Title : Mosque Accessibility and Utilization Enhancement

Problem : Many mosques face challenges related to attendance, accessibility, and facilities utilization. Understanding these patterns and optimizing mosque operations can improve engagement, better allocate resources, and enhance the overall experience for the community.

Goal : Will use data science techniques to adress the challenges so that create a more welcoming environment for the community.

```
!pip install dash
```

```
Collecting dash
  Downloading dash-2.12.0-py3-none-any.whl (10.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.4/10.4 MB 22.5 MB/s eta 0:00:00
Requirement already satisfied: Flask<2.3.0,>=1.0.4 in /usr/local/lib/python3.10/dist-packages (from dash) (2.2.5)
Collecting Werkzeug<2.3.0 (from dash)
  Downloading Werkzeug-2.2.3-py3-none-any.whl (233 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 233.6/233.6 kB 20.7 MB/s eta 0:00:00
Requirement already satisfied: plotly>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (5.15.0)
Collecting dash-html-components==2.0.0 (from dash)
  Downloading dash_html_components-2.0.0-py3-none-any.whl (4.1 kB)
Collecting dash-core-components==2.0.0 (from dash)
  Downloading dash_core_components-2.0.0-py3-none-any.whl (3.8 kB)
Collecting dash-table==5.0.0 (from dash)
  Downloading dash_table-5.0.0-py3-none-any.whl (3.9 kB)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.10/dist-packages (from dash) (4.7.1)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from dash) (2.31.0)
Collecting retrying (from dash)
  Downloading retrying-1.3.4-py3-none-any.whl (11 kB)
Collecting ansi2html (from dash)
  Downloading ansi2html-1.8.0-py3-none-any.whl (16 kB)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.10/dist-packages (from dash) (1.5.7)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from dash) (67.7.2)
Requirement already satisfied: Jinja2>=3.0 in /usr/local/lib/python3.10/dist-packages (from Flask<2.3.0,>=1.0.4->dash) (3.1.2)
Requirement already satisfied: itsdangerous>=2.0 in /usr/local/lib/python3.10/dist-packages (from Flask<2.3.0,>=1.0.4->dash) (2.1.2)
Requirement already satisfied: click>=8.0 in /usr/local/lib/python3.10/dist-packages (from Flask<2.3.0,>=1.0.4->dash) (8.1.6)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash) (8.2.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash) (23.1)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from Werkzeug<2.3.0->dash) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (3.2.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (2023.7.22)
Requirement already satisfied: six>=1.7.0 in /usr/local/lib/python3.10/dist-packages (from retrying->dash) (1.16.0)
Installing collected packages: dash-table, dash-html-components, dash-core-components, Werkzeug, retrying, ansi2html, dash
  Attempting uninstall: Werkzeug
    Found existing installation: Werkzeug 2.3.6
    Uninstalling Werkzeug-2.3.6:
      Successfully uninstalled Werkzeug-2.3.6
Successfully installed Werkzeug-2.2.3 ansi2html-1.8.0 dash-2.12.0 dash-core-components-2.0.0 dash-html-components-2.0.0 dash-table-5.0.0
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import dash
from dash import dcc, html
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
```

```
np.random.seed(42)

num_attendance_records = 500

attendance_data = []

for i in range(num_attendance_records):
    attendance_data.append({
        'Date': pd.Timestamp(np.random.choice(pd.date_range(start='2023-01-01', end='2023-12-31'))),
        'Prayer_Time': np.random.choice(['Fajr', 'Dhuhr', 'Asr', 'Maghrib', 'Isha'] , p=[0.1 , 0.2 , 0.3 , 0.2 , 0.2]),
         'Gender': np.random.choice(['Male', 'Female']) ,
        'Age_Group': np.random.choice(['Youth', 'Adult', 'Elderly']),
        'Transport_Mode': np.random.choice(['Car', 'Public Transport', 'Walking', 'Bicycle']),
        'Facility_Usage': np.random.choice(['Prayer Hall', 'Classroom', 'Meeting Room', 'Library'])
    })
```

```
attendance_df = pd.DataFrame(attendance_data)

attendance_df.to_csv('mosque_attendance_data.csv', index=False)


data = pd.read_csv("/content/mosque_attendance_data.csv")


data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Date            500 non-null    object
 1   Prayer_Time     500 non-null    object
 2   Gender          500 non-null    object
 3   Age_Group       500 non-null    object
 4   Transport_Mode  500 non-null    object
 5   Facility_Usage  500 non-null    object
dtypes: object(6)
memory usage: 23.6+ KB
```

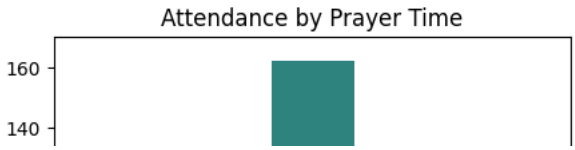
```
data.isna().sum()
```

```
Date              0
Prayer_Time       0
Gender            0
Age_Group         0
Transport_Mode    0
Facility_Usage    0
dtype: int64
```

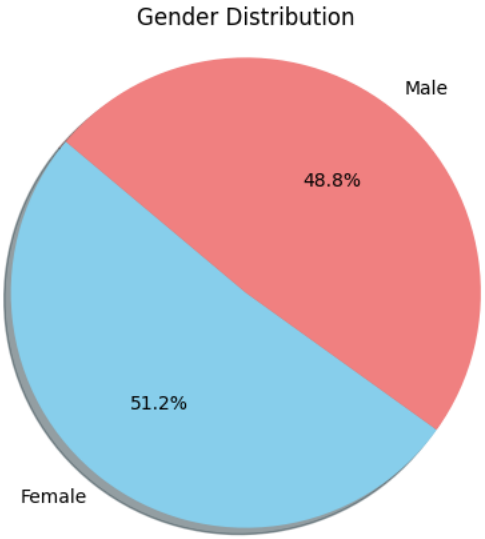
```
#check for duplicates
duplicates = data[data.duplicated(keep=False)]
if not duplicates.empty:
  print("Duplicate rows found:")
  print(duplicates)
else:
  print("No duplicates found.")
```

```
Duplicate rows found:
           Date Prayer_Time  Gender Age_Group Transport_Mode Facility_Usage
153  2023-07-14         Asr  Female     Adult            Car      Classroom
361  2023-07-14         Asr  Female     Adult            Car      Classroom
```

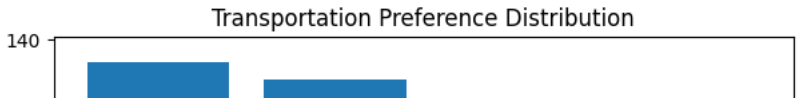
```
data = data.drop_duplicates()


# Visualize attendance patterns by prayer time
plt.figure(figsize=(5, 5))
sns.countplot(x='Prayer_Time', data=attendance_df, palette='viridis')
plt.title('Attendance by Prayer Time')
plt.xlabel('Prayer Time')
plt.ylabel('Attendance Count')
plt.show()
```



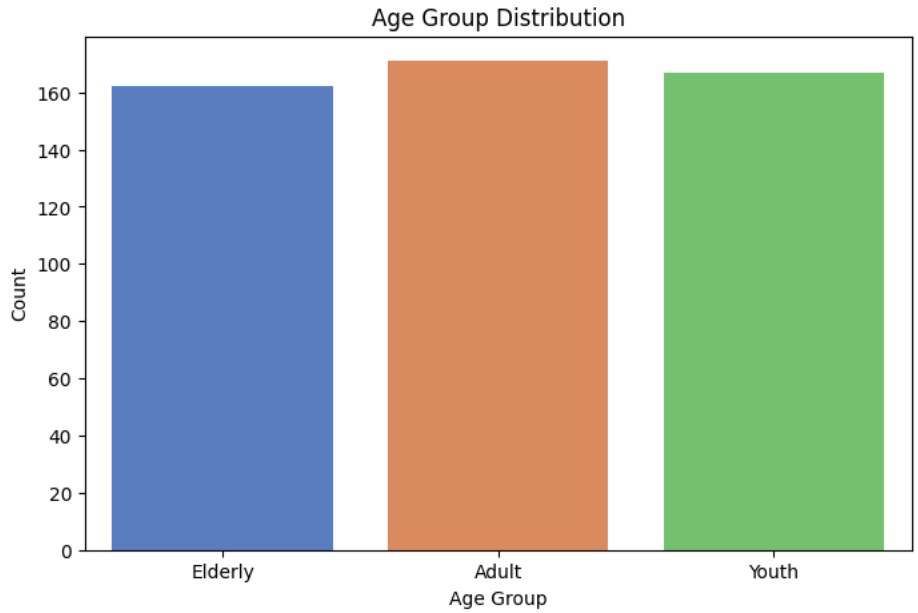

```
gender_counts = attendance_df['Gender'].value_counts()
labels = gender_counts.index
sizes = gender_counts.values
colors = ['skyblue', 'lightcoral']

plt.figure(figsize=(5, 5))
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=140)
plt.title('Gender Distribution')
plt.axis('equal')
plt.show()
```




```
transportation_counts = data['Transport_Mode'].value_counts()

# Create a bar graph
plt.bar(transportation_counts.index, transportation_counts.values)

# Add labels and title
plt.xlabel('Transportation Types')
plt.ylabel('Number of Attendees')
plt.title('Transportation Preference Distribution')

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




```
# Visualize age group distribution
plt.figure(figsize=(8, 5))
sns.countplot(x='Age_Group', data=attendance_df, palette='muted')
plt.title('Age Group Distribution')
plt.xlabel('Age Group')
plt.ylabel('Count')
plt.show()
```



Now i will make a dash app which will allow mosque administrators to monitor attendance patterns and demographic insights in real time.

```
# Initialize the Dash app
app = dash.Dash(__name__)

# Define the layout of the dashboard
app.layout = html.Div([
    html.H1("Mosque Attendance Dashboard"),

    dcc.Graph(
        id='prayer-time-attendance',
        figure=px.bar(
            attendance_df,
            x='Prayer_Time',
            color='Gender',
            barmode='group',
            title='Attendance by Prayer Time'
        )
    ),

    dcc.Graph(
        id='age-group-distribution',
        figure=px.pie(
            attendance_df,
            names='Age_Group',
            title='Age Group Distribution'
        )
    )
])

# Run the app
if __name__ == '__main__':
    app.run_server(debug=True)
```

Attendance Prediction Model

```
# Prepare features and target variable
X = attendance_df[['Gender', 'Age_Group']]
y = attendance_df['Prayer_Time']

# Encode categorical variables
X = pd.get_dummies(X, columns=['Gender', 'Age_Group'], drop_first=True)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Define hyperparameters for Random Forest
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Initialize the model
rf_model = RandomForestClassifier(random_state=42)

# Perform grid search for hyperparameter tuning
grid_search = GridSearchCV(estimator=rf_model, param_grid=param_grid, cv=3, scoring='accuracy', n_jobs=-1)
grid_search.fit(X_train_scaled, y_train)

# Get the best parameters from grid search
best_params = grid_search.best_params_
print("Best Parameters:", best_params)

# Evaluate the model on the test set
y_pred = grid_search.predict(X_test_scaled)
classification_rep = classification_report(y_test, y_pred)
print("Classification Report:\n", classification_rep)
```

```
Best Parameters: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 50}
Classification Report:
               precision    recall  f1-score   support

          Asr       0.40      0.72      0.51        39
        Dhuhr       0.00      0.00      0.00        17
         Fajr       0.00      0.00      0.00        16
         Isha       0.20      0.40      0.27        15
      Maghrib       0.00      0.00      0.00        13

     accuracy                           0.34       100
    macro avg       0.12      0.22      0.16       100
 weighted avg       0.19      0.34      0.24       100
```

Documentation

1. Data Collection I genetrated random data to get mosque attendance and transportation preferences. I then used data to demonstrate the functionality of the solution.
2. Data Preprocessing Data preprocessing included checking for null values and duplicate records. No null values were found, and duplicate records were removed to ensure data integrity.
3. Data Analysis and Visualization Various data analysis methods were employed to gain insights into the generated data. Visualization techniques were used to present findings, including:
    - Gender distribution among attendees
    - Age group distribution of attendees
    - Transportation Preference Distribution
    - Attendance patterns by prayer time
4. Dashboard Development: A real-time dashboard was developed to provide mosque administrators with insights into attendance patterns and demographic distributions. The dashboard includes interactive graphs:

    - Prayer time attendance graph
    - Age group distribution pie chart
5. Attendance Prediction Model: An attendance prediction model was built to predict attendance based on gender and age group. Due to the random nature of the data, the model's accuracy was not optimal, highlighting the need for more representative data.
6. Challenges Faced: Challenges encountered during the project include:

- Dealing with low accuracy of the attendance prediction model
- Generating meaningful random data

✓ 0s completed at 1:09 PM