Asthama Disease Prediction using Machine Learning

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
```

In [2]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
df = pd.read_csv("asthama.csv")
```

In [4]:

```
df.head()
```

Out[4]:

	Tiredness	Dry-Cough	Difficulty-in-Breathing	Sore-Throat	None_Sympton	Pains	Nasal-Congestion	Runny-Nose	None_E
0	1	1	1	1	0	1	1	1	
1	1	1	1	1	0	1	1	1	
2	1	1	1	1	0	1	1	1	
3	1	1	1	1	0	1	1	1	
4	1	1	1	1	0	1	1	1	

In [5]:

```
df.tail()
```

Out[5]:

	Tiredness	Dry-Cough	Difficulty-in-Breathing	Sore-Throat	None_Sympton	Pains	Nasal-Congestion	Runny-Nose	N
316795	0	0	0	0	1	0	0	0	
316796	0	0	0	0	1	0	0	0	
316797	0	0	0	0	1	0	0	0	
316798	0	0	0	0	1	0	0	0	
316799	0	0	0	0	1	0	0	0	

In [6]:

```
df.shape
```

Out[6]:

```
(316800, 19)
```

In [7]:

```
df.columns
```

Out[7]:

```
Index(['Tiredness', 'Dry-Cough', 'Difficulty-in-Breathing', 'Sore-Throat',
       'None_Sympton', 'Pains', 'Nasal-Congestion', 'Runny-Nose',
       'None_Experiencing', 'Age_0-9', 'Age_10-19', 'Age_20-24', 'Age_25-5
9',
       'Age_60+', 'Gender_Female', 'Gender_Male', 'Severity_Mild',
       'Severity_Moderate', 'Severity_None'],
      dtype='object')
```

In [8]:

```
df.duplicated().sum()
```

Out[8]:

```
311040
```

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
Tiredness                  0
Dry-Cough                  0
Difficulty-in-Breathing    0
Sore-Throat                0
None_Sympton               0
Pains                      0
Nasal-Congestion           0
Runny-Nose                 0
None_Experiencing          0
Age_0-9                    0
Age_10-19                  0
Age_20-24                  0
Age_25-59                  0
Age_60+                    0
Gender_Female              0
Gender_Male                0
Severity_Mild              0
Severity_Moderate          0
Severity_None              0
dtype: int64
```

In [10]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 316800 entries, 0 to 316799
Data columns (total 19 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Tiredness               316800 non-null  int64
 1   Dry-Cough               316800 non-null  int64
 2   Difficulty-in-Breathing 316800 non-null  int64
 3   Sore-Throat             316800 non-null  int64
 4   None_Sympton            316800 non-null  int64
 5   Pains                   316800 non-null  int64
 6   Nasal-Congestion        316800 non-null  int64
 7   Runny-Nose              316800 non-null  int64
 8   None_Experiencing       316800 non-null  int64
 9   Age_0-9                 316800 non-null  int64
 10  Age_10-19               316800 non-null  int64
 11  Age_20-24               316800 non-null  int64
 12  Age_25-59               316800 non-null  int64
 13  Age_60+                 316800 non-null  int64
 14  Gender_Female           316800 non-null  int64
 15  Gender_Male             316800 non-null  int64
 16  Severity_Mild           316800 non-null  int64
 17  Severity_Moderate       316800 non-null  int64
 18  Severity_None           316800 non-null  int64
dtypes: int64(19)
memory usage: 45.9 MB
```

In [11]:

```
df.describe()
```

Out[11]:

	Tiredness	Dry-Cough	Difficulty-in-Breathing	Sore-Throat	None_Sympton	
count	316800.000000	316800.000000	316800.000000	316800.000000	316800.000000	316800.
mean	0.500000	0.562500	0.500000	0.312500	0.062500	0.
std	0.500001	0.496079	0.500001	0.463513	0.242062	0.
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.
50%	0.500000	1.000000	0.500000	0.000000	0.000000	0.
75%	1.000000	1.000000	1.000000	1.000000	0.000000	1.
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.

In [12]:

```
df.nunique()
```

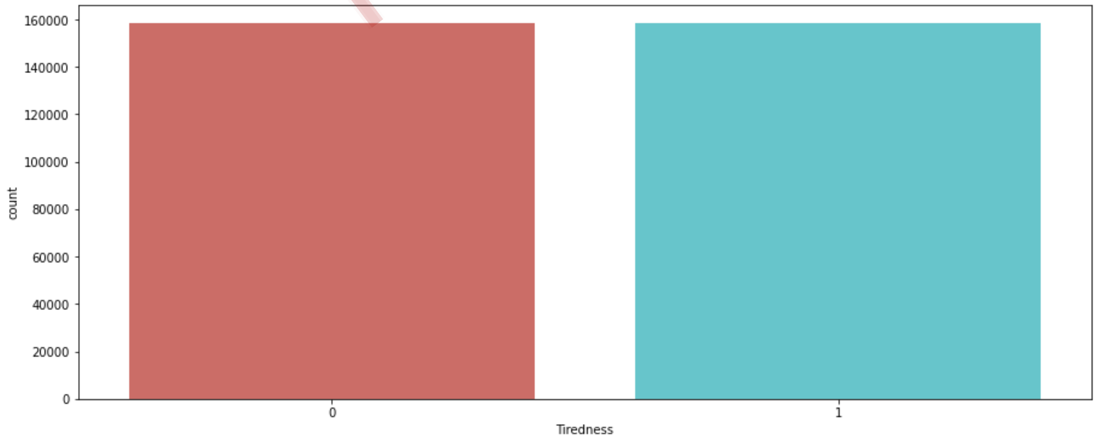
Out[12]:

```
Tiredness                  2
Dry-Cough                  2
Difficulty-in-Breathing    2
Sore-Throat                2
None_Sympton               2
Pains                      2
Nasal-Congestion           2
Runny-Nose                 2
None_Experiencing          2
Age_0-9                    2
Age_10-19                  2
Age_20-24                  2
Age_25-59                  2
Age_60+                    2
Gender_Female              2
Gender_Male                2
Severity_Mild              2
Severity_Moderate          2
Severity_None              2
dtype: int64
```

In [13]:

```
for i in df.columns:
    print('Countplot for:', i)
    plt.figure(figsize=(15,6))
    sns.countplot(x = df[i], data = df, palette = 'hls')
    plt.show()
    print('\n')
```

```
Countplot for: Tiredness
```

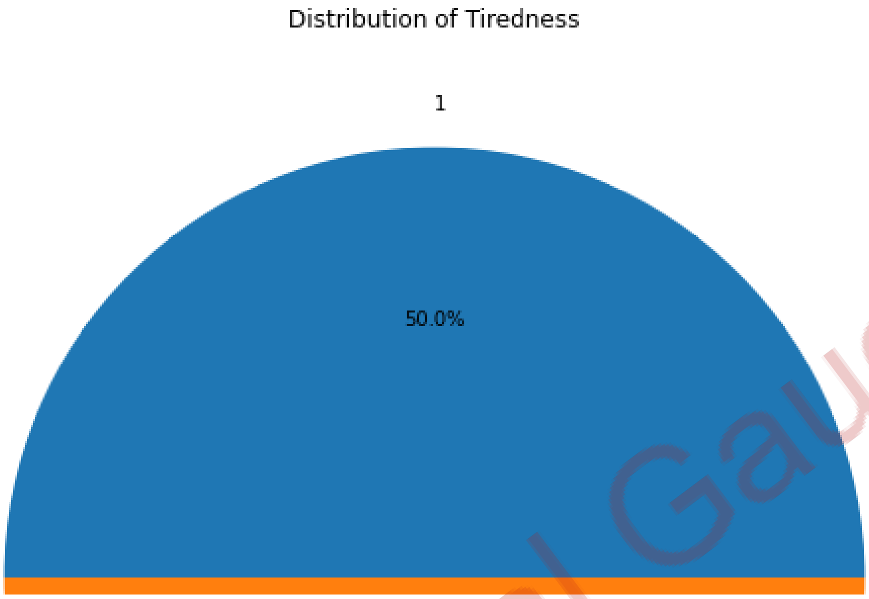


```
Countplot for: Dry-Cough
```

In [14]:

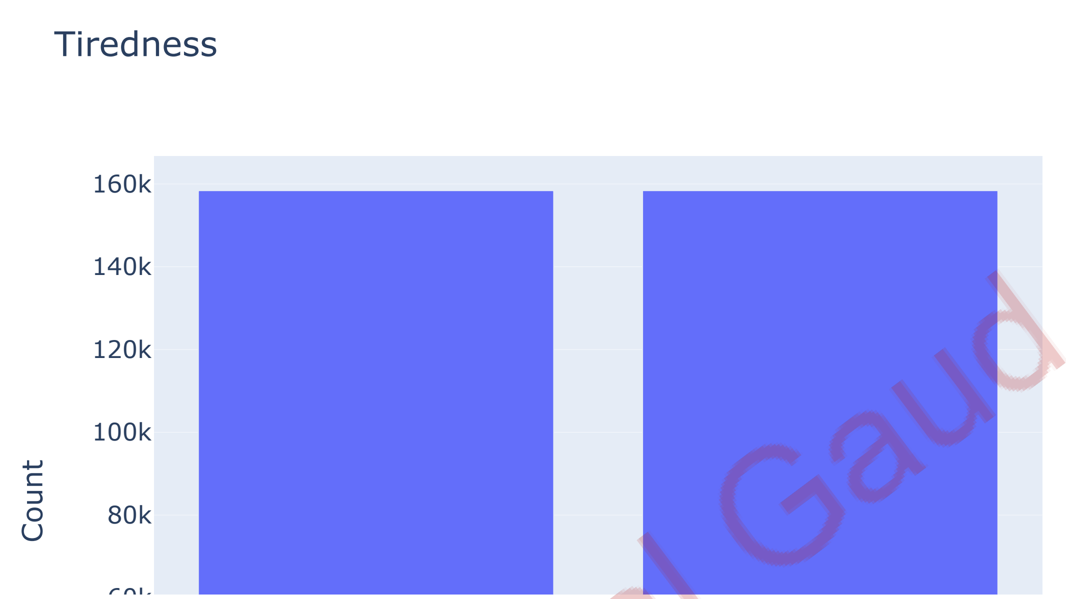
```
for i in df.columns:
    print('Pie plot for:', i)
    plt.figure(figsize=(20, 10))
    df[i].value_counts().plot(kind='pie', autopct='%1.1f%%')
    plt.title('Distribution of ' + i)
    plt.ylabel('')
    plt.show()
    print('\n')
```

```
Pie plot for: Tiredness
```



In [15]:

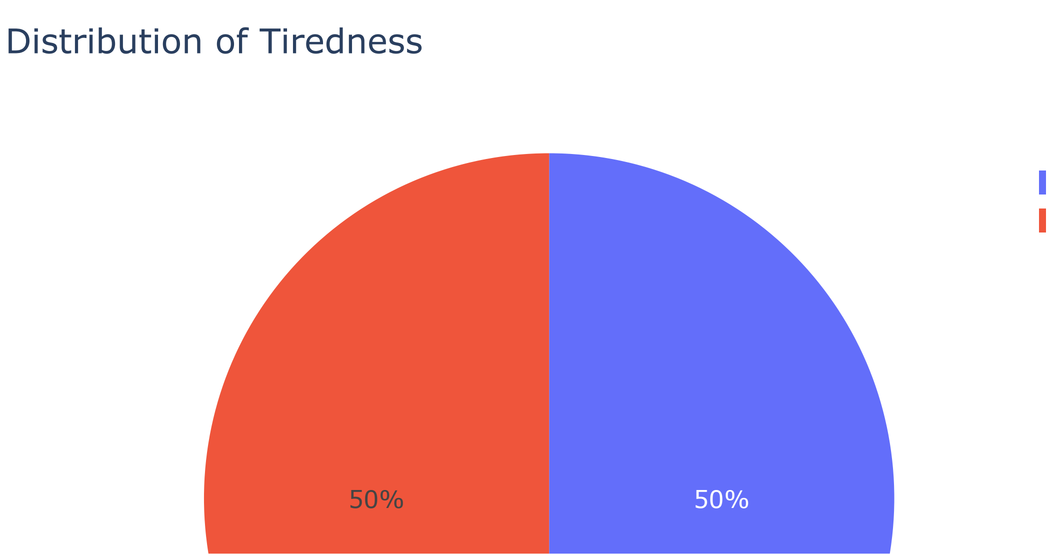
```
for i in df.columns:
    fig = go.Figure(data=[go.Bar(x=df[i].value_counts().index,
                                 y=df[i].value_counts())])
    fig.update_layout(
        title=i,
        xaxis_title=i,
        yaxis_title="Count")
    fig.show()
```



In [16]:

```
for i in df.columns:
    print('Pie plot for:', i)
    fig = px.pie(df, names=i, title='Distribution of ' + i)
    fig.show()
    print('\n')
```

```
Pie plot for: Tiredness
```



In [17]:

```
df = df.drop(['Severity_None'], axis = 1)
```

In [18]:

```
df['Asthma_Severity'] = df[['Severity_Mild', 'Severity_Moderate']].idxmax(axis=1)
```

In [19]:

```
severity_mapping = {
    'Severity_Mild': 'Mild',
    'Severity_Moderate': 'Moderate'
}
```

In [20]:

```
df['Asthma_Severity'] = df['Asthma_Severity'].map(severity_mapping)
```

In [21]:

```
df['Asthma_Severity']
```

Out[21]:

```
0             Mild
1             Mild
2             Mild
3         Moderate
4         Moderate
            ...
316795        Mild
316796        Mild
316797        Mild
316798        Mild
316799        Mild
Name: Asthma_Severity, Length: 316800, dtype: object
```

In [22]:

```
severity_mapping = {
    'Mild': 0,
    'Moderate': 1
}
```

In [23]:

```
df['Asthma_Severity'] = df['Asthma_Severity'].map(severity_mapping)
```

In [24]:

```
df['Asthma_Severity'].value_counts()
```

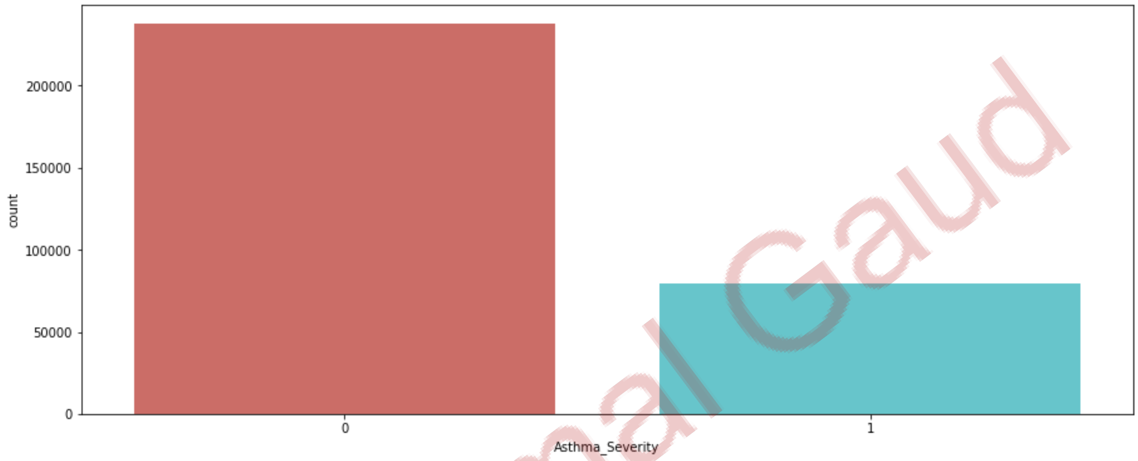
Out[24]:

```
0    237600
1     79200
Name: Asthma_Severity, dtype: int64
```

In [25]:

```
print('Countplot for:', 'Asthma_Severity')
plt.figure(figsize=(15,6))
sns.countplot(x = df['Asthma_Severity'], data = df, palette = 'hls')
plt.show()
```

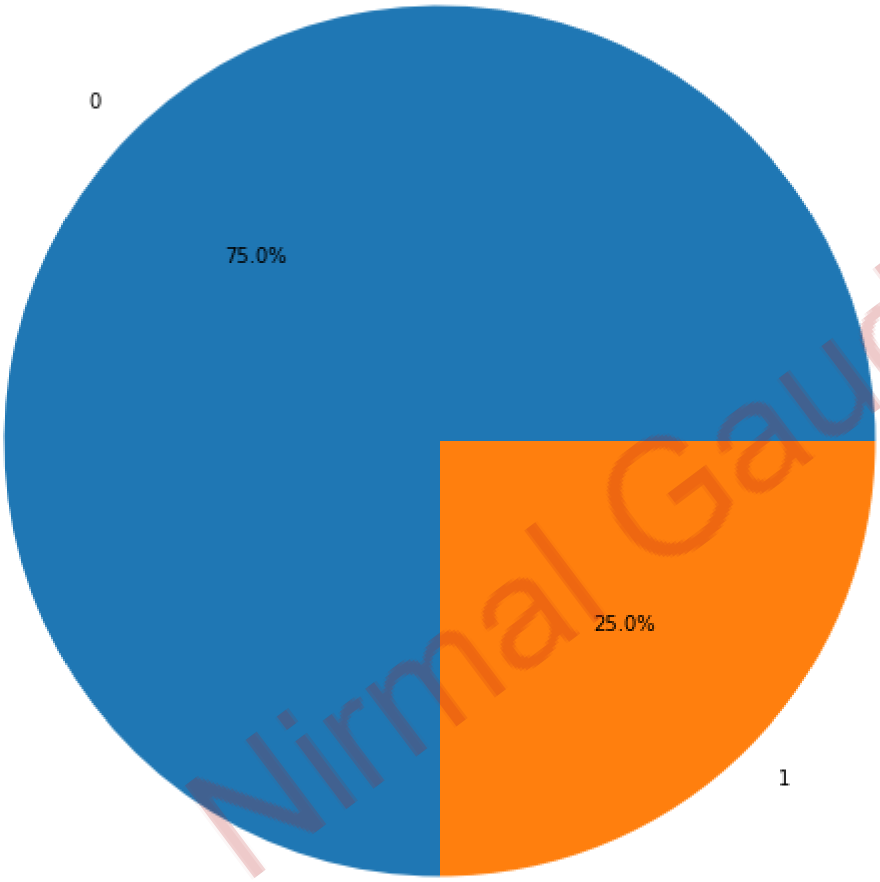
```
Countplot for: Asthma_Severity
```



In [26]:

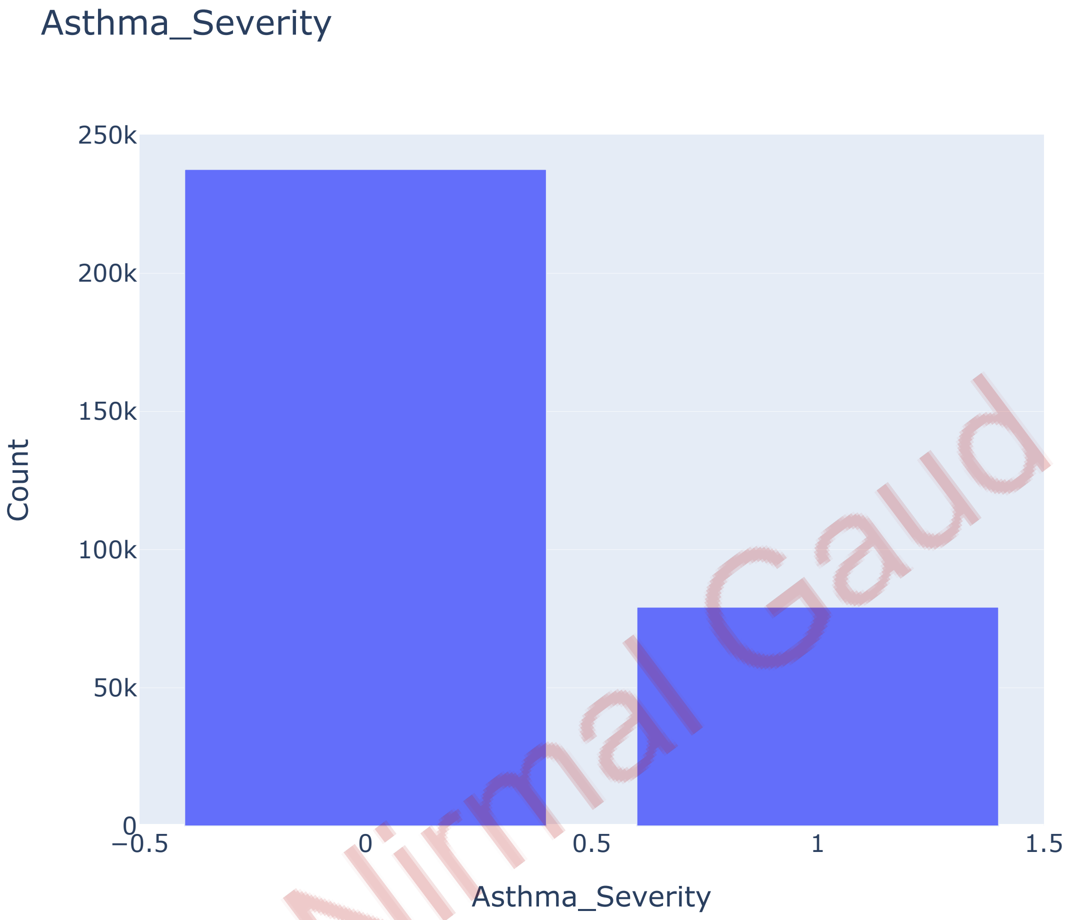
```
print('Pie plot for:', 'Asthma_Severity')
plt.figure(figsize=(20, 10))
df['Asthma_Severity'].value_counts().plot(kind='pie', autopct='%1.1f%%')
plt.ylabel('')
plt.show()
```

```
Pie plot for: Asthma_Severity
```



In [27]:

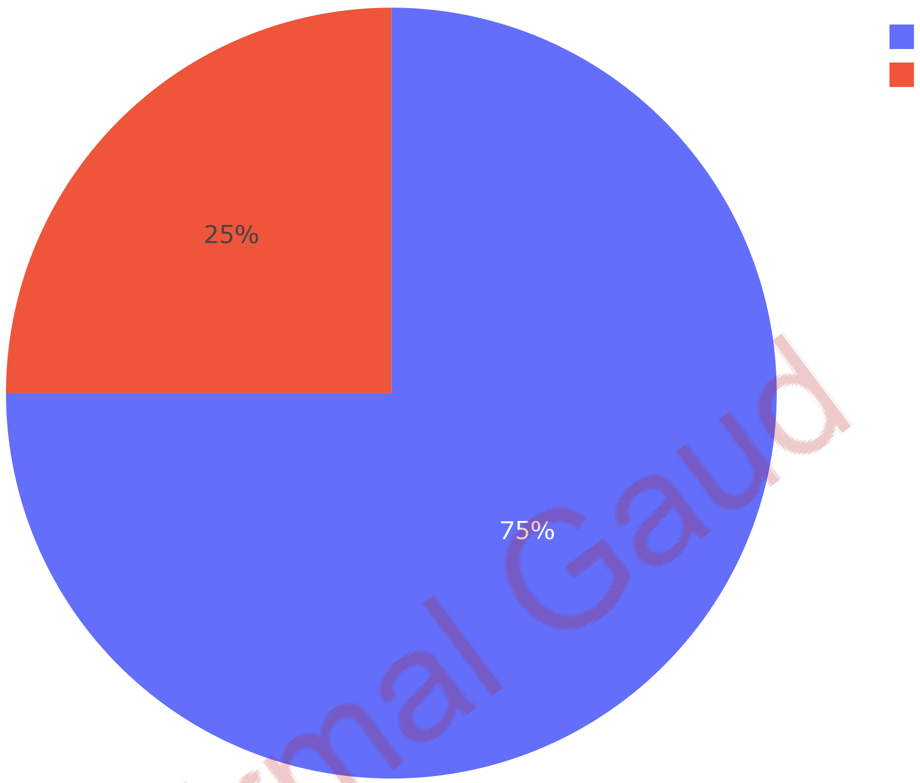
```
fig = go.Figure(data=[go.Bar(x=df['Asthma_Severity'].value_counts().index,
                              y=df['Asthma_Severity'].value_counts())])
fig.update_layout(title='Asthma_Severity', xaxis_title='Asthma_Severity',
                  yaxis_title="Count")
fig.show()
```



In [28]:

```
print('Pie plot for:', 'Asthma_Severity')
fig = px.pie(df, names='Asthma_Severity')
fig.show()
```

```
Pie plot for: Asthma_Severity
```



In [29]:

```
df.columns
```

Out[29]:

```
Index(['Tiredness', 'Dry-Cough', 'Difficulty-in-Breathing', 'Sore-Throat',
       'None_Sympton', 'Pains', 'Nasal-Congestion', 'Runny-Nose',
       'None_Experiencing', 'Age_0-9', 'Age_10-19', 'Age_20-24', 'Age_25-5
9',
       'Age_60+', 'Gender_Female', 'Gender_Male', 'Severity_Mild',
       'Severity_Moderate', 'Asthma_Severity'],
      dtype='object')
```

In [30]:

```
severity_levels = ['Severity_Mild', 'Severity_Moderate']
severity_distribution = df[severity_levels].sum()
```

In [31]:

```
severity_distribution
```

Out[31]:

```
Severity_Mild        79200
Severity_Moderate    79200
dtype: int64
```

In [32]:

```
df = df.drop(['Severity_Mild','Severity_Moderate'], axis = 1)
```

In [33]:

```
symptoms = ['Tiredness', 'Dry-Cough', 'Difficulty-in-Breathing', 'Sore-Throat', 'Pains',
symptom_counts = df[symptoms].sum()
```

In [34]:

```
symptom_counts
```

Out[34]:

```
Tiredness                  158400
Dry-Cough                  178200
Difficulty-in-Breathing    158400
Sore-Throat                 99000
Pains                      115200
Nasal-Congestion           172800
Runny-Nose                 172800
dtype: int64
```

In [35]:

```
age_groups = ['Age_0-9', 'Age_10-19', 'Age_20-24', 'Age_25-59', 'Age_60+']
age_distribution = df[age_groups].sum()
```

In [36]:

```
age_distribution
```

Out[36]:

```
Age_0-9      63360
Age_10-19    63360
Age_20-24    63360
Age_25-59    63360
Age_60+      63360
dtype: int64
```

In [37]:

```
gender_groups = ['Gender_Female', 'Gender_Male']
gender_distribution = df[gender_groups].sum()
```

In [38]:

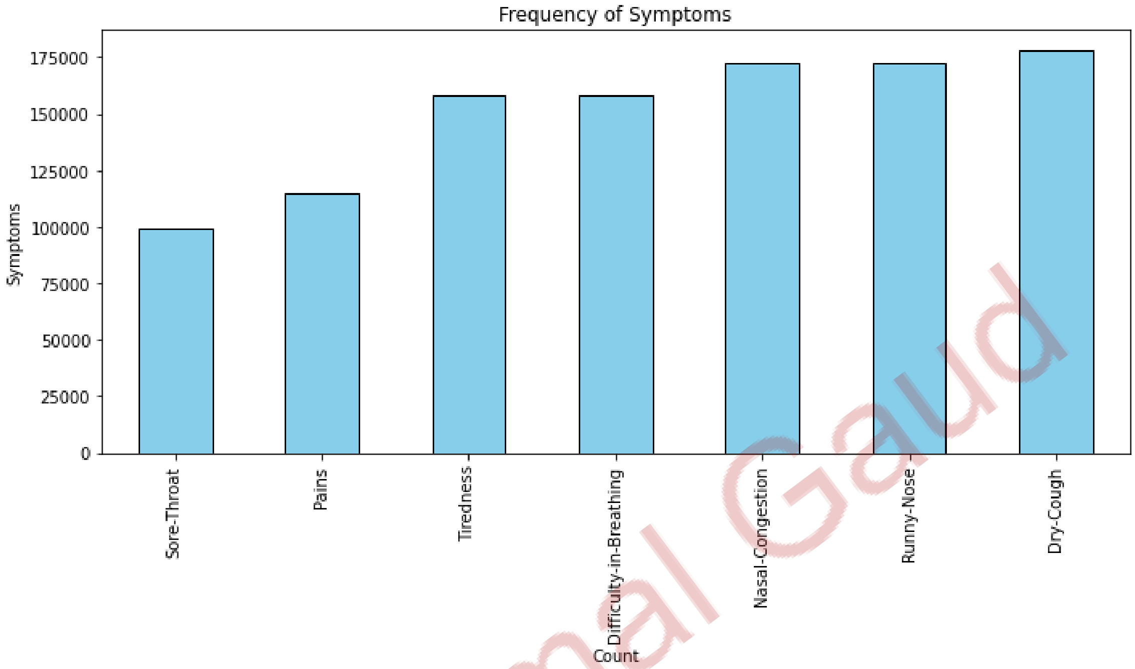
```
gender_distribution
```

Out[38]:

```
Gender_Female    105600
Gender_Male      105600
dtype: int64
```

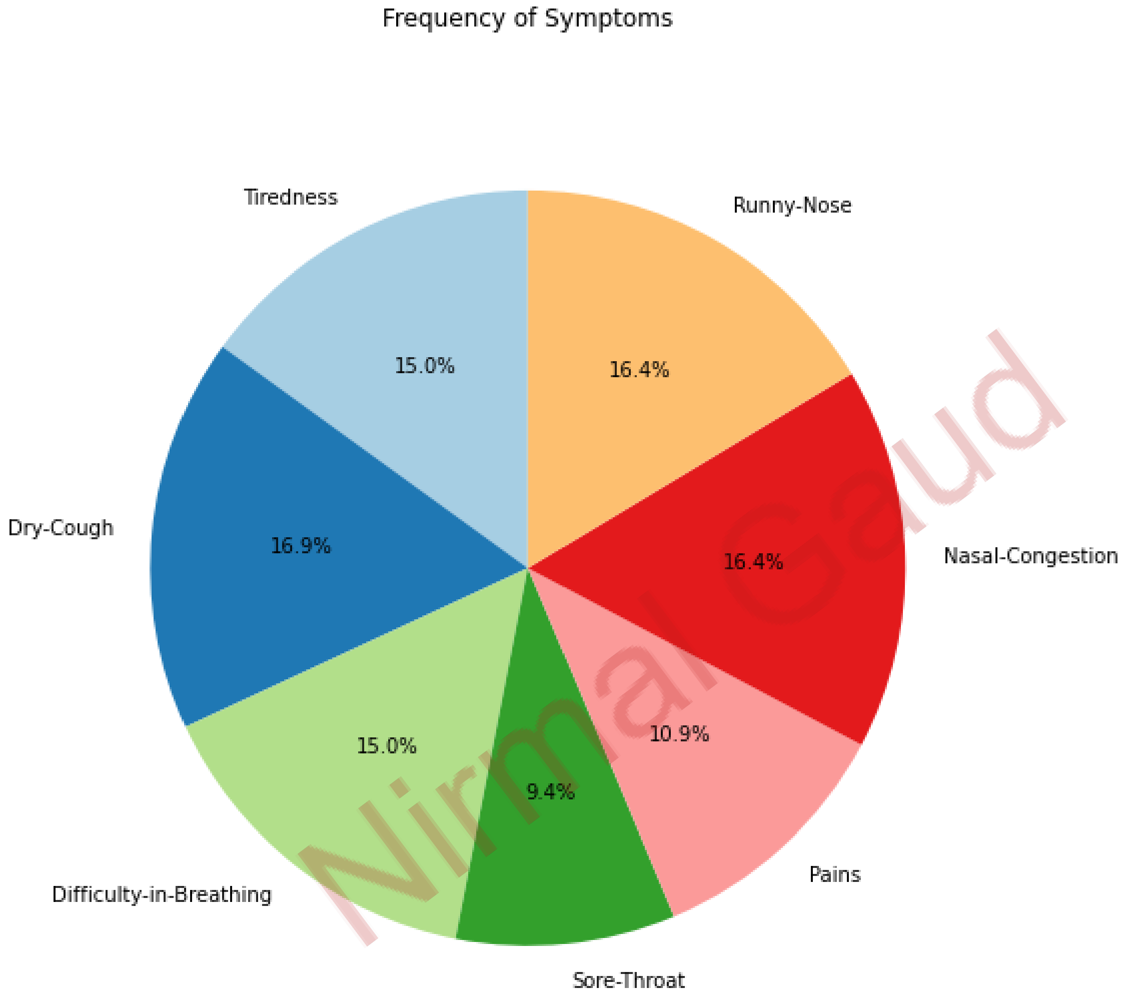
In [39]:

```
fig, ax = plt.subplots(figsize=(10, 6))

symptom_counts.sort_values().plot(kind='bar', ax=ax, color='skyblue',
                                  edgecolor='black')
ax.set_title('Frequency of Symptoms')
ax.set_xlabel('Count')
ax.set_ylabel('Symptoms')

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



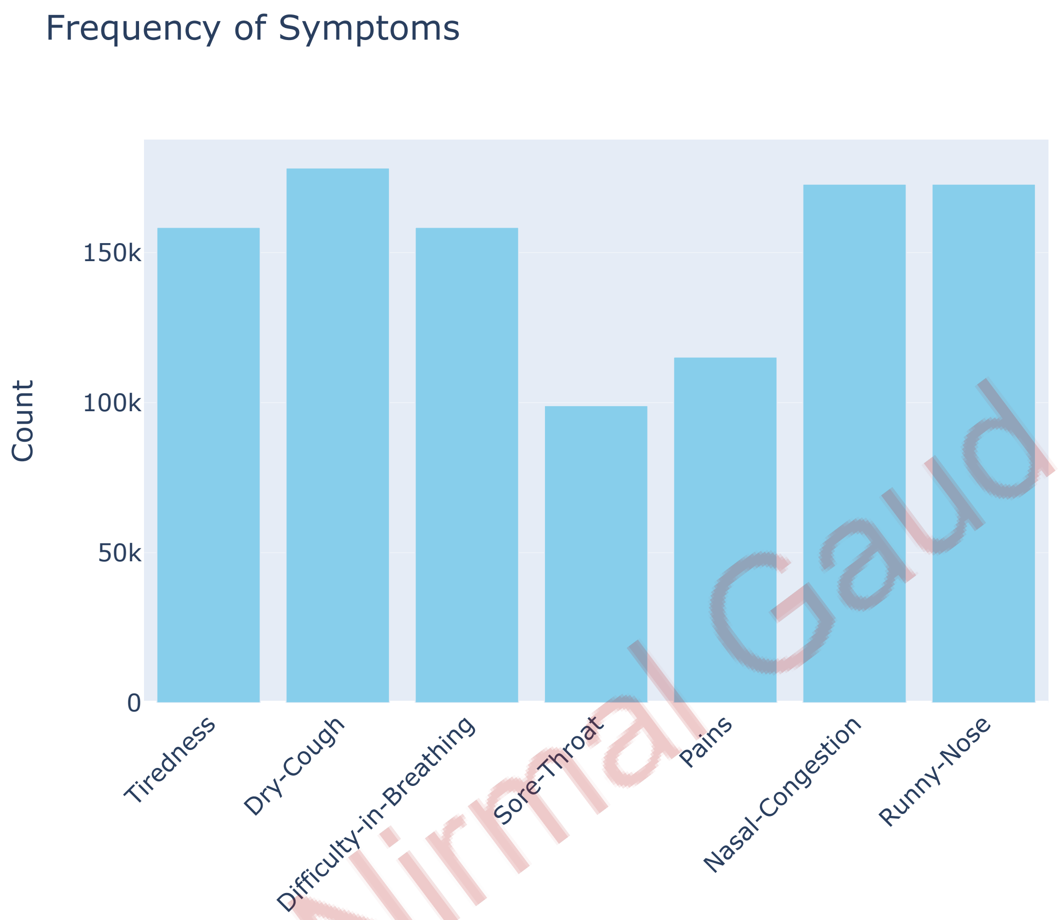
In [40]:

```
fig, ax = plt.subplots(figsize=(8, 8))
ax.pie(symptom_counts, labels=symptom_counts.index, autopct='%1.1f%%',
       startangle=90,
       colors=plt.cm.Paired(range(len(symptom_counts))))
ax.set_title('Frequency of Symptoms')
plt.axis('equal')
plt.tight_layout()
plt.show()
```



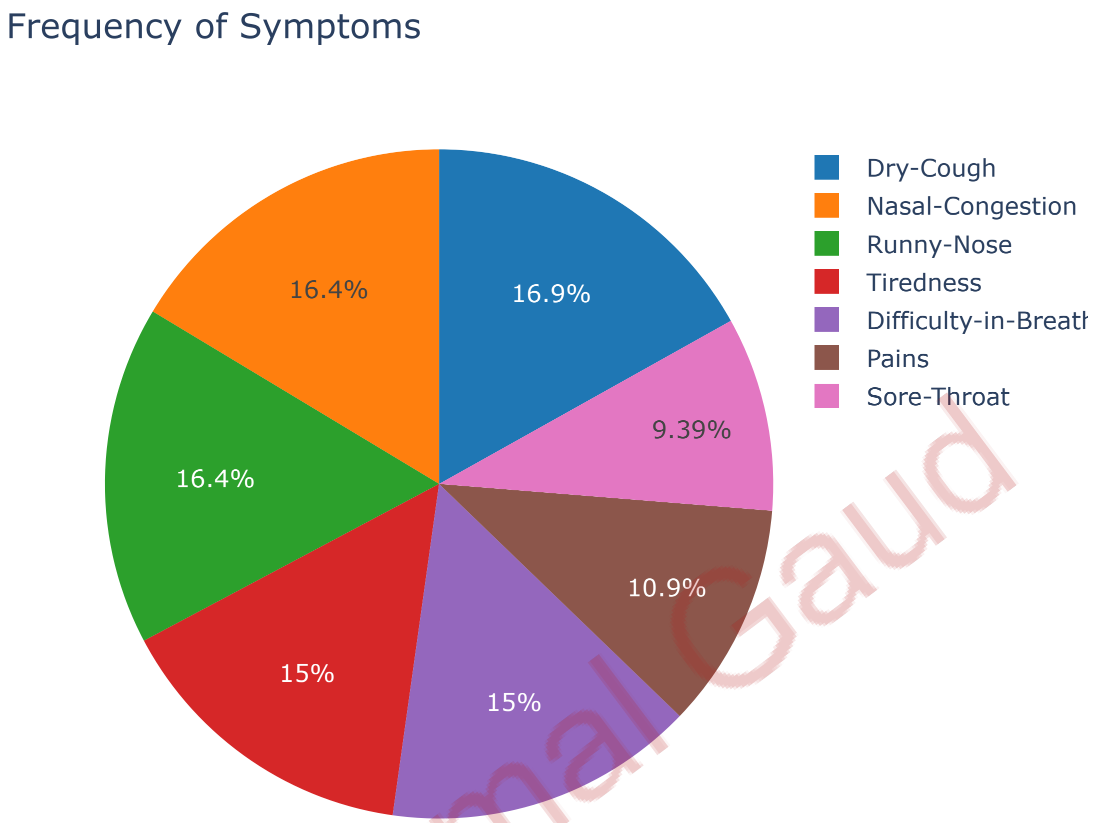
In [41]:

```
fig = px.bar(symptom_counts, x=symptom_counts.index, y=symptom_counts.values,
             labels={'x': 'Symptoms', 'y': 'Count'},
             title='Frequency of Symptoms', color_discrete_sequence=['skyblue'])
fig.update_layout(xaxis_tickangle=-45)
fig.show()
```



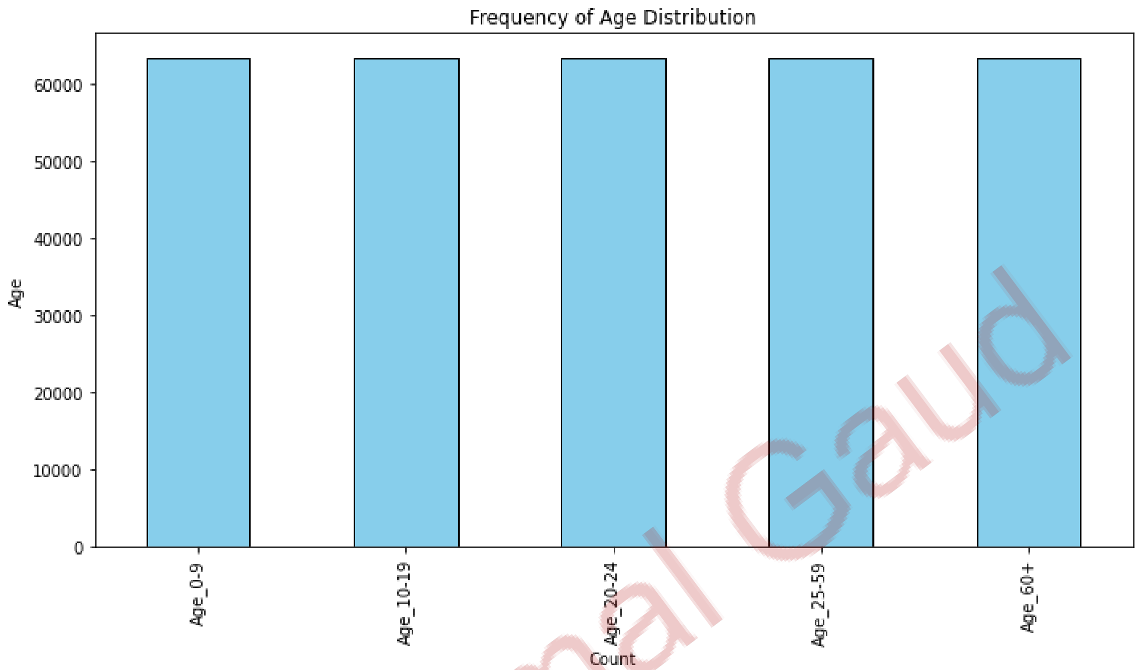
In [42]:

```
custom_colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e37
fig = px.pie(symptom_counts, values=symptom_counts.values, names=symptom_counts.index,
             title='Frequency of Symptoms', color_discrete_sequence=custom_colors)
fig.show()
```



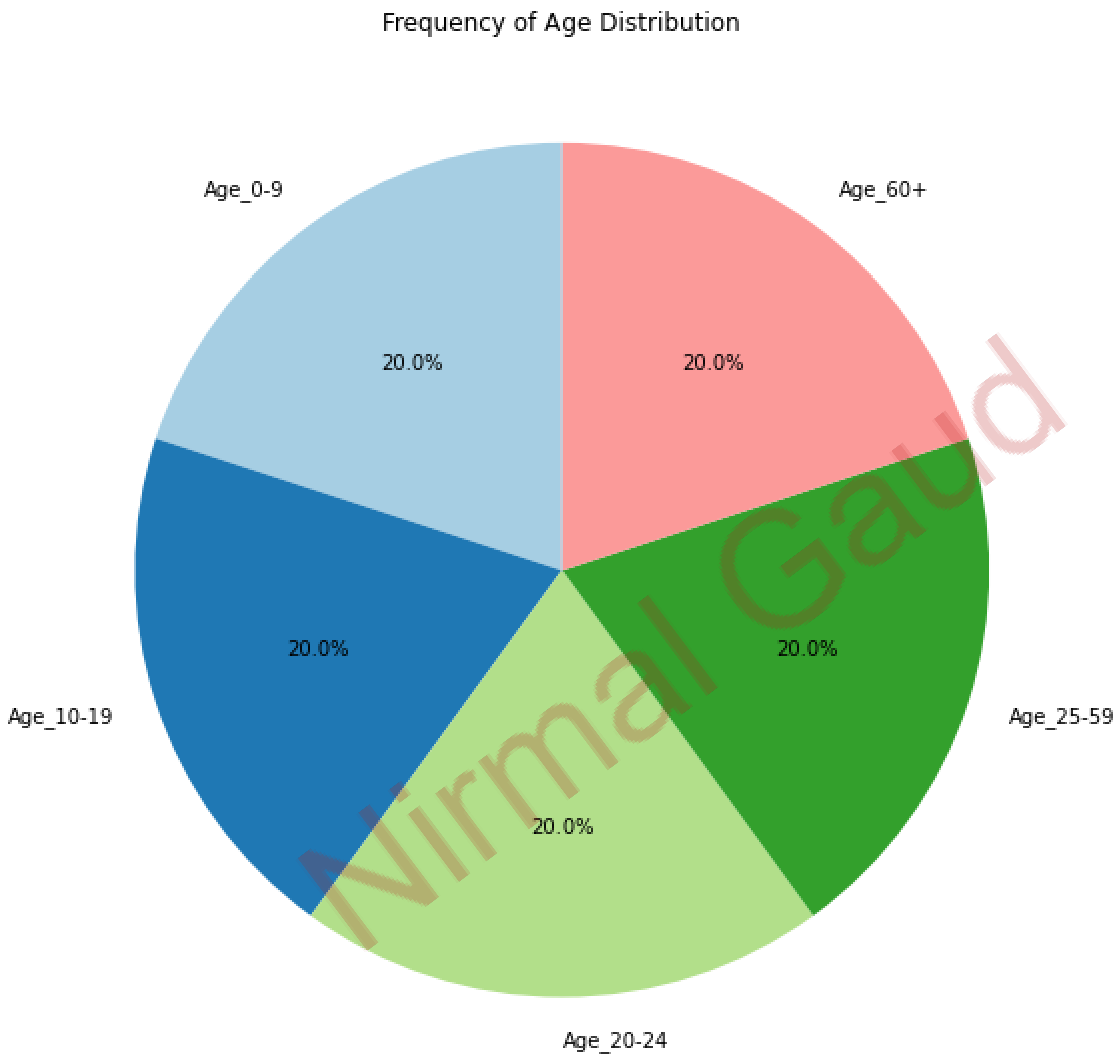
In [43]:

```
fig, ax = plt.subplots(figsize=(10, 6))

age_distribution.sort_values().plot(kind='bar', ax=ax, color='skyblue',
                                    edgecolor='black')
ax.set_title('Frequency of Age Distribution')
ax.set_xlabel('Count')
ax.set_ylabel('Age')

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



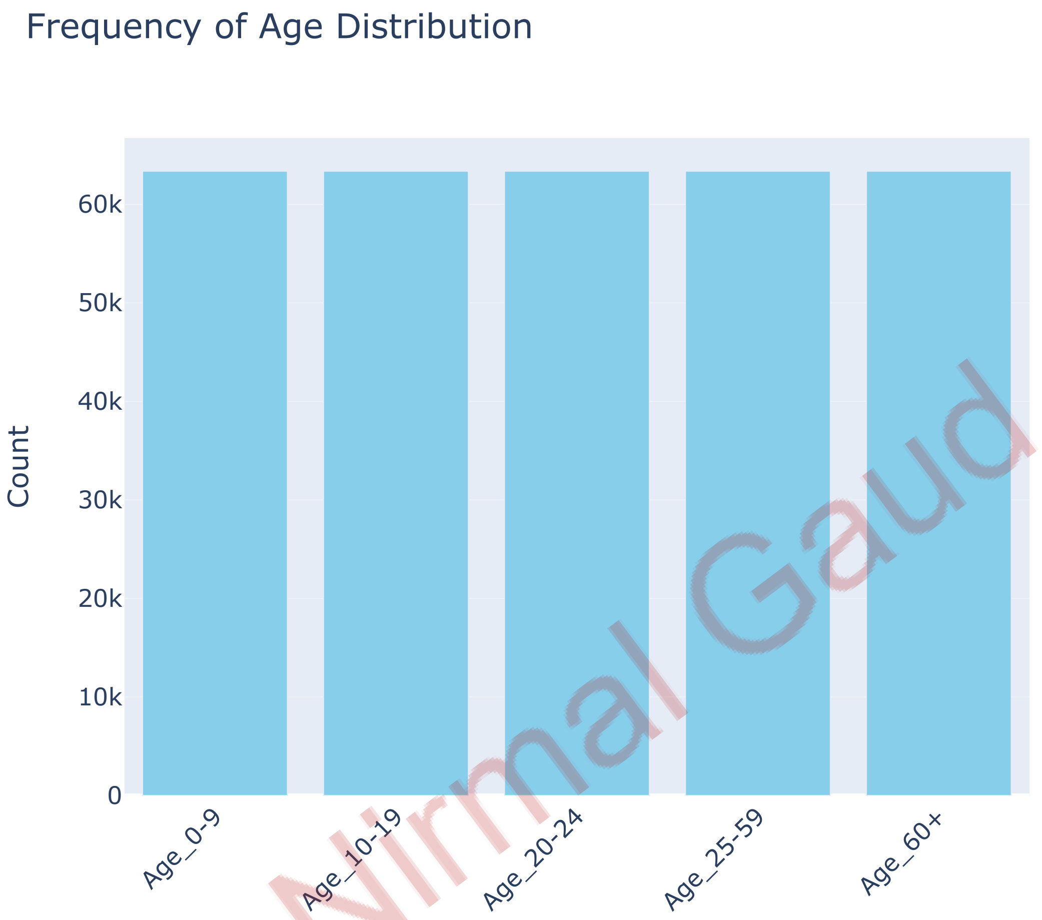
In [44]:

```
fig, ax = plt.subplots(figsize=(8, 8))
ax.pie(age_distribution, labels=age_distribution.index, autopct='%1.1f%%',
       startangle=90,
       colors=plt.cm.Paired(range(len(age_distribution))))
ax.set_title('Frequency of Age Distribution')
plt.axis('equal')
plt.tight_layout()
plt.show()
```



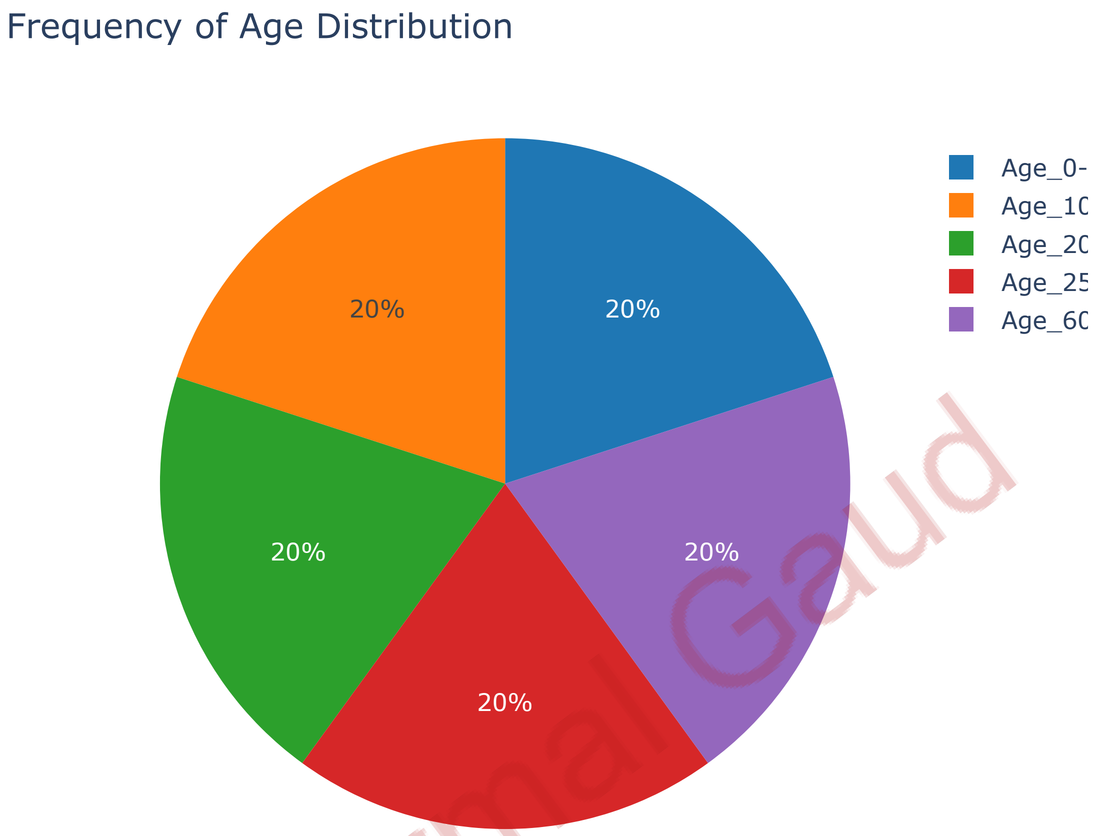
In [45]:

```
fig = px.bar(age_distribution, x=age_distribution.index, y=age_distribution.values,
             labels={'x': 'Age Distribution', 'y': 'Count'},
             title='Frequency of Age Distribution', color_discrete_sequence=['skyblue'])
fig.update_layout(xaxis_tickangle=-45)
fig.show()
```



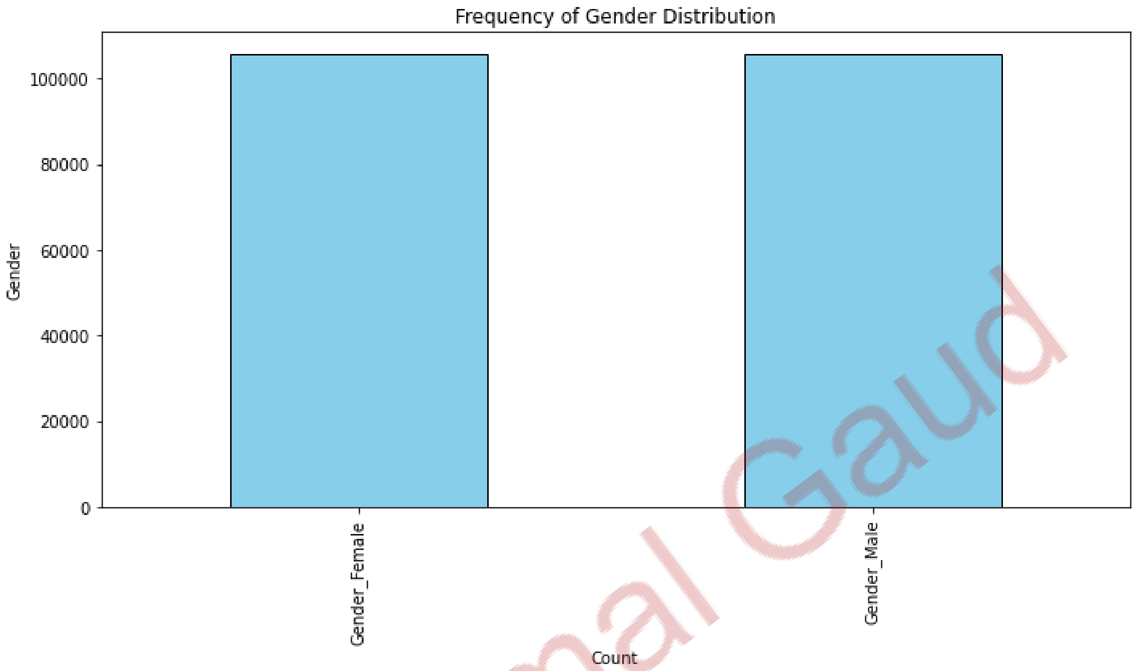
In [46]:

```
custom_colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e37
fig = px.pie(age_distribution, values=age_distribution.values, names=age_distribution.in
             title='Frequency of Age Distribution', color_discrete_sequence=custom_color
fig.show()
```



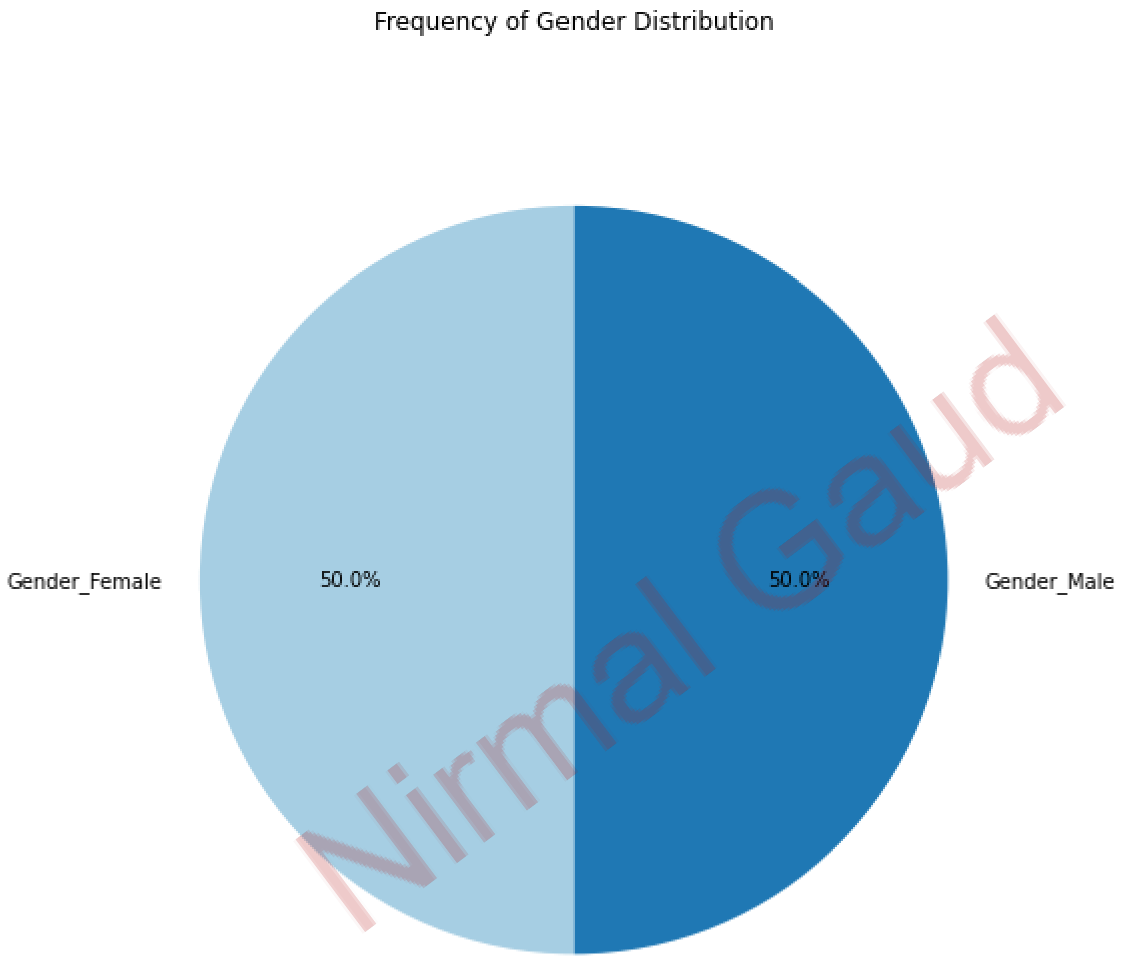
In [47]:

```
fig, ax = plt.subplots(figsize=(10, 6))

gender_distribution.sort_values().plot(kind='bar', ax=ax, color='skyblue',
                                        edgecolor='black')
ax.set_title('Frequency of Gender Distribution')
ax.set_xlabel('Count')
ax.set_ylabel('Gender')

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



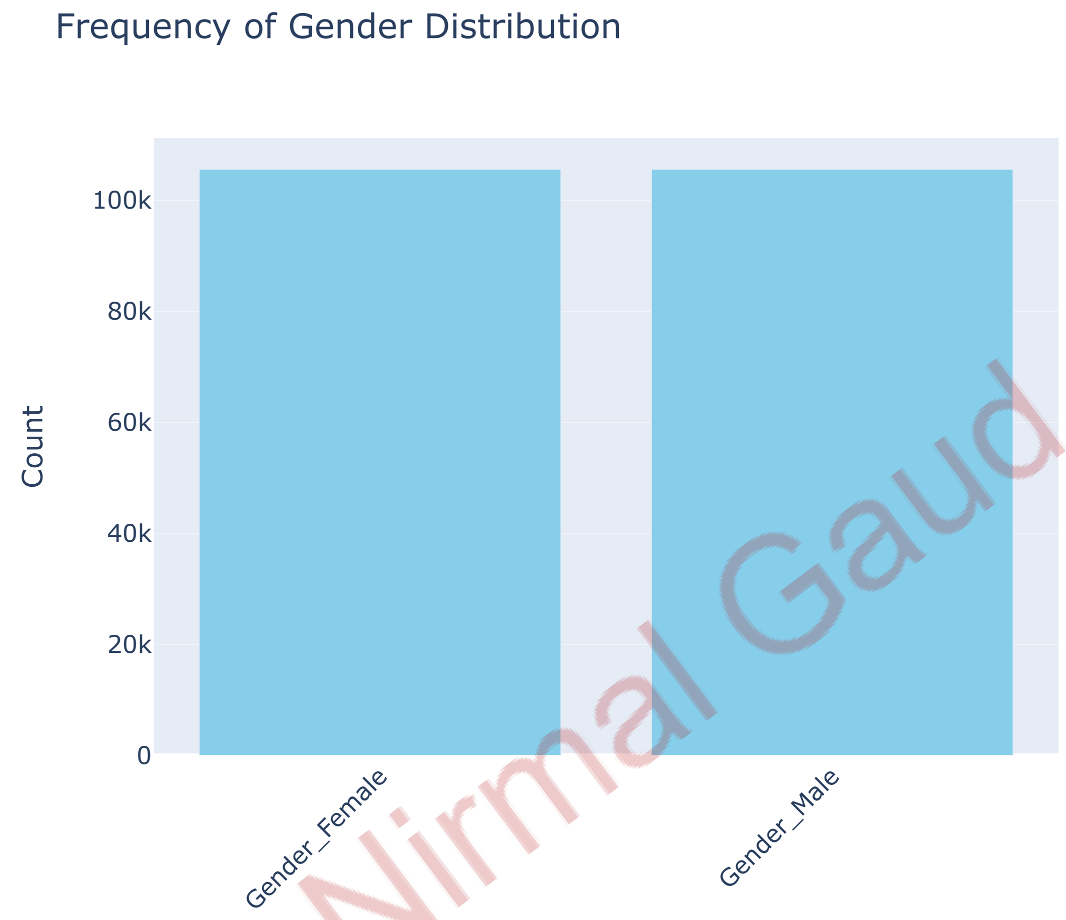
In [48]:

```
fig, ax = plt.subplots(figsize=(8, 8))
ax.pie(gender_distribution, labels=gender_distribution.index, autopct='%1.1f%%',
       startangle=90,
       colors=plt.cm.Paired(range(len(age_distribution))))
ax.set_title('Frequency of Gender Distribution')
plt.axis('equal')
plt.tight_layout()
plt.show()
```



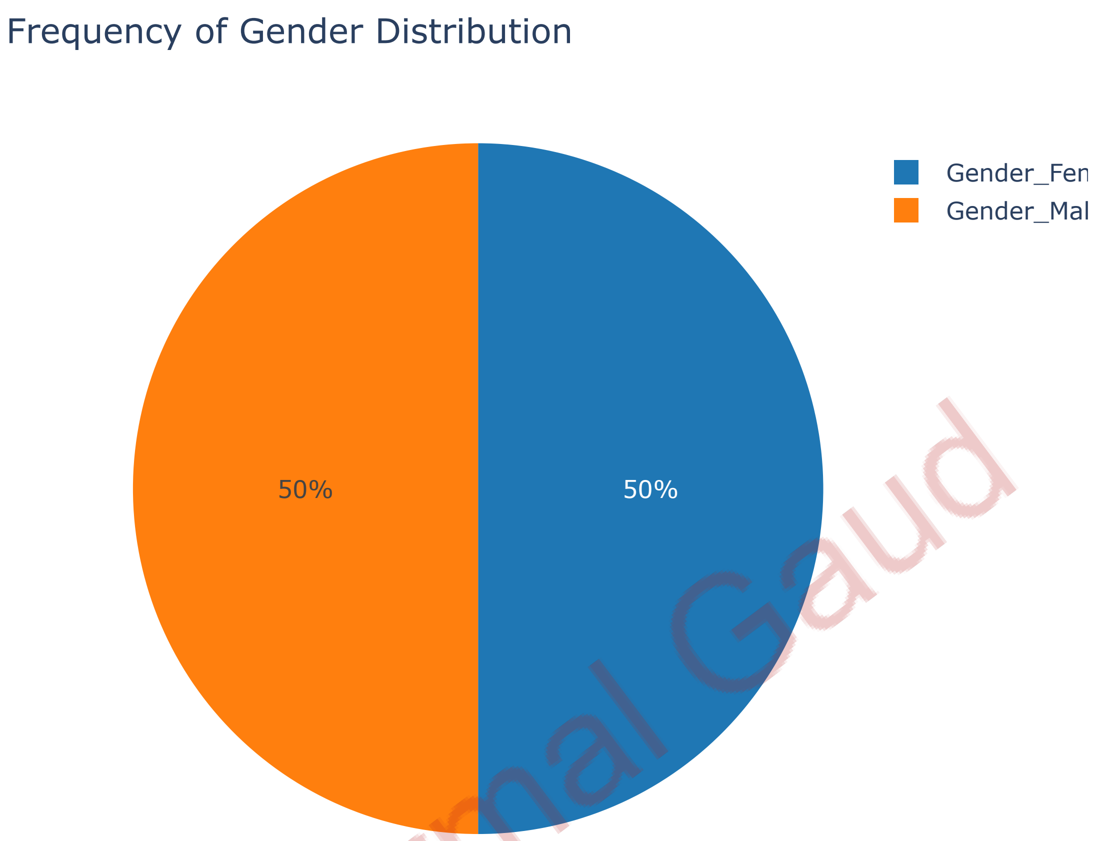
In [49]:

```
fig = px.bar(gender_distribution, x=gender_distribution.index, y=gender_distribution.val
             labels={'x': 'Gender Distribution', 'y': 'Count'},
             title='Frequency of Gender Distribution', color_discrete_sequence=['skyblue
fig.update_layout(xaxis_tickangle=-45)
fig.show()
```



In [50]:

```
custom_colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e37
fig = px.pie(gender_distribution, values=gender_distribution.values, names=gender_distri
             title='Frequency of Gender Distribution', color_discrete_sequence=custom_co
fig.show()
```



In [51]:

```
corr = df[symptoms].corr()
```

In [52]:

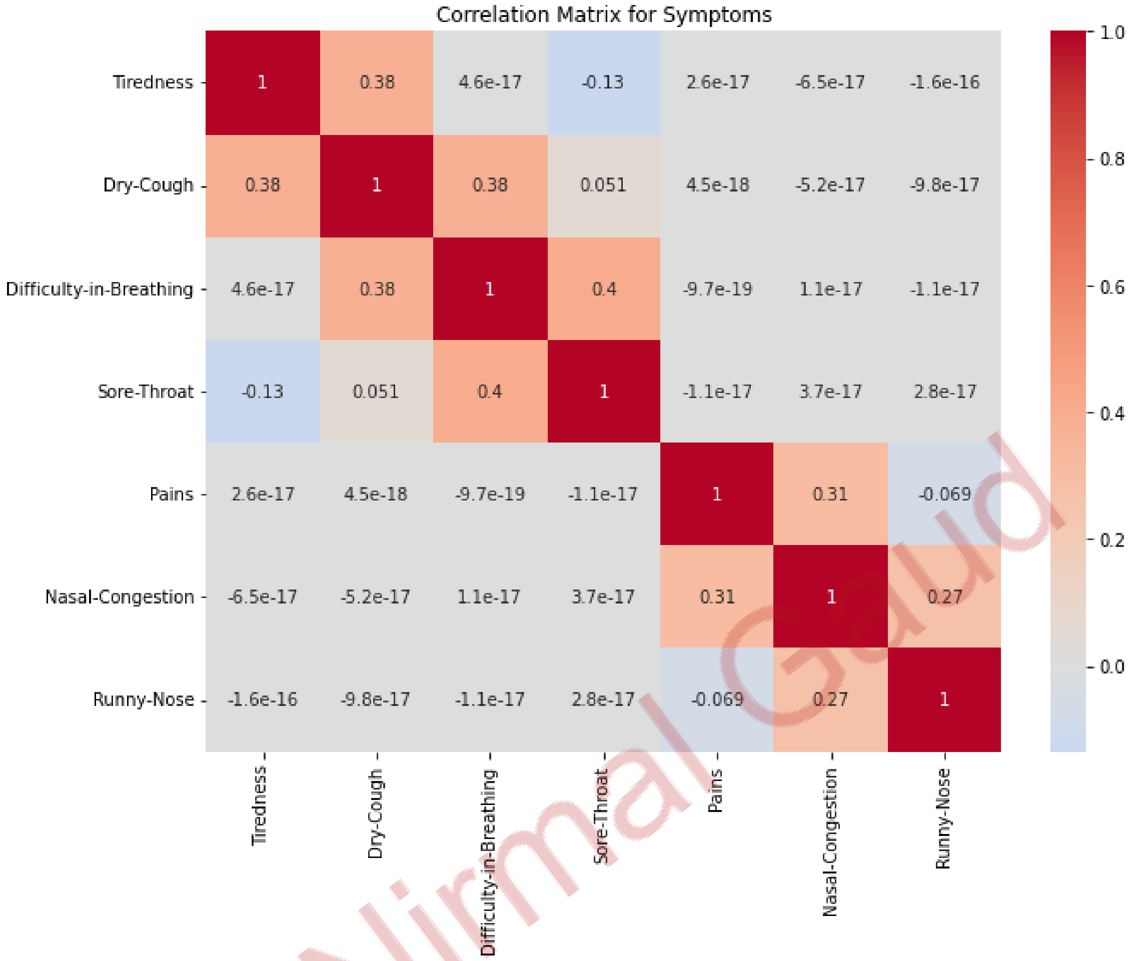
```
corr
```

Out[52]:

	Tiredness	Dry-Cough	Difficulty-in-Breathing	Sore-Throat	Pains	N Conge
Tiredness	1.000000e+00	3.779645e-01	4.629153e-17	-1.348400e-01	2.612420e-17	-6.526
Dry-Cough	3.779645e-01	1.000000e+00	3.779645e-01	5.096472e-02	4.519459e-18	-5.180
Difficulty-in-Breathing	4.629153e-17	3.779645e-01	1.000000e+00	4.045199e-01	-9.663757e-19	1.07676
Sore-Throat	-1.348400e-01	5.096472e-02	4.045199e-01	1.000000e+00	-1.063529e-17	3.70577
Pains	2.612420e-17	4.519459e-18	-9.663757e-19	-1.063529e-17	1.000000e+00	3.10529
Nasal-Congestion	-6.526807e-17	-5.180861e-17	1.076763e-17	3.705775e-17	3.105295e-01	1.00000
Runny-Nose	-1.561936e-16	-9.767619e-17	-1.091121e-17	2.844633e-17	-6.900656e-02	2.66666

In [53]:

```
fig, ax = plt.subplots(figsize=(10, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm', center=0, ax=ax)
ax.set_title('Correlation Matrix for Symptoms')
plt.tight_layout()
plt.show()
```



In [54]:

```
severity_symptom_means = df.groupby(by=[col for col in df.columns if "Asthma_Severity" i
```

In [55]:

```
severity_symptom_means
```

Out[55]:

	Tiredness	Dry-Cough	Difficulty-in-Breathing	Sore-Throat	Pains	Nasal-Congestion	Runny-Nose
Asthma_Severity							
0	0.5	0.5625	0.5	0.3125	0.363636	0.545455	0.545455
1	0.5	0.5625	0.5	0.3125	0.363636	0.545455	0.545455

In [56]:

```
severity_symptom_means = severity_symptom_means.transpose()
```

In [57]:

```
severity_symptom_means
```

Out[57]:

Asthma_Severity	0	1
Tiredness	0.500000	0.500000
Dry-Cough	0.562500	0.562500
Difficulty-in-Breathing	0.500000	0.500000
Sore-Throat	0.312500	0.312500
Pains	0.363636	0.363636
Nasal-Congestion	0.545455	0.545455
Runny-Nose	0.545455	0.545455

In [58]:

```
data = {
    'Asthma_Severity': [0, 1, 2],
    'Tiredness': [0.500000, 0.500000, 0.500000],
    'Dry-Cough': [0.562500, 0.562500, 0.562500],
    'Difficulty-in-Breathing': [0.500000, 0.500000, 0.500000],
    'Sore-Throat': [0.312500, 0.312500, 0.312500],
    'Pains': [0.363636, 0.363636, 0.363636],
    'Nasal-Congestion': [0.545455, 0.545455, 0.545455],
    'Runny-Nose': [0.545455, 0.545455, 0.545455]
}
```

In [59]:

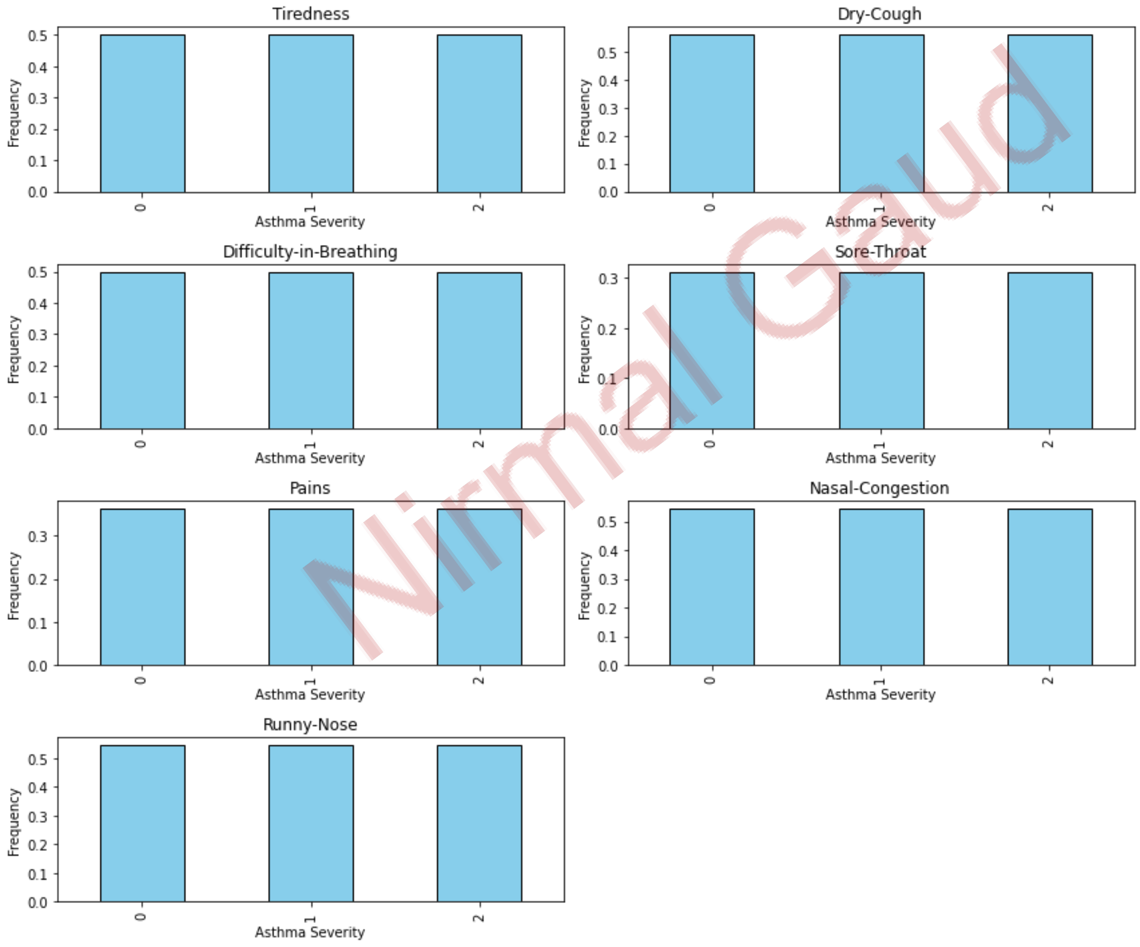
```
df1 = pd.DataFrame(data)
```

In [60]:

```
df1.set_index('Asthma_Severity', inplace=True)
```

In [61]:

```
fig, axes = plt.subplots(nrows=4, ncols=2, figsize=(12, 10))
symptoms = ['Tiredness', 'Dry-Cough', 'Difficulty-in-Breathing', 'Sore-Throat', 'Pains',

for i, symptom in enumerate(symptoms):
    row, col = divmod(i, 2)
    df1[symptom].plot(kind='bar', ax=axes[row, col], color='skyblue', edgecolor='black')
    axes[row, col].set_title(symptom)
    axes[row, col].set_xlabel('Asthma Severity')
    axes[row, col].set_ylabel('Frequency')

if len(symptoms) < len(axes.flat):
    for i in range(len(symptoms), len(axes.flat)):
        fig.delaxes(axes.flatten()[i])

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



In [62]:

```
df
```

Out[62]:

	Tiredness	Dry-Cough	Difficulty-in-Breathing	Sore-Throat	None_Sympton	Pains	Nasal-Congestion	Runny-Nose	N
0	1	1	1	1	0	1	1	1	
1	1	1	1	1	0	1	1	1	
2	1	1	1	1	0	1	1	1	
3	1	1	1	1	0	1	1	1	
4	1	1	1	1	0	1	1	1	
...	...	...	...	...	...	...	...	...	
316795	0	0	0	0	1	0	0	0	
316796	0	0	0	0	1	0	0	0	
316797	0	0	0	0	1	0	0	0	
316798	0	0	0	0	1	0	0	0	
316799	0	0	0	0	1	0	0	0	

316800 rows × 17 columns

In [63]:

```
X = df.drop(['Asthma_Severity'], axis = 1)
y = df['Asthma_Severity']
```

In [64]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [65]:

```
clf = RandomForestClassifier(n_estimators=100, random_state=42)
```

In [66]:

```
clf.fit(X, y)
```

Out[66]:

```
▾         RandomForestClassifier
RandomForestClassifier(random_state=42)
```

In [67]:

```
feature_importances = clf.feature_importances_
```

In [68]:

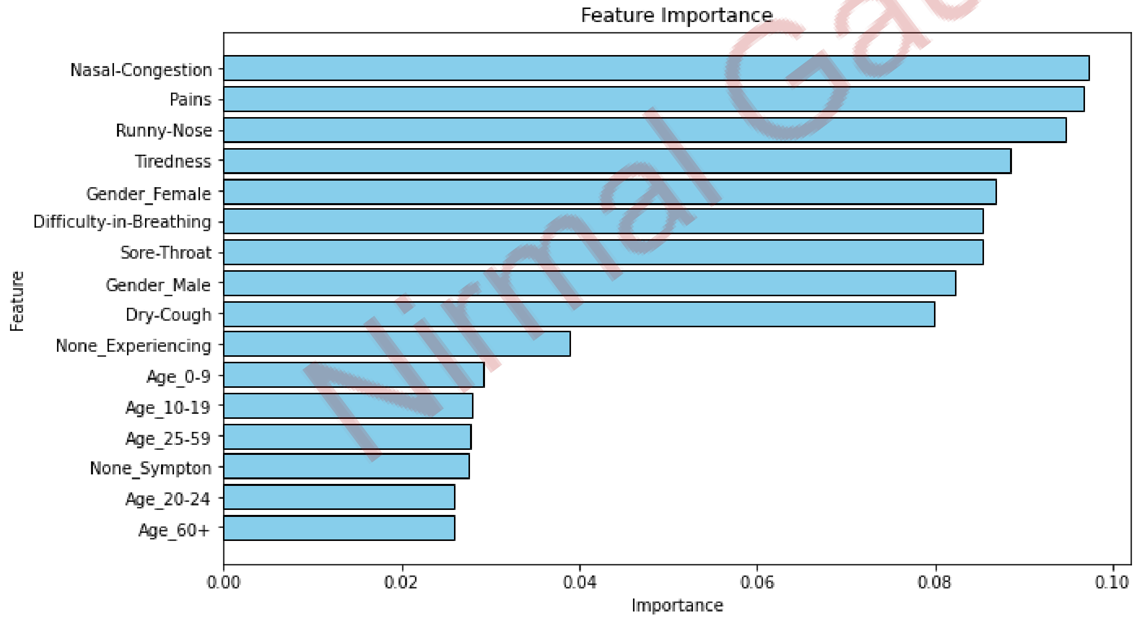
```
feature_importance_df = pd.DataFrame({'Feature': X.columns, 'Importance': feature_import
```

In [69]:

```
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=Fal
```

In [70]:

```
plt.figure(figsize=(10, 6))
plt.barh(feature_importance_df['Feature'], feature_importance_df['Importance'], color='s
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.title('Feature Importance')
plt.gca().invert_yaxis()
```



In [71]:

```
X = df[symptoms + gender_groups]
```

In [72]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
```

In [73]:

```
X_train,X_test,y_train,y_test=train_test_split(X,
                                               y,
                                               test_size=0.2,
                                               stratify = y,
                                               random_state=42)
```

In [74]:

```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [75]:

```
lr = LogisticRegression(random_state=42)
```

In [76]:

```
lr.fit(X_train, y_train)
```

Out[76]:

```
▾         LogisticRegression
LogisticRegression(random_state=42)
```

In [77]:

```
y_pred_lr = lr.predict(X_test)
```

In [78]:

```
accuracy = accuracy_score(y_test, y_pred_lr)
print(f'Accuracy: {accuracy:.2f}')
```

```
Accuracy: 0.75
```

In [79]:

```
dt = DecisionTreeClassifier()
```

In [80]:

```
model_dt = dt.fit(X_train, y_train)
```

In [81]:

```
y_pred_dt = model_dt.predict(X_test)
```

In [82]:

```
accuracy = accuracy_score(y_test, y_pred_dt)
print(f'Accuracy: {accuracy:.2f}')
```

```
Accuracy: 0.75
```

In [83]:

```
import xgboost as xgb
```

In [84]:

```
clf = xgb.XGBClassifier(
    objective='multi:softmax',
    num_class=len(pd.unique(y)),
    eval_metric='mlogloss',
    use_label_encoder=False
)
```

In [85]:

```
clf.fit(X_train, y_train)
```

Out[85]:

```
▾                                XGBClassifier
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytre
e=1,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric='mlogloss', gamma=0, gpu_id=-1,
              grow_policy='depthwise', importance_type=None,
              interaction_constraints='', learning_rate=0.300000012,
              max_bin=256, max_cat_to_onehot=4, max_delta_step=0, max_
depth=6,
              max_leaves=0, min_child_weight=1, missing=nan,
```

In [86]:

```
y_pred_xgb = clf.predict(X_test)
```

In [87]:

```
accuracy = accuracy_score(y_test, y_pred_xgb)
print(f'Accuracy: {accuracy:.2f}')
```

```
Accuracy: 0.75
```

Thanks !!!