task-5

May 15, 2025

0.1 Task 5: Streamlit Dashboard (Optional Submission)

To enhance the decision-making process for both internal teams and external stakeholders, I developed a lightweight but impactful **interactive Streamlit dashboard** for exploring regional solar deployment suitability.

0.1.1 Key Features

1.

0.1.2 Region Selector

- A user can select any region from a dropdown menu to view:
  - Solar irradiance
  - Grid access %
  - Electricity cost
  - Infrastructure index
  - Terrain ruggedness
  - Computed **Solar Access Score**
- Metrics are displayed using Streamlit's `st.metric()` cards for clarity and visual appeal.

2.

0.1.3 Ranked Region Table

- A dynamic, sortable table ranks all 50 regions based on their **Solar Access Score** — a composite metric derived from irradiance, grid access, cost, and infrastructure.
- The top 10 regions are visualized in a horizontal **bar chart** (using Plotly) for quick identification of high-impact zones.

3.

0.1.4 Region Profile Radar Chart

- A polar chart allows users to compare the selected region's performance against **benchmark maximums** for each metric.
- This helps identify which factors are strengths or weaknesses for a given location.

4.

0.1.5 Embedded Strategic Summary

- A brief written recommendation recap (from Task 4) is included directly in the app, summarizing:
  - The top 3 regions for pilot projects
  - Key reasoning behind those choices
  - Remaining unknowns and next steps

0.1.6 Benefits

- **Interactive Decision Support**: Enables faster, visual filtering of candidate regions.
- **Stakeholder Ready**: Suitable for sharing with government partners or internal strategy teams.
- **Modular & Extensible**: Can easily be connected to live datasets or geospatial APIs in future phases.

[ ]:
```python
import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px

# ---------- Load & Prepare Data ----------
df = pd.read_csv("PrimeFrontier_SolarDeploymentDataset.csv")

from sklearn.preprocessing import MinMaxScaler

def compute_solar_access_score(df):
    scaler = MinMaxScaler()
    scaled = scaler.fit_transform(df[[
        'Solar_Irradiance_kWh_m2_day',
        'Grid_Access_Percent',
        'Infrastructure_Index',
        'Electricity_Cost_USD_per_kWh'
    ]])
    df[['Norm_Irradiance', 'Norm_GridAccess', 'Norm_Infra', 'Norm_Cost']] =␣
↪scaled
    df['Inverse_Grid'] = 1 - df['Norm_GridAccess']
    df['Solar_Access_Score'] = (
        0.35 * df['Norm_Irradiance'] +
        0.25 * df['Inverse_Grid'] +
        0.20 * df['Norm_Infra'] +
        0.20 * df['Norm_Cost']
    )
    return df

df = compute_solar_access_score(df)
```

```python
df_sorted = df.sort_values(by="Solar_Access_Score", ascending=False)

# ---------- Streamlit UI ----------
st.set_page_config(layout="wide")
st.title("  Prime Frontier Group - Solar Site Dashboard")

# Sidebar Region Selector
region = st.sidebar.selectbox("  Select a Region", df["Region"].unique())
selected = df[df["Region"] == region].squeeze()

# ---------- Region Metric Cards ----------
st.subheader(f"  Metrics for {region}")

col1, col2, col3 = st.columns(3)
col1.metric("Solar Irradiance", f"{selected['Solar_Irradiance_kWh_m2_day']} kWh/m²/day")
col2.metric("Grid Access", f"{selected['Grid_Access_Percent']}%")
col3.metric("Electricity Cost", f"${selected['Electricity_Cost_USD_per_kWh']} / kWh")

col4, col5, col6 = st.columns(3)
col4.metric("Infrastructure Index", f"{selected['Infrastructure_Index']}")
col5.metric("Terrain Ruggedness", f"{selected['Terrain_Ruggedness_Score']}")
col6.metric("  Solar Access Score", f"{round(selected['Solar_Access_Score'], 3)}")

# ---------- Bar Chart: Top 10 Regions ----------
st.markdown("###   Top 10 Regions by Solar Access Score")
top10 = df_sorted[["Region", "Solar_Access_Score"]].head(10)
fig1 = px.bar(top10, x='Solar_Access_Score', y='Region', orientation='h', color='Solar_Access_Score',
              title="Top 10 Solar Suitability Rankings", height=400)
fig1.update_layout(yaxis={'categoryorder': 'total ascending'})
st.plotly_chart(fig1, use_container_width=True)

# ---------- Radar Chart: Selected Region vs Max Values ----------
st.markdown("###   Regional Profile vs Benchmark")

radar_df = pd.DataFrame({
    'Metric': ['Solar Irradiance', 'Grid Access', 'Electricity Cost', 'Infrastructure', 'Ruggedness'],
    'Selected Region': [
        selected['Solar_Irradiance_kWh_m2_day'],
        selected['Grid_Access_Percent'],
        selected['Electricity_Cost_USD_per_kWh'],
        selected['Infrastructure_Index'],
        selected['Terrain_Ruggedness_Score']
```

```
    ],
    'Max Value': [
        df['Solar_Irradiance_kWh_m2_day'].max(),
        df['Grid_Access_Percent'].max(),
        df['Electricity_Cost_USD_per_kWh'].max(),
        df['Infrastructure_Index'].max(),
        df['Terrain_Ruggedness_Score'].max()
    ]
})
fig2 = px.line_polar(radar_df, r='Selected Region', theta='Metric',␣
↪line_close=True, title="Region Profile vs Benchmark")
fig2.add_scatterpolar(r=radar_df['Max Value'], theta=radar_df['Metric'],␣
↪fill='none', name='Max Value')
st.plotly_chart(fig2, use_container_width=True)

# ---------- Strategic Summary ----------
st.markdown("###  Strategic Summary")
st.info("""
- **Region_32, Region_7, and Region_3** are top candidates for solar pilot␣
↪deployment based on high Solar Access Scores.
- These areas combine high irradiance, elevated energy cost, and limited grid␣
↪access.
- Next step: Validate on-ground logistics, community readiness, and policy␣
↪alignment.
""")
```

0.1.7 Strategic Summary

- **Region_32, Region_7, and Region_3** are top candidates for solar pilot deployment based on high Solar Access Scores.
- These areas combine high irradiance, elevated energy cost, and limited grid access.
- Next step: Validate on-ground logistics, community readiness, and policy alignment.