C_b_2 : Recherche de modèles : Prophet

Utilisons maintenant le modèle développé par Facebook (Meta) pour modéliser des séries temporelles.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
import locale
import calendar
import holidays
from rich import print
from datetime import date
from references import *
from src import *

from prophet import Prophet

from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error as mae
from sklearn.metrics import mean_squared_error as mse
from sklearn.model_selection import TimeSeriesSplit

import warnings

warnings.filterwarnings("ignore")
plt.style.use("fivethirtyeight")
pd.options.mode.chained_assignment = None
pd.set_option("display.max_columns", 500)
pd.set_option("display.width", 1000)

locale.setlocale(locale.LC_ALL, "fr_CA.UTF-8")
```

```
Importing plotly failed. Interactive plots will not work.
```

Out[1]:
```
'fr_CA.UTF-8'
```

In [2]:
```
# %load_ext jupyter_black

import black
import jupyter_black

jupyter_black.load(
    lab=True,
    line_length=55,
    target_version=black.TargetVersion.PY311,
)
```

Prophet

> Prophet is a framework for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust in regard to missing data and shifts in trend, and typically handles outliers well.

Exemple qui a inspiré ce sujet

Rob Mulla

- https://youtu.be/j0eioK5edqg?si=jW2169K9XA5xdbPP
- https://www.kaggle.com/code/robikscube/time-series-forecasting-with-prophet-yt/notebook

Autre exemple avec variables multiples :

Exemple / explications simples : https://stackoverflow.com/questions/54544285/is-it-possible-to-do-multivariate-multi-step-forecasting-using-fb-prophet

Import des données et créations des caractéristiques (*features*)

In [3]:
```
(
    df,
    InfoDates,
) = import_and_create_features_no_categorical(
    lags=[
        1,
        2,
        3,
        4,
        6,
        24,
        364 * 24 * 1,
        364 * 24 * 2,
```

```
        364 * 24 * 3,
    ],
    fenetres=[1, 2, 3, 4, 6, 8, 12, 16, 24],
    fin="20221231",
    getInfoDate=True,
)

df = df.dropna()

FEATURES = df.columns.to_list()[
    1:
]  # Enlevons MW en première colonne
```

Transformation pour le modèle Prophet

La cible doit être `y` et les valeurs de temps `ds`.

In [4]:
```
df["ds"] = df.index
df = df.rename(columns={"MW": "y"})
```

In [5]:
```
df.head()
```

Out[5]:

date	y	Temp	hourofday	quarter	year	dayofyear	dayofmonth	weekof
2021-12-23 21:00:00	32859.50	-12.3	21	4	2021	357	23	
2021-12-23 22:00:00	32274.67	-12.8	22	4	2021	357	23	
2021-12-23 23:00:00	31169.27	-12.8	23	4	2021	357	23	
2021-12-24 00:00:00	30591.23	-13.1	0	4	2021	358	24	
2021-12-24 01:00:00	30032.52	-13.2	1	4	2021	358	24	

Création des ensembles d'apprentissage et validation

In [6]:
```
date_slit = "2022-01-01"
```

```
df_train = df.iloc[df.index < date_slit]
df_test = df.iloc[df.index >= date_slit]
```

Création du modèle sans les caractéristiques

Le modèle utilisera seulement les dates, mais effectuera une déduction des caractéristiques de date par lui-même.

In [7]:
```
m = Prophet()

m.fit(df_train)

forecast = m.predict(df_test.drop(columns="y"))
```

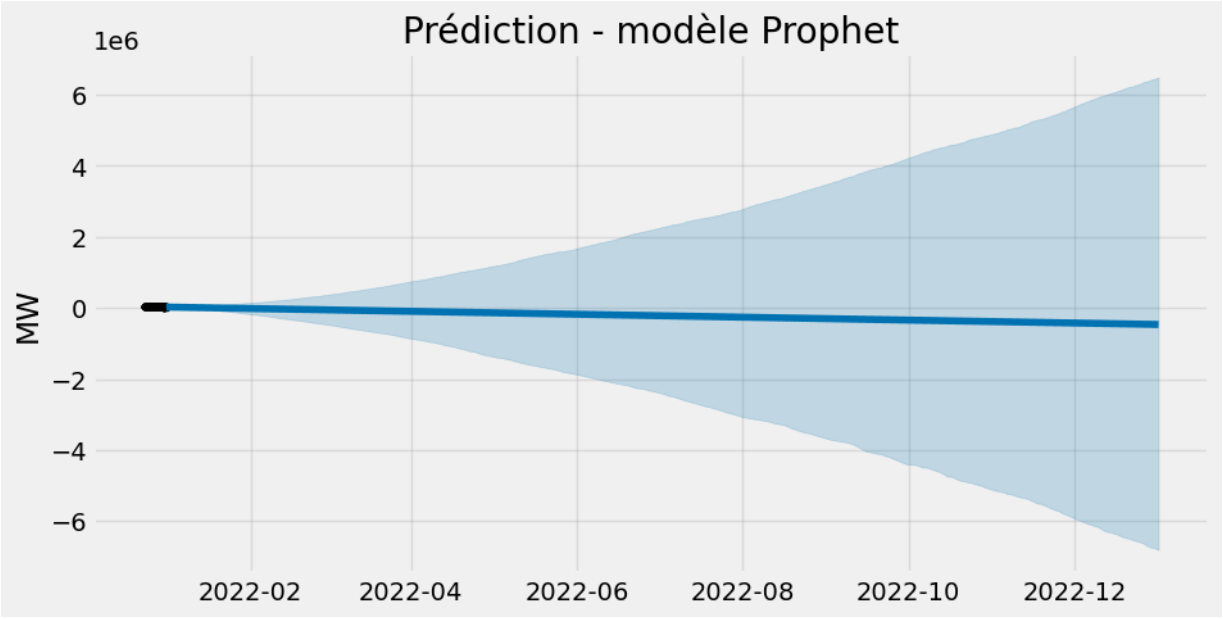
```
INFO:prophet:Disabling yearly seasonality. Run prophet with yearly_seasonali
ty=True to override this.
INFO:prophet:Disabling weekly seasonality. Run prophet with weekly_seasonali
ty=True to override this.
DEBUG:cmdstanpy:input tempfile: /var/folders/h1/zbfgvczd009f52sgp7ld62500000
gn/T/tmpno48uf50/egihck8i.json
DEBUG:cmdstanpy:input tempfile: /var/folders/h1/zbfgvczd009f52sgp7ld62500000
gn/T/tmpno48uf50/vpix3pxa.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/opt/homebrew/anaconda3/envs/sci1402/lib/pyt
hon3.11/site-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed
=49498', 'data', 'file=/var/folders/h1/zbfgvczd009f52sgp7ld62500000gn/T/tmpn
o48uf50/egihck8i.json', 'init=/var/folders/h1/zbfgvczd009f52sgp7ld62500000g
n/T/tmpno48uf50/vpix3pxa.json', 'output', 'file=/var/folders/h1/zbfgvczd009f
52sgp7ld62500000gn/T/tmpno48uf50/prophet_modelt952vqa_/prophet_model-2023120
4222843.csv', 'method=optimize', 'algorithm=lbfgs', 'iter=10000']
22:28:43 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
22:28:43 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```

In [8]:
```
forecast.head()
```

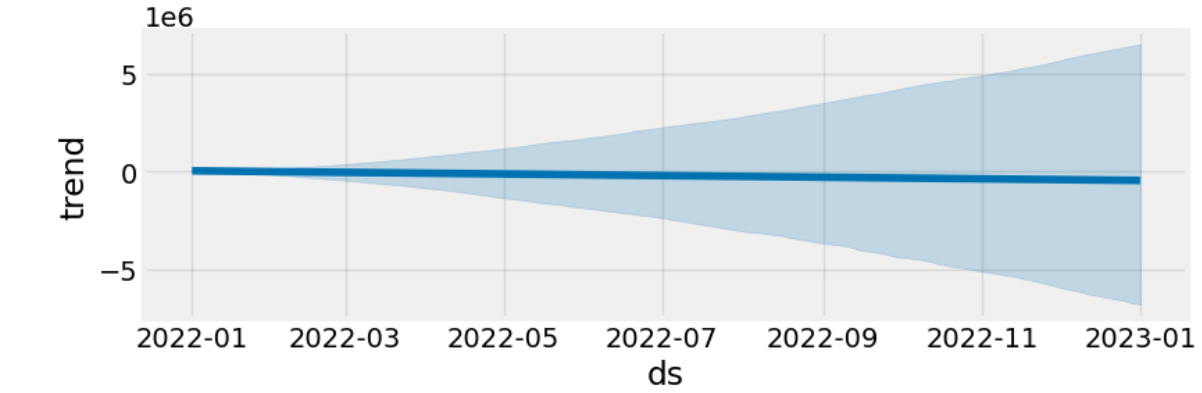
Out[8]:

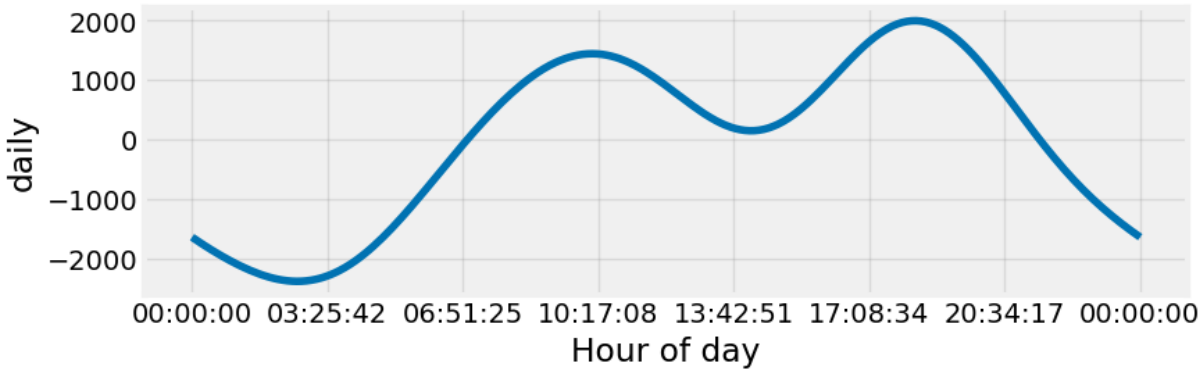
	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_uppe
0	2022-01-01 00:00:00	24947.323573	22694.373939	23910.008162	24947.323573	24947.32357
1	2022-01-01 01:00:00	24891.227646	22205.515272	23461.679801	24887.871974	24894.58762
2	2022-01-01 02:00:00	24835.131718	21862.931665	23144.183827	24818.485673	24852.62682
3	2022-01-01 03:00:00	24779.035790	21775.415084	23044.902682	24749.026419	24817.15075
4	2022-01-01 04:00:00	24722.939862	21961.639554	23241.580182	24674.674855	24784.06041

In [9]:
```
fig, ax = plt.subplots(figsize=(10, 5))
fig = m.plot(forecast, ax=ax)
ax.set_title("Prédiction - modèle Prophet")
ax.set_ylabel("MW")
ax.set_xlabel("")
plt.show()
```



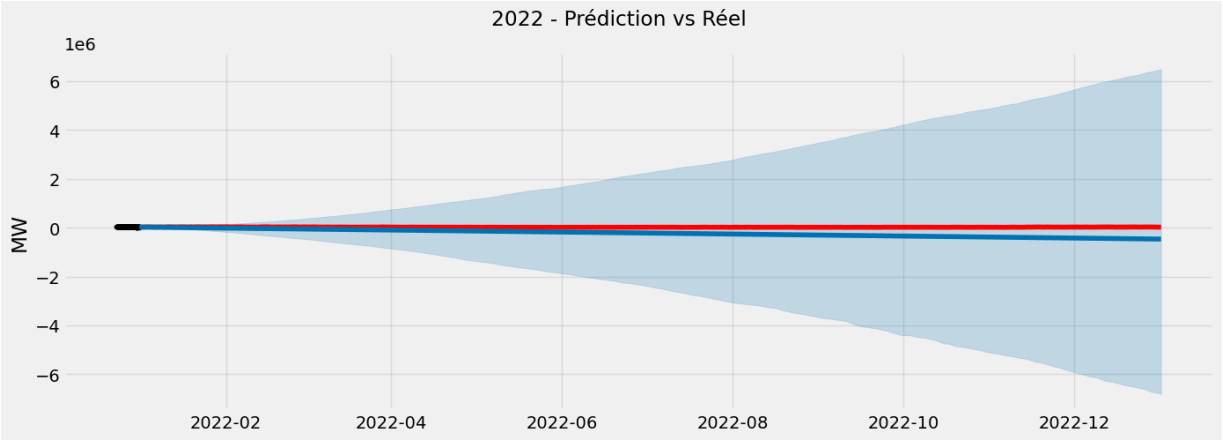
In [10]:
```
fig = m.plot_components(forecast)
plt.show()
```





Comparaison avec les données réelles

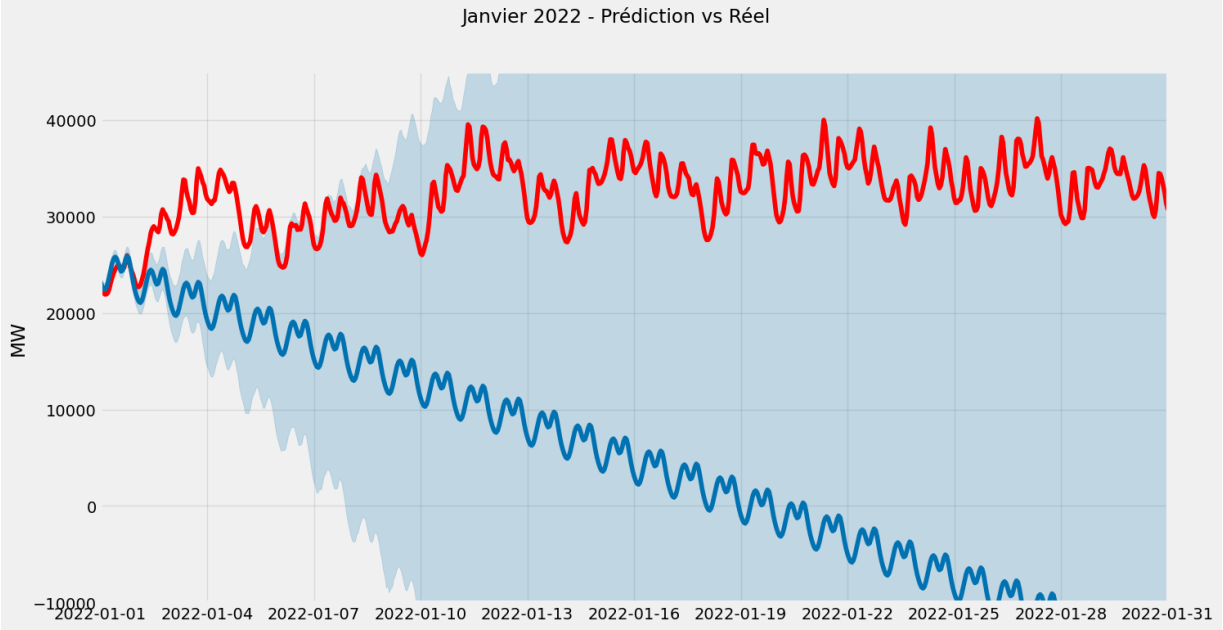
In [11]:
```
f, ax = plt.subplots(figsize=(15, 5))
ax.plot(
    df_test.index, df_test["y"], color="r"
)  # Données rééelles
fig = m.plot(forecast, ax=ax)  # Prédictions
ax.set_ylabel("MW")
ax.set_xlabel("")
plot = plt.suptitle("2022 - Prédiction vs Réel")
```



Nous ne pouvons pas voir grand-chose avec cette échelle.

In [12]:
```
fig, ax = plt.subplots(figsize=(15, 8))
ax.plot(df_test.index, df_test["y"], color="r")
fig = m.plot(forecast, ax=ax)
```

```
ax.set_xbound(
    lower=datetime(2022, 1, 1),
    upper=datetime(2022, 1, 31),
)
ax.set_ylim(-10_000, 45_000)
ax.set_ylabel("MW")
ax.set_xlabel("")
plot = plt.suptitle(
    "Janvier 2022 - Prédiction vs Réel"
)
```



Nous pouvons voir que le modèle décroche complètement. Il est en mesure de reproduire les variations quotidiennes (la double bosse de demande), mais la tendance est grandement à la baisse, projetant des valeurs de demande sous zéro après une vingtaine de jours.

Cela ne fonctionne pas : essayons avec l'ajout de caractéristiques.

Création d'un modèle avec le delat T et la moyenne mobile de la température sur 24h

Ajoutons de 2 caractéristiques : `Temp_MOYMOBILE_t-24h` et `DT_18`

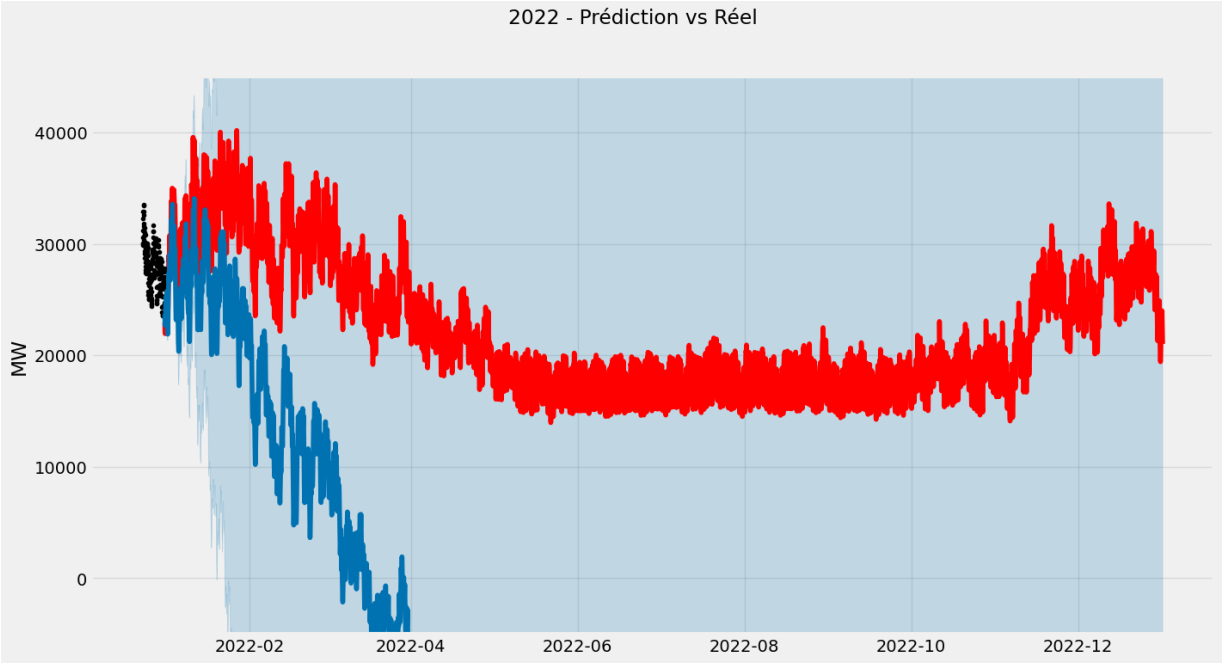
Comme cela fonctionnait bien dans l'autre modèle (importance relative élevée), utilisons une caractéristique supplémentaire dans ce modèle.

In [13]:
```
m = Prophet()

m.add_regressor("Temp_MOYMOBILE_t-24h")
m.add_regressor("DT_18")

m.fit(df_train)
```

```
forecast = m.predict(df_test.drop(columns="y"))
```

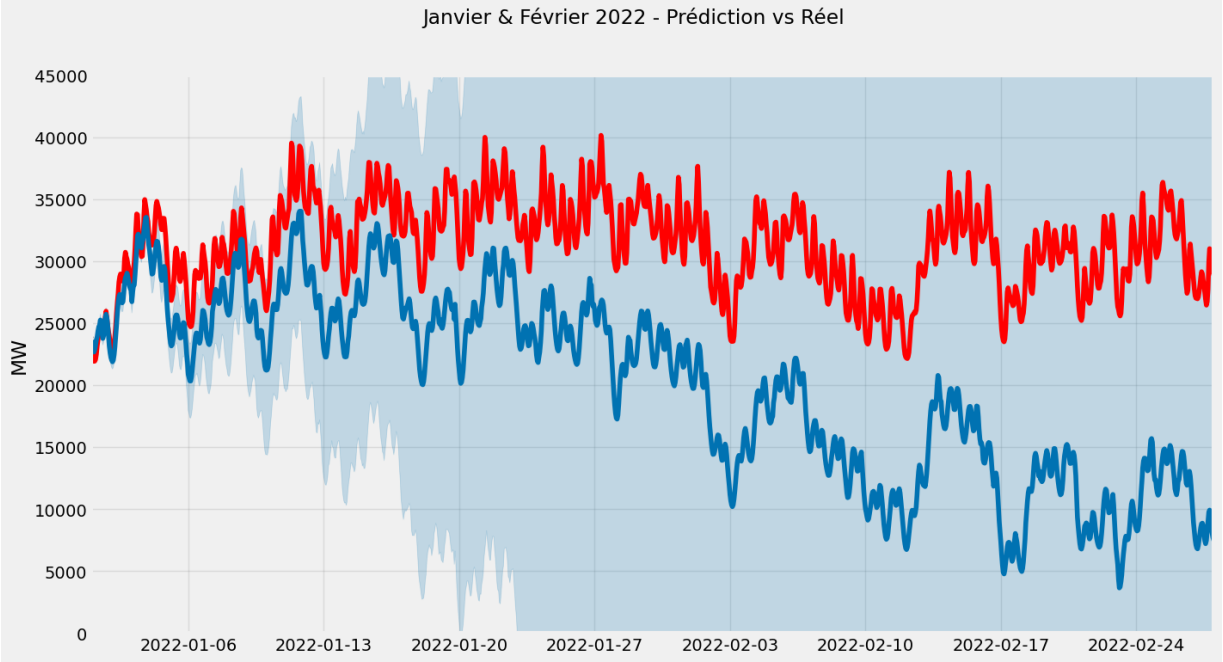
```
INFO:prophet:Disabling yearly seasonality. Run prophet with yearly_seasonali
ty=True to override this.
INFO:prophet:Disabling weekly seasonality. Run prophet with weekly_seasonali
ty=True to override this.
DEBUG:cmdstanpy:input tempfile: /var/folders/h1/zbfgvczd009f52sgp7ld62500000
gn/T/tmpno48uf50/rhcobyyx.json
DEBUG:cmdstanpy:input tempfile: /var/folders/h1/zbfgvczd009f52sgp7ld62500000
gn/T/tmpno48uf50/nywyup81.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/opt/homebrew/anaconda3/envs/sci1402/lib/pyt
hon3.11/site-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed
=99050', 'data', 'file=/var/folders/h1/zbfgvczd009f52sgp7ld62500000gn/T/tmpn
o48uf50/rhcobyyx.json', 'init=/var/folders/h1/zbfgvczd009f52sgp7ld62500000g
n/T/tmpno48uf50/nywyup81.json', 'output', 'file=/var/folders/h1/zbfgvczd009f
52sgp7ld62500000gn/T/tmpno48uf50/prophet_modelqhvzoc97/prophet_model-2023120
4222845.csv', 'method=optimize', 'algorithm=lbfgs', 'iter=10000']
22:28:45 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
22:28:45 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```

In [14]:
```
fig, ax = plt.subplots(figsize=(15, 8))
ax.plot(df_test.index, df_test["y"], color="r")
fig = m.plot(forecast, ax=ax)
ax.set_ylabel("MW")
ax.set_xlabel("")
ax.set_ylim(-5_000, 45_000)

plot = plt.suptitle("2022 - Prédiction vs Réel")
```



Le modèle décroche encore une fois après un mois et demi environ.

In [15]:
```
fig, ax = plt.subplots(figsize=(15, 8))
ax.plot(df_test.index, df_test["y"], color="r")
fig = m.plot(forecast, ax=ax)
ax.set_xbound(
    lower=datetime(2022, 1, 1),
    upper=datetime(2022, 2, 28),
)
ax.set_ylim(0, 45_000)
ax.set_ylabel("MW")
ax.set_xlabel("")
plot = plt.suptitle(
    "Janvier & Février 2022 – Prédiction vs Réel"
)
```



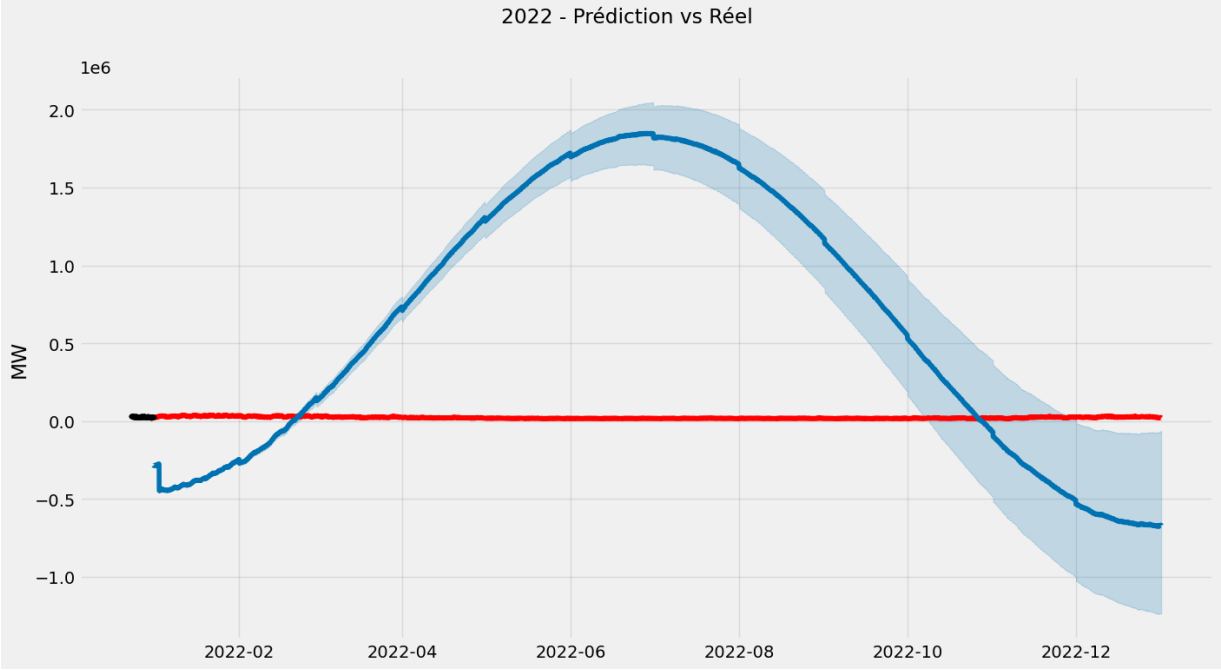
Création d'un modèle avec toutes les caractéristiques

Comme le modèle s'est amélioré avec l'ajout de caractéristiques, essayons de les ajouter toutes.

In [16]:
```
m = Prophet()

for feature in FEATURES:
    m.add_regressor(feature)

m.fit(df_train)

forecast = m.predict(df_test.drop(columns="y"))
```

```
INFO:prophet:Disabling yearly seasonality. Run prophet with yearly_seasonali
ty=True to override this.
INFO:prophet:Disabling weekly seasonality. Run prophet with weekly_seasonali
ty=True to override this.
DEBUG:cmdstanpy:input tempfile: /var/folders/h1/zbfgvczd009f52sgp7ld62500000
gn/T/tmpno48uf50/awodpg7b.json
DEBUG:cmdstanpy:input tempfile: /var/folders/h1/zbfgvczd009f52sgp7ld62500000
gn/T/tmpno48uf50/3ii5qbvh.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/opt/homebrew/anaconda3/envs/sci1402/lib/pyt
hon3.11/site-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed
=56402', 'data', 'file=/var/folders/h1/zbfgvczd009f52sgp7ld62500000gn/T/tmpn
o48uf50/awodpg7b.json', 'init=/var/folders/h1/zbfgvczd009f52sgp7ld62500000g
n/T/tmpno48uf50/3ii5qbvh.json', 'output', 'file=/var/folders/h1/zbfgvczd009f
52sgp7ld62500000gn/T/tmpno48uf50/prophet_modelujvcaqyi/prophet_model-2023120
4222846.csv', 'method=optimize', 'algorithm=lbfgs', 'iter=10000']
22:28:46 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
22:28:46 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```

In [17]:
```
fig, ax = plt.subplots(figsize=(15, 8))
ax.plot(df_test.index, df_test["y"], color="r")
fig = m.plot(forecast, ax=ax)
ax.set_ylabel("MW")
ax.set_xlabel("")
plot = plt.suptitle("2022 - Prédiction vs Réel")
```



L'ajout de toutes les caractéristiques a complètement bousillé le modèle qui est décroché.

Pour arriver à un meilleur résultat, il faudrait faire des essais moins radicaux (2, 5, 20 caractéristiques ?)

Conclusions

Avec ces résultats préliminaires, le temps qui passe dans le projet et les résultats extrêmement intéressants que nous avons obtenus avec **XGBoost**, je prends la décision d'arrêter la modélisation avec Prophet dès maintenant.

Notre temps sera mieux utilisé à peaufiner le modèle de XGBoost que de rendre celui-ci potable.