Alzeimer's Disease Clinical Trials

Import Library

In [4]:
```
## Import Library
import json
import pandas as pd
import os
import streamlit as st
import plotly.express as px
import matplotlib.pyplot as plt
```

Data scripting from Clinical trial.gov

In [2]:
```
# Initialize a list to collect data
data_list = []
```

In [4]:
```
#Loop through each JSON file in the current directory
for filename in os.listdir(os.getcwd()):
    if filename.endswith(".json"):
        file_path = os.path.join(os.getcwd(), filename)
        with open(file_path, 'r', encoding='utf-8') as file:
            data = json.load(file)
            id_module = data['protocolSection']['identificationModule']
            desc_module = data['protocolSection']['descriptionModule']
            contacts_locations_module = data['protocolSection'].get('contactsLocationsModule', {})
            sponsor_module = data['protocolSection'].get('sponsorCollaboratorsModule', {})
            responsible_party = sponsor_module.get('responsibleParty', {})
            additional_pi_section = data.get('additionalPISection', [{}])

            # Extract required data from JSON file
            nct_id = id_module.get("nctId", "")
            detailed_description = desc_module.get("detailedDescription", "")

            # Extracting PI information from various sections
            overall_officials = contacts_locations_module.get("overallOfficials", [])
            if overall_officials:
                pi_info = overall_officials[0]
                pi = pi_info.get("name", "")
                role = pi_info.get("role", "")
                affiliation = pi_info.get("affiliation", "")
            elif responsible_party:
                pi = responsible_party.get("investigatorFullName", "")
                role = responsible_party.get("investigatorTitle", "")
                affiliation = responsible_party.get("investigatorAffiliation", "")
            elif additional_pi_section:
                pi_info = additional_pi_section[0]
                pi = pi_info.get("name", "")
                affiliation = pi_info.get("affiliation", "")
                role = ""
# Append the processed data to the list
            data_list.append({
                "nct_id": nct_id,
                "detailed_description": detailed_description,
                "pi": pi,
                "role": role,
                "affiliation": affiliation
            })

            # Convert the list of dictionaries to a DataFrame
            df_json = pd.DataFrame(data_list)

            csv_file = "ctg-studies.csv"
            df_csv = pd.read_csv(csv_file)
            df_csv.rename(columns={'NCT Number': 'nct_id'}, inplace=True)

            # Merge the DataFrames
            merged_df = pd.merge(df_json, df_csv, on='nct_id', how='left')

            # Save the merged DataFrame to an Excel file
            merged_df.to_excel("merged_output.xlsx", index=False)
```

Data processing

In [ ]:
```
file_path = 'merged_output.xlsx'
df_merged = pd.read_excel(file_path)
```

Data Cleaning

In [ ]:
```
cleaned_df = df_merged.dropna(subset=['Phases','Sponsor','Start Date','affiliation', 'pi'])
cleaned_df
```

In [ ]:
```
# Convert the 'Start Date' column to datetime format
cleaned_df['Start Date'] = pd.to_datetime(cleaned_df['Start Date'], errors='coerce')
# Convert the 'Primary Completion Date' column to datetime format
cleaned_df['Primary Completion Date'] = pd.to_datetime(cleaned_df['Start Date'], errors='coerce')
cleaned_df['Completion Year'] = cleaned_df['Primary Completion Date'].dt.year

# Phase 2-4 industry sponsered study completed in 2024 to 2027
df_selected = cleaned_df[(cleaned_df['Funder Type'].str.upper() == 'INDUSTRY') &
                         (cleaned_df['Phases'].isin(['PHASE2', 'PHASE3','PHASE4']))&
                         (cleaned_df['Study Status']!= 'WITHDRAWN')
                         ]

df_selectedC = df_selected.dropna(axis=1, how='all')
df_selectedC
```

In [ ]:
```
df_selectedC = df_selectedC.sort_values(by='Primary Completion Date', ascending=True)
df_selectedC
```

Dashboard

In [ ]:
```
# Title of the dashboard
st.title("AD Clinical Trial Dashboard")

# Filter the data for studies complete in 2023-2034
trials_2023_2034 = df_selectedC[(df_selectedC['Completion Date'] >= '2023-01-01') & (df_selectedC['Completion Date'] <= '2034-12-31')]

# Count the number of trials in 2023-2034
num_trials_2023_2034 = trials_2023_2034.shape[0]

# Streamlit section to show the number of trials from 2023-2034
st.subheader("Number of Trials complete in 2023-2034")
st.metric(label="Trials complete (2023-2030)", value=num_trials_2023_2034)

# Plot 1: Pie Chart for Phases will complete in 2023-2034
st.subheader("Phases Distribution of Trials complete in 2023-2034")
phase_counts = trials_2023_2034['Phases'].value_counts()
fig1, ax1 = plt.subplots()
ax1.pie(phase_counts, labels=phase_counts.index, autopct='%1.1f%%', startangle=90)
ax1.axis('equal')
st.pyplot(fig1)

# Plot 2: Bar Chart for Sponsor vs. Phases (complete in 2023-2034)
st.subheader("Sponsor vs. Phases of Trials Complete in 2023-2034")
sponsor_phase_counts = trials_2023_2034.groupby(['Sponsor', 'Phases']).size().reset_index(name='Counts')
fig2 = px.bar(sponsor_phase_counts, x='Sponsor', y='Counts', color='Phases', barmode='group',
             title="Number of Studies per Sponsor by Phases")
st.plotly_chart(fig2)

# Plot3: Bar Chart for Conditions
st.subheader("Trials by Condition (complete in 2023-2034)")
condition_counts = trials_2023_2034.groupby('Conditions').size().reset_index(name='Counts')
fig3 = px.bar(condition_counts, x='Conditions', y='Counts',
             title="Number of Studies per Condition")
st.plotly_chart(fig3)

# New Plot 4: Number of Studies Expected to Complete Between 2023 and 2034
st.subheader("Number of Studies Expected to Complete Between 2023 and 2034")

# Group by year and sponsor
date_grouped_df = trials_2023_2034.groupby(['Completion Year', 'Sponsor']).size().reset_index(name='Count')

# Create a bar chart using Plotly
fig4 = px.bar(date_grouped_df, x='Completion Year', y='Count', color='Sponsor', barmode='group',
             title="Studies Expected to Complete Between 2023 and 2034 by Sponsor")

st.plotly_chart(fig4)

# Footer
st.write("### Data Source: https://clinicaltrials.gov")
```