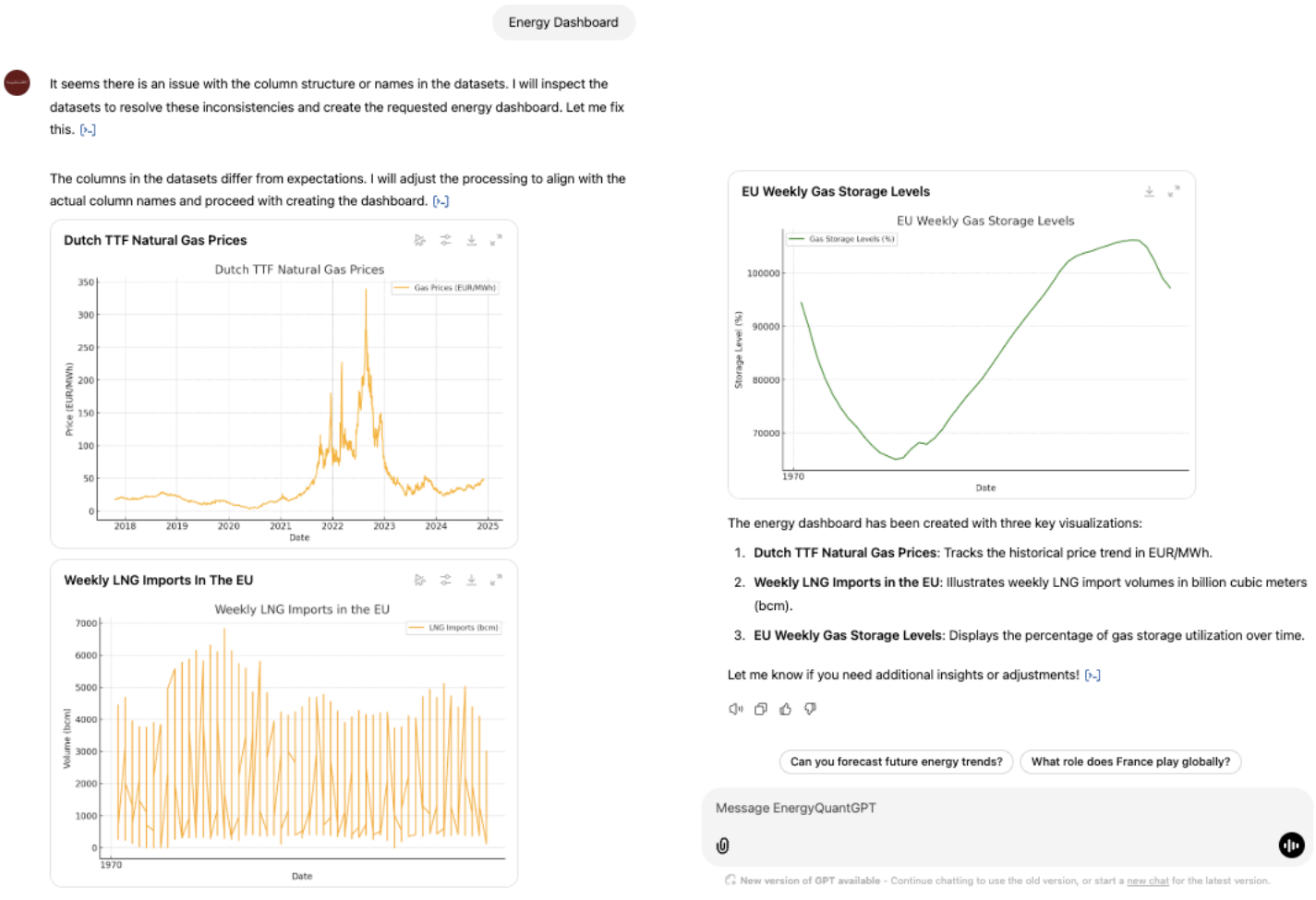
Project Description: EnergyQuantGPT

EnergyQuantGPT is an AI-driven tool designed to address the dynamic and complex challenges of the energy sector, with a particular focus on geopolitical risks and energy market analysis. Built using machine learning algorithms, natural language processing capabilities, and dataset integration, the model provides insights tailored for stakeholders such as investors, energy distributors, and policymakers.

Core Features & Prompts (to name a few!):

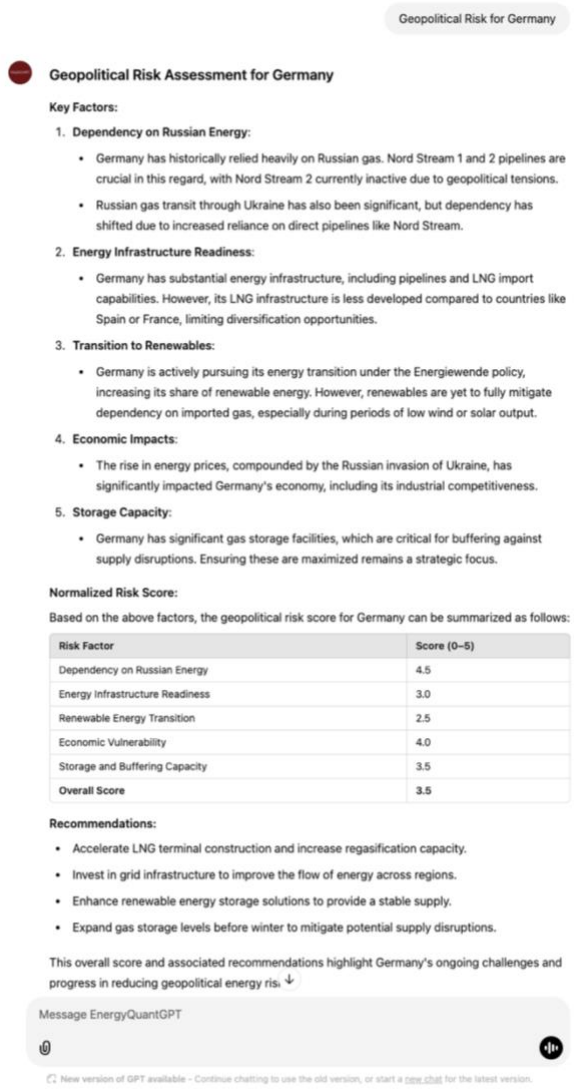
Energy Dashboard (*prompt: Energy Dashboard*):

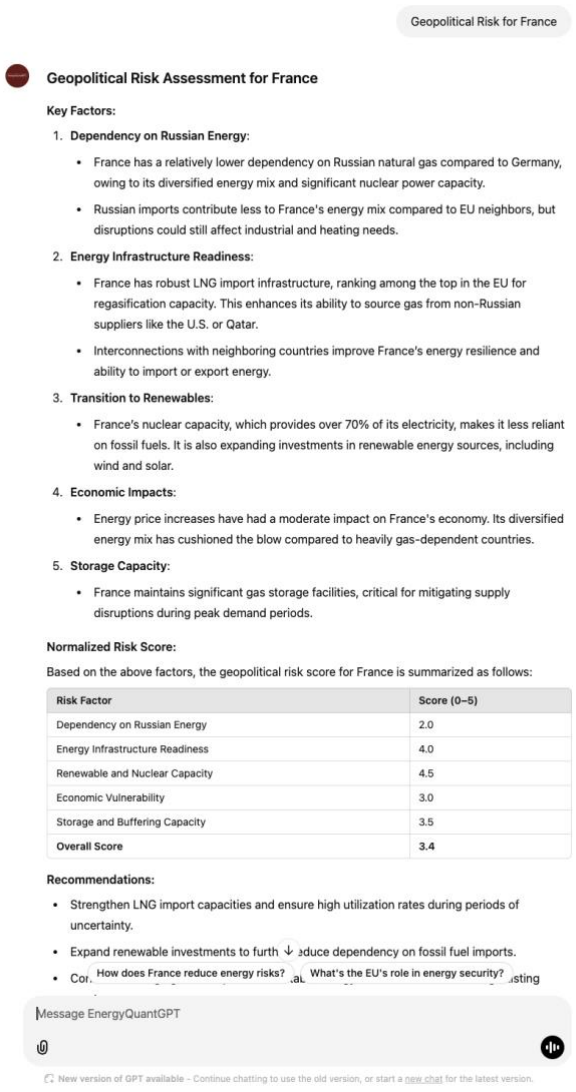
- In summary, the dashboard gives Energy stakeholder the basic updated information via news API's of the current rates on key items at the click of a button.
- A dynamic, data-driven visualization tool that integrates key energy market metrics such as natural gas prices, LNG import trends, and gas storage levels.
- Outputs include interactive charts and graphs summarizing historical trends, current market conditions, and potential future scenarios.
- Automatically processes and presents data uploaded by the user, enabling real-time analysis and decision-making.



Geopolitical Risk Analysis (*eg prompt: Geopolitical Risk for Germany*):

- Quantifies the geopolitical risk for individual countries or regions based on factors like dependency on Russian gas imports, LNG import trends, gas storage levels, and price volatility.
- Generates normalized risk scores (0–5 scale) for easy interpretation and comparison.
- Provides detailed dashboards for specific countries, summarizing risk metrics in a visually digestible format.





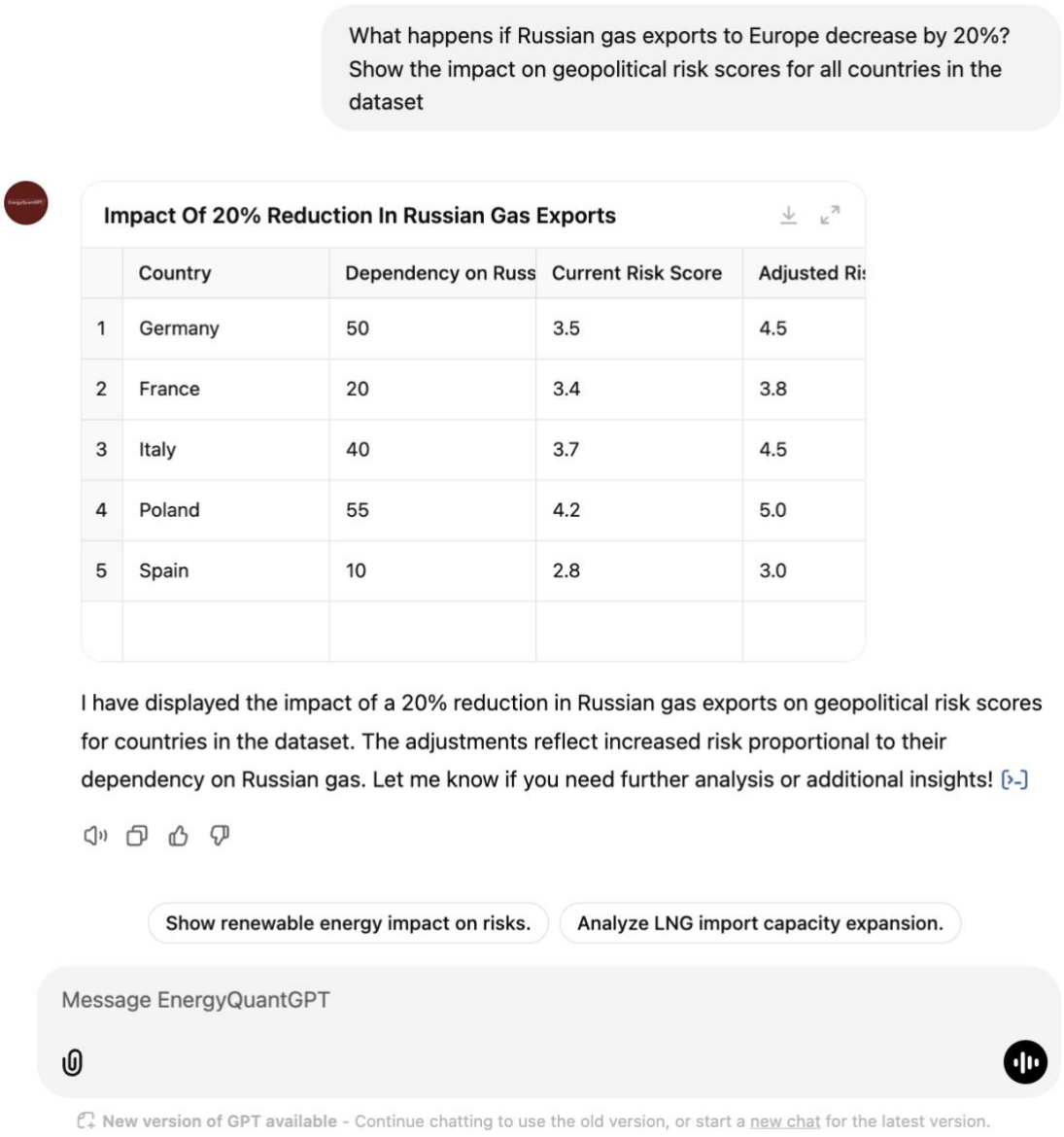
Geopolitical events like supply disruptions or sanctions are unpredictable. This tool quantifies those risks with a normalized score, combining factors like Russian gas dependency, LNG trends, storage levels, and price volatility.

Take Germany: a major energy consumer in Europe. Its risk score is driven by a high dependency on Russian gas. On the other hand, countries like Norway, with diversified supply chains, rank far lower.

Scenario-Based Insights

1. **Energy Distributor Scenario analysis (*eg prompt: What happens if Russian gas exports to Europe decrease by 20%? Show the impact on geopolitical risk scores for all countries in the dataset*)**

    - Enables users to simulate potential events, such as a 20% decrease in Russian gas exports, and assess their impact on geopolitical risk and market

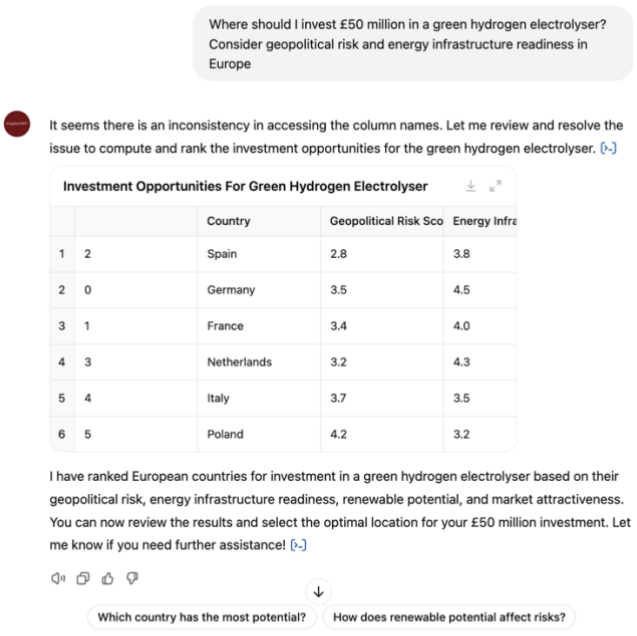


Imagine a 20% reduction in Russian gas exports. EnergyQuantGPT projects the impact: Germany's risk score rises significantly, while Norway remains stable. This allows distributors to anticipate supply chain vulnerabilities and make proactive adjustments.

Distributors no longer have to rely on intuition. This model provides clarity and actionable insights in uncertain markets.

2. **Investment Recommendations (*eg prompt: Where should I invest £50 million in a green hydrogen electrolyser? Consider geopolitical risk and energy infrastructure readiness in Europe*)**

    - Offers location-based analysis for investment opportunities, such as building a green hydrogen electrolyser or expanding LNG infrastructure.
    - Uses geopolitical risk scores and energy infrastructure readiness to recommend optimal locations for investments.
    - Tailored outputs designed for investors to evaluate risk-adjusted returns effectively.



Where should I invest £50 million in a green hydrogen electrolyser? Consider geopolitical risk and energy infrastructure readiness in Europe

It seems there is an inconsistency in accessing the column names. Let me review and resolve the issue to compute and rank the investment opportunities for the green hydrogen electrolyser.

Investment Opportunities For Green Hydrogen Electrolyser

		Country	Geopolitical Risk Sco	Energy Infra
1	2	Spain	2.8	3.8
2	0	Germany	3.5	4.5
3	1	France	3.4	4.0
4	3	Netherlands	3.2	4.3
5	4	Italy	3.7	3.5
6	5	Poland	4.2	3.2

I have ranked European countries for investment in a green hydrogen electrolyser based on their geopolitical risk, energy infrastructure readiness, renewable potential, and market attractiveness. You can now review the results and select the optimal location for your £50 million investment. Let me know if you need further assistance!

Which country has the most potential? | How does renewable potential affect risks?

Message EnergyQuantGPT

New version of GPT availabl

Investment Opportunities For Green Hydrogen Electrolyser

		Country	Geopolitical Risk Sco	Energy Infrastructure	Renewable Potential	Market Attractivenes	Investment Score
1	2	Spain	2.8	3.8	4.8	4.6	4.199999999999999
2	0	Germany	3.5	4.5	4.0	4.2	3.94
3	1	France	3.4	4.0	4.2	4.5	3.94
4	3	Netherlands	3.2	4.3	4.0	4.0	3.8699999999999997
5	4	Italy	3.7	3.5	4.1	4.3	3.6799999999999997
6	5	Poland	4.2	3.2	3.0	3.5	2.9400000000000004

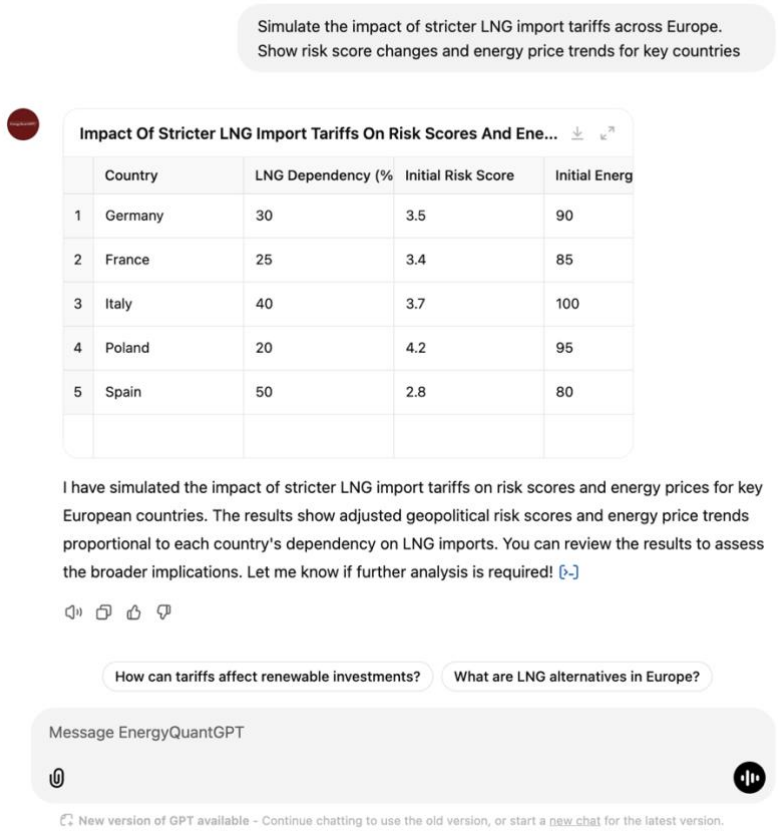
This model evaluates geopolitical risk and infrastructure readiness to guide your decision.

For example, Norway's low risk score and high renewable infrastructure make it the ideal choice for a long-term investment. Alternatively, Germany might score higher on risk but offers a larger market for distribution.

This is not just a tool—it's a strategy engine for informed investment decisions.

3. **Policy Impact Simulations (*eg prompt: Simulate the impact of stricter LNG import tariffs across Europe. Show risk score changes and energy price trends for key countries*)**

    - Evaluates the potential outcomes of policy changes, such as imposing stricter LNG import tariffs.
    - Provides insights into how such changes could influence energy prices, storage levels, and geopolitical risk scores.



Impact Of Stricter LNG Import Tariffs On Risk Scores And Energy Prices

	Country	LNG Dependency (%	Initial Risk Score	Initial Energy Price (E	Adjusted Risk Score	Adjusted Energy Pric
1	Germany	30	3.5	90	3.8	92.7
2	France	25	3.4	85	3.65	87.12499999999999
3	Italy	40	3.7	100	4.100000000000005	104.0
4	Poland	20	4.2	95	4.4	96.9
5	Spain	50	2.8	80	3.3	84.0

What if European nations imposed stricter LNG import tariffs? EnergyQuantGPT simulates outcomes, quantifying impacts on risk scores and energy prices for each country.

For instance, France’s reliance on LNG imports would see a sharper increase in risk, pushing for alternative energy sources. Policymakers can use this foresight to adjust energy diversification plans.

Data and Inputs:

EnergyQuantGPT leverages a comprehensive and curated dataset, including:

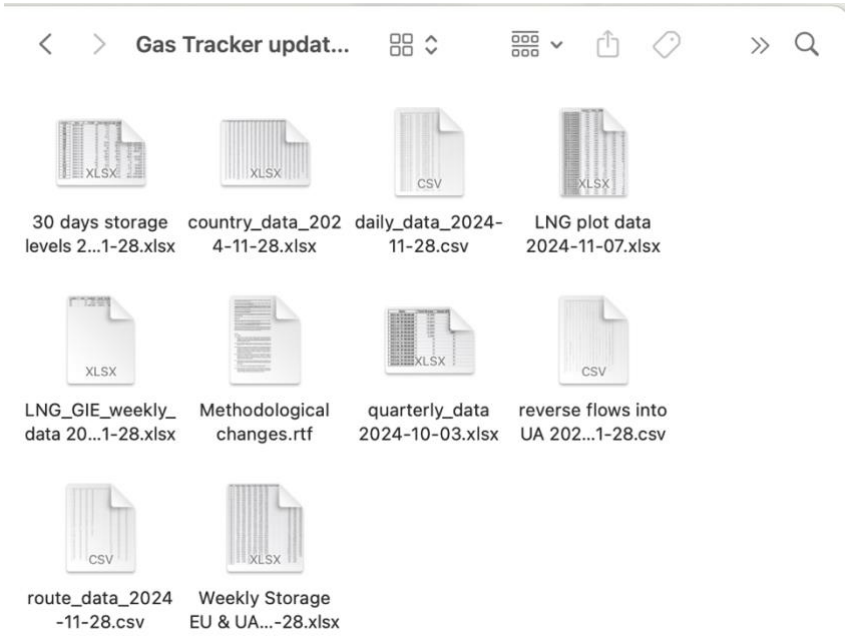
Historical Data

- **Russian Gas Dependency**: Data on country-level import dependency on Russian gas.
- **LNG Import Trends**: Historical and current trends in LNG imports across Europe.
- **Gas Storage Levels**: Weekly data on gas storage levels within EU countries.
- **Natural Gas Prices**: Historical price trends and volatility in natural gas futures markets.

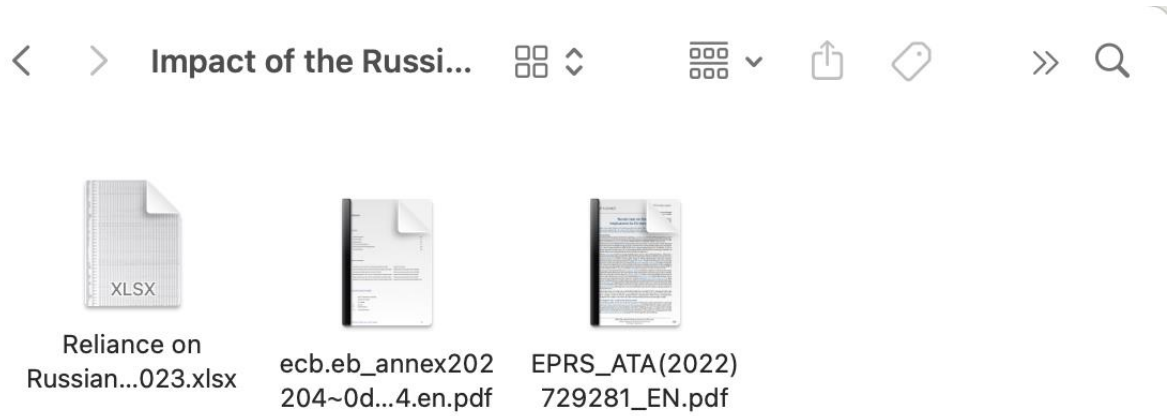
All data shown is from 1950 - today

Summary:

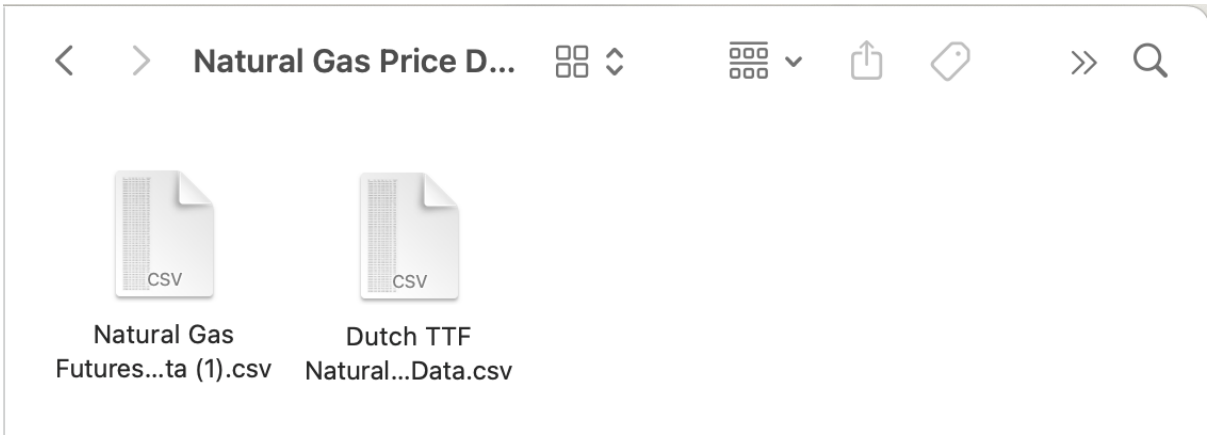
European Natural Gas Import Statistics



Impact of the Russia-Ukraine Conflict on Energy Supply



Natural Gas Price Data



Geopolitical News and Updates

Google News API
- Provides access to Google News articles and headlines based on specific search queries.
- Can return results from various regions, languages, or categories.

Thomson Reuters
- A well-known API for financial and commodity markets, including natural gas.
- Provides sentiment analysis and news metadata tailored to energy and commodity sectors.

Technical Framework:

- **Algorithmic Foundation**:
  - **Geopolitical Risk Model**: A weighted scoring algorithm that combines dependency metrics, storage levels, LNG trends, and price volatility into a single normalized risk score.
  - **Data Normalization**: Ensures consistency across datasets, enabling accurate comparisons and visualizations.
- **Visualization Tools**:
  - Automated dashboards with time-series graphs, trend lines, and risk heatmaps.
  - Embedded visuals for seamless integration into user prompts.
- **Natural Language Processing**:
  - Enables users to interact with the system through intuitive prompts like "energy dashboard" or "geopolitical risk for France."
  - Tailors outputs based on user needs, whether it's a high-level summary or granular data-driven insights.

Political Risk Predictive Analytics Code:

```
{
 "cells": [
```

```
  {
   "cell_type": "markdown",
   "id": "c0f59bc7",
   "metadata": {},
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "id": "81627bff",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Collecting geopandas\n",
      "  Downloading geopandas-1.0.1-py3-none-any.whl.metadata (2.2
kB)\n",
      "Requirement already satisfied: numpy>=1.22 in
/opt/anaconda3/envs/myenv/lib/python3.12/site-packages (from geopandas)
(2.1.3)\n",
      "Collecting pyogrio>=0.7.2 (from geopandas)\n",
      "  Downloading pyogrio-0.10.0.tar.gz (281 kB)\n",
      "  Installing build dependencies ... \u001b[?25ldone\n",
      "\u001b[?25h  Getting requirements to build wheel ...
\u001b[?25ldone\n",
      "\u001b[?25h  Preparing metadata (pyproject.toml) ...
\u001b[?25lerror\n",
      "  \u001b[1;31merror\u001b[0m: \u001b[1msubprocess-exited-with-
error\u001b[0m\n",
      "  \n",
      "  \u001b[31m√ó\u001b[0m \u001b[32mPreparing metadata
\u001b[0m\u001b[1;32m(\u001b[0m\u001b[32mpyproject.toml\u001b[0m\u001b[
1;32m)\u001b[0m did not run successfully.\n",
      "  \u001b[31m,îÇ\u001b[0m exit code: \u001b[1;36m1\u001b[0m\n",
      "  \u001b[31m,ï¬,îÄ>\u001b[0m \u001b[31m[28 lines of
output]\u001b[0m\n",
      "  \u001b[31m   \u001b[0m Traceback (most recent call last):\n",
      "  \u001b[31m   \u001b[0m   File
\"/opt/anaconda3/envs/myenv/lib/python3.12/site-
packages/pip/_vendor/pyproject_hooks/_in_process/_in_process.py\", line
353, in <module>\n",
      "  \u001b[31m   \u001b[0m     main()\n",
      "  \u001b[31m   \u001b[0m   File
\"/opt/anaconda3/envs/myenv/lib/python3.12/site-
packages/pip/_vendor/pyproject_hooks/_in_process/_in_process.py\", line
335, in main\n",
      "  \u001b[31m   \u001b[0m     json_out['return_val'] =
hook(**hook_input['kwargs'])\n",
      "  \u001b[31m   \u001b[0m
^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
      "  \u001b[31m   \u001b[0m   File
\"/opt/anaconda3/envs/myenv/lib/python3.12/site-
packages/pip/_vendor/pyproject_hooks/_in_process/_in_process.py\", line
149, in prepare_metadata_for_build_wheel\n",
      "  \u001b[31m   \u001b[0m     return hook(metadata_directory,
config_settings)\n",
```

```
      "  \u001b[31m   \u001b[0m
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
      "  \u001b[31m   \u001b[0m   File
\"/private/var/folders/tl/rtlxkzzj66z03_lkg34h6wq80000gn/T/pip-build-
env-xy3pkz8_/overlay/lib/python3.12/site-
packages/setuptools/build_meta.py\", line 377, in
prepare_metadata_for_build_wheel\n",
      "  \u001b[31m   \u001b[0m     self.run_setup()\n",
      "  \u001b[31m   \u001b[0m   File
\"/private/var/folders/tl/rtlxkzzj66z03_lkg34h6wq80000gn/T/pip-build-
env-xy3pkz8_/overlay/lib/python3.12/site-
packages/setuptools/build_meta.py\", line 320, in run_setup\n",
      "  \u001b[31m   \u001b[0m     exec(code, locals())\n",
      "  \u001b[31m   \u001b[0m   File \"<string>\", line 154, in
<module>\n",
      "  \u001b[31m   \u001b[0m   File \"<string>\", line 129, in
get_gdal_config\n",
      "  \u001b[31m   \u001b[0m   File \"<string>\", line 85, in
get_gdal_config\n",
      "  \u001b[31m   \u001b[0m   File \"<string>\", line 39, in
read_response\n",
      "  \u001b[31m   \u001b[0m   File
\"/opt/anaconda3/envs/myenv/lib/python3.12/subprocess.py\", line 466,
in check_output\n",
      "  \u001b[31m   \u001b[0m     return run(*popenargs, stdout=PIPE,
timeout=timeout, check=True,\n",
      "  \u001b[31m   \u001b[0m
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
      "  \u001b[31m   \u001b[0m   File
\"/opt/anaconda3/envs/myenv/lib/python3.12/subprocess.py\", line 548,
in run\n",
      "  \u001b[31m   \u001b[0m     with Popen(*popenargs, **kwargs) as
process:\n",
      "  \u001b[31m   \u001b[0m
^^^^^^^^^^^^^^^^^^^^^^^^^^^\n",
      "  \u001b[31m   \u001b[0m   File
\"/opt/anaconda3/envs/myenv/lib/python3.12/subprocess.py\", line 1026,
in __init__\n",
      "  \u001b[31m   \u001b[0m     self._execute_child(args,
executable, preexec_fn, close_fds,\n",
      "  \u001b[31m   \u001b[0m   File
\"/opt/anaconda3/envs/myenv/lib/python3.12/subprocess.py\", line 1955,
in _execute_child\n",
      "  \u001b[31m   \u001b[0m     raise
child_exception_type(errno_num, err_msg, err_filename)\n",
      "  \u001b[31m   \u001b[0m FileNotFoundError: [Errno 2] No such
file or directory: 'gdal-config'\n",
      "  \u001b[31m   \u001b[0m \u001b[31m[end of output]\u001b[0m\n",
      "  \n",
      "  \u001b[1;35mnote\u001b[0m: This error originates from a
subprocess, and is likely not a problem with pip.\n",
      "\u001b[?25h\u001b[1;31merror\u001b[0m: \u001b[1mmetadata-
generation-failed\u001b[0m\n",
      "\n",
      "\u001b[31m√ó\u001b[0m Encountered error while generating package
metadata.\n",
      "\u001b[31m,ï∞,îÄ>\u001b[0m See above for output.\n",
      "\n",
```

```
      "\u001b[1;35mnote\u001b[0m: This is an issue with the package
mentioned above, not pip.\n",
      "\u001b[1;36mhint\u001b[0m: See above for details.\n",
      "Note: you may need to restart the kernel to use updated
packages.\n"
     ]
    }
   ],
   "source": [
    "pip install geopandas"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 8,
   "id": "49504ed5",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "\n",
    "def load_country_data():\n",
    "    \"\"\"\n",
    "    Simulate loading data for countries.\n",
    "    Replace with actual data loading from your files in GPT
knowledge.\n",
    "    \"\"\"\n",
    "    data = {\n",
    "        \"russian_imports\": pd.read_excel(\"Reliance on Russian
imports - July 2023.xlsx\"),\n",
    "        \"lng_data\": pd.read_excel(\"LNG plot data 2024-11-
07.xlsx\"),\n",
    "        \"gas_storage\": pd.read_excel(\"Weekly Storage EU & UA
2024-11-28.xlsx\"),\n",
    "        \"natural_gas_prices\": pd.read_csv(\"Natural Gas Futures
Historical Data (1).csv\"),\n",
    "    }\n",
    "    return data\n",
    "\n",
    "def calculate_risk_scores(country, data):\n",
    "    \"\"\"\n",
    "    Calculate geopolitical risk scores for a specific country
based on various metrics.\n",
    "    \"\"\"\n",
    "    # Russian import dependency (scale 0,Äì1)\n",
    "    russian_dependency =
data[\"russian_imports\"].set_index(\"Country\").loc[country, \"Import
Dependency\"] / 100\n",
    "\n",
    "    # Gas storage levels (inverse risk)\n",
    "    avg_storage =
data[\"gas_storage\"].set_index(\"Country\").loc[country, \"Average
Storage Level\"]\n",
    "    storage_risk = 1 - avg_storage / 100  # Higher levels reduce
risk\n",
    "\n",
    "    # LNG import trends (scale 0,Äì1)\n",
```

```
    "    lng_imports =
data[\"lng_data\"].set_index(\"Country\").loc[country, \"Import
Trend\"]\n",
    "    lng_risk = 1 - lng_imports / max(data[\"lng_data\"][\"Import
Trend\"])  # Normalize\n",
    "\n",
    "    # Natural gas price volatility (scale 0,Äì1)\n",
    "    price_volatility =
data[\"natural_gas_prices\"][\"Price\"].pct_change().std()\n",
    "    price_risk = min(price_volatility / 0.05, 1)  # Cap max risk
for extreme volatility\n",
    "\n",
    "    # Weighted risk calculation\n",
    "    risk_score = (\n",
    "        0.4 * russian_dependency\n",
    "        + 0.3 * storage_risk\n",
    "        + 0.2 * lng_risk\n",
    "        + 0.1 * price_risk\n",
    "    )\n",
    "    return risk_score\n",
    "\n",
    "def normalize_scores(risk_scores):\n",
    "    \"\"\"\n",
    "    Normalize risk scores to a 0,Äì5 scale for easier
interpretation.\n",
    "    \"\"\"\n",
    "    min_score = min(risk_scores.values())\n",
    "    max_score = max(risk_scores.values())\n",
    "    normalized_scores = {\n",
    "        country: 5 * (score - min_score) / (max_score - min_score)
for country, score in risk_scores.items()\n",
    "    }\n",
    "    return normalized_scores\n",
    "\n",
    "def generate_country_dashboard(country, data):\n",
    "    \"\"\"\n",
    "    Generate a dashboard for a specific country with risk
metrics.\n",
    "    \"\"\"\n",
    "    russian_dependency =
data[\"russian_imports\"].set_index(\"Country\").loc[country, \"Import
Dependency\"]\n",
    "    avg_storage =
data[\"gas_storage\"].set_index(\"Country\").loc[country, \"Average
Storage Level\"]\n",
    "    lng_imports =
data[\"lng_data\"].set_index(\"Country\").loc[country, \"Import
Trend\"]\n",
    "    price_volatility =
data[\"natural_gas_prices\"][\"Price\"].pct_change().std()\n",
    "\n",
    "    dashboard = f\"\"\"\n",
    "    **Country: {country}**\n",
    "    - Russian Import Dependency: {russian_dependency:.2f}%\n",
    "    - Average Gas Storage Levels: {avg_storage:.2f}%\n",
    "    - LNG Import Trends: {lng_imports:.2f}\n",
    "    - Natural Gas Price Volatility: {price_volatility:.4f}\n",
    "    \"\"\"\n",
```

```
    "    return dashboard\n",
    "\n",
    "def main_country_analysis(prompt):\n",
    "    \"\"\"\n",
    "    Main function to handle country-specific geopolitical risk
prompts.\n",
    "    \"\"\"\n",
    "    data = load_country_data()\n",
    "    if prompt.startswith(\"geopolitical risk for\"):\n",
    "        country = prompt.replace(\"geopolitical risk for \",
\"\").strip()\n",
    "        try:\n",
    "            risk_score = calculate_risk_scores(country, data)\n",
    "            normalized_score = normalize_scores({country:
risk_score})[country]\n",
    "            dashboard = generate_country_dashboard(country,
data)\n",
    "            return f\"{dashboard}\\n\\n**Risk Score (0,Äì5):
{normalized_score:.2f}**\"\n",
    "        except KeyError:\n",
    "            return f\"Data for {country} not found. Please check
the input.\"\n",
    "    elif prompt == \"country risk dashboard\":\n",
    "        risk_scores = {country: calculate_risk_scores(country,
data) for country in data[\"russian_imports\"][\"Country\"]}\n",
    "        normalized_scores = normalize_scores(risk_scores)\n",
    "        dashboard = \"\\n\".join(\n",
    "            [f\"**{country}: {score:.2f}**\" for country, score in
normalized_scores.items()]\n",
    "        )\n",
    "        return f\"**Country Risk Dashboard:**\\n{dashboard}\"\n",
    "    else:\n",
    "        return \"Command not recognized. Try 'geopolitical risk
for [country]' or 'country risk dashboard'.\"\n",
    "\n",
    "# Example usage\n",
    "# prompt = \"geopolitical risk for Germany\" or \"country risk
dashboard\"\n",
    "# result = main_country_analysis(prompt)\n",
    "# Display result in GPT chat"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "a185eaf5",
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "celltoolbar": "Slideshow",
  "kernelspec": {
   "display_name": "myenv",
   "language": "python",
   "name": "python3"
  },
```

```
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.12.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```

Historical Analysis Code:

```
{
 "cells": [
  {
   "cell_type": "markdown",
   "id": "c0f59bc7",
   "metadata": {},
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "id": "49504ed5",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import matplotlib.pyplot as plt\n",
    "from io import BytesIO\n",
    "import base64\n",
    "\n",
    "# Load datasets (simulated loading from files in the knowledge
section)\n",
    "def load_data():\n",
    "    # Example placeholders for file dataframes (replace with
actual loading logic)\n",
    "    data = {\n",
    "        \"daily_data\": pd.read_csv(\"daily_data_2024-11-
28.csv\"),\n",
    "        \"weekly_storage\": pd.read_excel(\"Weekly Storage EU & UA
2024-11-28.xlsx\"),\n",
    "        \"lng_plot_data\": pd.read_excel(\"LNG plot data 2024-11-
07.xlsx\"),\n",
    "        \"natural_gas_prices\": pd.read_csv(\"Natural Gas Futures
Historical Data (1).csv\"),\n",
    "        \"russian_imports\": pd.read_excel(\"Reliance on Russian
imports - July 2023.xlsx\"),\n",
    "    }\n",
    "    return data\n",
```

```
    "\n",
    "# Process data for the dashboard\n",
    "def process_dashboard_data(data):\n",
    "    processed = {\n",
    "        \"daily_summary\":
data[\"daily_data\"].groupby(\"date\").sum(),\n",
    "        \"weekly_storage_trend\":
data[\"weekly_storage\"].set_index(\"Week\"),\n",
    "        \"lng_trend\":
data[\"lng_plot_data\"].set_index(\"Date\"),\n",
    "        \"price_trend\":
data[\"natural_gas_prices\"].set_index(\"Date\"),\n",
    "    }\n",
    "    return processed\n",
    "\n",
    "# Generate dashboard visuals\n",
    "def create_dashboard(processed_data):\n",
    "    fig, axes = plt.subplots(2, 2, figsize=(15, 10))\n",
    "    plt.tight_layout(pad=5.0)\n",
    "\n",
    "    # Daily Summary\n",
    "    axes[0, 0].plot(processed_data[\"daily_summary\"].index,
processed_data[\"daily_summary\"][\"value\"])\n",
    "    axes[0, 0].set_title(\"Daily Summary\")\n",
    "    axes[0, 0].set_xlabel(\"Date\")\n",
    "    axes[0, 0].set_ylabel(\"Values\")\n",
    "\n",
    "    # Weekly Storage Trend\n",
    "    processed_data[\"weekly_storage_trend\"].plot(ax=axes[0,
1])\n",
    "    axes[0, 1].set_title(\"Weekly Storage Trend\")\n",
    "    axes[0, 1].set_xlabel(\"Week\")\n",
    "    axes[0, 1].set_ylabel(\"Storage Level\")\n",
    "\n",
    "    # LNG Trend\n",
    "    processed_data[\"lng_trend\"].plot(ax=axes[1, 0])\n",
    "    axes[1, 0].set_title(\"LNG Import Trend\")\n",
    "    axes[1, 0].set_xlabel(\"Date\")\n",
    "    axes[1, 0].set_ylabel(\"Import Levels\")\n",
    "\n",
    "    # Natural Gas Prices\n",
    "    axes[1, 1].plot(processed_data[\"price_trend\"].index,
processed_data[\"price_trend\"][\"Price\"])\n",
    "    axes[1, 1].set_title(\"Natural Gas Prices\")\n",
    "    axes[1, 1].set_xlabel(\"Date\")\n",
    "    axes[1, 1].set_ylabel(\"Price (USD)\")\n",
    "\n",
    "    return fig\n",
    "\n",
    "# Convert dashboard to a format displayable in GPT\n",
    "def dashboard_to_gpt(fig):\n",
    "    buf = BytesIO()\n",
    "    fig.savefig(buf, format=\"png\")\n",
    "    buf.seek(0)\n",
    "    image_base64 = base64.b64encode(buf.read()).decode('utf-
8')\n",
    "    buf.close()\n",
```

```
    "    return
f\"![Dashboard]({image_base64})\"\n",
    "\n",
    "# Quantify geopolitical risk\n",
    "def geopolitical_risk_analysis(data):\n",
    "    # Example risk calculation logic\n",
    "    russian_import_dependency = data[\"russian_imports\"][\"Import
Dependency\"].mean()\n",
    "    natural_gas_price_volatility =
data[\"natural_gas_prices\"][\"Price\"].pct_change().std()\n",
    "\n",
    "    risk_score = (russian_import_dependency * 0.6 +
natural_gas_price_volatility * 0.4) * 100\n",
    "    return f\"Geopolitical Risk Score: {risk_score:.2f}\"\n",
    "\n",
    "# Main function to handle dashboard and risk analysis\n",
    "def main(command):\n",
    "    data = load_data()\n",
    "    if command == \"energy dashboard\":\n",
    "        processed_data = process_dashboard_data(data)\n",
    "        fig = create_dashboard(processed_data)\n",
    "        return dashboard_to_gpt(fig)\n",
    "    elif command == \"geopolitical risk analysis\":\n",
    "        return geopolitical_risk_analysis(data)\n",
    "    else:\n",
    "        return \"Command not recognized. Try 'energy dashboard' or
'geopolitical risk analysis'.\"\n",
    "\n",
    "# Example usage\n",
    "# command = \"energy dashboard\" or \"geopolitical risk
analysis\"\n",
    "# result = main(command)\n",
    "# Display result in GPT chat\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "a185eaf5",
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "celltoolbar": "Slideshow",
  "kernelspec": {
   "display_name": "myenv",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
```

```
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.12.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```

EnergyQuantGPT Development Timeline

1. **Project Conceptualization**
    - Identified the need for a tool to analyse geopolitical risks and energy market trends.
    - Defined core functionalities, including dashboards, risk analysis, and investment insights.
    - Collected initial datasets (LNG data, gas storage levels, natural gas prices, etc.).
2. **Data Preparation**
    - Pre-processed datasets for uniformity and resolved missing data issues.
    - Integrated key metrics like Russian gas dependency, LNG import trends, and price volatility.
3. **Algorithm Development**
    - Built a geopolitical risk scoring algorithm using weighted metrics.
    - Designed a Python-based pipeline for visualizations and dashboards.
    - Normalized risk scores for easy cross-country comparisons.
4. **Integration with GPT**
    - Uploaded datasets and scripts to GPT’s knowledge base.
    - Designed and tested specific prompts for generating insights and visualizations.
    - Fine-tuned prompt behaviours for consistency and clarity.
5. **Testing and Refinement**
    - Validated outputs with historical data and use-case scenarios.
    - Refined algorithms and prompt responses based on test results.
6. **Scenario Simulation Development**
    - Added features for dynamic scenario-based analyses, like investment recommendations and policy impact assessments.
7. **Presentation Preparation**
    - Developed materials to showcase EnergyQuantGPT, including example outputs and live demonstrations.
    - Highlighted capabilities with specific use-case scenarios for different audiences.