In [1]:
```
import pandas as pd
from darts import TimeSeries
```

In [2]:
```
df = pd.read_csv("MY3_May_2023_KNN_Imputed.csv")
df['Datetime'] = pd.to_datetime(df['Datetime'])
df.set_index('Datetime', inplace=True)
df_resampled = df.resample('1H').mean()

test_length = 24 * 7
train_length = len(df_resampled) - test_length

df_train = df_resampled.iloc[:train_length]
df_test = df_resampled.iloc[train_length - 24 : train_length + test_length]
```

In [3]:
```
targets = ['field1','field2','field3','field4','field7','field8']
covariates = ['field5','field6']

X_train_df = df_train[covariates]
Y_train_df = df_train[targets]

X_test_df = df_test[covariates]
Y_test_df = df_test[targets]
```

In [4]:
```
X_train = TimeSeries.from_dataframe(df_train[covariates])
Y_train = TimeSeries.from_dataframe(df_train[targets])

X_test = TimeSeries.from_dataframe(df_test[covariates])
Y_test = TimeSeries.from_dataframe(df_test[targets])
```

In [5]:
```
from darts.models import RegressionModel
from darts.metrics import mae
import warnings
warnings.filterwarnings("ignore")

n = len(Y_test)

# Initialize the Regression Model
modelrm = RegressionModel(lags=24, lags_past_covariates=24, output_chunk_length=24 )

# Train the model on the training data
modelrm.fit(Y_train, past_covariates=X_train)
```

Out[5]:
```
RegressionModel(lags=24, lags_past_covariates=24, lags_future_covariates=None, output_chunk_length=24, add_e
ncoders=None, model=None, multi_models=True, use_static_covariates=True)
```

In [6]:
```
predrm1 = modelrm.predict(n, past_covariates = X_test)

predictions = TimeSeries.pd_dataframe(predrm1)
predictions
```

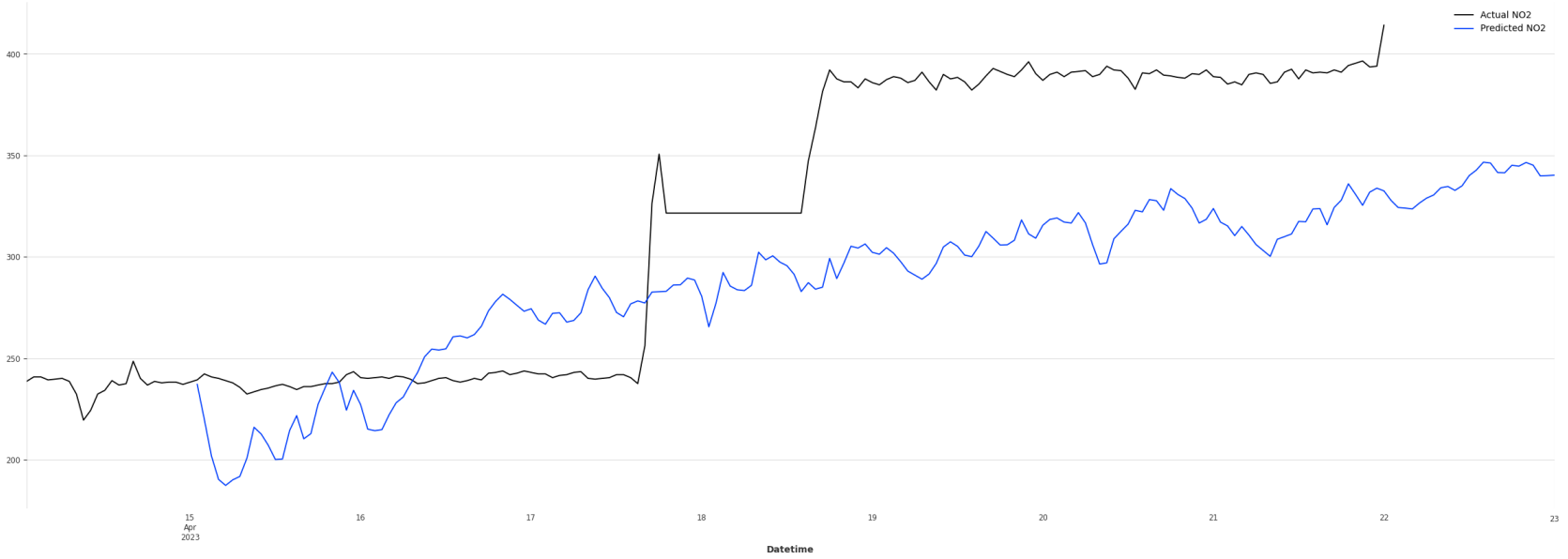
Out[6]:

component	field1	field2	field3	field4	field7	field8
Datetime						
2023-04-15 01:00:00	237.150143	130.141162	945.208720	278.078840	45.455573	56.134433
2023-04-15 02:00:00	219.937800	132.593991	886.518707	263.296294	43.536159	53.481290
2023-04-15 03:00:00	201.969550	126.420432	832.437854	251.274744	41.120225	50.005764
2023-04-15 04:00:00	190.333917	119.722306	798.367510	252.944625	39.118672	47.206874
2023-04-15 05:00:00	187.350764	132.227558	781.573712	261.866541	37.827292	45.549918
...	...	...	...	...	...	...
2023-04-22 20:00:00	346.466232	187.155772	824.720900	400.267623	14.448441	16.186752
2023-04-22 21:00:00	345.146252	186.843016	831.730251	398.349890	14.595799	16.362469
2023-04-22 22:00:00	339.853802	182.484661	831.613886	392.314483	14.949997	16.750197
2023-04-22 23:00:00	340.000479	179.700998	833.657662	391.235403	15.432536	17.340623
2023-04-23 00:00:00	340.228358	180.261696	841.617300	393.292502	15.847821	17.833834

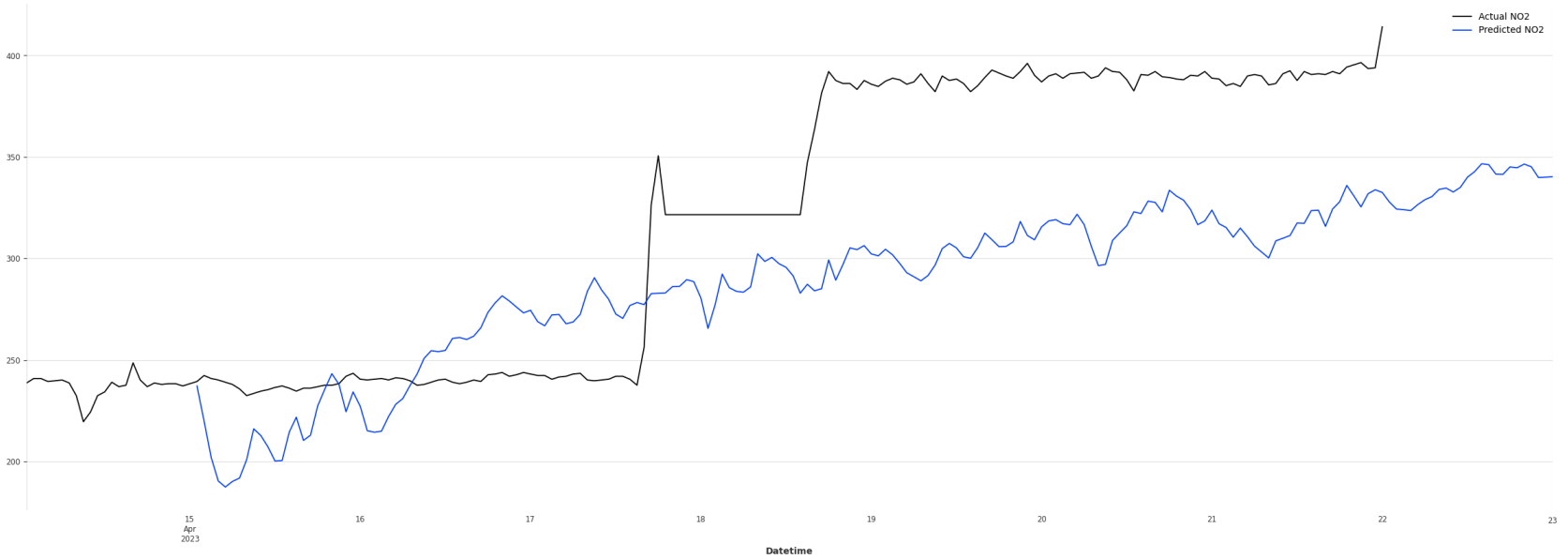
192 rows × 6 columns

In [7]:
```
from darts.metrics import rmse, mae


# Convert actual and predicted values to TimeSeries
actual_series_list = [TimeSeries.from_dataframe(Y_test_df[[target]]) for target in targets]
predicted_series_list = [TimeSeries.from_dataframe(predictions[[target]]) for target in targets]

# Calculate RMSE and MAE for each target field
rmse_values = []
mae_values = []

for actual, predicted in zip(actual_series_list, predicted_series_list):
    rmse_value = rmse(actual, predicted)
    mae_value = mae(actual, predicted)
    rmse_values.append(rmse_value)
    mae_values.append(mae_value)

# Create a DataFrame to store the results for LightGBM
results_df = pd.DataFrame({
    'Field': targets,
    'RMSE_RM': rmse_values,
    'MAE_RM': mae_values
})

# Save the LightGBM results to a CSV file
results_df.to_csv('Regression_multi_results.csv', index=False)
```

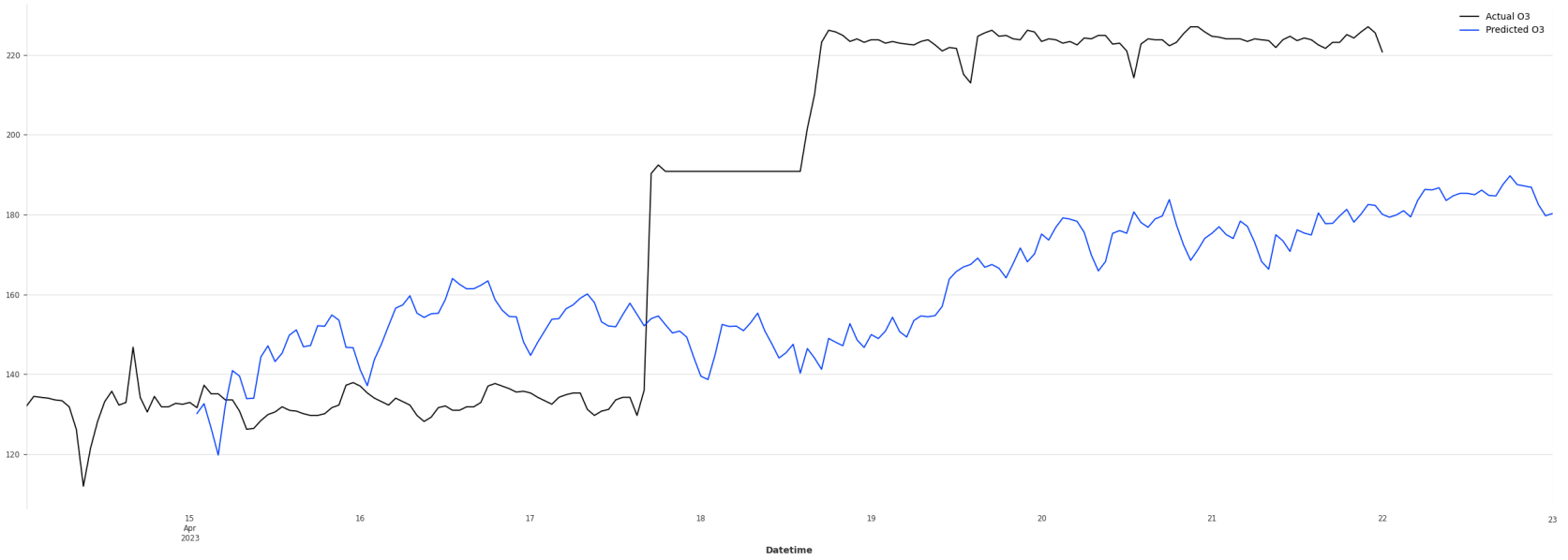
In [8]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(30,10))
Y_test_df['field1'].plot(label='Actual NO2')
predictions['field1'].plot(label='Predicted NO2')
plt.legend()
plt.show()
```



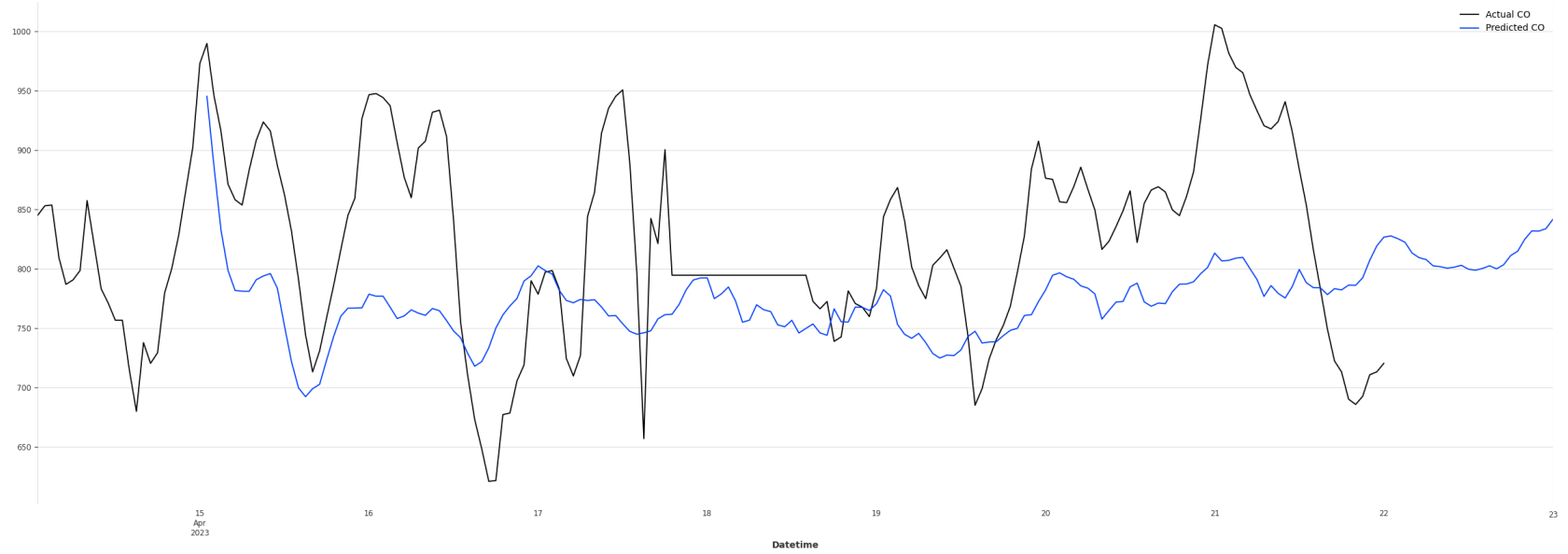
In [9]:
```
plt.figure(figsize=(30,10))
Y_test_df['field1'].plot(label='Actual NO2')
predictions['field1'].plot(label='Predicted NO2')
plt.legend()
plt.show()
```



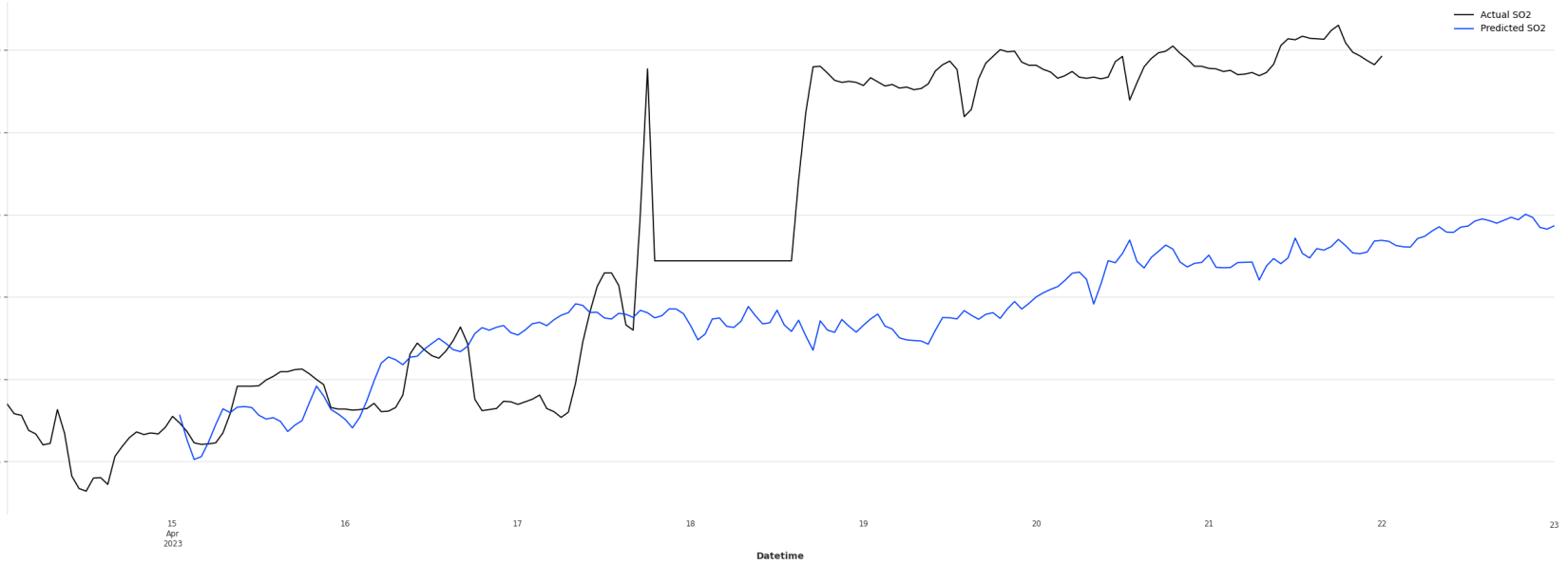
In [10]:
```
plt.figure(figsize=(30,10))
Y_test_df['field2'].plot(label='Actual O3')
predictions['field2'].plot(label='Predicted O3')
plt.legend()
plt.show()
```



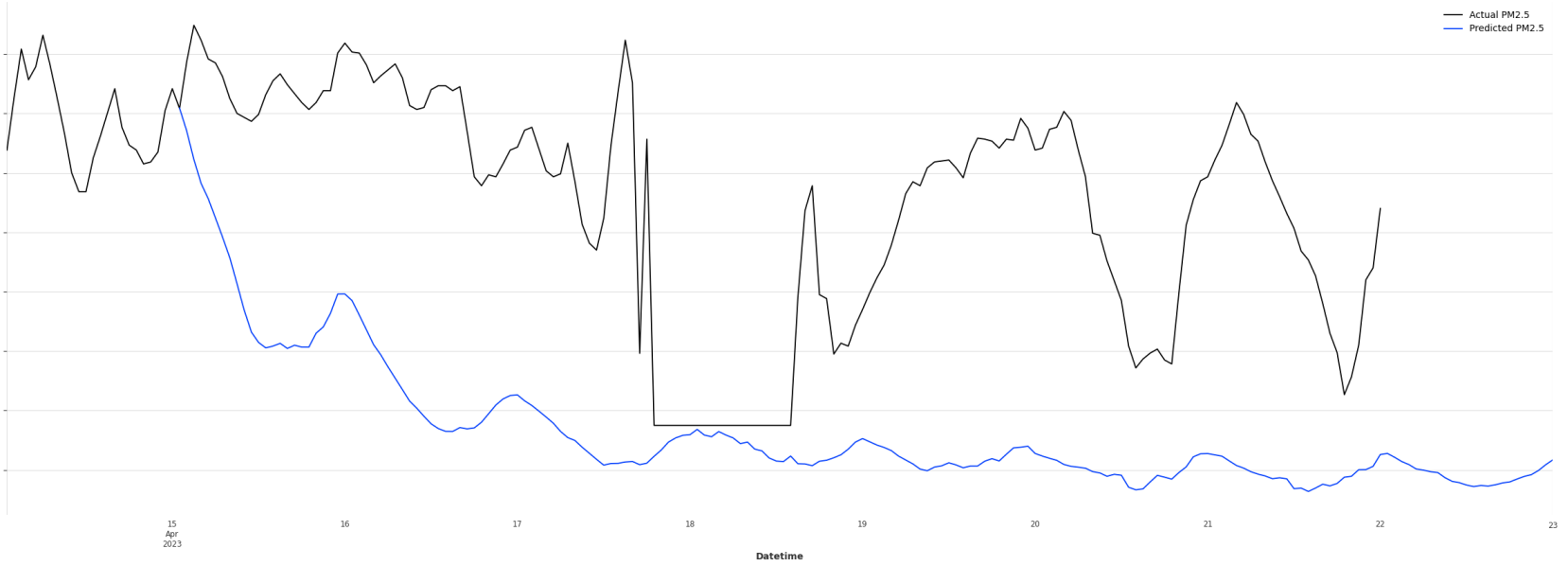
In [11]:
```
plt.figure(figsize=(30,10))
Y_test_df['field3'].plot(label='Actual CO')
predictions['field3'].plot(label='Predicted CO')
plt.legend()
plt.show()
```



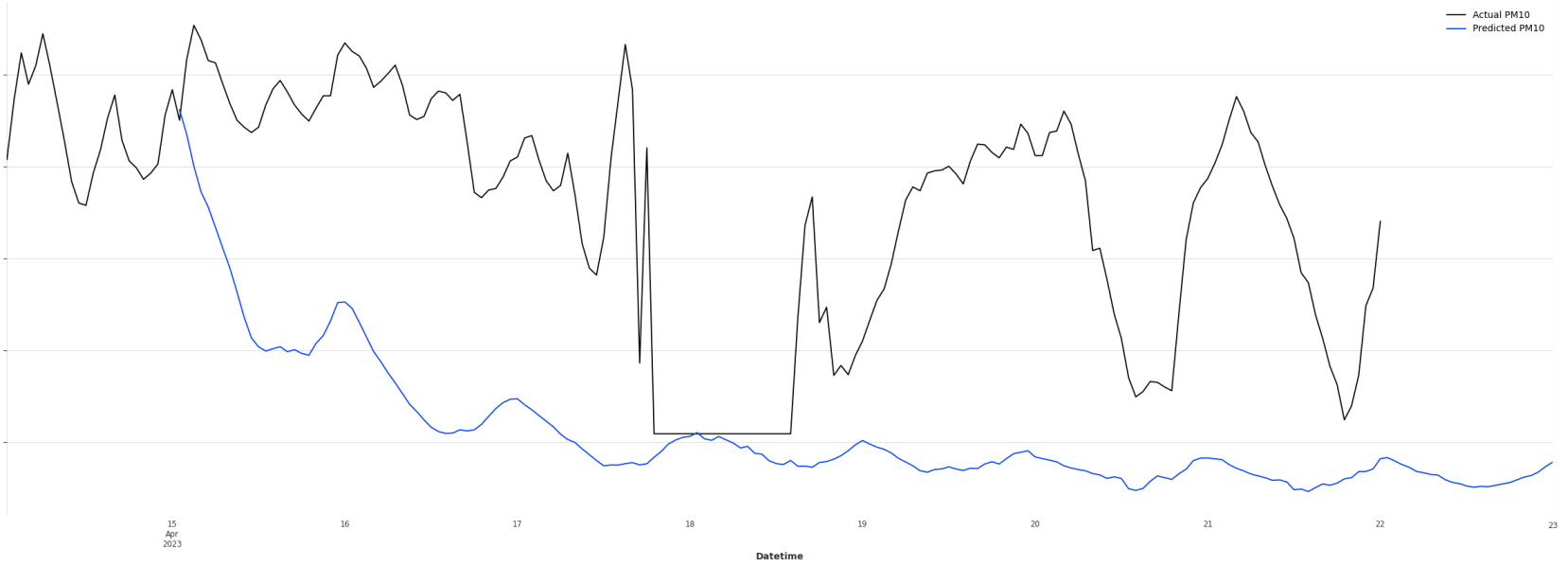
In [12]:
```
plt.figure(figsize=(30,10))
Y_test_df['field4'].plot(label='Actual SO2')
predictions['field4'].plot(label='Predicted SO2')
plt.legend()
plt.show()
```



In [13]:
```
plt.figure(figsize=(30,10))
Y_test_df['field7'].plot(label='Actual PM2.5')
predictions['field7'].plot(label='Predicted PM2.5')
plt.legend()
plt.show()
```



In [14]:
```
plt.figure(figsize=(30,10))
Y_test_df['field8'].plot(label='Actual PM10')
predictions['field8'].plot(label='Predicted PM10')
plt.legend()
plt.show()
```



In [ ]: