender Patients')
plt.xlabel('Reactions')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

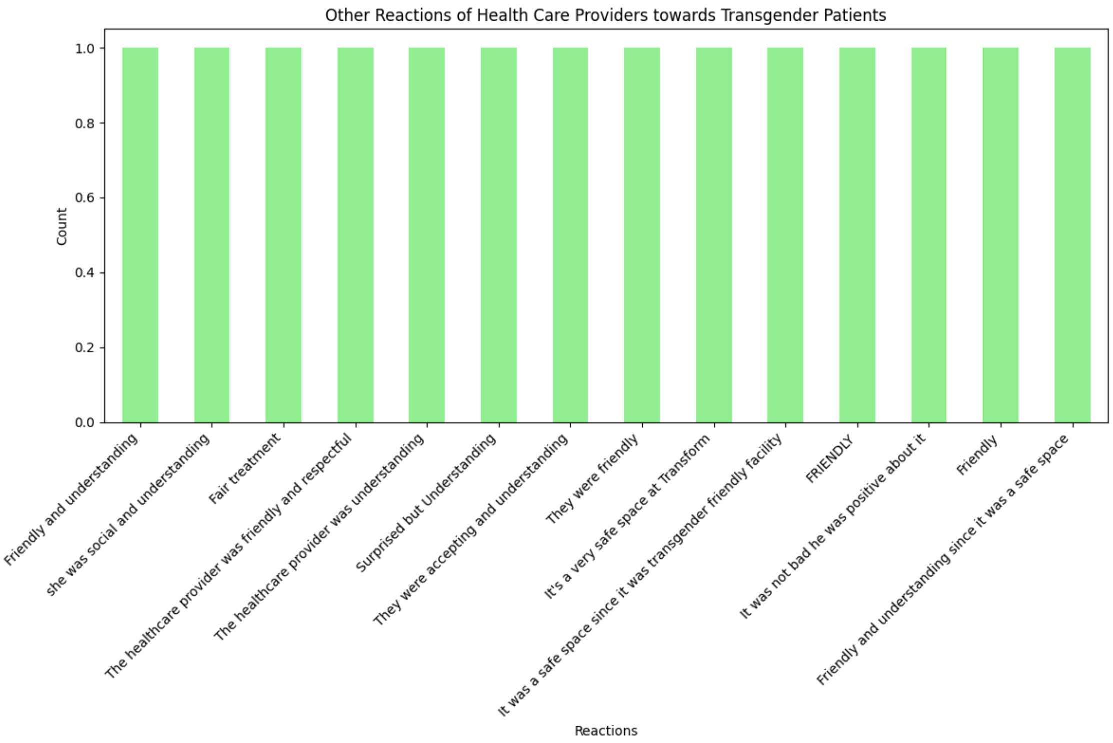
```
Was unfriendly and disrespectful    23
Other (specify)                     14
Refused to provide care services     8
Verbally abusive                     3
Don’t know                           1
```



```
[181]: import pandas as pd
import matplotlib.pyplot as plt

# Assuming the column name is 'Specify other reaction' in kisumu_respondents

# Clean up the responses if necessary (strip spaces)
other_reactions = nairobi_respondents['Specify other reaction'].str.strip()

# Count the occurrences of each response
```

```
other_reaction_counts = other_reactions.value_counts()

# Print the counts
print(other_reaction_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(12, 8))
other_reaction_counts.plot(kind='bar', color='lightgreen')
plt.title('Other Reactions of Health Care Providers towards Transgender␣
  ↪Patients')
plt.xlabel('Reactions')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

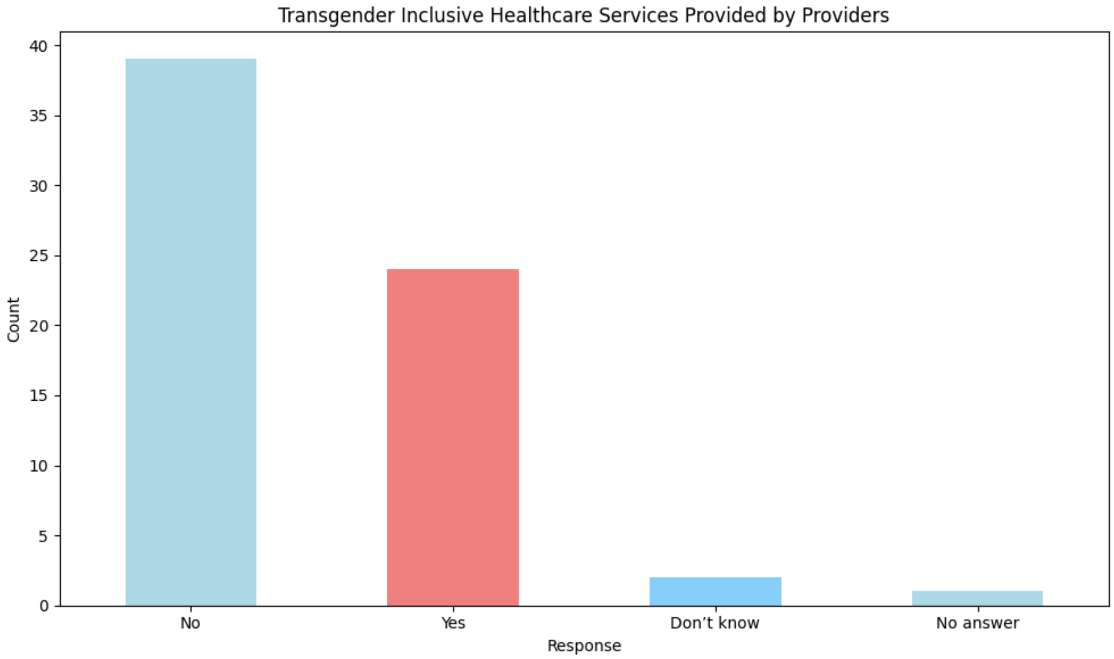
```
Friendly and understanding                                        1
she was social and understanding                                  1
Fair treatment                                                    1
The healthcare provider was friendly and respectful               1
The healthcare provider was understanding                         1
Surprised but Understanding                                       1
They were accepting and understanding                             1
They were friendly                                                1
It's a very safe space at Transform                               1
It was a safe space since it was transgender friendly facility    1
FRIENDLY                                                          1
It was not bad he was positive about it                           1
Friendly                                                          1
Friendly and understanding since it was a safe space              1
```



[182]:

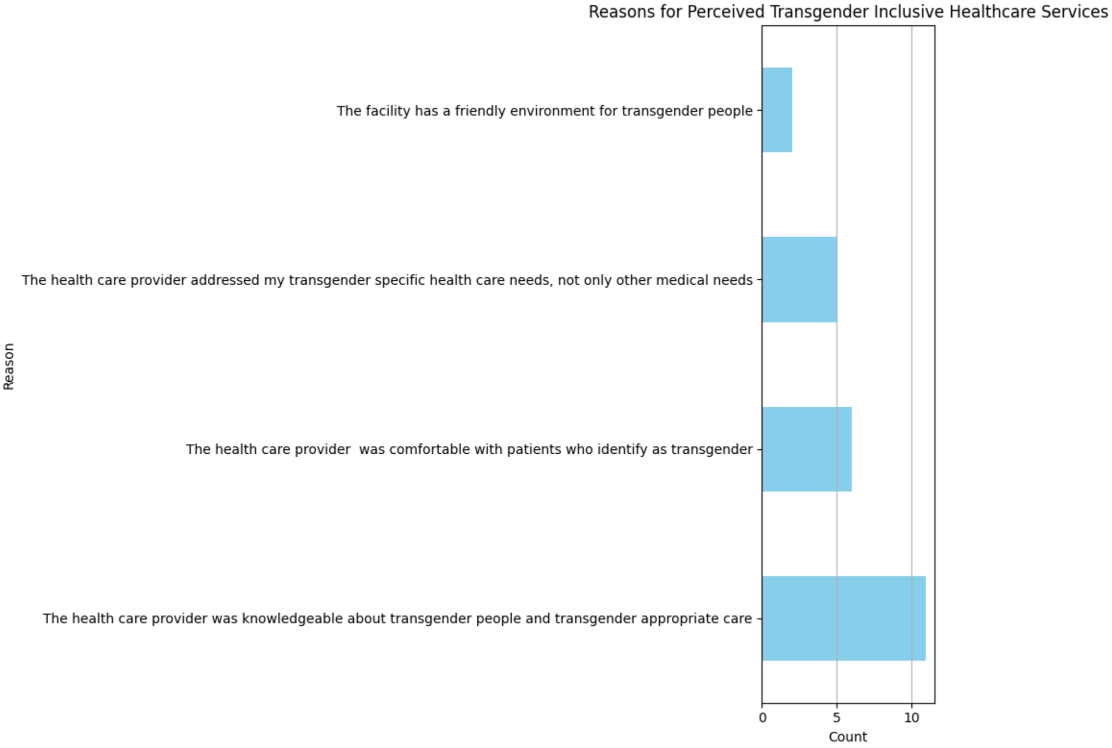
```
import pandas as pd
import matplotlib.pyplot as plt

# Assuming the column name is '43. Did the health care provider provide you␣
↪with transgender inclusive healthcare services?'

# Clean up the responses if necessary (strip spaces)
transgender_healthcare = nairobi_respondents['43. Did the health care provider␣
↪provide you with  transgender inclusive healthcare services?'].str.strip()

# Count the occurrences of each response
transgender_healthcare_counts = transgender_healthcare.value_counts()

# Print the counts
print(transgender_healthcare_counts.to_string(index=True))

# Plotting the grouped bar chart
plt.figure(figsize=(10, 6))
transgender_healthcare_counts.plot(kind='bar', color=['lightblue',␣
↪'lightcoral', 'lightskyblue'])
plt.title('Transgender Inclusive Healthcare Services Provided by Providers')
plt.xlabel('Response')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

```
No            39
Yes           24
Don't know     2
No answer      1
```



[183]:
```
transgender_inclusive_reasons = nairobi_respondents['44. If Yes, why do you␣
↪think that the service(s) provided were transgender inclusive?'].str.strip()

# Count the occurrences of each response
transgender_inclusive_reasons_counts = transgender_inclusive_reasons.
↪value_counts()

# Print the counts
print(transgender_inclusive_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
transgender_inclusive_reasons_counts.plot(kind='barh', color='skyblue')
plt.title('Reasons for Perceived Transgender Inclusive Healthcare Services')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

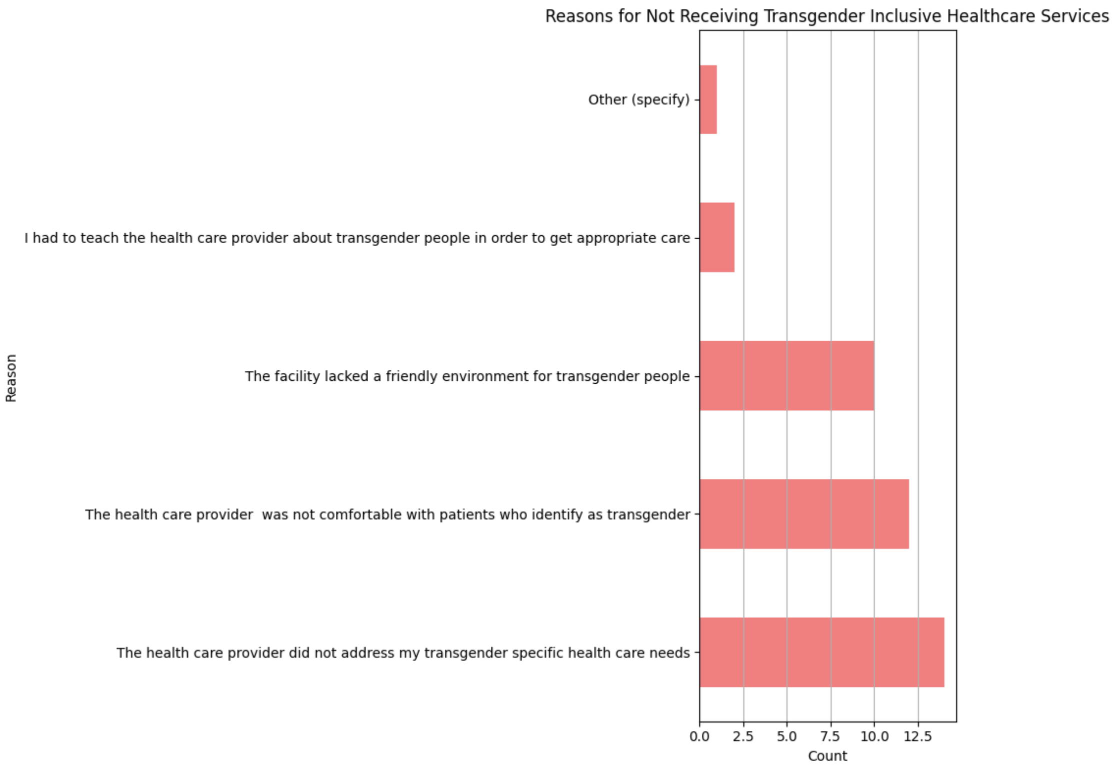
```
The health care provider was knowledgeable about transgender people and
transgender appropriate care          11
The health care provider  was comfortable with patients who identify as
transgender                            6
The health care provider addressed my transgender specific health care needs,
not only other medical needs     5
The facility has a friendly environment for transgender people
2
```



```
[184]: import pandas as pd
       import matplotlib.pyplot as plt

       # Assuming the column name is '45. If No, why do you think the health care␣
        ↪provider did not provide you with transgender inclusive health care?'

       # Clean up the responses if necessary (strip spaces)
       not_transgender_inclusive_reasons = nairobi_respondents['45. If No, why do you␣
        ↪think the health care provider did not provide you with transgender␣
        ↪inclusive health care?'].str.strip()
```

```python
# Count the occurrences of each response
not_transgender_inclusive_reasons_counts = not_transgender_inclusive_reasons.
 ↪value_counts()

# Print the counts
print(not_transgender_inclusive_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
not_transgender_inclusive_reasons_counts.plot(kind='barh', color='lightcoral')
plt.title('Reasons for Not Receiving Transgender Inclusive Healthcare Services')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

```
The health care provider did not address my transgender specific health care
needs                   14
The health care provider  was not comfortable with patients who identify as
transgender              12
The facility lacked a friendly environment for transgender people
10
I had to teach the health care provider about transgender people in order to get
appropriate care     2
Other (specify)
1
```

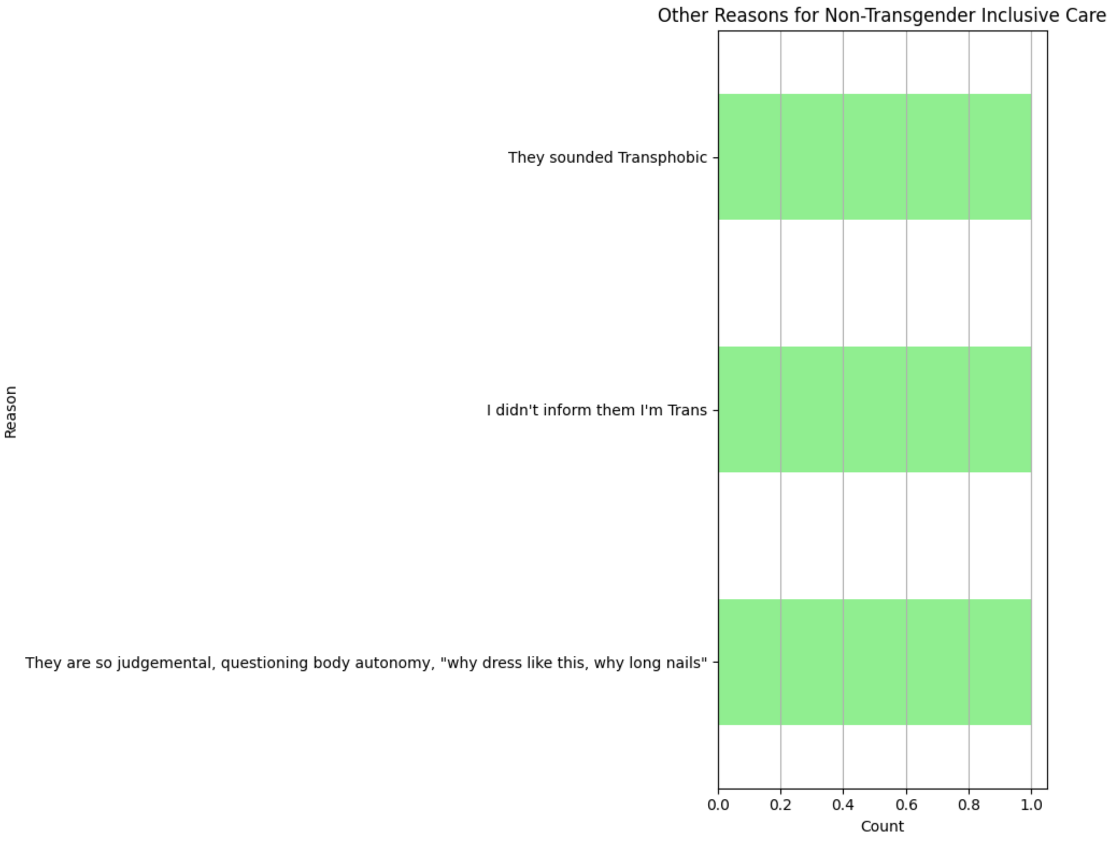


[191]:

```
other_trans_inclusive_reasons = kisumu_respondents['Specify other - NP␣
 ↪TransInclusive'].str.strip()

# Count the occurrences of each response
other_trans_inclusive_reasons_counts = other_trans_inclusive_reasons.
 ↪value_counts()

# Print the counts
print(other_trans_inclusive_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
other_trans_inclusive_reasons_counts.plot(kind='barh', color='lightgreen')
plt.title('Other Reasons for Non-Transgender Inclusive Care')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

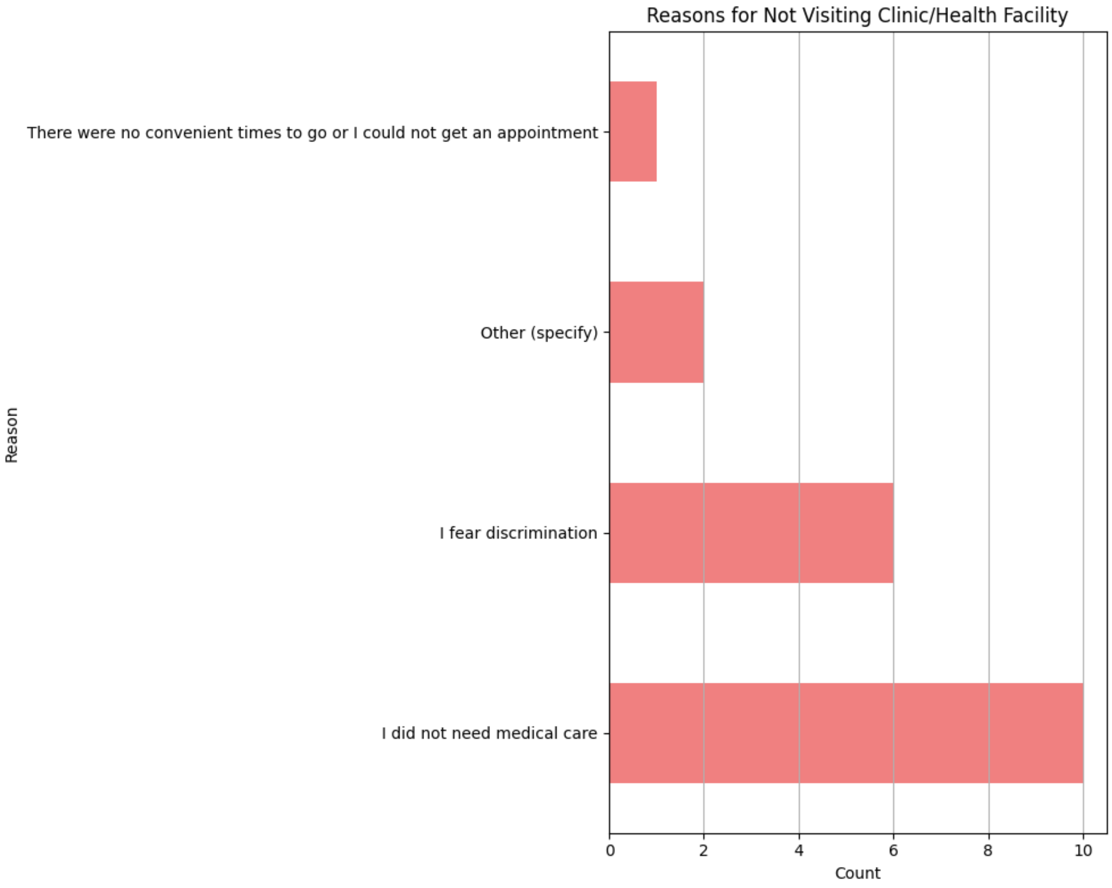
```
They are so judgemental, questioning body autonomy, "why dress like this, why
long nails"    1
I didn't inform them I'm Trans
```

```
1
They sounded Transphobic
1
```



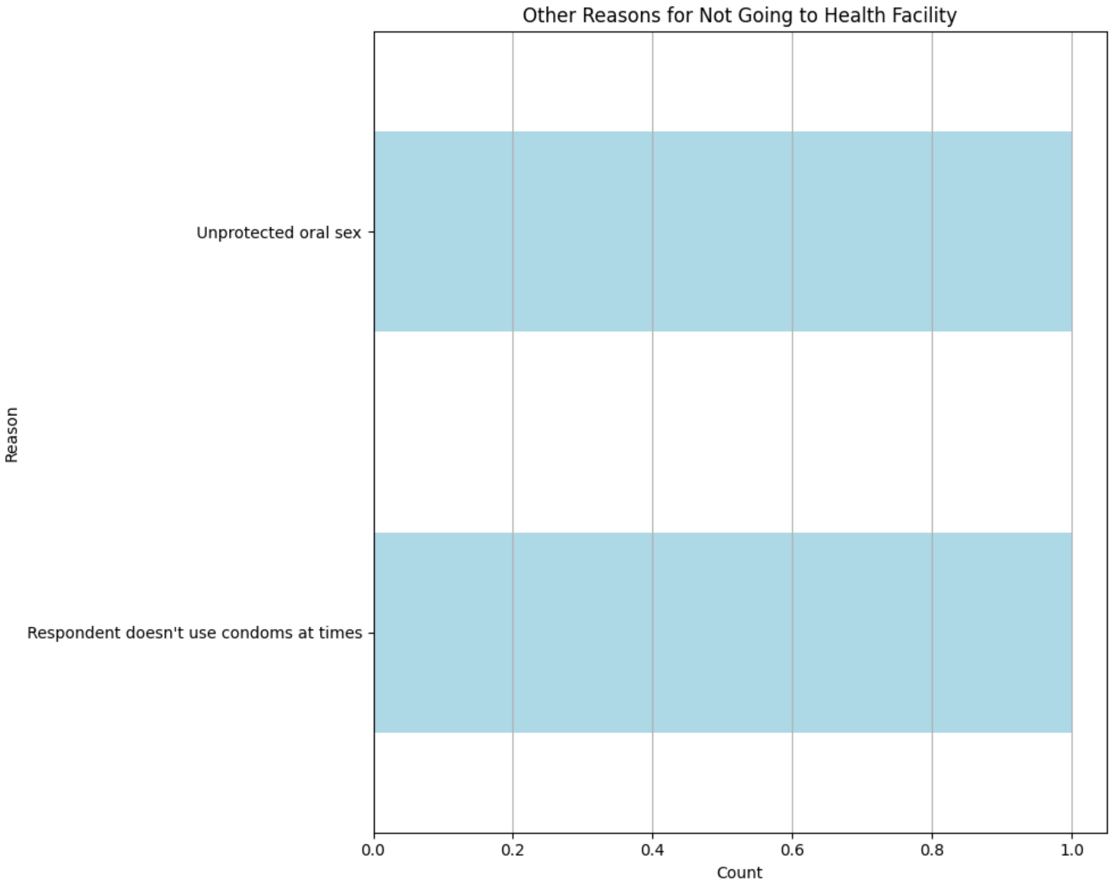
[192]:
```
no_visit_reasons = nairobi_respondents['46. If No , why have you not gone to a␣
 ↪clinic/ health facility?'].str.strip()

# Count the occurrences of each response
no_visit_reasons_counts = no_visit_reasons.value_counts()

# Print the counts
print(no_visit_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
no_visit_reasons_counts.plot(kind='barh', color='lightcoral')
plt.title('Reasons for Not Visiting Clinic/Health Facility')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

```
I did not need medical care                                          10
I fear discrimination                                                 6
Other (specify)                                                       2
There were no convenient times to go or I could not get an appointment    1
```



[198]:
```
other_reasons = nairobi_respondents['Specify other reason'].str.strip()

# Count the occurrences of each response
other_reasons_counts = other_reasons.value_counts()

# Print the counts
print(other_reasons_counts.to_string(index=True))

# Plotting the bar chart
plt.figure(figsize=(10, 8))
other_reasons_counts.plot(kind='barh', color='lightblue')
plt.title('Other Reasons for Not Going to Health Facility')
plt.xlabel('Count')
plt.ylabel('Reason')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

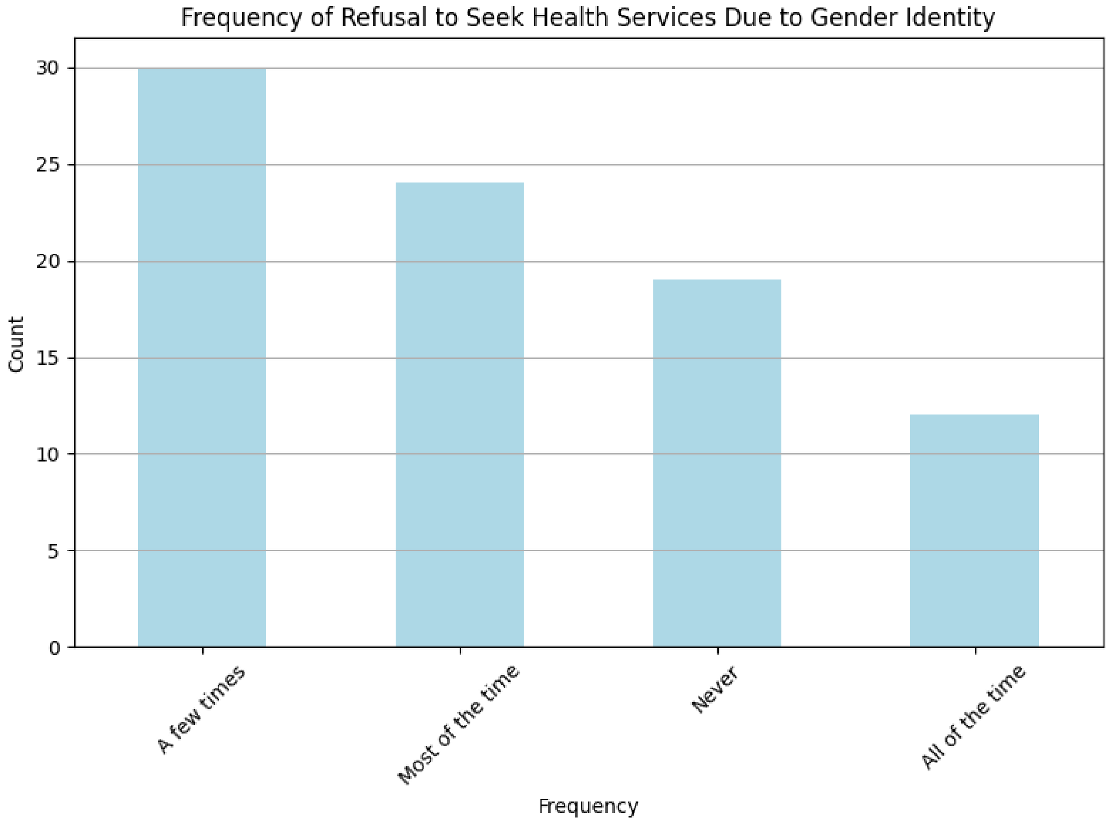
```
Respondent doesn't use condoms at times    1
Unprotected oral sex                       1
```



```
[199]: refusal_responses = nairobi_respondents['47. In the last 3 months, how often␣
        ↪did you refuse to seek health services because of your gender identity?'].
        ↪str.strip()

       # Count the occurrences of each response
       refusal_counts = refusal_responses.value_counts()

       # Print the counts
       print(refusal_counts.to_string())
```

```
# Plotting the bar chart
plt.figure(figsize=(8, 6))
refusal_counts.plot(kind='bar', color='lightblue')
plt.title('Frequency of Refusal to Seek Health Services Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
A few times          30
Most of the time     24
Never                19
All of the time      12
```

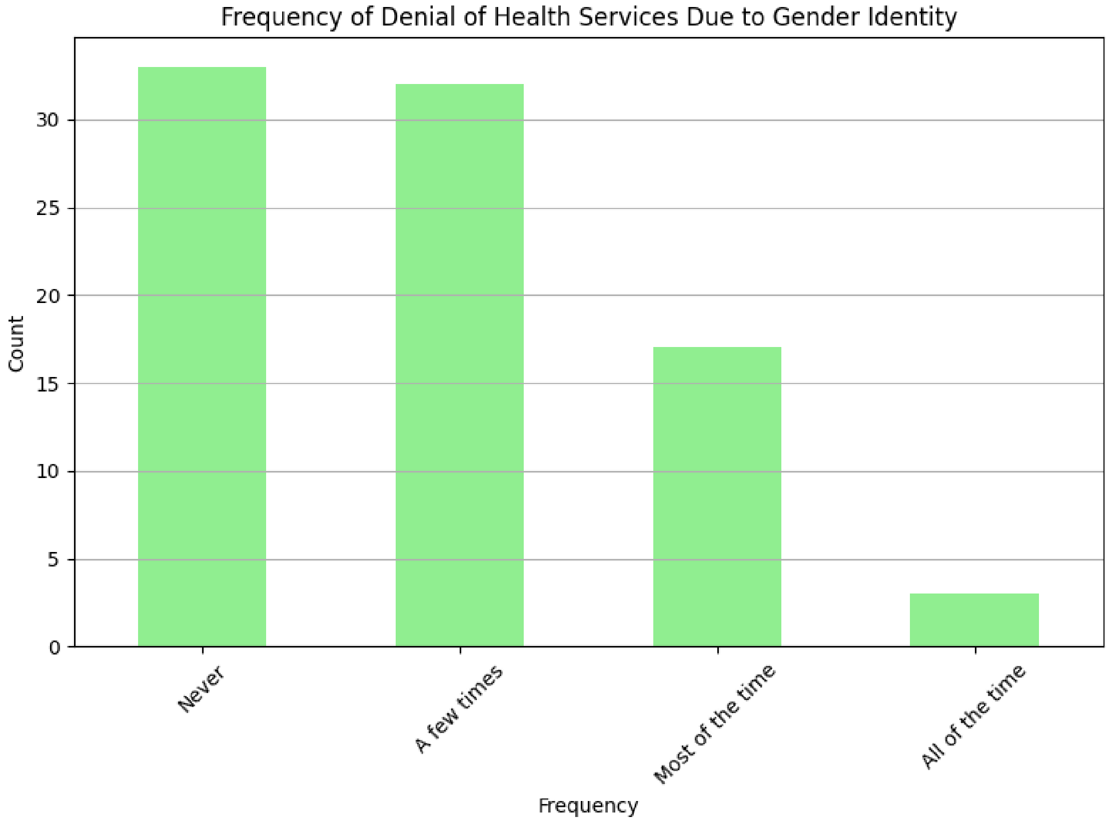


[200]:

```
denial_responses = nairobi_respondents['48. In the last 3months, how often were␣
↪you denied health services because of your gender identity?'].str.strip()

# Count the occurrences of each response
denial_counts = denial_responses.value_counts()
```

```python
# Print the counts
print(denial_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
denial_counts.plot(kind='bar', color='lightgreen')
plt.title('Frequency of Denial of Health Services Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

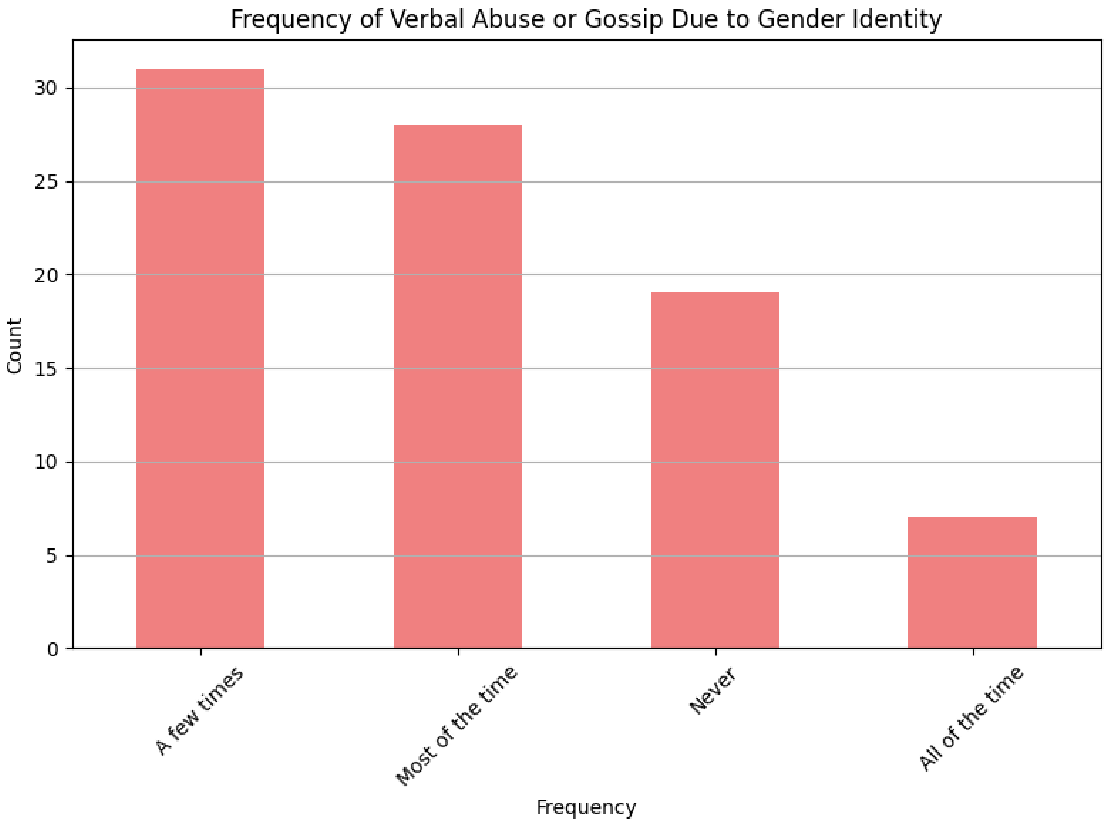
```
Never               33
A few times         32
Most of the time    17
All of the time      3
```



[201]:

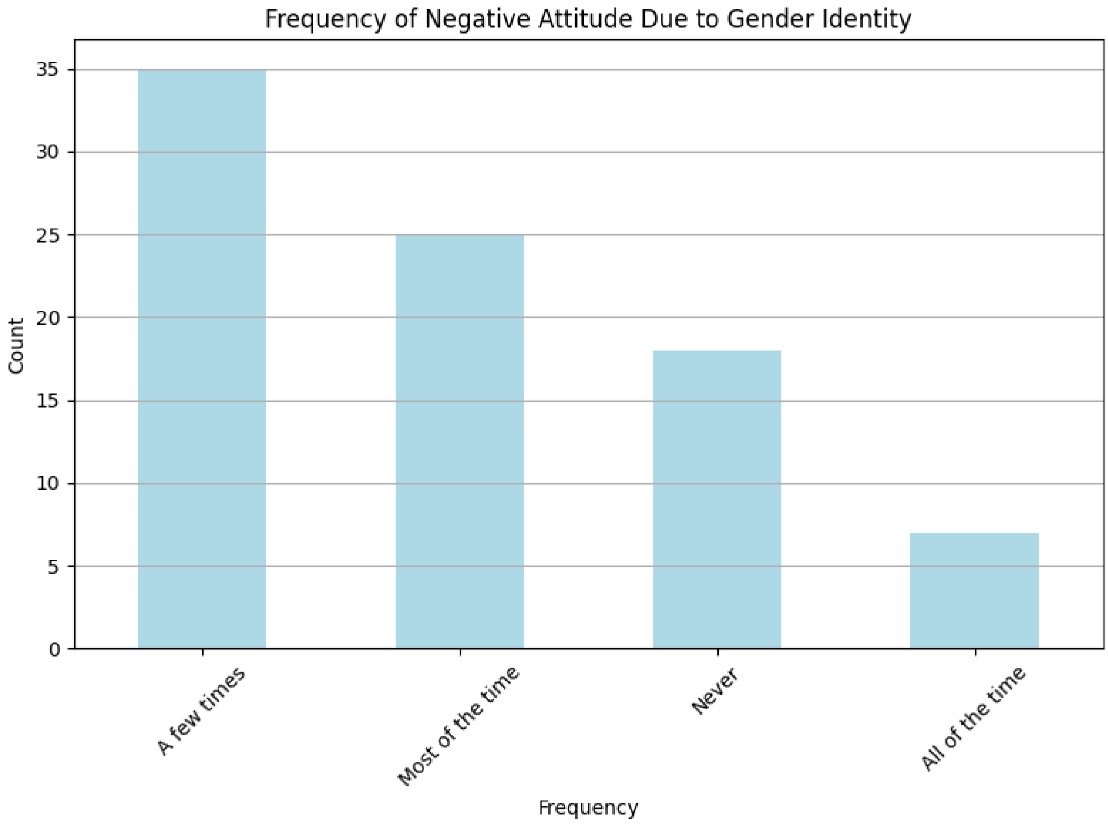
```
abuse_responses = nairobi_respondents['49. In the last 3 months, how often were␣
 ↪you gossiped or verbally abused at a health facility because of your gender␣
 ↪identity?'].str.strip()

# Count the occurrences of each response
abuse_counts = abuse_responses.value_counts()

# Print the counts
print(abuse_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
abuse_counts.plot(kind='bar', color='lightcoral')
plt.title('Frequency of Verbal Abuse or Gossip Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
A few times         31
Most of the time    28
Never               19
All of the time      7
```



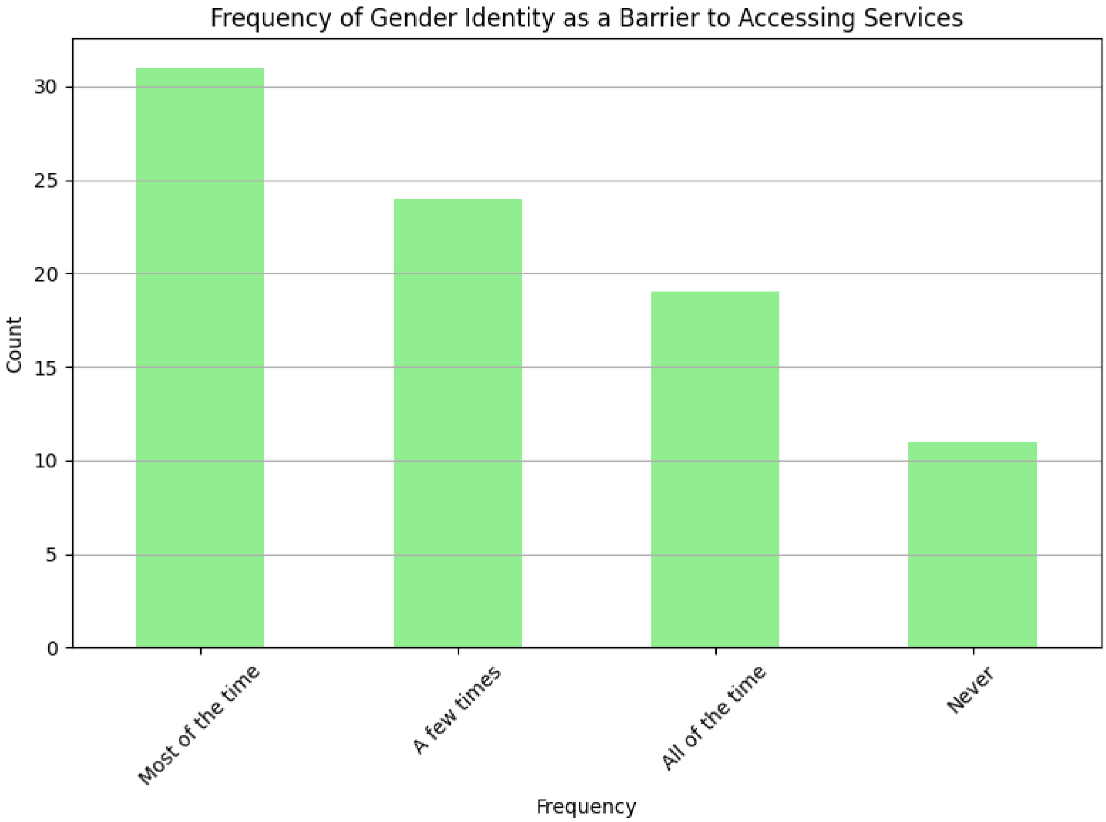
[202]:
```
attitude_responses = nairobi_respondents['50. In the last 3 months, how often␣
↪did you feel that a health care worker was having  negative attitude towards␣
↪you because of your gender identity?'].str.strip()

# Count the occurrences of each response
attitude_counts = attitude_responses.value_counts()

# Print the counts
print(attitude_counts.to_string())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
attitude_counts.plot(kind='bar', color='lightblue')
plt.title('Frequency of Negative Attitude Due to Gender Identity')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

```
A few times        35
Most of the time   25
Never              18
All of the time     7
```



```
[203]: barrier_responses = nairobi_respondents['51. In the last 3 months, how often␣
        ↪did you feel your gender identity  was  a barrier to accessing services at a␣
        ↪health facility?'].str.strip()

       # Count the occurrences of each response
       barrier_counts = barrier_responses.value_counts()

       # Print the counts
       print(barrier_counts.to_string())

       # Plotting the bar chart
       plt.figure(figsize=(8, 6))
       barrier_counts.plot(kind='bar', color='lightgreen')
       plt.title('Frequency of Gender Identity as a Barrier to Accessing Services')
       plt.xlabel('Frequency')
       plt.ylabel('Count')
       plt.xticks(rotation=45)
       plt.grid(axis='y')
       plt.tight_layout()
       plt.show()
```

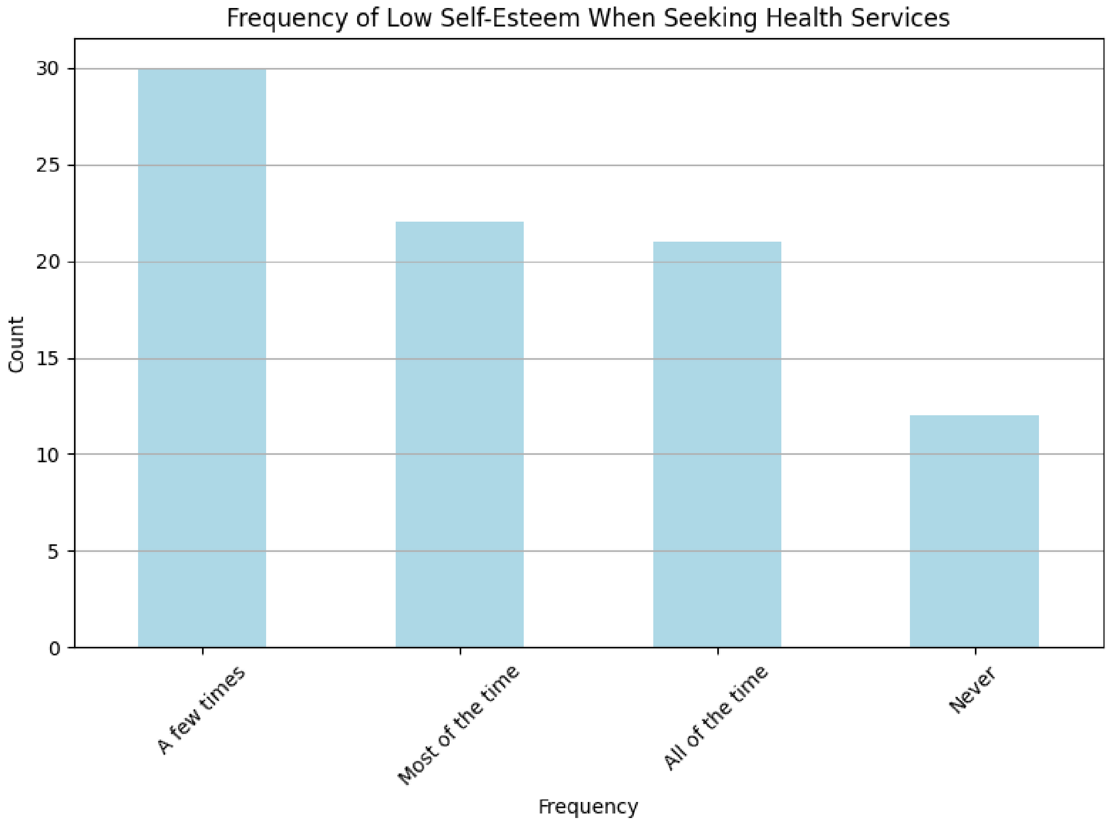
```
Most of the time    31
A few times         24
All of the time     19
Never               11
```



[204]:
```
self_esteem_responses = nairobi_respondents['53. In the last 3, how often did␣
↪you experience low self-esteem when seeking services at a health facility?'].
↪str.strip()

# Count the occurrences of each response
self_esteem_counts = self_esteem_responses.value_counts()

# Print the counts
print(self_esteem_counts.to_string())

# Plotting the bar chart
```

```
plt.figure(figsize=(8, 6))
self_esteem_counts.plot(kind='bar', color='lightblue')
plt.title('Frequency of Low Self-Esteem When Seeking Health Services')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

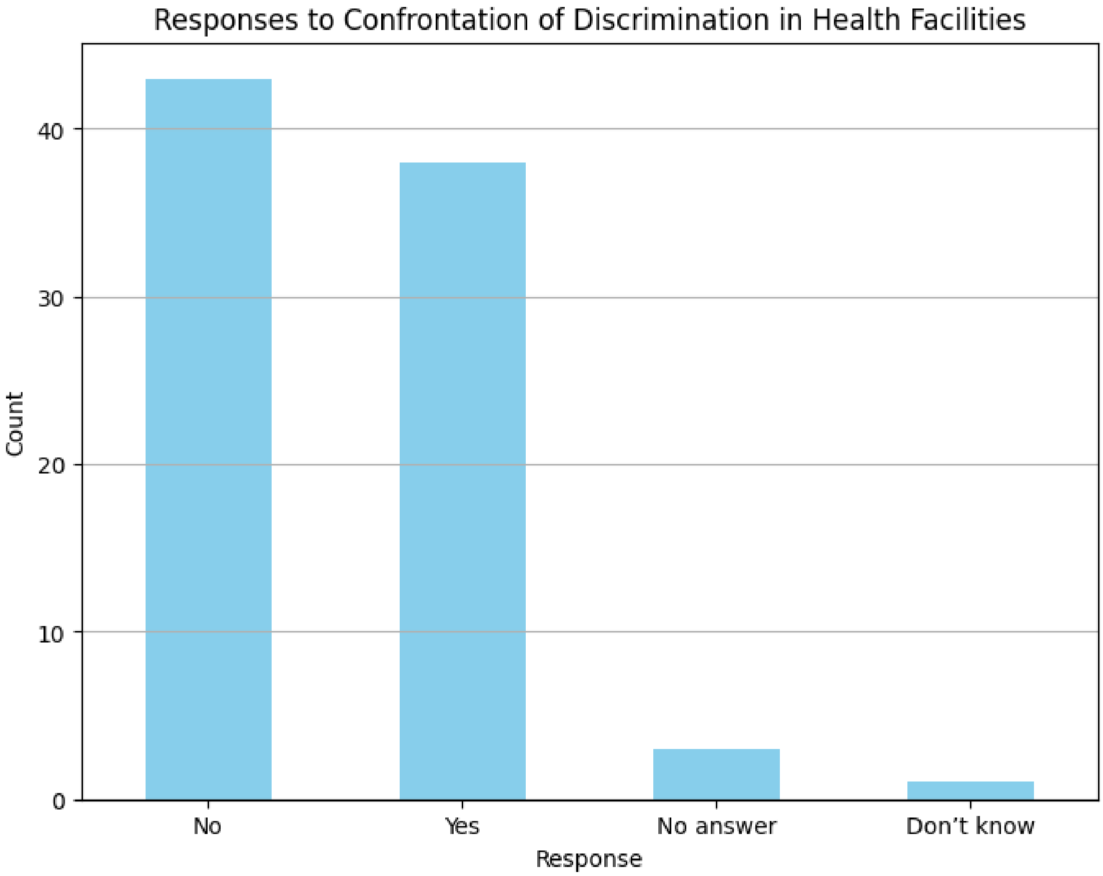
```
A few times        30
Most of the time   22
All of the time    21
Never              12
```



[205]:

```
response_counts = nairobi_respondents['54. In the last 3 months, did you ever␣
↪confront, challenge or educate a health care worker in a facility who was␣
↪Stigmatising and/or discriminating against you or other transgender women␣
↪within the facility?'].value_counts()

# Plotting the responses
```

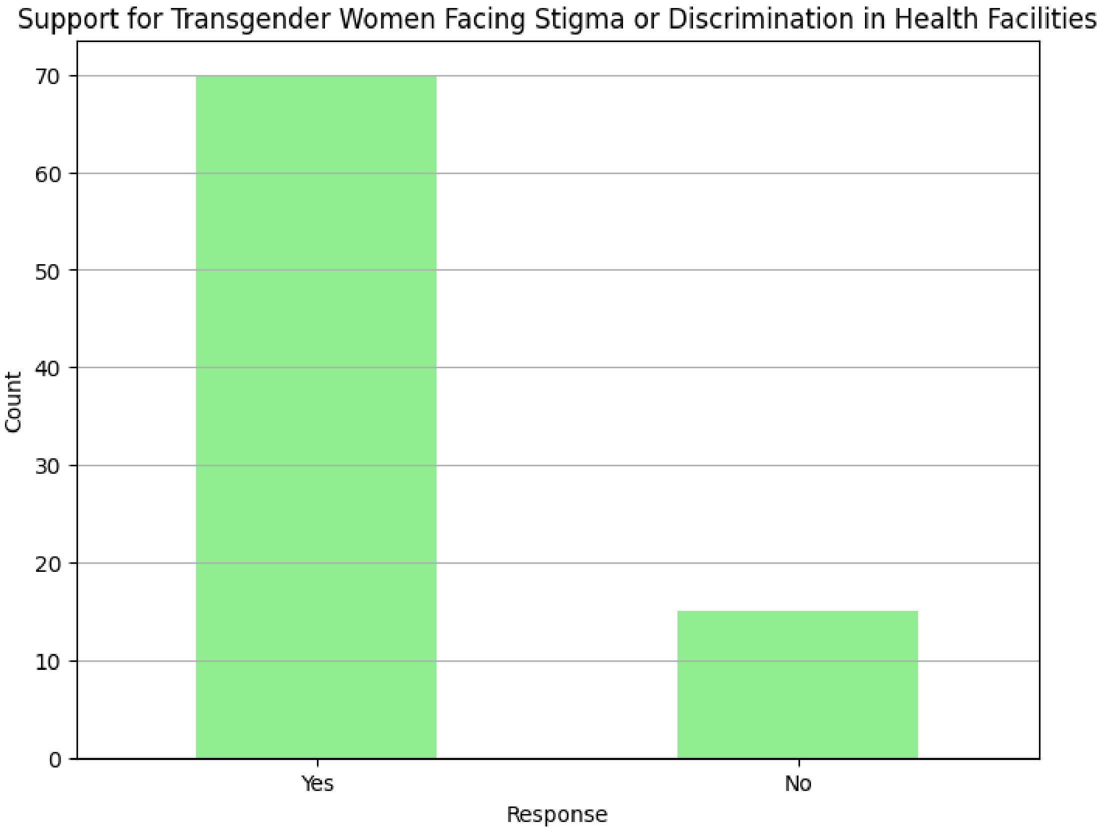
```
plt.figure(figsize=(8, 6))
response_counts.plot(kind='bar', color='skyblue')
plt.title('Responses to Confrontation of Discrimination in Health Facilities')
plt.xlabel('Response')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.grid(axis='y')

plt.show()
```



[206]:

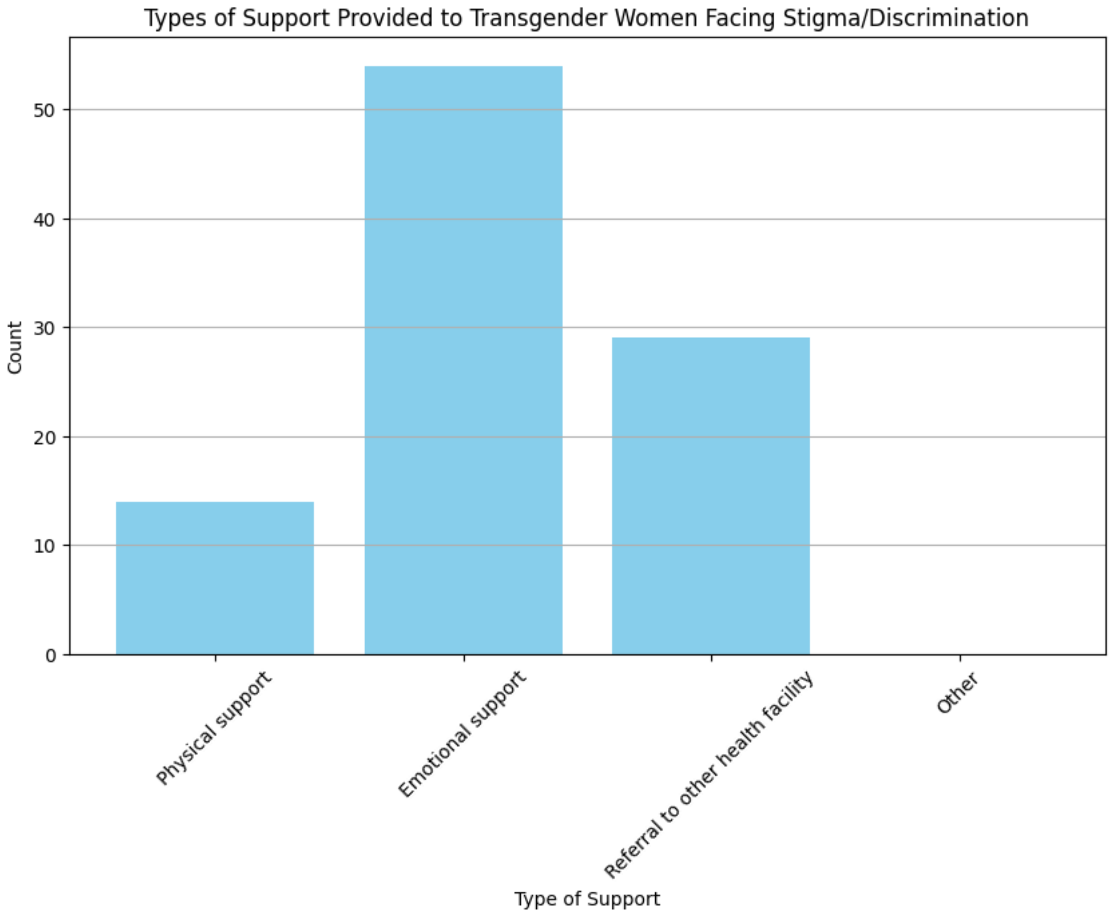
```
response_counts_advocacy = nairobi_respondents['55. In the last 3  months,did␣
↪you support any transgender woman/ women who has/have been stigmatised and/
↪or discriminated against in a health care facility?'].value_counts()

# Plotting the responses
plt.figure(figsize=(8, 6))
response_counts_advocacy.plot(kind='bar', color='lightgreen')
plt.title('Support for Transgender Women Facing Stigma or Discrimination in␣
↪Health Facilities')
```

```
plt.xlabel('Response')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.grid(axis='y')

plt.show()
```



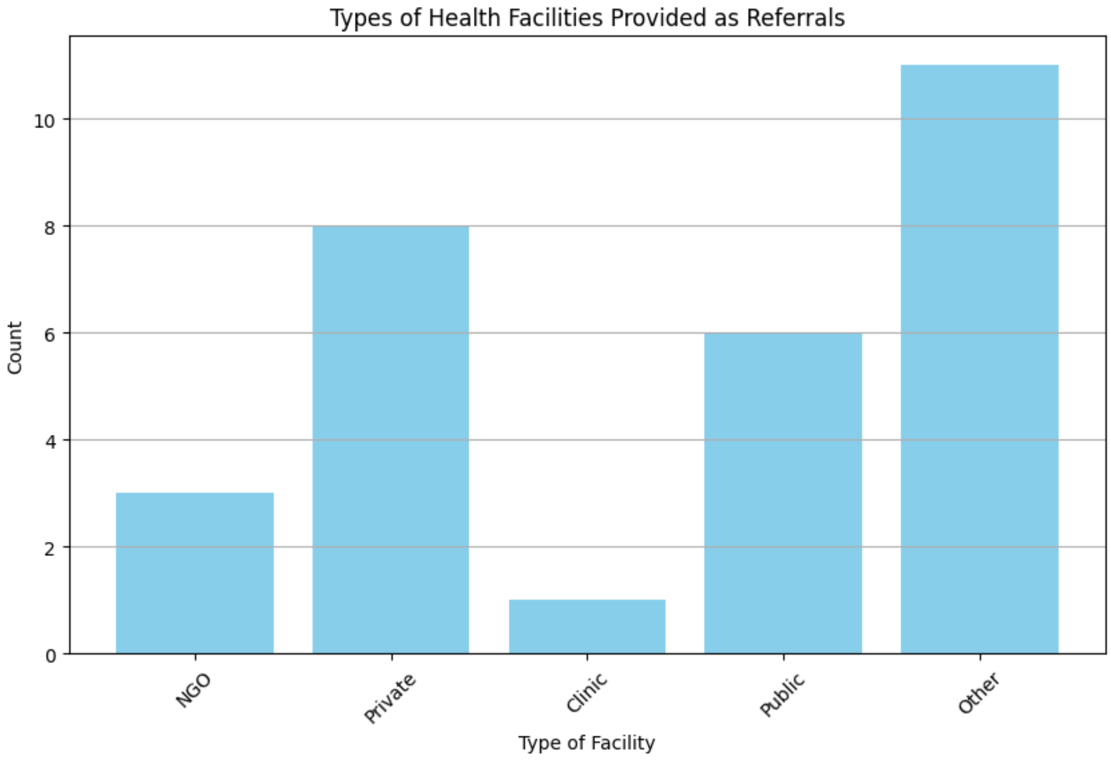
[207]:

```
support_responses = nairobi_respondents['56. If Yes, what types of support did␣
 ↪you provide?'].dropna().str.strip()

# Splitting the responses by their content
types_of_support = {
    'Physical support': 0,
    'Emotional support': 0,
    'Referral to other health facility': 0,
    'Other': 0
}

# Counting each type of support
for response in support_responses:
    if 'Physical support' in response:
        types_of_support['Physical support'] += 1
    if 'Emotional support' in response:
        types_of_support['Emotional support'] += 1
    if 'Referral to other health facility' in response:
        types_of_support['Referral to other health facility'] += 1
    if 'other(' in response.lower():
        types_of_support['Other'] += 1

# Plotting the results
plt.figure(figsize=(10, 6))
plt.bar(types_of_support.keys(), types_of_support.values(), color='skyblue')
plt.title('Types of Support Provided to Transgender Women Facing Stigma/
 ↪Discrimination')
plt.xlabel('Type of Support')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')

plt.show()
```



[209]:

```python
facility_responses = nairobi_respondents['Specify type of facility'].dropna().str.strip()

# Cleaning and standardizing facility types
facility_counts = defaultdict(int)

for response in facility_responses:
    # Standardize variations and categorize
    if 'public' in response.lower():
        facility_counts['Public'] += 1
    elif 'private' in response.lower():
        facility_counts['Private'] += 1
    elif 'ngo' in response.lower():
        facility_counts['NGO'] += 1
    elif 'clinic' in response.lower():
        facility_counts['Clinic'] += 1
    elif 'hospital' in response.lower():
        facility_counts['Hospital'] += 1
    else:
        facility_counts['Other'] += 1

# Plotting the results
plt.figure(figsize=(10, 6))
plt.bar(facility_counts.keys(), facility_counts.values(), color='skyblue')
plt.title('Types of Health Facilities Provided as Referrals')
plt.xlabel('Type of Facility')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y')

plt.show()
```



[219]:

```python
other_support_responses = nairobi_respondents['Specify other type of support'].str.strip().dropna()

# Counting the occurrences of each response
other_support_counts = other_support_responses.value_counts()

# Printing the counts
print(other_support_counts.to_string())

# Check if the series is empty before plotting
if not other_support_counts.empty:
    # Plotting the bar chart
    plt.figure(figsize=(10, 6))
    other_support_counts.plot(kind='bar', color='lightblue')
    plt.title('Types of Support Provided to Transgender Women')
    plt.xlabel('Types of Support')
    plt.ylabel('Count')
    plt.xticks(rotation=45, ha='right')
    plt.grid(axis='y')
    plt.tight_layout()
    plt.show()
else:
    print("No data to plot.")
```

```
Series([], )
No data to plot.
```

[ ]:

```

```

[ ]:

```

```