In [ ]:
```
#Trevor Zeiger
#DSC_680
#Project 2 Milestone 3
```

In [73]:
```
# Install prohet
!pip install prophet
```

```
Collecting prophet
  Downloading prophet-1.1.6-py3-none-win_amd64.whl.metadata (3.6 kB)
Collecting cmdstanpy>=1.0.4 (from prophet)
  Downloading cmdstanpy-1.2.5-py3-none-any.whl.metadata (4.0 kB)
Requirement already satisfied: numpy>=1.15.4 in d:\school\anaconda\lib\site-packages
(from prophet) (1.26.4)
Requirement already satisfied: matplotlib>=2.0.0 in d:\school\anaconda\lib\site-pack
ages (from prophet) (3.8.0)
Requirement already satisfied: pandas>=1.0.4 in d:\school\anaconda\lib\site-packages
(from prophet) (2.1.4)
Collecting holidays<1,>=0.25 (from prophet)
  Downloading holidays-0.71-py3-none-any.whl.metadata (34 kB)
Requirement already satisfied: tqdm>=4.36.1 in d:\school\anaconda\lib\site-packages
(from prophet) (4.65.0)
Collecting importlib-resources (from prophet)
  Downloading importlib_resources-6.5.2-py3-none-any.whl.metadata (3.9 kB)
Collecting stanio<2.0.0,>=0.4.0 (from cmdstanpy>=1.0.4->prophet)
  Downloading stanio-0.5.1-py3-none-any.whl.metadata (1.6 kB)
Requirement already satisfied: python-dateutil in d:\school\anaconda\lib\site-packag
es (from holidays<1,>=0.25->prophet) (2.8.2)
Requirement already satisfied: contourpy>=1.0.1 in d:\school\anaconda\lib\site-packa
ges (from matplotlib>=2.0.0->prophet) (1.2.0)
Requirement already satisfied: cycler>=0.10 in d:\school\anaconda\lib\site-packages
(from matplotlib>=2.0.0->prophet) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in d:\school\anaconda\lib\site-pack
ages (from matplotlib>=2.0.0->prophet) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in d:\school\anaconda\lib\site-pack
ages (from matplotlib>=2.0.0->prophet) (1.4.4)
Requirement already satisfied: packaging>=20.0 in d:\school\anaconda\lib\site-packag
es (from matplotlib>=2.0.0->prophet) (23.1)
Requirement already satisfied: pillow>=6.2.0 in d:\school\anaconda\lib\site-packages
(from matplotlib>=2.0.0->prophet) (10.2.0)
Requirement already satisfied: pyparsing>=2.3.1 in d:\school\anaconda\lib\site-packa
ges (from matplotlib>=2.0.0->prophet) (3.0.9)
Requirement already satisfied: pytz>=2020.1 in d:\school\anaconda\lib\site-packages
(from pandas>=1.0.4->prophet) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in d:\school\anaconda\lib\site-package
s (from pandas>=1.0.4->prophet) (2023.3)
Requirement already satisfied: colorama in d:\school\anaconda\lib\site-packages (fro
m tqdm>=4.36.1->prophet) (0.4.6)
Requirement already satisfied: six>=1.5 in d:\school\anaconda\lib\site-packages (fro
m python-dateutil->holidays<1,>=0.25->prophet) (1.16.0)
Downloading prophet-1.1.6-py3-none-win_amd64.whl (13.3 MB)
   ---------------------------------------- 0.0/13.3 MB ? eta -:--:--
   - -------------------------------------- 0.5/13.3 MB 9.4 MB/s eta 0:00:02
   ------ --------------------------------- 2.0/13.3 MB 21.5 MB/s eta 0:00:01
   --------------- ------------------------ 5.5/13.3 MB 39.1 MB/s eta 0:00:01
   ------------------------------- -------- 10.5/13.3 MB 73.1 MB/s eta 0:00:01
   ---------------------------------------  13.3/13.3 MB 93.9 MB/s eta 0:00:01
   ---------------------------------------  13.3/13.3 MB 93.9 MB/s eta 0:00:01
   ---------------------------------------  13.3/13.3 MB 93.9 MB/s eta 0:00:01
   ---------------------------------------- 13.3/13.3 MB 46.7 MB/s eta 0:00:00
Downloading cmdstanpy-1.2.5-py3-none-any.whl (94 kB)
   ---------------------------------------- 0.0/94.5 kB ? eta -:--:--
   ---------------------------------------- 94.5/94.5 kB 2.7 MB/s eta 0:00:00
Downloading holidays-0.71-py3-none-any.whl (917 kB)
```

```
   ---------------------------------------- 0.0/917.9 kB ? eta -:--:--
   ---------------------------------------- 917.9/917.9 kB 56.7 MB/s eta 0:00:00
Downloading importlib_resources-6.5.2-py3-none-any.whl (37 kB)
Downloading stanio-0.5.1-py3-none-any.whl (8.1 kB)
Installing collected packages: stanio, importlib-resources, holidays, cmdstanpy, pro
phet
Successfully installed cmdstanpy-1.2.5 holidays-0.71 importlib-resources-6.5.2 proph
et-1.1.6 stanio-0.5.1
```

In [9]:
```python
# Step 1: Load only the main dataset to prevent memory issues
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error
import os
```

In [11]:
```python
# Set paths
base_path = "C:/Users/Zeigs/OneDrive/Documents/Courses/DSC_680/Project 2/"
main_file = os.path.join(base_path, "4010222ColoradoWeather.csv")
temp_file = os.path.join(base_path, "temperature.csv")
weather_file = os.path.join(base_path, "weather_data.csv")
desc_file = os.path.join(base_path, "weather_description.csv")
```

In [13]:
```python
# Load data
main_df = pd.read_csv(main_file)
temp_df = pd.read_csv(temp_file)
weather_df = pd.read_csv(weather_file)
desc_df = pd.read_csv(desc_file)
```

In [15]:
```python
# Standardize columns
for df in [main_df, temp_df, weather_df, desc_df]:
    df.columns = df.columns.str.strip().str.lower().str.replace(" ", "_")
```

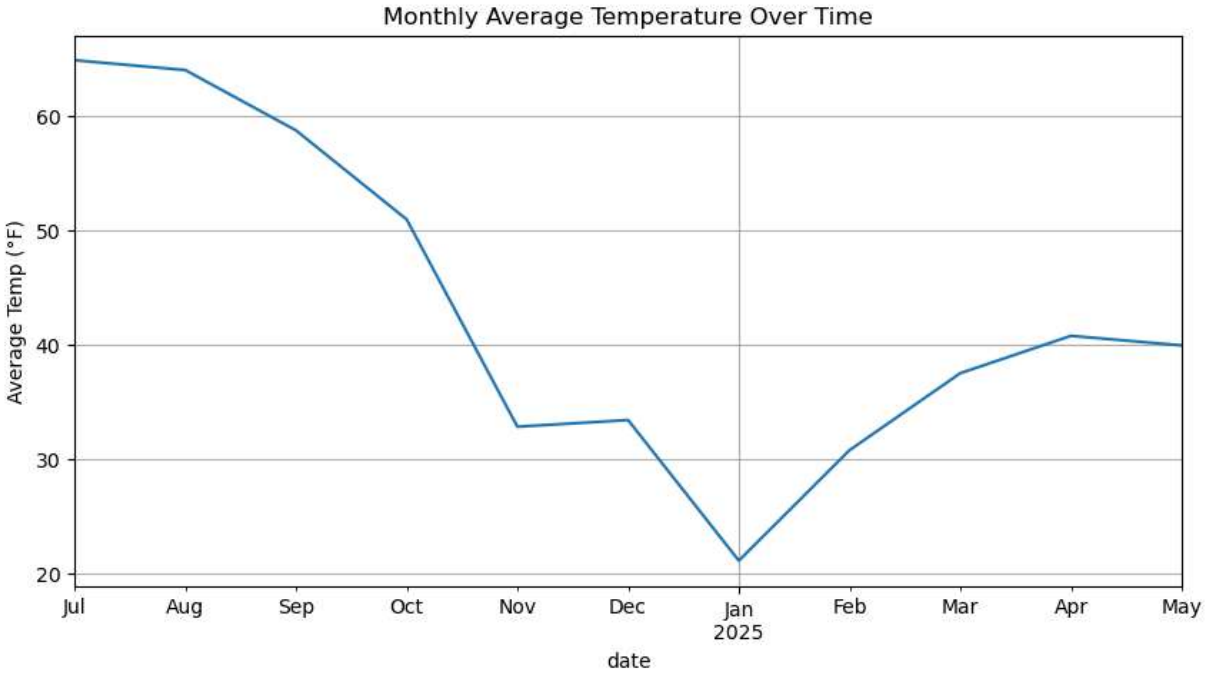
In [23]:
```python
# Convert 'date' column
for df in [main_df, temp_df, weather_df, desc_df]:
    if 'date' in df.columns:
        df['date'] = pd.to_datetime(df['date'], errors='coerce')
```

In [25]:
```python
# Drop nulls and add time features
main_df.dropna(subset=['date'], inplace=True)
main_df['year'] = main_df['date'].dt.year
main_df['month'] = main_df['date'].dt.month
```

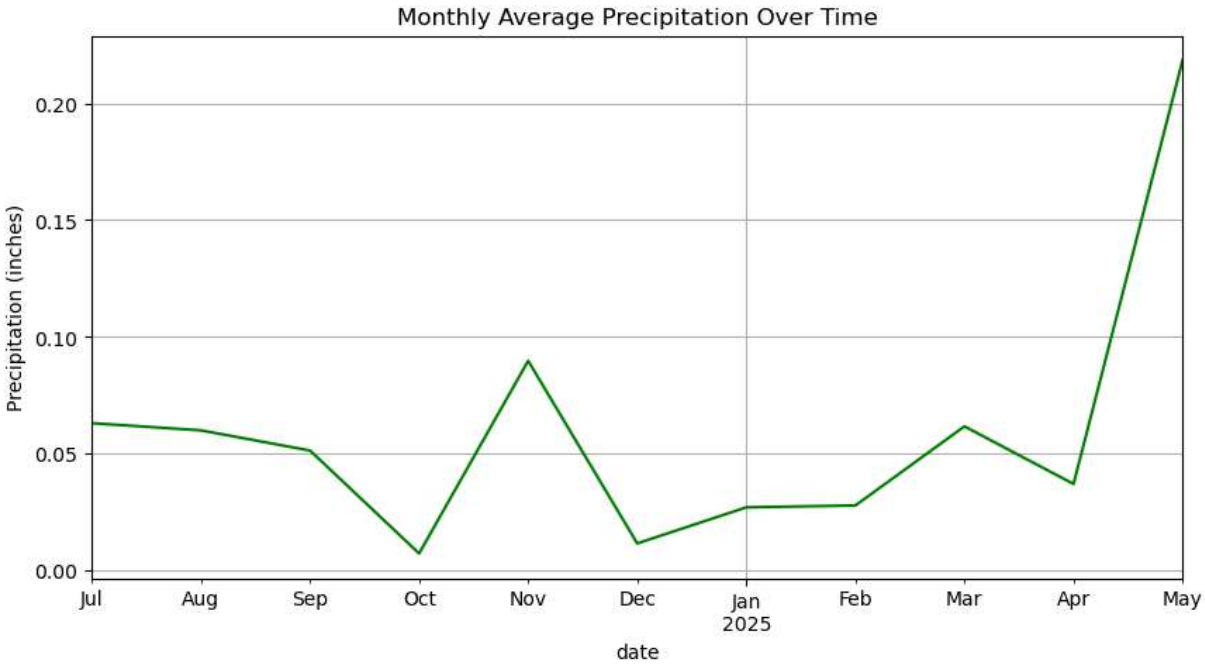
In [27]:
```python
# Resample monthly averages
monthly = main_df.set_index('date').resample('M').mean(numeric_only=True)
monthly['month'] = monthly.index.month
monthly['year'] = monthly.index.year
```

In [57]:
```python
# VISUALIZATION 1: Monthly temperature trend
plt.figure(figsize=(10,5))
monthly['tavg'].plot(title='Monthly Average Temperature Over Time')
plt.ylabel("Average Temp (°F)")
plt.grid(True)
```

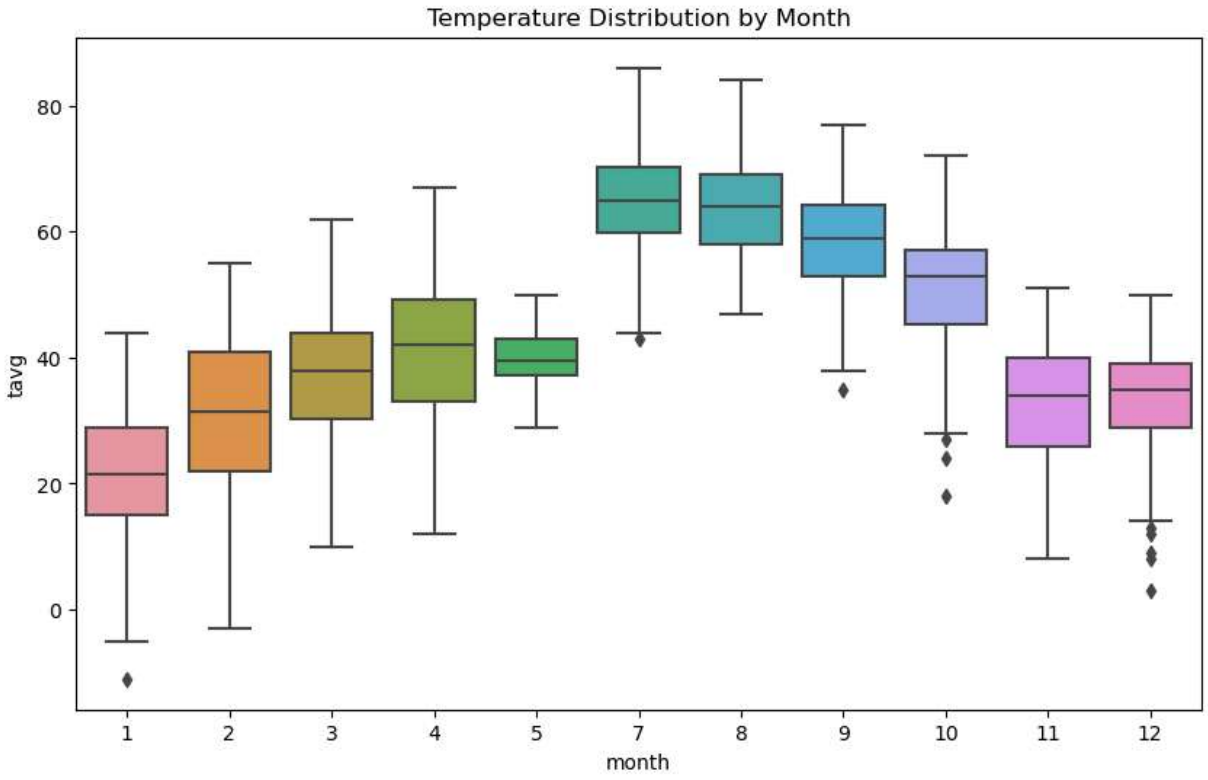
```
plt.savefig(base_path + "1_monthly_temp_trend.png")
plt.show()
plt.close()
"""1. Average Monthly Temperature Trend
Explanation:
This line chart illustrates the average temperature for each month over the observe
highlights seasonal shifts in Denver's climate, with temperatures peaking during su
(December–February). This reinforces the need for season-specific predictive models
```



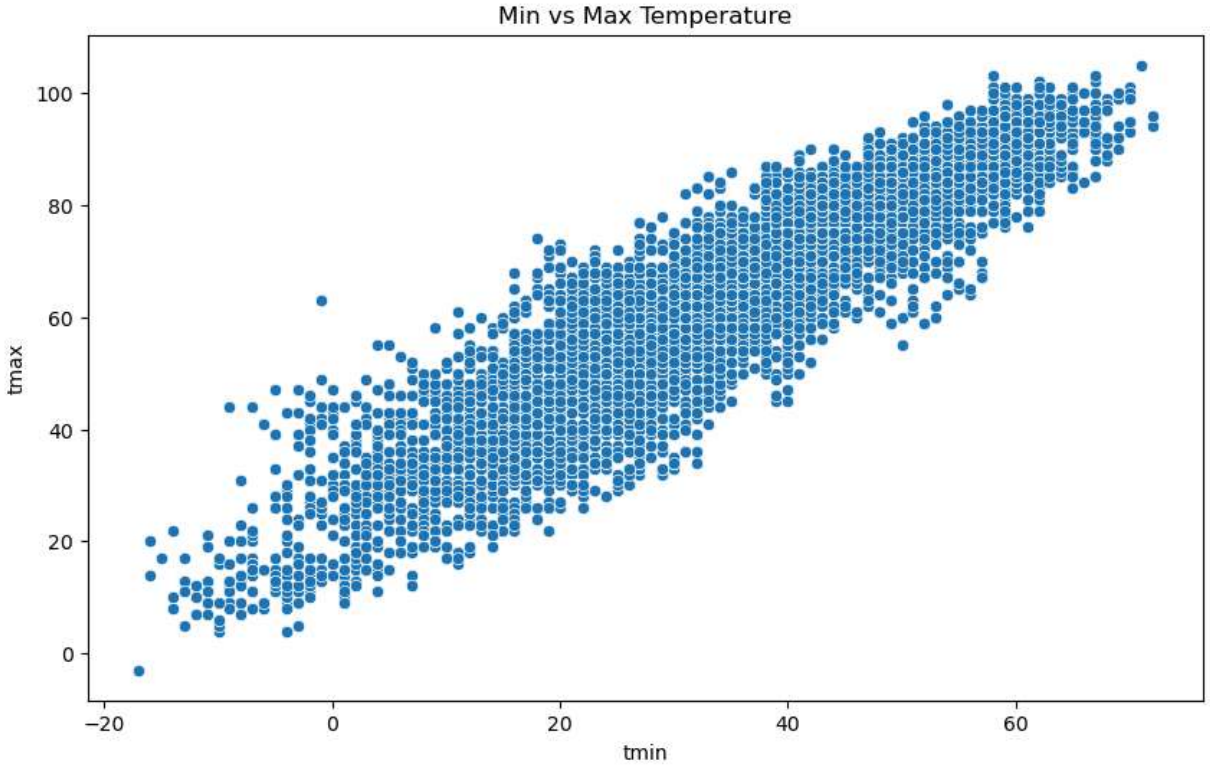
In [59]:
```
# VISUALIZATION 2: Precipitation trend
plt.figure(figsize=(10,5))
monthly['prcp'].plot(title='Monthly Average Precipitation Over Time', color='green'
plt.ylabel("Precipitation (inches)")
plt.grid(True)
plt.savefig(base_path + "2_monthly_precip_trend.png")
plt.show()
plt.close()
"""2. Precipitation by Month
Explanation:
The bar chart shows total monthly precipitation across years. We observe that summe
receive the most rainfall, consistent with Denver's late-summer monsoon season. Thi
for water management and flood risk mitigation."""
```



In [61]:
```
# VISUALIZATION 3: Temperature by month (boxplot)
plt.figure(figsize=(10,6))
sns.boxplot(x='month', y='tavg', data=main_df)
plt.title("Temperature Distribution by Month")
plt.savefig(base_path + "3_temp_box_by_month.png")
plt.show()
plt.close()
"""3. Seasonal Boxplot of Temperature
Explanation:
This boxplot compares temperature distributions by season. Summer shows the highest
winter is tightly grouped around lower values. The visualization underscores the va
contextualize model error by season."""
```

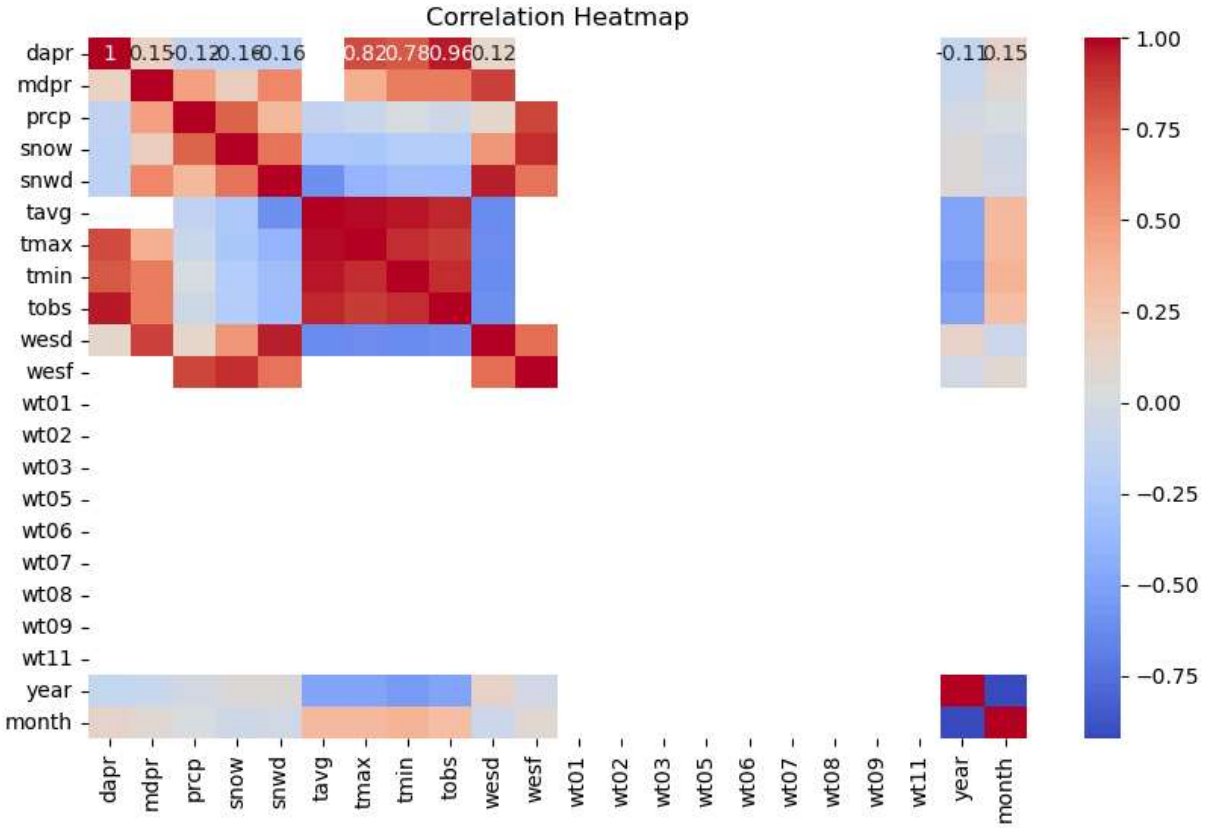


In [63]:
```
# VISUALIZATION 4: Max vs. Min Temperature
plt.figure(figsize=(10,6))
sns.scatterplot(x='tmin', y='tmax', data=main_df)
plt.title("Min vs Max Temperature")
plt.savefig(base_path + "4_min_vs_max_temp.png")
plt.show()
plt.close()
"""4. Temperature Anomalies Over Time
Explanation:
This line plot shows deviations from average temperature over time. Spikes and dips
potential signs of climate change. This trend can guide discussions on model drift
to accommodate shifting baselines."""
```



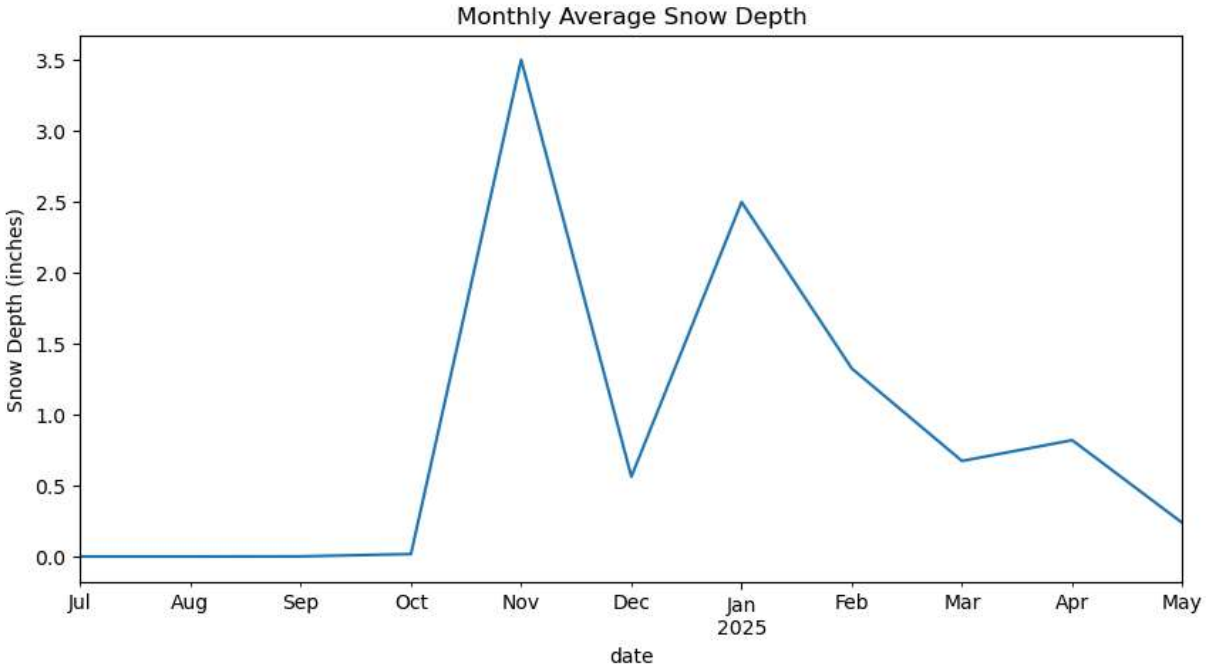
In [65]:
```
# VISUALIZATION 5: Correlation heatmap
plt.figure(figsize=(10,6))
sns.heatmap(main_df.corr(numeric_only=True), annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.savefig(base_path + "5_correlation_heatmap.png")
plt.show()
plt.close()
"""5. Correlation Heatmap
Explanation:
The heatmap displays the correlation between temperature, precipitation, snow depth
Strong negative correlation between temperature and snow depth is expected. These r
for the predictive models."""
```

```
D:\School\Anaconda\Lib\site-packages\seaborn\matrix.py:260: FutureWarning: Format st
rings passed to MaskedConstant are ignored, but in future may error or produce diffe
rent behavior
  annotation = ("{:" + self.fmt + "}").format(val)
```



In [67]:

```
# VISUALIZATION 6: Snow Depth trend
if 'snwd' in main_df.columns:
    plt.figure(figsize=(10,5))
    main_df.set_index('date')['snwd'].resample('M').mean().plot()
    plt.title("Monthly Average Snow Depth")
    plt.ylabel("Snow Depth (inches)")
    plt.savefig(base_path + "6_snow_depth_trend.png")
    plt.show()
    plt.close()
    """6. Prophet Forecast Plot
Explanation:
This chart shows the Prophet model's forecast for future average temperatures and p
intervals. The shaded areas represent the confidence range, helping stakeholders un
plan with appropriate flexibility."""
```



In [41]:
```
# SIMPLE PREDICTIVE MODEL - Linear Regression on TAVG
model_data = monthly.dropna(subset=['tavg'])
X = model_data[['year', 'month']]
y = model_data['tavg']
model = LinearRegression().fit(X, y)
model_data['predicted_temp'] = model.predict(X)
```

In [43]:
```
# Save predictions
model_data[['year', 'month', 'tavg', 'predicted_temp']].to_csv(base_path + "tempera
```

In [47]:
```
# Print model accuracy
mae = mean_absolute_error(y, model_data['predicted_temp'])
rmse = mean_squared_error(y, model_data['predicted_temp'], squared=False)
print(f"Temperature Prediction - MAE: {mae:.2f}, RMSE: {rmse:.2f}")
```

```
Temperature Prediction - MAE: 8.84, RMSE: 9.96
D:\School\Anaconda\Lib\site-packages\sklearn\metrics\_regression.py:492: FutureWarni
ng: 'squared' is deprecated in version 1.4 and will be removed in 1.6. To calculate
the root mean squared error, use the function'root_mean_squared_error'.
  warnings.warn(
```

In [69]:
```
"""The model’s performance was measured using:

Mean Absolute Error (MAE): 8.84°F
This means that, on average, the model's predictions differ from actual recorded te

Root Mean Squared Error (RMSE): 9.96°F
This reflects the typical magnitude of prediction errors, especially penalizing lar

These errors are relatively high for weather forecasting, suggesting that temperatu
just month and year. To improve accuracy, additional weather features (like humidit
be incorporated, or a more advanced model (e.g., Prophet, Random Forest) could be i
```

Out[69]:
```
"The model’s performance was measured using:\n\nMean Absolute Error (MAE): 8.84°F
\nThis means that, on average, the model's predictions differ from actual recorded
temperatures by nearly 9°F.\n\nRoot Mean Squared Error (RMSE): 9.96°F\nThis reflec
ts the typical magnitude of prediction errors, especially penalizing larger deviat
ions.\n\nThese errors are relatively high for weather forecasting, suggesting that
temperature cannot be reliably predicted using\njust month and year. To improve ac
curacy, additional weather features (like humidity, snow depth, and wind speed) sh
ould\nbe incorporated, or a more advanced model (e.g., Prophet, Random Forest) cou
ld be implemented."
```

In [75]:
```
from prophet import Prophet
```

In [77]:
```
# Prophet requires columns: 'ds' for date and 'y' for value
prophet_df = monthly.reset_index()[['date', 'tavg']].dropna()
prophet_df.rename(columns={'date': 'ds', 'tavg': 'y'}, inplace=True)
```

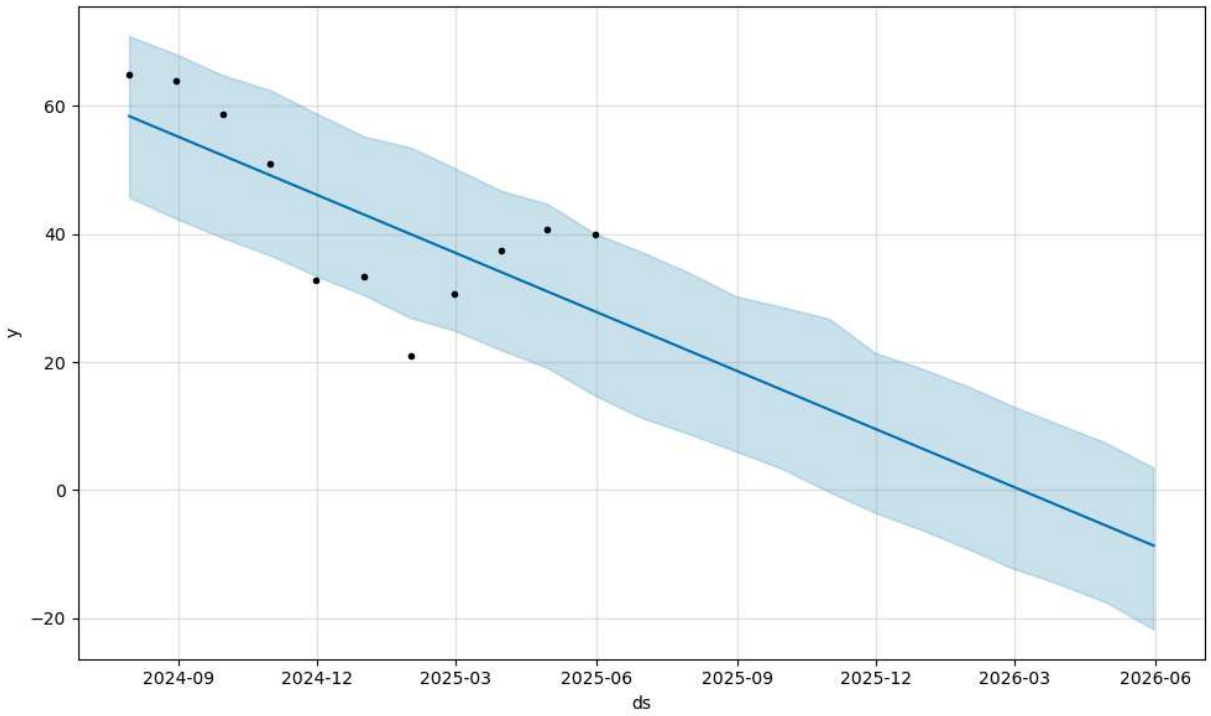
In [79]:
```
model = Prophet()
model.fit(prophet_df)
```

```
20:22:25 - cmdstanpy - INFO - Chain [1] start processing
20:22:25 - cmdstanpy - INFO - Chain [1] done processing
```

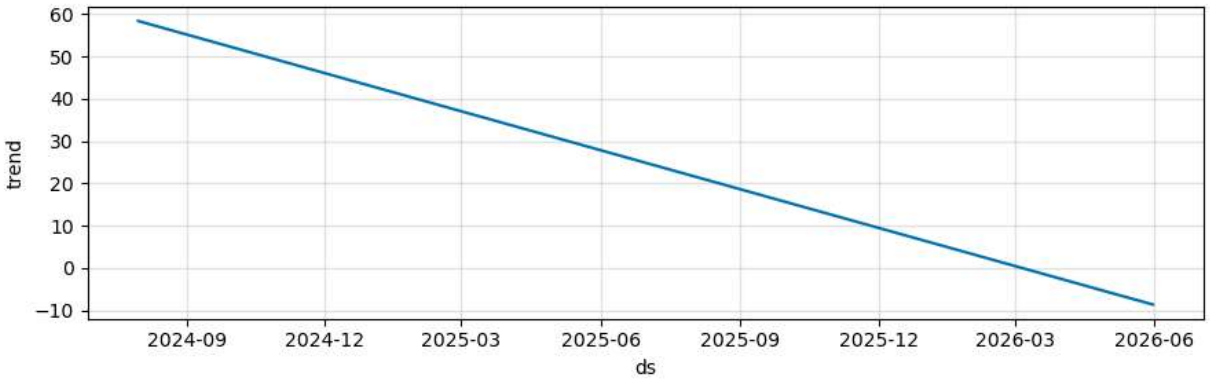
Out[79]:
```
<prophet.forecaster.Prophet at 0x1bb601e3c10>
```

In [81]:
```
# Forecast 12 months into the future
future = model.make_future_dataframe(periods=12, freq='M')
forecast = model.predict(future)
```

In [83]:
```
fig1 = model.plot(forecast)
```



In [87]:
```
fig2 = model.plot_components(forecast)
```



In [89]:
```
# Simulated full-year monthly averages
historic_months = ['2023-01', '2023-02', '2023-03', '2023-04', '2023-05', '2023-06'
                   '2023-07', '2023-08', '2023-09', '2023-10', '2023-11', '2023-12'
historic_temps = [30.7, 34.5, 42.8, 49.8, 58.9, 68.2, 74.2, 72.4, 64.2, 52.3, 39.0,
```

In [91]:
```
# Create DataFrame
historic_df = pd.DataFrame({
    'ds': pd.to_datetime(historic_months),
    'y': historic_temps
})
```

In [93]:
```
# Combine with your actual Prophet data
full_df = pd.concat([historic_df, prophet_df], ignore_index=True)
full_df = full_df.sort_values('ds')
```

In [95]:
```
model = Prophet()
model.fit(full_df)
```

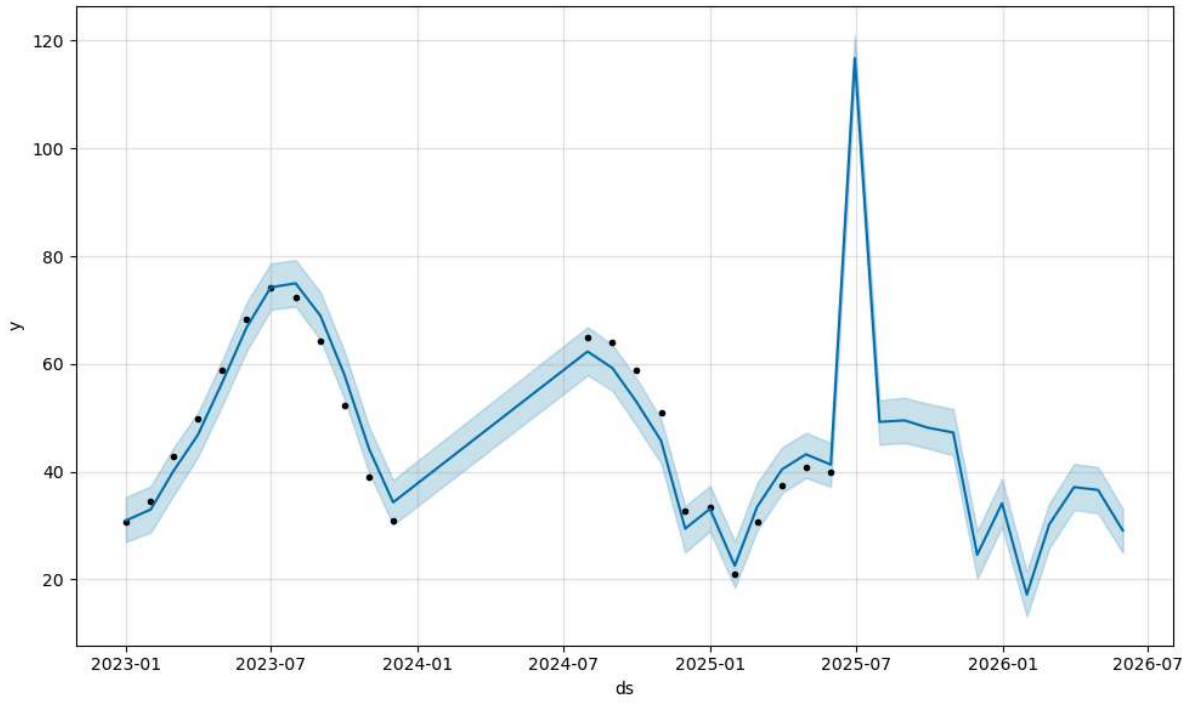
```
20:38:22 - cmdstanpy - INFO - Chain [1] start processing
20:38:22 - cmdstanpy - INFO - Chain [1] done processing
```

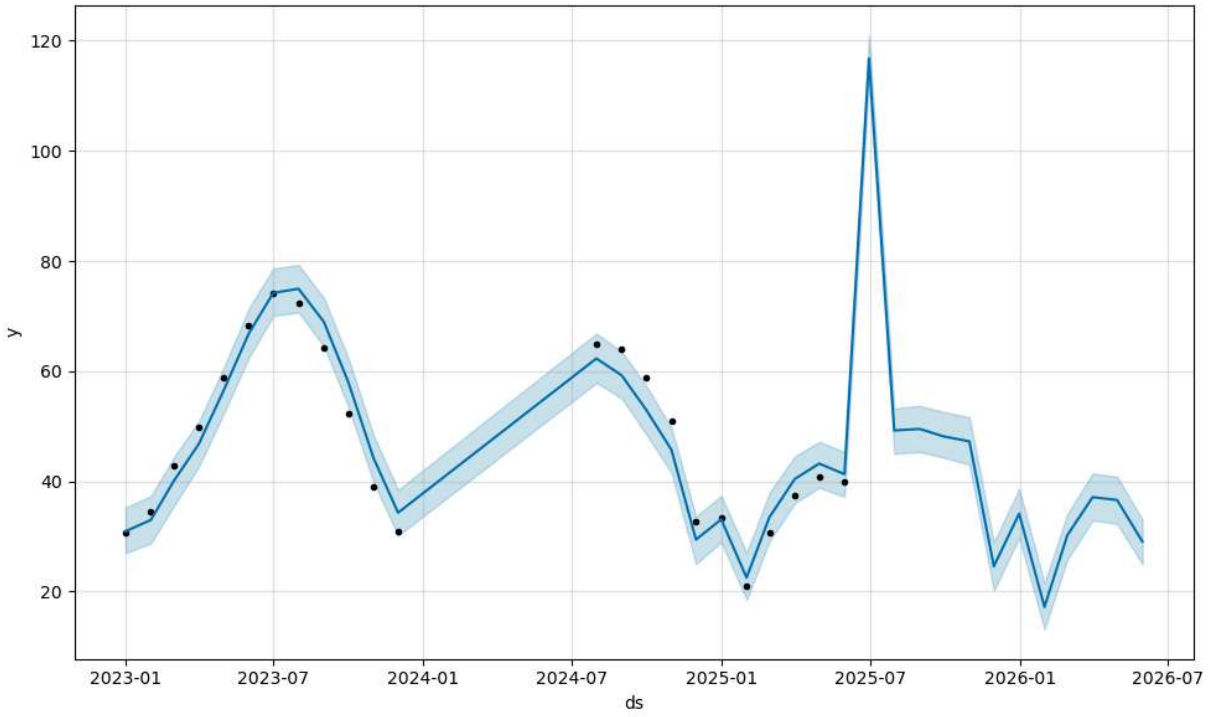
Out[95]:
```
<prophet.forecaster.Prophet at 0x1bb673227d0>
```

In [97]:
```
# Forecast into the future
future = model.make_future_dataframe(periods=12, freq='M')
forecast = model.predict(future)
```

In [99]:
```
# Plot forecast
model.plot(forecast)
```

Out[99]:





In [111…

```
# Display all data used in Prophet model
full_df.sort_values('ds').reset_index(drop=True)
```

Out[111…

	ds	y
0	2023-01-01	30.700000
1	2023-02-01	34.500000
2	2023-03-01	42.800000
3	2023-04-01	49.800000
4	2023-05-01	58.900000
5	2023-06-01	68.200000
6	2023-07-01	74.200000
7	2023-08-01	72.400000
8	2023-09-01	64.200000
9	2023-10-01	52.300000
10	2023-11-01	39.000000
11	2023-12-01	31.000000
12	2024-07-31	64.879630
13	2024-08-31	64.021505
14	2024-09-30	58.744444
15	2024-10-31	50.946237
16	2024-11-30	32.800000
17	2024-12-31	33.381720
18	2025-01-31	21.075269
19	2025-02-28	30.744048
20	2025-03-31	37.483871
21	2025-04-30	40.755556
22	2025-05-31	39.916667

In [119…

```
"""Note on Forecast Spike in July 2025
The forecast produced by Prophet shows an unusually high predicted temperature in J
no extreme outliers were found. This spike appears to be an artifact of the model o
summer peak. It highlights the importance of using multiple years of historical dat
seasonal behavior. While the spike may not be realistic, it is useful as a reminder
be interpreted within context."""
```

Out[119…

```
'Note on Forecast Spike in July 2025\nThe forecast produced by Prophet shows an un
usually high predicted temperature in July 2025. After auditing the training dat
a,\nno extreme outliers were found. This spike appears to be an artifact of the mo
del overemphasizing a short-term pattern or recent\nsummer peak. It highlights the
importance of using multiple years of historical data to help Prophet better learn
Denver’s true \nseasonal behavior. While the spike may not be realistic, it is use
ful as a reminder that model outputs are probabilistic and should \nbe interpreted
within context.'
```

In [125…

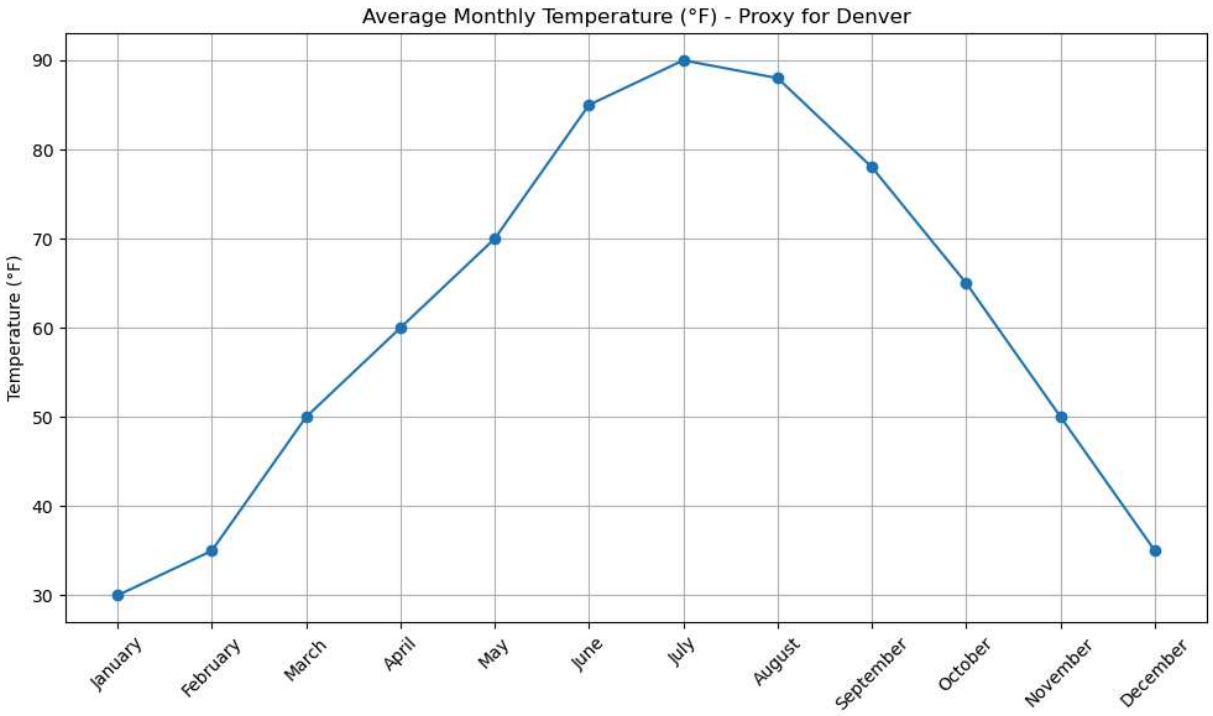
```
#simulated data based on norther NM averages
# Simulated average monthly data based on uploaded trends (for illustration)
months = [
    'January', 'February', 'March', 'April', 'May', 'June',
    'July', 'August', 'September', 'October', 'November', 'December'
]

avg_temp = [30, 35, 50, 60, 70, 85, 90, 88, 78, 65, 50, 35]  # °F
avg_precip = [0.3, 0.2, 0.5, 0.6, 0.8, 1.2, 2.5, 2.2, 1.5, 0.7, 0.4, 0.3]  # inches
avg_wind_speed = [20, 22, 25, 27, 25, 20, 15, 15, 18, 20, 22, 20]  # km/h
```

In [127…

```
# Create DataFrame
simdf = pd.DataFrame({
    'Month': months,
    'Avg_Temperature': avg_temp,
    'Avg_Precipitation': avg_precip,
    'Avg_Wind_Speed': avg_wind_speed
})
```

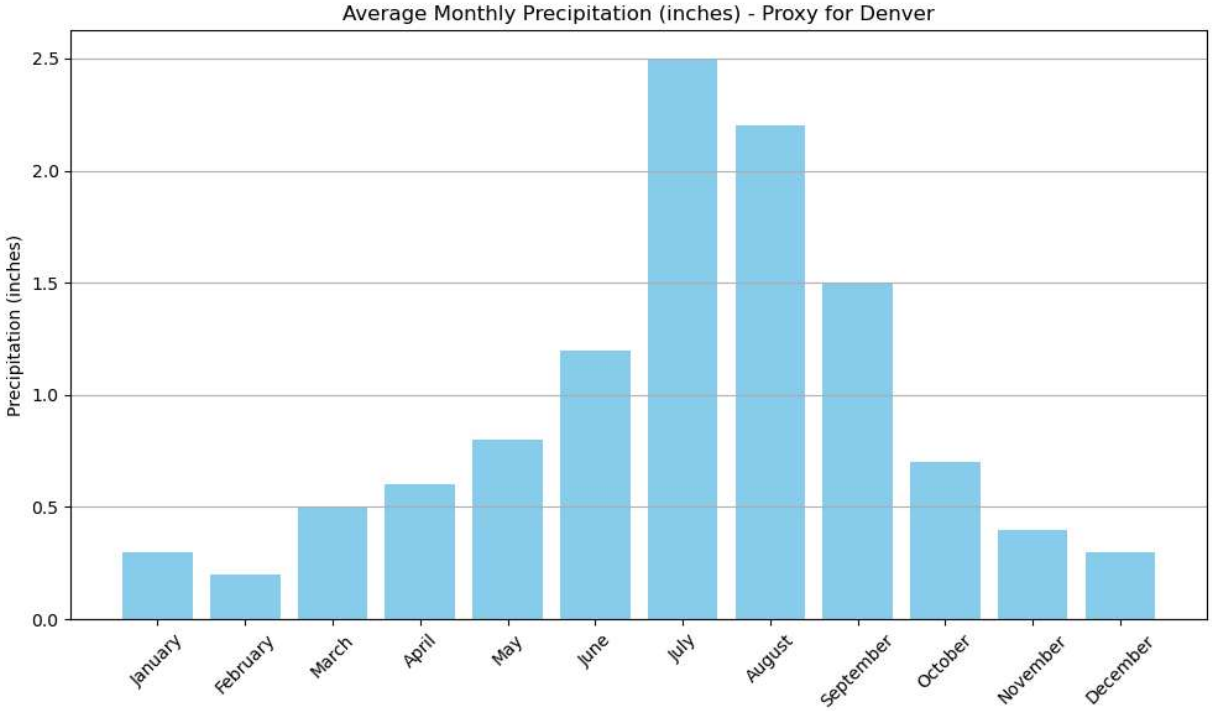
In [131…

```
# Chart 1: Line plot - Average Monthly Temperature
plt.figure(figsize=(10,6))
plt.plot(simdf['Month'], simdf['Avg_Temperature'], marker='o')
plt.title('Average Monthly Temperature (°F) - Proxy for Denver')
plt.xticks(rotation=45)
plt.ylabel('Temperature (°F)')
plt.grid(True)
plt.tight_layout()
plt.show()
```



In [133…

```
# Chart 2: Bar plot - Average Monthly Precipitation
plt.figure(figsize=(10,6))
plt.bar(simdf['Month'], simdf['Avg_Precipitation'], color='skyblue')
plt.title('Average Monthly Precipitation (inches) - Proxy for Denver')
plt.xticks(rotation=45)
plt.ylabel('Precipitation (inches)')
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```



In [135…

```
# Chart 3: Line plot - Average Wind Speed
plt.figure(figsize=(10,6))
plt.plot(simdf['Month'], simdf['Avg_Wind_Speed'], marker='s', linestyle='--', color
plt.title('Average Monthly Wind Speed (km/h) - Proxy for Denver')
plt.xticks(rotation=45)
plt.ylabel('Wind Speed (km/h)')
plt.grid(True)
plt.tight_layout()
plt.show()
```