Importing the Solar Consumption Csv

In [36]:
```
import pandas as pd
import numpy as np
df = pd.read_csv("solar_consuption.csv")
```

In [123]:
```
# processing the NaN values
df['Code'] = df['Code'].fillna(0)
df.head(10)
```

Out[123]:

	Entity	Code	Year	Electricity from solar (TWh)
0	Afghanistan	AFG	2000	0.0
1	Afghanistan	AFG	2001	0.0
2	Afghanistan	AFG	2002	0.0
3	Afghanistan	AFG	2003	0.0
4	Afghanistan	AFG	2004	0.0
5	Afghanistan	AFG	2005	0.0
6	Afghanistan	AFG	2006	0.0
7	Afghanistan	AFG	2007	0.0
8	Afghanistan	AFG	2008	0.0
9	Afghanistan	AFG	2009	0.0

In [118]:
```
df.info()
df.shape
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8683 entries, 0 to 8682
Data columns (total 4 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   Entity                         8683 non-null   object 
 1   Code                           8683 non-null   object 
 2   Year                           8683 non-null   int64  
 3   Electricity from solar (TWh)  8683 non-null   float64
dtypes: float64(1), int64(1), object(2)
memory usage: 271.5+ KB
```

Out[118]: (8683, 4)

In [39]:
```
# Getting the highest Solar consumption from the data
df.sort_values('Electricity from solar (TWh)', ascending=False)
```

Out[39]:

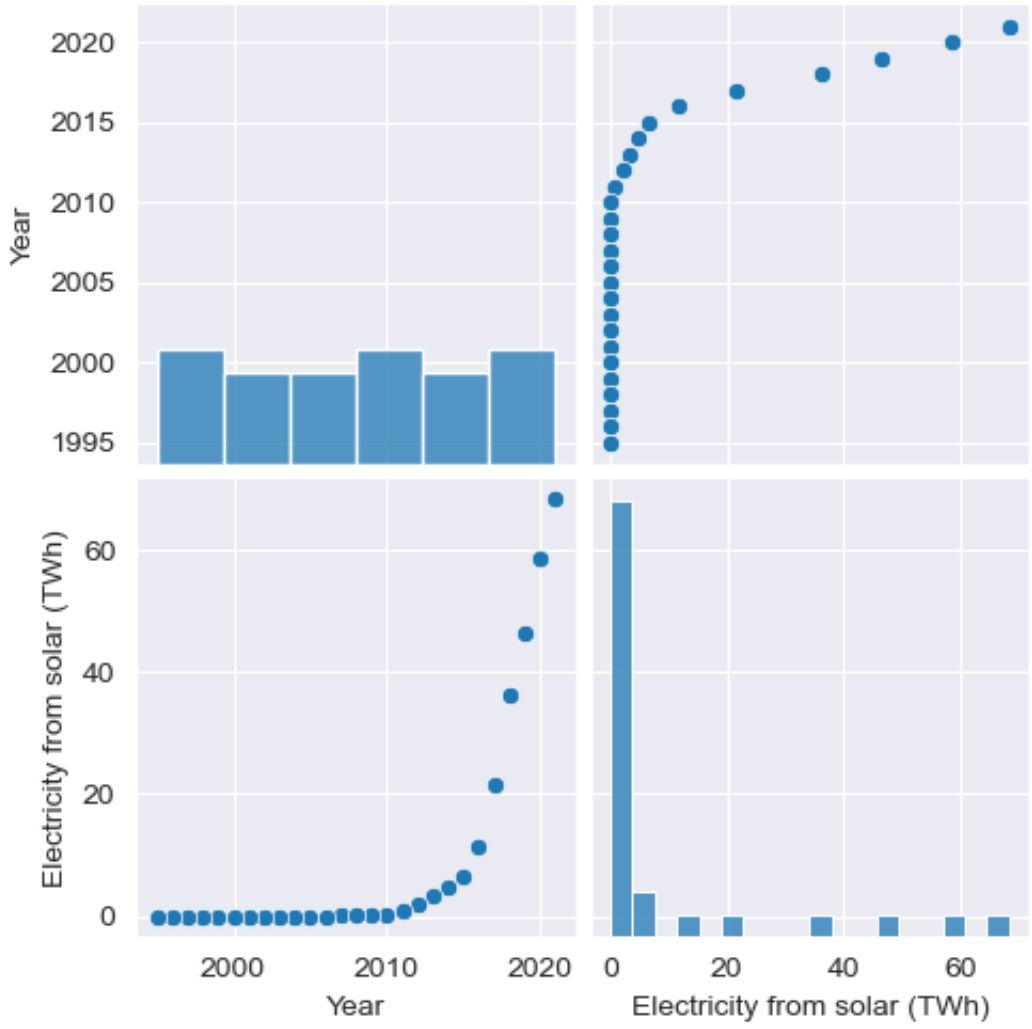
	Entity	Code	Year	Electricity from solar (TWh)
8616	World	OWID_WRL	2021	1040.50
2881	G20 (Ember)	0	2021	937.75
8615	World	OWID_WRL	2020	852.10
2880	G20 (Ember)	0	2020	775.70
8614	World	OWID_WRL	2019	701.19
...	...	...	...	...
4139	Kiribati	KIR	2014	0.00
4138	Kiribati	KIR	2013	0.00
4137	Kiribati	KIR	2012	0.00
4136	Kiribati	KIR	2011	0.00
0	Afghanistan	AFG	2000	0.00

8683 rows × 4 columns

In [148]:
```
# Filtering & getting descendig solar consumption data for India only
pd = df[df["Entity"].str.contains("India")]
pd = pd.sort_values('Electricity from solar (TWh)', ascending=False)
pd = pd[pd['Electricity from solar (TWh)'] != 0]
print(pd)
sns.pairplot(pd);
# pd.head(10)
```

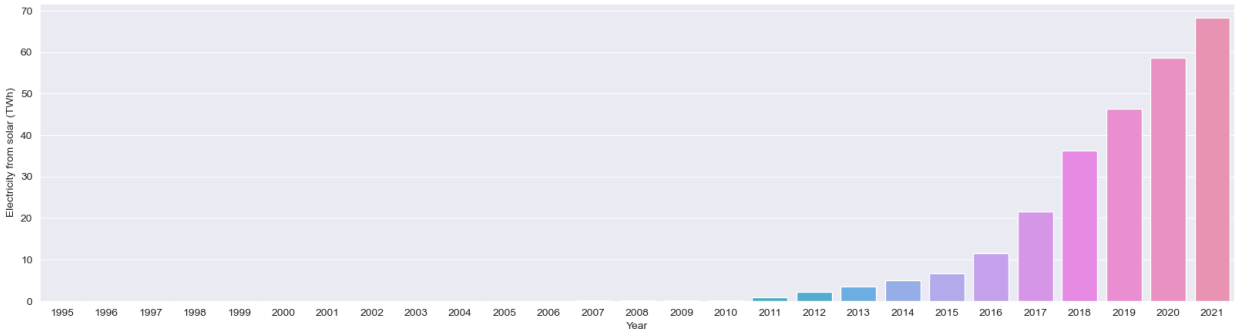
```
     Entity Code  Year  Electricity from solar (TWh)
3625  India  IND  2021                     68.310000
3624  India  IND  2020                     58.680000
3623  India  IND  2019                     46.270000
3622  India  IND  2018                     36.330000
3621  India  IND  2017                     21.540000
3620  India  IND  2016                     11.560000
3619  India  IND  2015                      6.570000
3618  India  IND  2014                      4.910000
3617  India  IND  2013                      3.430000
3616  India  IND  2012                      2.100000
3615  India  IND  2011                      0.830000
3614  India  IND  2010                      0.110000
3613  India  IND  2009                      0.080000
3611  India  IND  2007                      0.060000
3612  India  IND  2008                      0.060000
```

3609	India	IND	2005	0.020000
3608	India	IND	2004	0.020000
3607	India	IND	2003	0.020000
3610	India	IND	2006	0.010000
3606	India	IND	2002	0.010000
3605	India	IND	2001	0.010000
3604	India	IND	2000	0.010000
3603	India	IND	1999	0.006061
3602	India	IND	1998	0.006061
3601	India	IND	1997	0.006061
3600	India	IND	1996	0.006061
3599	India	IND	1995	0.001010



In [160]:
```
# barplot for India
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(20,5))
sns.set_style('darkgrid')
ind_efs = pd['Electricity from solar (TWh)']
ind_year = pd['Year']
sns.barplot(ind_year, ind_efs)
plt.show()
```

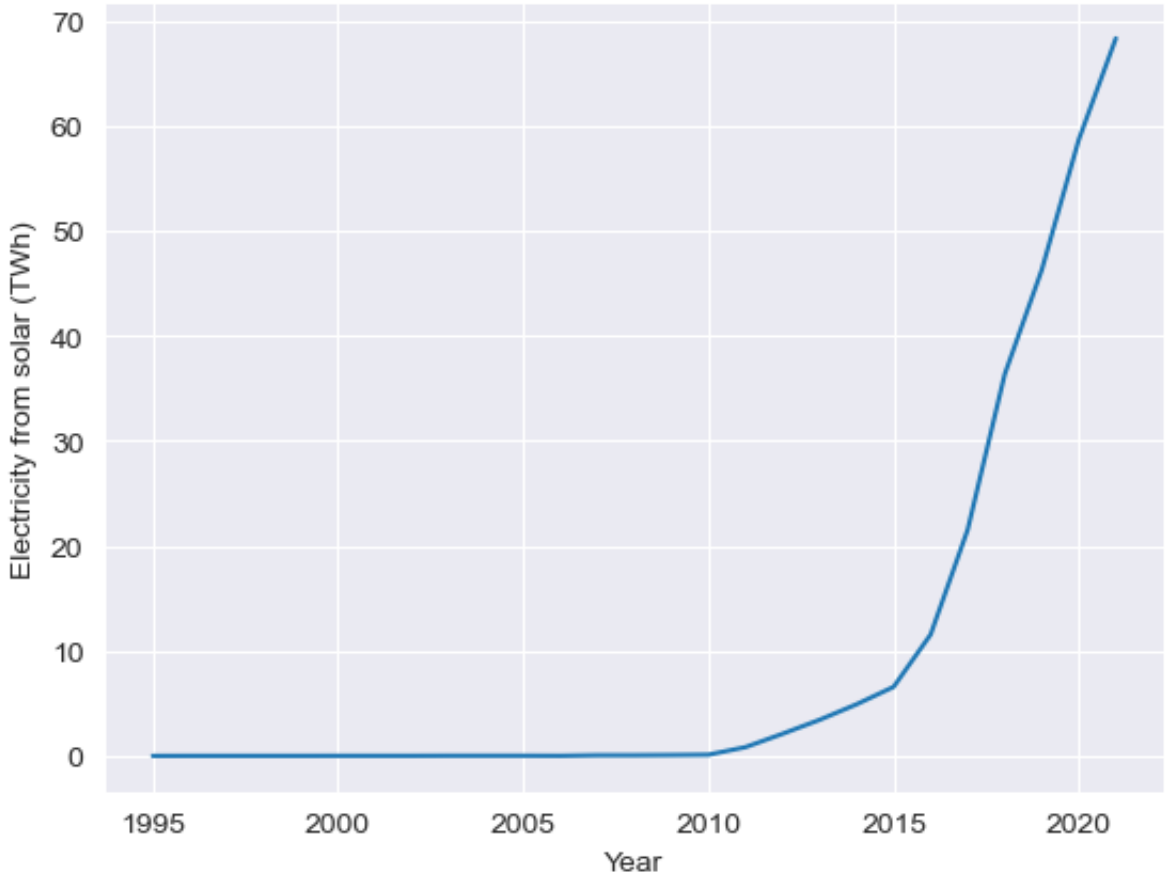
```
/Users/kathan/opt/anaconda3/lib/python3.9/site-packages/seaborn/_deco
rators.py:36: FutureWarning: Pass the following variables as keyword
args: x, y. From version 0.12, the only valid positional argument wil
l be `data`, and passing other arguments without an explicit keyword
will result in an error or misinterpretation.
  warnings.warn(
```



In [161]:
```
# line plot for india
sns.lineplot(ind_year, ind_efs)
```

```
/Users/kathan/opt/anaconda3/lib/python3.9/site-packages/seaborn/_deco
rators.py:36: FutureWarning: Pass the following variables as keyword
args: x, y. From version 0.12, the only valid positional argument wil
l be `data`, and passing other arguments without an explicit keyword
will result in an error or misinterpretation.
  warnings.warn(
```

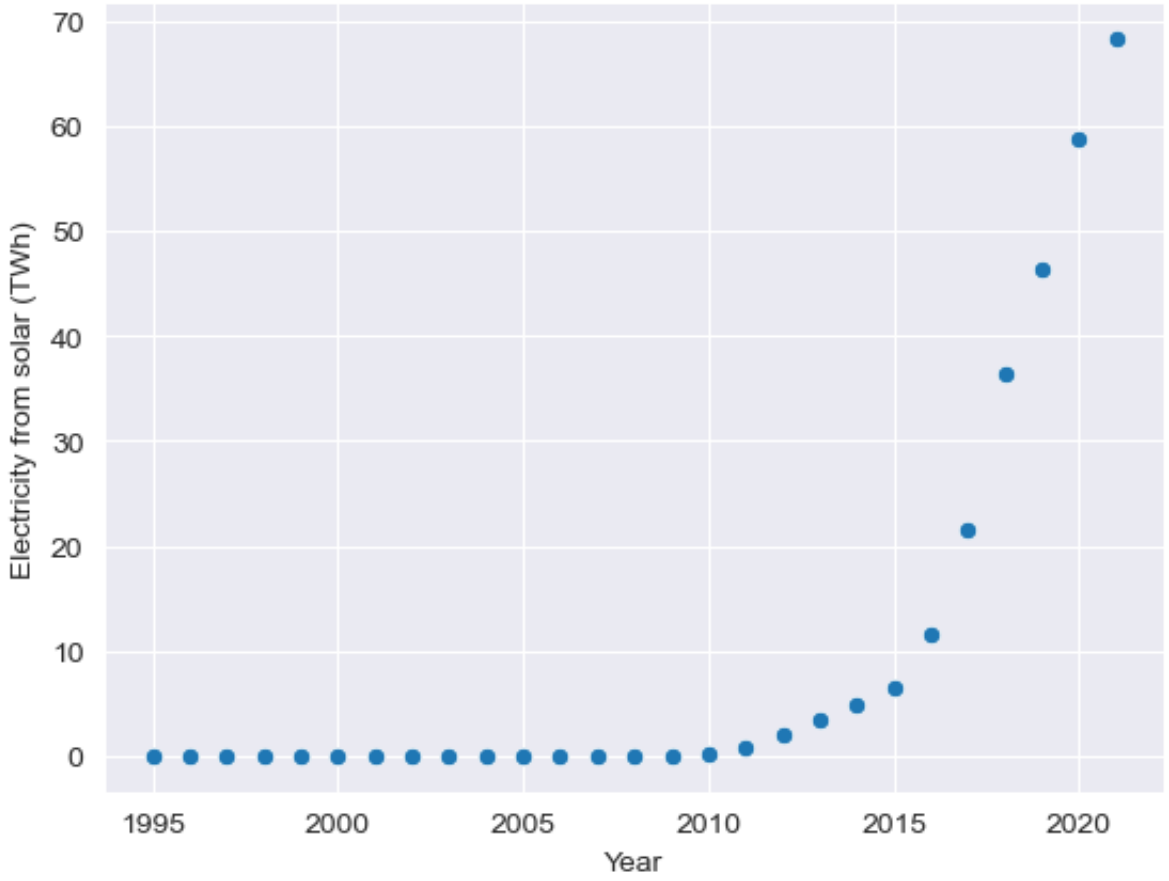
Out[161]:
```
<AxesSubplot:xlabel='Year', ylabel='Electricity from solar (TWh)'>
```



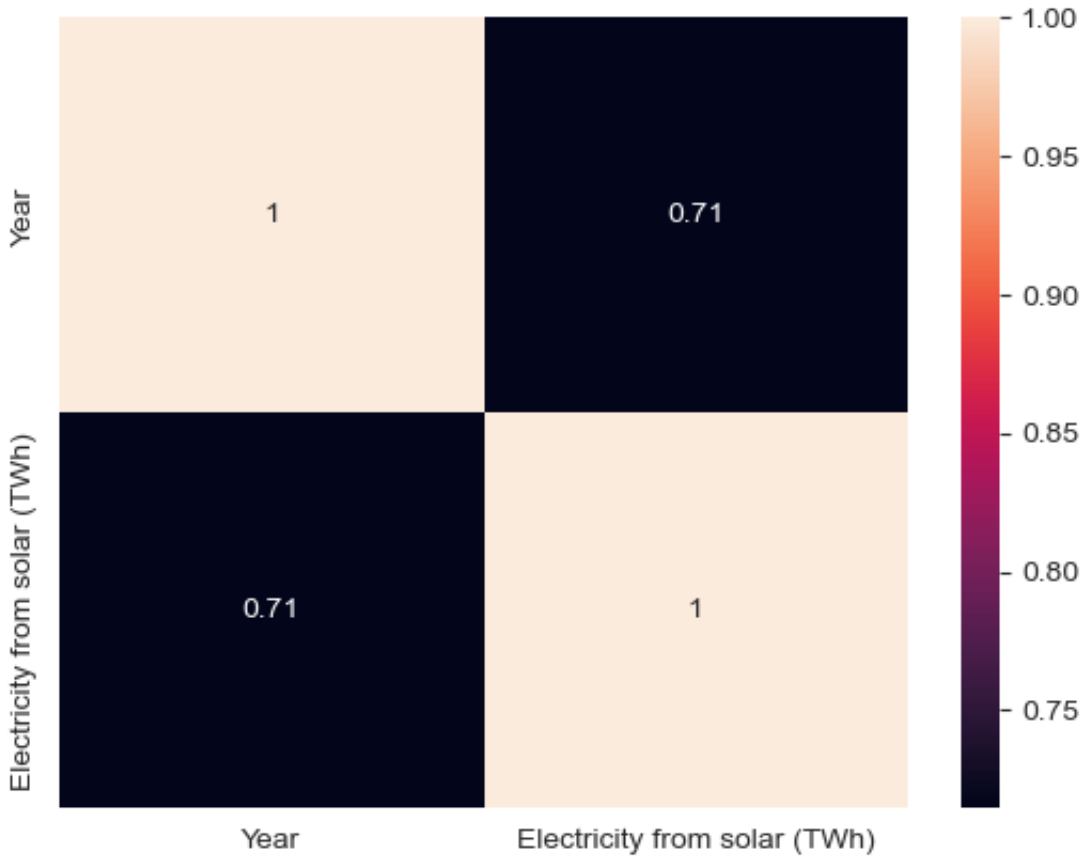
In [111]:
```
# scatterplot for india
sns.scatterplot(year, efs)
```

```
/Users/kathan/opt/anaconda3/lib/python3.9/site-packages/seaborn/_deco
rators.py:36: FutureWarning: Pass the following variables as keyword
args: x, y. From version 0.12, the only valid positional argument wil
l be `data`, and passing other arguments without an explicit keyword
will result in an error or misinterpretation.
  warnings.warn(
```

Out[111]: <AxesSubplot:xlabel='Year', ylabel='Electricity from solar (TWh)'>



In [114]:
```
corr_matrix = pd.corr()
sns.heatmap(corr_matrix, annot=True)
plt.show()
```



In [146]:
```
# getting highest solar consumption for usa only
us = df[df["Entity"].str.contains("United States")]
us = us.sort_values('Electricity from solar (TWh)', ascending=False)
us = us[us['Electricity from solar (TWh)'] != 0]
print(us)
```

```
                        Entity Code  Year  Electricity from solar
(TWh)
8217              United States  USA  2021                   164.
420000
8216              United States  USA  2020                   130.
720000
8215              United States  USA  2019                   106.
890000
8214              United States  USA  2018                    93.
360000
8213              United States  USA  2017                    77.
280000
8212              United States  USA  2016                    54.
870000
```

8211	United States	USA	2015	39.030000
8210	United States	USA	2014	28.920000
8209	United States	USA	2013	9.040000
8208	United States	USA	2012	4.330000
8207	United States	USA	2011	1.820000
8206	United States	USA	2010	1.210000
8205	United States	USA	2009	0.890000
8204	United States	USA	2008	0.860000
8203	United States	USA	2007	0.610000
8200	United States	USA	2004	0.580000
8201	United States	USA	2005	0.550000
8198	United States	USA	2002	0.550000
8192	United States	USA	1996	0.541341
8197	United States	USA	2001	0.540000
8193	United States	USA	1997	0.532696
8199	United States	USA	2003	0.530000
8194	United States	USA	1998	0.524659
8195	United States	USA	1999	0.518342
8191	United States	USA	1995	0.514472
8202	United States	USA	2006	0.510000
8190	United States	USA	1994	0.501873
8196	United States	USA	2000	0.490000
8187	United States	USA	1991	0.478253
8189	United States	USA	1993	0.475162
8188	United States	USA	1992	0.408844

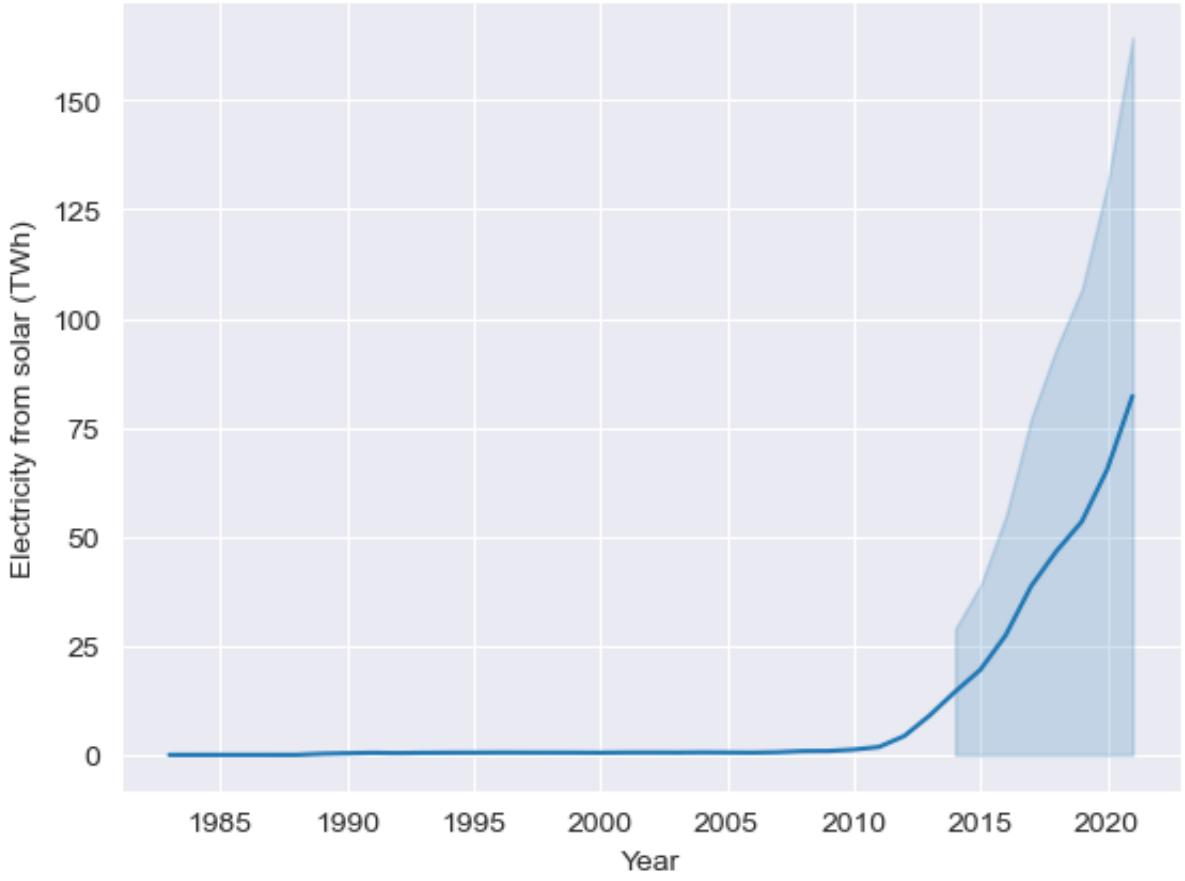
```
8186                United States  USA  1990                    0.
370795
8185                United States  USA  1989                    0.
253132
8239  United States Virgin Islands  VIR  2021                    0.
020000
8182                United States  USA  1986                    0.
014174
8181                United States  USA  1985                    0.
010737
8183                United States  USA  1987                    0.
010603
8232  United States Virgin Islands  VIR  2014                    0.
010000
8234  United States Virgin Islands  VIR  2016                    0.
010000
8238  United States Virgin Islands  VIR  2020                    0.
010000
8233  United States Virgin Islands  VIR  2015                    0.
010000
8237  United States Virgin Islands  VIR  2019                    0.
010000
8236  United States Virgin Islands  VIR  2018                    0.
010000
8235  United States Virgin Islands  VIR  2017                    0.
010000
8184                United States  USA  1988                    0.
009186
8180                United States  USA  1984                    0.
005301
8179                United States  USA  1983                    0.
003000
```

In [155]:
```
# Line chart for usa
import seaborn as sns
import matplotlib.pyplot as plt
# plt.figure(figsize=(20,5))
sns.set_style('darkgrid')
us_efs = us['Electricity from solar (TWh)']
us_year = us['Year']

sns.lineplot(us_year, us_efs)

plt.show()
```

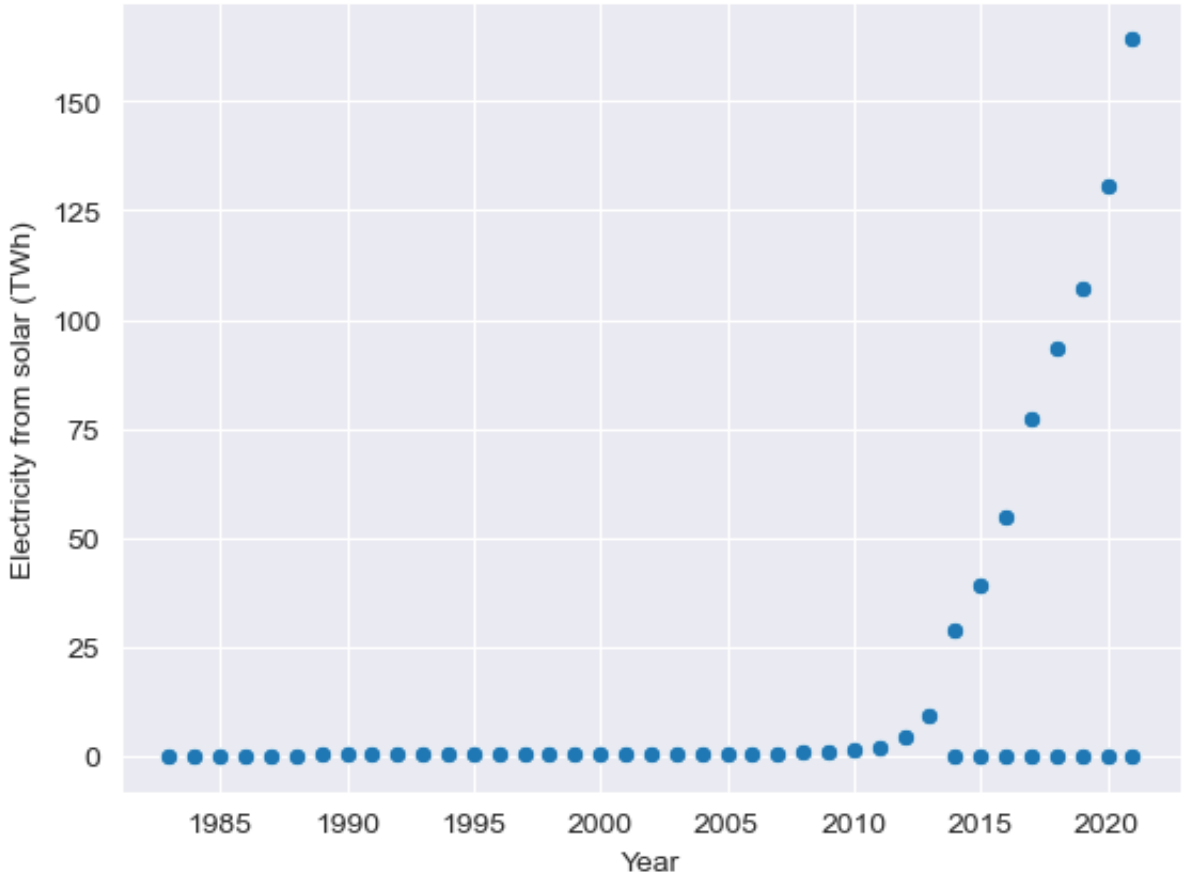
```
/Users/kathan/opt/anaconda3/lib/python3.9/site-packages/seaborn/_deco
rators.py:36: FutureWarning: Pass the following variables as keyword
args: x, y. From version 0.12, the only valid positional argument wil
l be `data`, and passing other arguments without an explicit keyword
will result in an error or misinterpretation.
  warnings.warn(
```



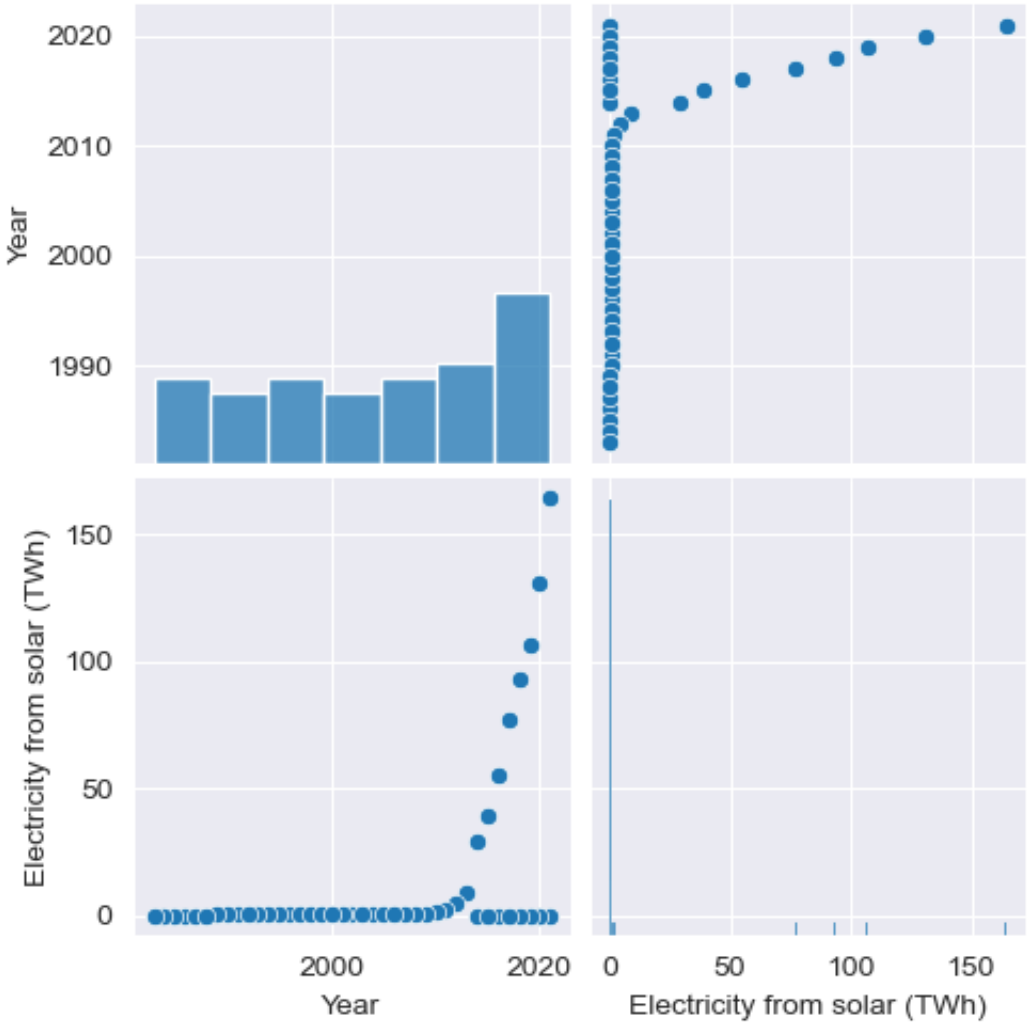
In [137]:
```
# scatterplot for usa
sns.scatterplot(us_year, us_efs)
```

```
/Users/kathan/opt/anaconda3/lib/python3.9/site-packages/seaborn/_deco
rators.py:36: FutureWarning: Pass the following variables as keyword
args: x, y. From version 0.12, the only valid positional argument wil
l be `data`, and passing other arguments without an explicit keyword
will result in an error or misinterpretation.
  warnings.warn(
```

Out[137]: <AxesSubplot:xlabel='Year', ylabel='Electricity from solar (TWh)'>



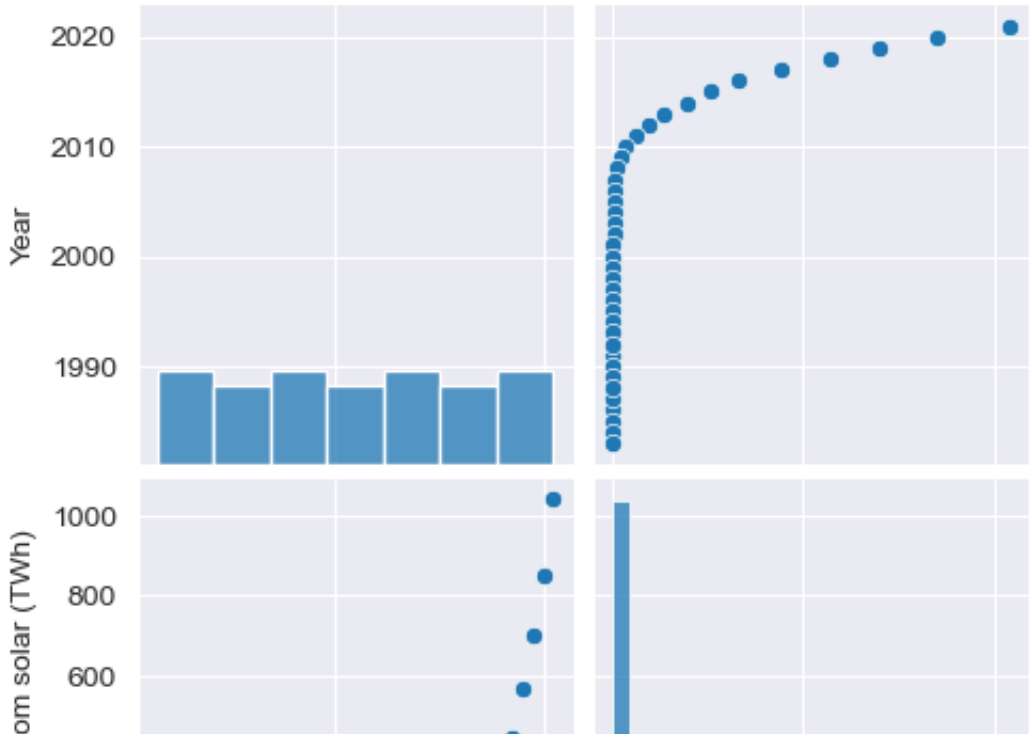
In [149]:
```
sns.pairplot(us);
```

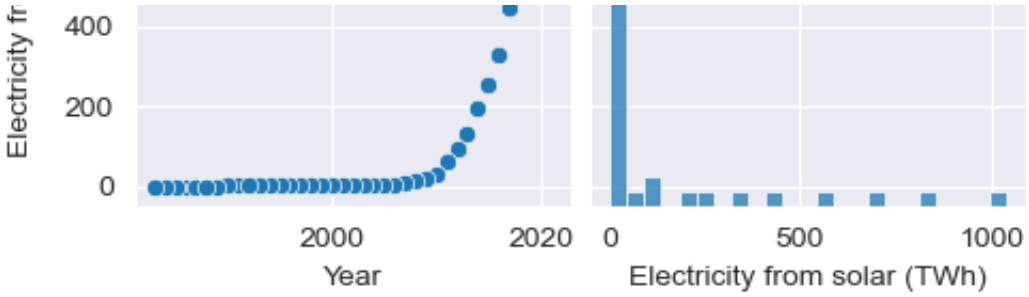


In [152]:
```
# highest to lowest consumption for world
world = df[df["Entity"].str.contains("World")]
world = world.sort_values('Electricity from solar (TWh)', ascending=Fa
world = world[world['Electricity from solar (TWh)'] != 0]
print(world)
```

	Entity	Code	Year	Electricity from solar (TWh)
8616	World	OWID_WRL	2021	1040.500000
8615	World	OWID_WRL	2020	852.100000
8614	World	OWID_WRL	2019	701.190000
8613	World	OWID_WRL	2018	570.570000
8612	World	OWID_WRL	2017	444.540000
8611	World	OWID_WRL	2016	329.150000
8610	World	OWID_WRL	2015	254.870000
8609	World	OWID_WRL	2014	196.460000
8608	World	OWID_WRL	2013	131.420000
8607	World	OWID_WRL	2012	95.180000
8606	World	OWID_WRL	2011	61.850000

8606	World	OWID_WRL	2011	31.850000
8605	World	OWID_WRL	2010	31.050000
8604	World	OWID_WRL	2009	19.190000
8603	World	OWID_WRL	2008	11.360000
8602	World	OWID_WRL	2007	6.920000
8601	World	OWID_WRL	2006	5.110000
8600	World	OWID_WRL	2005	3.780000
8599	World	OWID_WRL	2004	2.710000
8598	World	OWID_WRL	2003	2.070000
8597	World	OWID_WRL	2002	1.690000
8596	World	OWID_WRL	2001	1.350000
8595	World	OWID_WRL	2000	1.080000
8594	World	OWID_WRL	1999	0.905197
8593	World	OWID_WRL	1998	0.811789
8592	World	OWID_WRL	1997	0.749585
8591	World	OWID_WRL	1996	0.699208
8590	World	OWID_WRL	1995	0.638814
8589	World	OWID_WRL	1994	0.597014
8588	World	OWID_WRL	1993	0.556698
8586	World	OWID_WRL	1991	0.505203
8587	World	OWID_WRL	1992	0.466585
8585	World	OWID_WRL	1990	0.388295
8584	World	OWID_WRL	1989	0.262223
8581	World	OWID_WRL	1986	0.015184
8580	World	OWID_WRL	1985	0.011747
8582	World	OWID_WRL	1987	0.010603
8583	World	OWID_WRL	1988	0.010196
8579	World	OWID_WRL	1984	0.006311
8578	World	OWID_WRL	1983	0.003000




Electricity fr
400
200
0
2000
2020
Year
0
500
1000
Electricity from solar (TWh)

In [156]:
```
# Lineplot for world
import seaborn as sns
import matplotlib.pyplot as plt
# plt.figure(figsize=(20,5))
sns.set_style('darkgrid')
world_efs = world['Electricity from solar (TWh)']
world_year = world['Year']

sns.lineplot(world_year,world_efs)

plt.show()
```

```
/Users/kathan/opt/anaconda3/lib/python3.9/site-packages/seaborn/_deco
rators.py:36: FutureWarning: Pass the following variables as keyword
args: x, y. From version 0.12, the only valid positional argument wil
l be `data`, and passing other arguments without an explicit keyword
will result in an error or misinterpretation.
  warnings.warn(
```