In [29]:

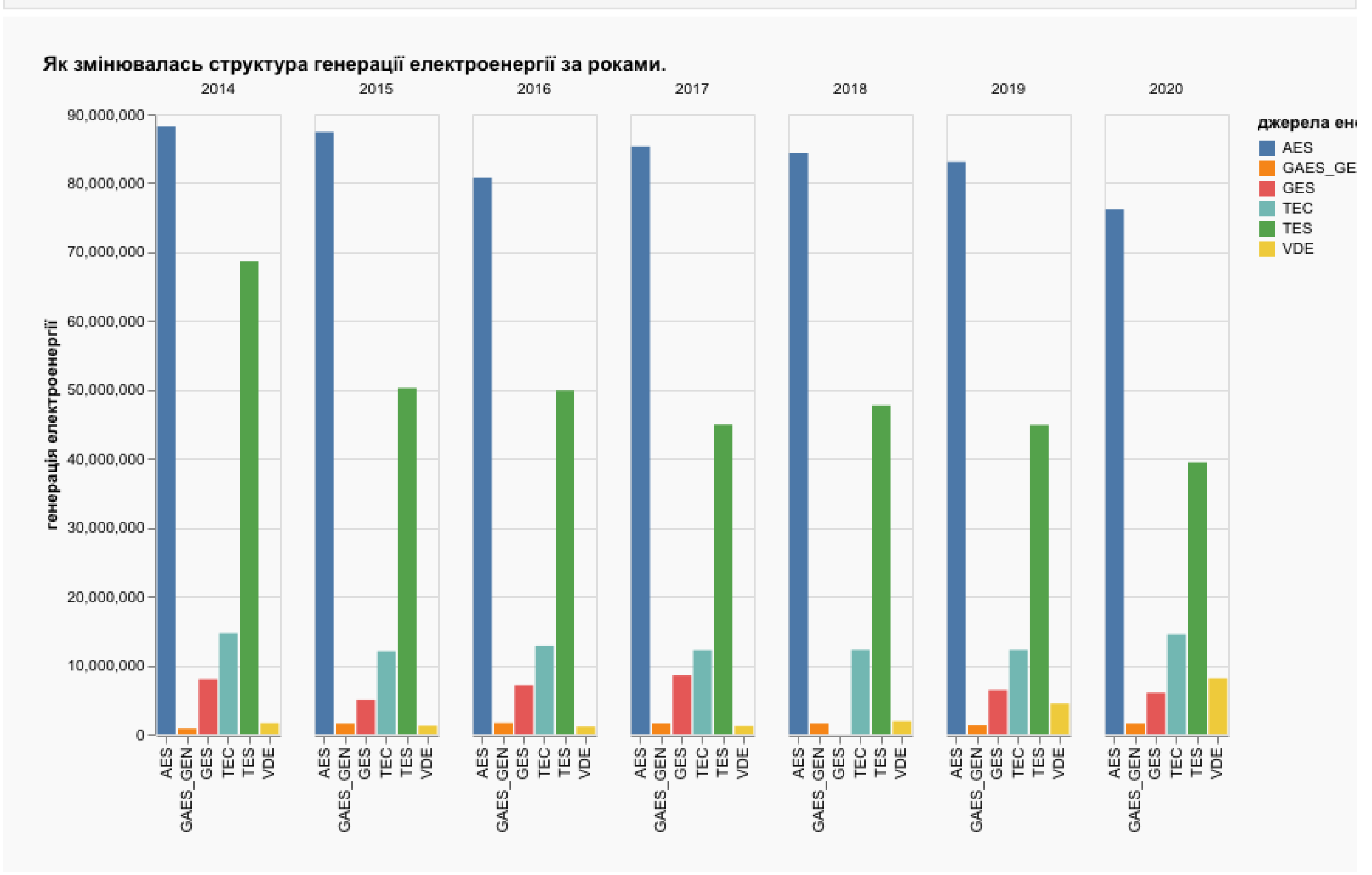
```python
import pandas as pd
import altair as alt
import pyppdf.patch_pyppeteer


# reding the data
df = pd.read_csv("2014-2020.csv")

df.drop(["GAES_PUMP", "UK_BLR_RUS","UK_EURO","UK_MLD", "Unnamed_1", "Unnamed_2"], axis = 1, inplace = True)


df['year'] = df["Час/Дата"].apply(lambda x: int(x[-4:]))



alt.data_transformers.disable_max_rows()


df_new = {"year": [], "type": [], "el": [], 'el_p':[]}


#for all years
for year in range(2014, 2021):
    #take all the data per year
    year_list = df[df["year"] == year]
    # year + type combination
    s = sum(year_list["AES"]) + sum(year_list["TEC"])+ sum(year_list["VDE"]) + sum(year_list["TES"]) + sum(year
    for exact_type in ["AES", "TEC", "VDE", "TES", "GES", "GAES_GEN"]:

        el_type_total = sum(year_list[exact_type])
        df_new["year"].append(year)
        df_new["type"].append(exact_type)
        df_new["el_p"].append(el_type_total/s*100)
        df_new["el"].append(el_type_total)

df_new = pd.DataFrame.from_dict(df_new)
#no NA
df_new["el"].fillna(0)

df1 = df_new
```

```
    year      type          el       el_p
0   2014       AES  88204418.0  48.481993
1   2014       TEC  14684731.0   8.071535
2   2014       VDE   1606386.0   0.882958
3   2014       TES  68605877.0  37.709558
4   2014       GES   7991074.0   4.392333
5   2014  GAES_GEN    839842.0   0.461623
6   2015       AES  87413763.0  55.511754
7   2015       TEC  12041074.0   7.646635
8   2015       VDE   1234936.0   0.784241
9   2015       TES  50259819.0  31.917293
10  2015       GES   4964478.0   3.152672
11  2015  GAES_GEN   1554857.0   0.987406
12  2016       AES  80762094.0  52.678526
13  2016       TEC  12841033.0   8.375794
14  2016       VDE   1093190.0   0.713053
15  2016       TES  49879543.0  32.534828
16  2016       GES   7113989.0   4.640227
17  2016  GAES_GEN   1621377.0   1.057572
18  2017       AES  85314258.0  55.488333
19  2017       TEC  12208062.0   7.940115
20  2017       VDE   1177032.0   0.765541
21  2017       TES  44945303.0  29.232393
22  2017       GES   8531952.0   5.549175
23  2017  GAES_GEN   1575099.0   1.024443
24  2018       AES  84351328.0        NaN
25  2018       TEC  12261018.0        NaN
26  2018       VDE   1863726.0        NaN
27  2018       TES  47747092.0        NaN
28  2018       GES         NaN        NaN
29  2018  GAES_GEN   1564867.0        NaN
30  2019       AES  83098265.0  54.521108
31  2019       TEC  12251850.0   8.038488
32  2019       VDE   4441801.0   2.914284
33  2019       TES  44877587.0  29.444367
34  2019       GES   6422154.0   4.213601
35  2019  GAES_GEN   1323192.0   0.868152
36  2020       AES  76210883.0  52.221973
37  2020       TEC  14536545.0   9.960875
38  2020       VDE   8124734.0   5.567310
39  2020       TES  39504024.0  27.069337
40  2020       GES   5998680.0   4.110475
41  2020  GAES_GEN   1561564.0   1.070030
```

In [57]:

```python
#- Як змінювалась структура генерації електроенергії за роками?

alt.Chart(df1).mark_bar().encode(
    x = alt.X('type:O', title = ''),
    y = alt.Y('el:Q', title = 'генерація електроенергії' ),
    color = alt.Color('type:N', title = 'джерела енергії'),
    column = alt.Column('year:O', title = '')
).properties(width = 80, height = 400, background = '#F9F9F9', padding = 25, title = 'Як змінювалась структура

# чому вибрала цей варіант.
# 6 років ми можемо зобразити як окремі графіки, це не ріже очі.
# bar chart добре показати взаємовіношення між елементами
# альтернативи.
# 1) line chart - нижній персентель значно менший ніж вищий, складно побачити його буде якщо це
# буде лінія + буде перекриття один одного.
# 2) area chart - складно побачити реальне співвідношення
# 3) staked bar chart - складно побачити реальне співвідношення, скаче
# pie chart - складно побачити реальне співвідношення, скаче

# графік можна нормалзуватищоб було краще видно зміну нижнього персентилю, але тоді не буде видно
#реального співвідношення.
# якщо ми робили цю генерація для звичайних людей, то не дуже зрозуміло що таке є AES, GES, TEC ...
```

Out[57]:



In [35]:

```python
##- Як залежить споживання електроенергії від дня року та години доби?

# new data for the list

df["month"] = df["Час/Дата"].apply(lambda x: int(x[-7:-5]))

df["day"] = df["Час/Дата"].apply(lambda x: x[-10:-8])

df["hour"] = df["Час/Дата"].apply(lambda x: x.split("-")[0])


df["no_year"] = df["Час/Дата"].apply(lambda x: x[:-5])
```

In [36]:

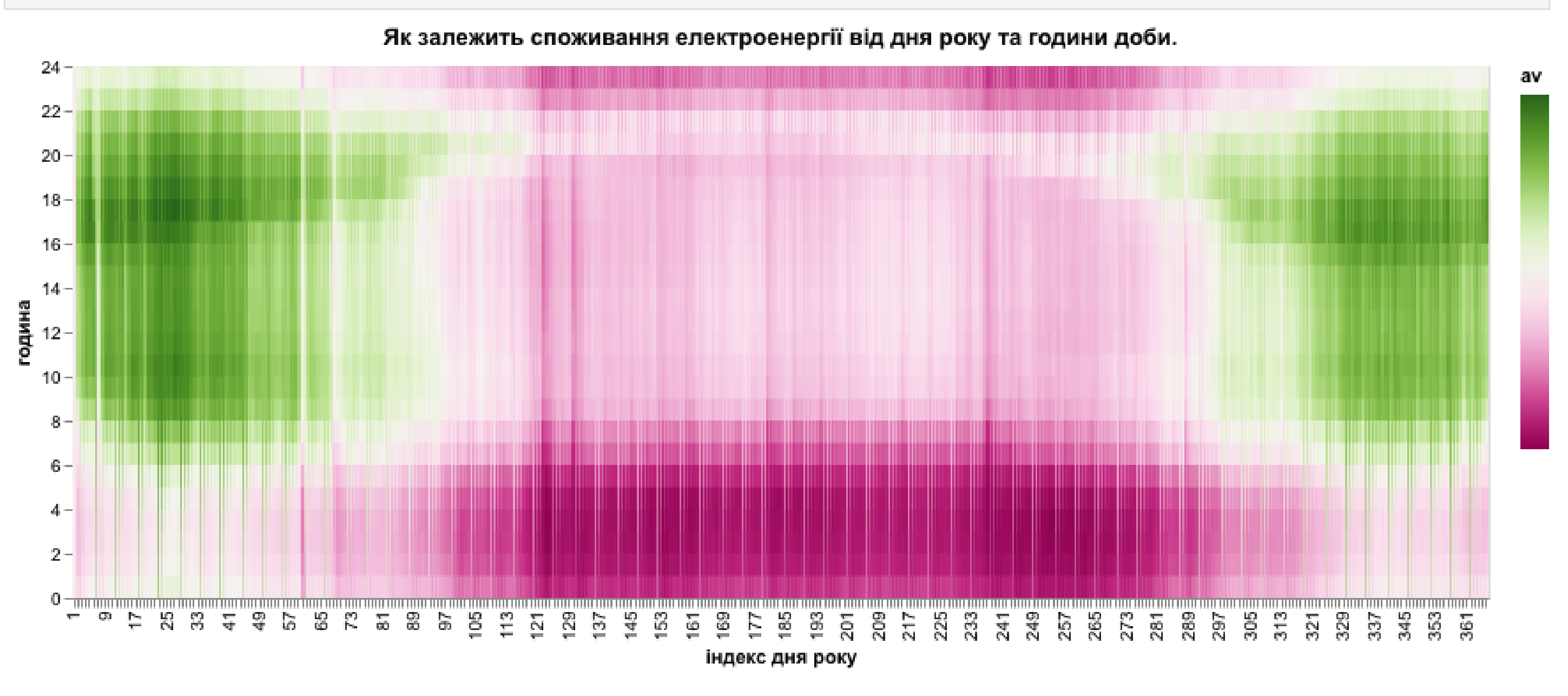
```python
#create a new list with day_of_year_n + hour + av_consumption

df2 = {"hour": [], "day": [], "av": []}
day = 367

for i in range(0, 8761):
    row = df.loc[i]
    cut = df[df["no_year"] == row["no_year"]]

    l = len(cut)
    av = sum(cut["CONSUMPTION"])/l

    df2["hour"].append(row["hour"])
    if row["hour"] == '24':
        day = day - 1
    df2["day"].append(day)
    df2["av"].append(av)

df2 = pd.DataFrame.from_dict(df2)


print(df2)
```

```
     hour  day            av
0      24  366  18122.571429
1      23  366  19178.428571
2      22  366  20010.571429
3      21  366  20662.285714
4      20  366  21152.000000
...   ...  ...           ...
8756    4    2  15869.000000
8757    3    2  15918.714286
8758    2    2  16115.714286
8759    1    2  16632.000000
8760   24    1  17408.285714

[8761 rows x 3 columns]
```

In [51]:

```python
#Як залежить споживання електроенергії від дня року та години доби?
alt.Chart(df2).mark_rect().encode(
    x = alt.X('day:O', title = "індекс дня року"),
    y = alt.Y('hour:Q', sort = alt.SortField('descending'), title ='година'),
    color = alt.Color('av:Q', scale = alt.Scale(scheme = 'pinkyellowgreen'))
).properties(width = 800, title = 'Як залежить споживання електроенергії від дня року та години доби.')

# heatmap так як вона дозволяє дивитись дані одразу по трьом кордининатам
# так як нам потрібно зобразити дуже великий набір даних, то найкреще розрізняти по найбільш читабельному
#параметру, коліру.
# можна спробувати mark_circle, але він не так добре читається
# можна зробити 24 чарти для кожної години дня, але це тільки зупсує все.
```

Out[51]:



In [52]:

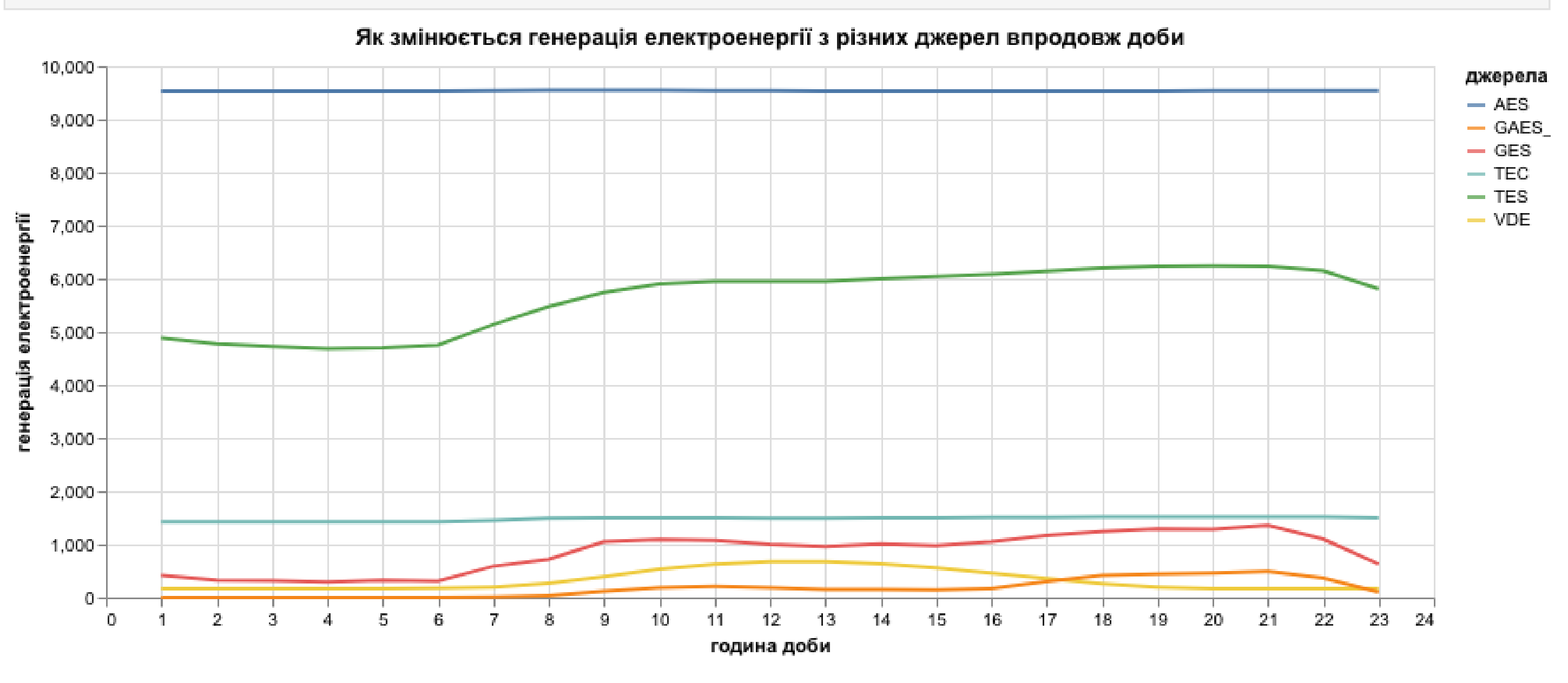
```python
#- Як змінюється генерація електроенергії з різних джерел впродовж доби?

df3 = {"hour": [], "type": [], "av": []}

for hour in range(1, 24):
    hour_list = df[df["hour"] == str(hour)]
    for exact_type in ["AES", "TEC", "VDE", "TES", "GES", "GAES_GEN"]:
        av = sum(hour_list[exact_type])/len(hour_list)
        df3["hour"].append(hour)
        df3["type"].append(exact_type)
        df3["av"].append(av)

df3 = pd.DataFrame.from_dict(df3)


print(df3)
```

```
     hour      type           av
0       1       AES  9533.777865
1       1       TEC  1428.560813
2       1       VDE   164.178334
3       1       TES  4883.335549
4       1       GES   413.657020
..    ...       ...          ...
133    23       TEC  1500.018772
134    23       VDE   164.157215
135    23       TES  5807.820493
136    23       GES   629.127493
137    23  GAES_GEN    98.544388

[138 rows x 3 columns]
```

In [55]:

```python
alt.Chart(df3).mark_line().encode(
    x = alt.X('hour:Q', title = "година доби"),
    y = alt.Y('av:Q', title = 'генерація електроенергії'),
    color = alt.Color('type:N', title = 'джерела енергії')
).properties(width = 750, title = 'Як змінюється генерація електроенергії з різних джерел впродовж доби')

#- Як змінюється генерація електроенергії з різних джерел впродовж доби?

# line chart добре показує зміни в часі.
# Один графік так як Для розділення по годинах не потрібно розділяти графіки так як це не читабельно
# Можна було спробувати гістограму для кожного типу електроенергії, але краще коли вони всі разом на одному гра
# та видно співвідношення.
# Barchart не дуже добре ідея так як він збиває співвідношення та порядок
# area - можн також збивати, особливо нижній персентіль. Вони перетинаються і будуть полігони посеред графіку.
```

Out[55]:



In [58]:

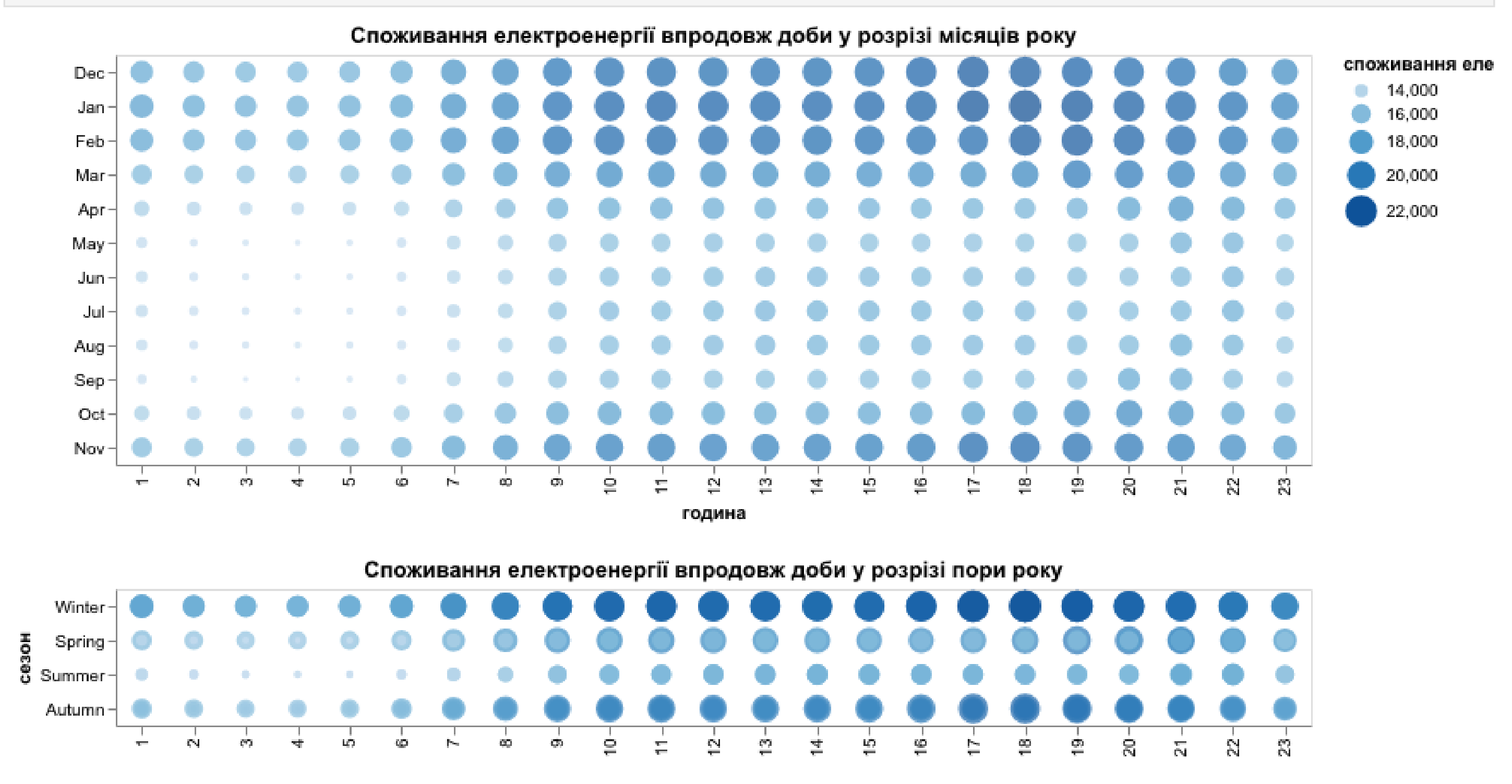
```python
#- Як змінюється споживання електроенергії впродовж доби у розрізі місяців року та пір року?

df4 = {"hour": [], "month": [], "av": []}

for hour in range(1, 24):
    hour_list = df[df["hour"] == str(hour)]
    for month in (1,2,3,4,5,6,7,8,9,10,11,12):
        month_hour_list = hour_list[hour_list["month"] == month]

        av = sum(month_hour_list["CONSUMPTION"])/len(month_hour_list)

        df4["hour"].append(hour)
        df4["month"].append(month)
        df4["av"].append(av)

df4 = pd.DataFrame.from_dict(df4)


print(df4)
```

```
     hour  month            av
0       1      1  17855.262673
1       1      2  17505.383838
2       1      3  16270.645161
3       1      4  14712.419048
4       1      5  13518.317972
..    ...    ...           ...
271    23      8  15252.290323
272    23      9  14992.033333
273    23     10  16605.069124
274    23     11  18082.919048
275    23     12  19053.244240

[276 rows x 3 columns]
```

In [71]:

```python
df4['text_month'] = df4['month'].map({1: 'Jan', 2: 'Feb',  3: 'Mar', 4: 'Apr', 5: 'May', 6: 'Jun', 7: 'Jul', 8:
df4['season'] = df4['month'].map({1: 'Winter', 2: 'Winter',  3: 'Spring', 4: 'Spring', 5: 'Spring', 6: 'Summer'



p1 = alt.Chart(df4).mark_circle().encode(
    y = alt.Y('text_month:O', title = "", sort=['Dec','Jan','Feb', 'Mar','Apr','May','Jun','Jul','Aug','Sep','O
    x = alt.X('hour:O', sort = alt.SortField('descending'), title ='година'),
    size = alt.Size('av',  scale = alt.Scale(zero = False)),
    color = alt.Color('av',  scale = alt.Scale(zero = False), title= 'споживання електроенергії')
).properties(width = 700, title = 'Споживання електроенергії впродовж доби у розрізі місяців року')

p2 = alt.Chart(df4).mark_circle().encode(
    x = alt.X('hour:O', title = "", sort=['Winter', 'Spring', 'Summer', 'Autumn']),
    y = alt.Y('season:O', sort = alt.SortField('descending'), title = 'сезон'),
    size = alt.Size('av',  scale = alt.Scale(zero = False)),
    color = alt.Color('av',  scale = alt.Scale(zero = False))
).properties(width = 700, title= 'Споживання електроенергії впродовж доби у розрізі пори року')

alt.vconcat(p1, p2)

#- Як змінюється споживання електроенергії впродовж доби у розрізі місяців року та пір року?

# Ті самі аргументи, що й для "споживання електроенергії від дня року та години доби"
# Але тут менше даних, тому можна спробувати розмірами та прозорістю передати значення чисел.

# Спочатку я хотіла промалювати вертикальні лінії на графіку p1 задля того щоб розрізнити пори року.
# Але через них гірше видно загальньу картину співвідношення Місяць-година-споживання.
# Тому година - сезон- споживання  це окремий графік
```

Out[71]:



In [72]:

```python
#- Як змінюється споживання електроенергії впродовж тижня?
df["weekday"] = df["day"]
day = 0
for i in range(0, 61368):
    row = df.loc[i]
    hour = row["hour"]
    if hour == '24':
        day = day + 1
    day = day%7
    df["weekday"][i] = day

df['weekday'] = df['weekday'].map({1: 'Thu', 2: 'Wed',  3: 'Tue', 4: 'Mon', 5: 'Sun', 6: 'Sat', 0: "Fri"})


df5 = {"weekday":[], "av" :[]}
for weekday in ['Mon', 'Sun','Sat', "Fri", 'Thu', 'Wed', 'Tue']:
    weekday_list = df[df["weekday"] == str(weekday)]
    av = sum(weekday_list["CONSUMPTION"])/len(weekday_list)
    df5["weekday"].append(weekday)
    df5["av"].append(av)


df5 = pd.DataFrame.from_dict(df5)
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages/ipykernel_launcher.py:10: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.
```

In [79]:

```python
alt.Chart(df5).mark_bar().encode(
    x = alt.X('weekday:O', title = "", sort = ['Mon', 'Tue', 'Wed', 'Thu',  "Fri", 'Sat','Sun']),
    y = alt.Y('av:Q', scale = alt.Scale(
                    zero = False), title = 'споживання електроенергії')
).properties(width = 850, title = 'Як змінюється споживання електроенергії впродовж тижня')


# barchart гарно показує співвідношення та зміни у часі.
# linechart , area chart теж можуть тут бути, це гарні варіанти
# недолік цієї візуалізації це те, що вона показує лише серенє значення за рік, що не є точним числом.
# насправіді, на графіку "споживання електроенергії від дня року та години доби" також видно, що
# два дня тижня регулярно даєть набагато менший резулат, ніж інші дні.
```

Out[79]: