ANALYSING OIL WELL PRODUCTION DATA

In [1]:
```
# importing the needed libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
import warnings
warnings.filterwarnings('ignore')
sns.set_style("whitegrid")
plt.rcParams['figure.figsize'] = (12, 6)
```

In [2]:
```
# loading the dataset
df = pd.read_excel("C://Users//quays//Desktop//Oil well.xlsx")
df_copy = df.copy() # making a copy of the original data
```

DATA INSPECTION AND CLEANING

In [3]:
```
# viewing the top 5 rows of the data
df.head()
```

Out[3]:

	Date	\nOil volume (m3/day)	Volume of liquid (m3/day)	\nGas volume (m3/day)	Water volume (m3/day)	Water cut (%)	\nWorking hours	\nDynamic level (m)	\nReservoir pressure (atm)
0	2013-01-01	49	70	13055	21	29	24	1819	214
1	2013-01-02	49	70	13055	21	29	24	1836	214
2	2013-01-03	49	70	13055	21	29	24	1788	214
3	2013-01-04	49	70	13055	21	29	24	1789	214
4	2013-01-05	44	70	11768	26	36	24	1825	214

In [4]:
```
# correcting some few columns names.
df.rename(columns={
    '\nOil volume (m3/day)': 'Oil volume (m3/day)','\nGas volume (m3/day)': 'Gas volume
'\nWorking hours':'Working hours','\nDynamic level (m)':'Dynamic level (m)','\nReservoir
'Reservoir pressure (atm)'}, inplace=True)
```

In [5]:
```
# checking for the null values
df.isnull().sum()
```

Out[5]:
```
Date                          0
Oil volume (m3/day)           0
Volume of liquid (m3/day)     0
Gas volume (m3/day)           0
Water volume (m3/day)         0
Water cut (%)                 0
Working hours                 0
Dynamic level (m)             0
Reservoir pressure (atm)      0
dtype: int64
```

In [6]:
```
# checking for duplicates
df.duplicated().sum()
```

Out[6]:
```
0
```

In [7]:
```
# shape of the data
df.shape
```

Out[7]:
```
(2939, 9)
```

In [8]:
```
# checking for the data types
df.dtypes
```

Out[8]:
```
Date                          datetime64[ns]
Oil volume (m3/day)                    int64
Volume of liquid (m3/day)              int64
Gas volume (m3/day)                    int64
Water volume (m3/day)                 object
Water cut (%)                          int64
Working hours                          int64
Dynamic level (m)                      int64
Reservoir pressure (atm)               int64
dtype: object
```

In [9]:
```
# dropping null column
df.dropna(inplace = True)
```

In [10]:
```
df.isnull().sum()
```

Out[10]:
```
Date                          0
Oil volume (m3/day)           0
Volume of liquid (m3/day)     0
Gas volume (m3/day)           0
Water volume (m3/day)         0
Water cut (%)                 0
Working hours                 0
Dynamic level (m)             0
Reservoir pressure (atm)      0
dtype: int64
```

In [11]:
```
# checking for the columns data types
df.dtypes
```

Out[11]:
```
Date                          datetime64[ns]
Oil volume (m3/day)                    int64
Volume of liquid (m3/day)              int64
Gas volume (m3/day)                    int64
Water volume (m3/day)                 object
Water cut (%)                          int64
Working hours                          int64
Dynamic level (m)                      int64
Reservoir pressure (atm)               int64
dtype: object
```

In [12]:
```
# removing the white spaces, replacing null values with nan and convert column to numeri
s = df["Water volume (m3/day)"].astype(str).str.strip().replace('', np.nan)
df["Water volume (m3/day)"] = pd.to_numeric(s, errors="coerce").astype("Int64")
```

In [13]:
```
# checking for null values
df.isnull().sum()
```

Out[13]:
```
Date                          0
Oil volume (m3/day)           0
Volume of liquid (m3/day)     0
Gas volume (m3/day)           0
Water volume (m3/day)         1
Water cut (%)                 0
Working hours                 0
Dynamic level (m)             0
Reservoir pressure (atm)      0
dtype: int64
```

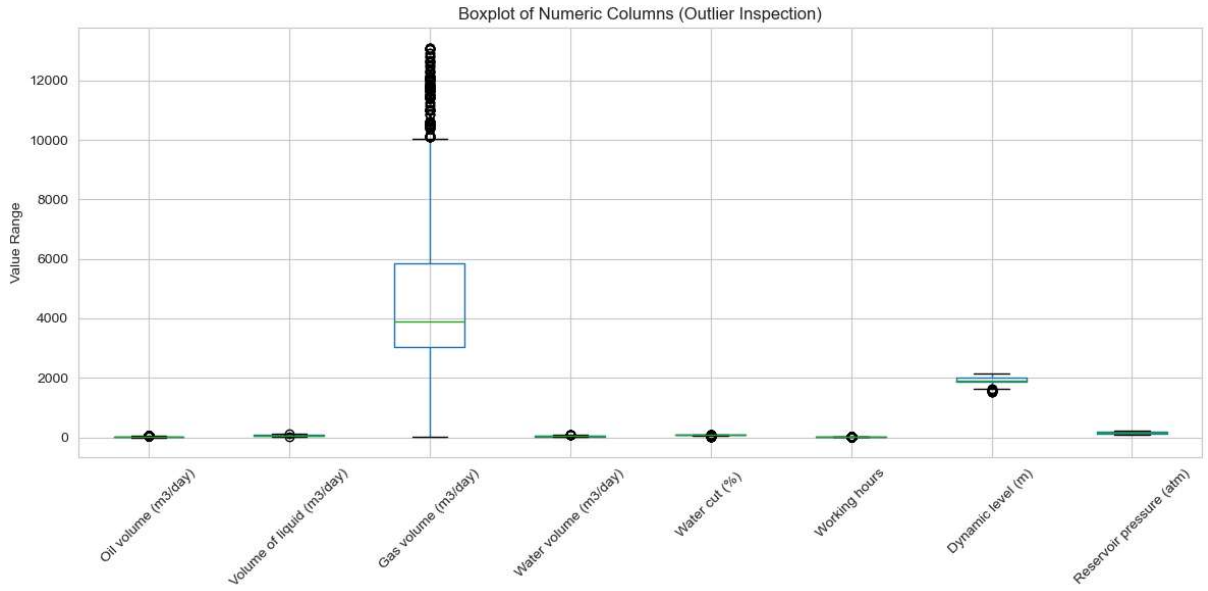
In [14]:
```
# dropping the null containing row
df.dropna(inplace = True)
```

In [15]:
```
df.head()
```

Out[15]:

	Date	Oil volume (m3/day)	Volume of liquid (m3/day)	Gas volume (m3/day)	Water volume (m3/day)	Water cut (%)	Working hours	Dynamic level (m)	Reservoir pressure (atm)
0	2013-01-01	49	70	13055	21	29	24	1819	214
1	2013-01-02	49	70	13055	21	29	24	1836	214
2	2013-01-03	49	70	13055	21	29	24	1788	214
3	2013-01-04	49	70	13055	21	29	24	1789	214
4	2013-01-05	44	70	11768	26	36	24	1825	214

In [16]:
```
# checking for outliers

# Select numeric columns only (exclude 'Date')
numeric_cols = df.select_dtypes(include='number').columns
if 'Date' in numeric_cols:
    numeric_cols = numeric_cols.drop('Date')

# Plot boxplots for visual outlier inspection
plt.figure(figsize=(12, 6))
df[numeric_cols].boxplot(rot=45)
plt.title("Boxplot of Numeric Columns (Outlier Inspection)")
plt.ylabel("Value Range")
plt.tight_layout()
plt.show()
```



In [17]:
```
# removing the outliers from the data

# Select numeric columns (exclude 'Date' if present)
numeric_cols = df.select_dtypes(include=[np.number]).columns.tolist()
if 'Date' in numeric_cols:
    numeric_cols = [c for c in numeric_cols if c != 'Date']

# Copy the data
df_no_outliers = df.copy()

# Remove outliers using IQR method
for col in numeric_cols:
    Q1 = df_no_outliers[col].quantile(0.25)
    Q3 = df_no_outliers[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    df_no_outliers = df_no_outliers[
        (df_no_outliers[col] >= lower_bound) & (df_no_outliers[col] <= upper_bound)
    ]

# Print only shapes
print("Original data shape:", df.shape)
print("New data shape after removing outliers:", df_no_outliers.shape)
```

```
Original data shape: (2938, 9)
New data shape after removing outliers: (2106, 9)
```

In [18]:
```
# reviewing the data
df.head()
```

Out[18]:

	Date	Oil volume (m3/day)	Volume of liquid (m3/day)	Gas volume (m3/day)	Water volume (m3/day)	Water cut (%)	Working hours	Dynamic level (m)	Reservoir pressure (atm)
0	2013-01-01	49	70	13055	21	29	24	1819	214
1	2013-01-02	49	70	13055	21	29	24	1836	214
2	2013-01-03	49	70	13055	21	29	24	1788	214
3	2013-01-04	49	70	13055	21	29	24	1789	214
4	2013-01-05	44	70	11768	26	36	24	1825	214

FEATURE ENGINEERING

In [19]:
```
# creating new columns

# 1. Month and Year columns
df['Month'] = df['Date'].dt.month
df['Year'] = df['Date'].dt.year

# 2. Days Since Start
df['Days_Since_Start'] = (df['Date'] - df['Date'].min()).dt.days

# 3. Water-Oil Ratio (WOR)
df['WOR'] = df['Water volume (m3/day)'] / df['Oil volume (m3/day)']

# 4. Cumulative Oil Production
df['Cumulative_Oil'] = df['Oil volume (m3/day)'].cumsum()

# 5. Gas-Oil Ratio (GOR)
df['GOR'] = df['Gas volume (m3/day)'] / df['Oil volume (m3/day)']

# 6. Pressure Derivative
df['Pressure_Derivative'] = df['Reservoir pressure (atm)'].diff()

# 7. Moving Averages
df['Oil_MA_7'] = df['Oil volume (m3/day)'].rolling(window=7).mean()
df['Pressure_MA_30'] = df['Reservoir pressure (atm)'].rolling(window=30).mean()
```

EXPLORATORY DATA ANALYSIS

In [20]:
```
df.columns.tolist()
```

Out[20]:
```
['Date',
 'Oil volume (m3/day)',
 'Volume of liquid (m3/day)',
 'Gas volume (m3/day)',
 'Water volume (m3/day)',
 'Water cut (%) ',
 'Working hours',
 'Dynamic level (m)',
 'Reservoir pressure (atm)',
 'Month',
 'Year',
 'Days_Since_Start',
 'WOR',
 'Cumulative_Oil',
 'GOR',
 'Pressure_Derivative',
 'Oil_MA_7',
 'Pressure_MA_30']
```

In [21]:
```
df.head()
```

Out[21]:

	Date	Oil volume (m3/day)	Volume of liquid (m3/day)	Gas volume (m3/day)	Water volume (m3/day)	Water cut (%)	Working hours	Dynamic level (m)	Reservoir pressure (atm)	Month	Year	Days_Si
0	2013-01-01	49	70	13055	21	29	24	1819	214	1	2013	
1	2013-01-02	49	70	13055	21	29	24	1836	214	1	2013	
2	2013-01-03	49	70	13055	21	29	24	1788	214	1	2013	
3	2013-01-04	49	70	13055	21	29	24	1789	214	1	2013	
4	2013-01-05	44	70	11768	26	36	24	1825	214	1	2013	

In [22]:
```
df.isnull().sum()
```

Out[22]:
```
Date                          0
Oil volume (m3/day)           0
Volume of liquid (m3/day)     0
Gas volume (m3/day)           0
Water volume (m3/day)         0
Water cut (%)                 0
Working hours                 0
Dynamic level (m)             0
Reservoir pressure (atm)      0
Month                         0
Year                          0
Days_Since_Start              0
WOR                           0
Cumulative_Oil                0
GOR                           0
Pressure_Derivative           1
Oil_MA_7                      6
Pressure_MA_30               29
dtype: int64
```

In [23]:
```
# Fill NaN values with the first available moving average value
df['Oil_MA_7'] = df['Oil_MA_7'].fillna(method='bfill')
df['Pressure_MA_30'] = df['Pressure_MA_30'].fillna(method='bfill')
df['Pressure_Derivative'] = df['Pressure_Derivative'].fillna(method='bfill')
```

In [24]:
```
df.head()
```

Out[24]:

	Date	Oil volume (m3/day)	Volume of liquid (m3/day)	Gas volume (m3/day)	Water volume (m3/day)	Water cut (%)	Working hours	Dynamic level (m)	Reservoir pressure (atm)	Month	Year	Days_Si
0	2013-01-01	49	70	13055	21	29	24	1819	214	1	2013	
1	2013-01-02	49	70	13055	21	29	24	1836	214	1	2013	
2	2013-01-03	49	70	13055	21	29	24	1788	214	1	2013	
3	2013-01-04	49	70	13055	21	29	24	1789	214	1	2013	
4	2013-01-05	44	70	11768	26	36	24	1825	214	1	2013	

In [25]:
```
# information about the data
df.info()
```

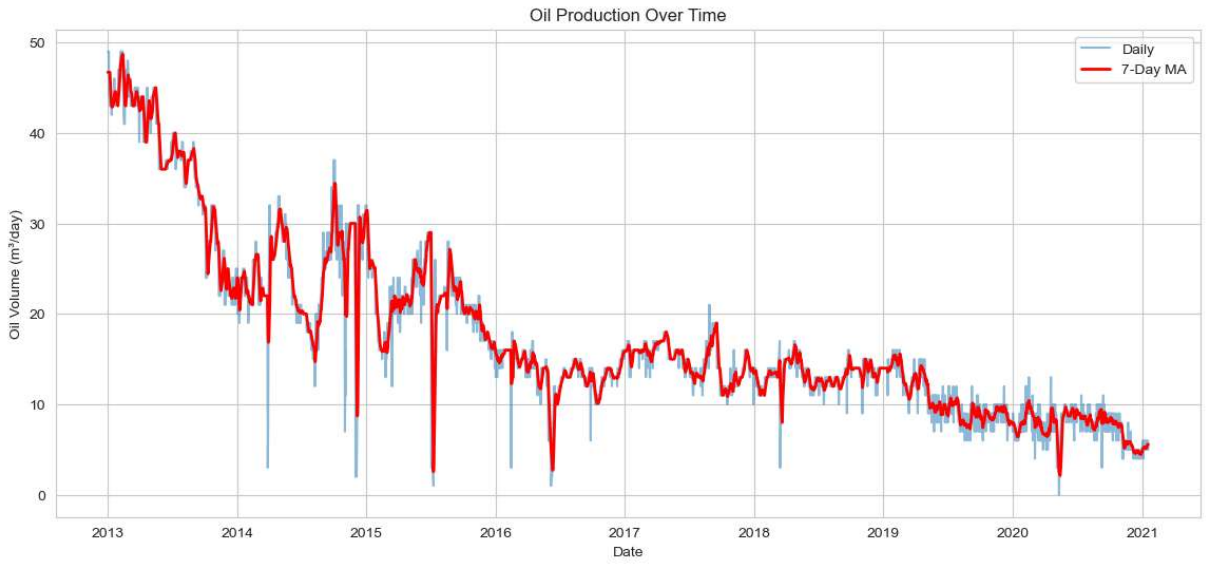
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2938 entries, 0 to 2938
Data columns (total 18 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Date                       2938 non-null   datetime64[ns]
 1   Oil volume (m3/day)        2938 non-null   int64
 2   Volume of liquid (m3/day)  2938 non-null   int64
 3   Gas volume (m3/day)        2938 non-null   int64
 4   Water volume (m3/day)      2938 non-null   Int64
 5   Water cut (%)              2938 non-null   int64
 6   Working hours              2938 non-null   int64
 7   Dynamic level (m)          2938 non-null   int64
 8   Reservoir pressure (atm)   2938 non-null   int64
 9   Month                      2938 non-null   int64
 10  Year                       2938 non-null   int64
 11  Days_Since_Start           2938 non-null   int64
 12  WOR                        2938 non-null   Float64
 13  Cumulative_Oil             2938 non-null   int64
 14  GOR                        2938 non-null   float64
 15  Pressure_Derivative        2938 non-null   float64
 16  Oil_MA_7                   2938 non-null   float64
 17  Pressure_MA_30             2938 non-null   float64
dtypes: Float64(1), Int64(1), datetime64[ns](1), float64(4), int64(11)
memory usage: 441.8 KB
```

In [26]:
```
# dataset statistics
df.describe()
```

Out[26]:

	Oil volume (m3/day)	Volume of liquid (m3/day)	Gas volume (m3/day)	Water volume (m3/day)	Water cut (%)	Working hours	Dynamic level (m)	Reservo pressu (atn
count	2938.000000	2938.000000	2938.000000	2938.0	2938.000000	2938.000000	2938.000000	2938.00000
mean	17.615385	59.460177	4727.757318	41.833901	70.706263	22.344112	1930.440095	157.00000
std	9.678442	18.636057	2596.101282	13.05598	9.515529	3.039917	114.522092	32.90653
min	0.000000	12.000000	4.000000	9.0	29.000000	7.000000	1529.000000	100.00000
25%	11.000000	50.000000	3040.750000	33.0	69.000000	22.000000	1855.000000	129.00000
50%	15.000000	58.000000	3908.500000	43.0	73.000000	24.000000	1890.000000	157.00000
75%	22.000000	74.000000	5843.000000	50.0	76.000000	24.000000	2008.000000	185.00000
max	49.000000	113.000000	13113.000000	99.0	100.000000	24.000000	2137.000000	214.00000

In [27]:
```
# daily oil production trend
plt.figure(figsize=(14, 6))
plt.plot(df['Date'], df['Oil volume (m3/day)'], alpha=0.5, label='Daily')
plt.plot(df['Date'], df['Oil_MA_7'], linewidth=2, label='7-Day MA', color='red')
plt.title('Oil Production Over Time')
plt.xlabel('Date')
plt.ylabel('Oil Volume (m³/day)')
plt.legend()
plt.show()
```



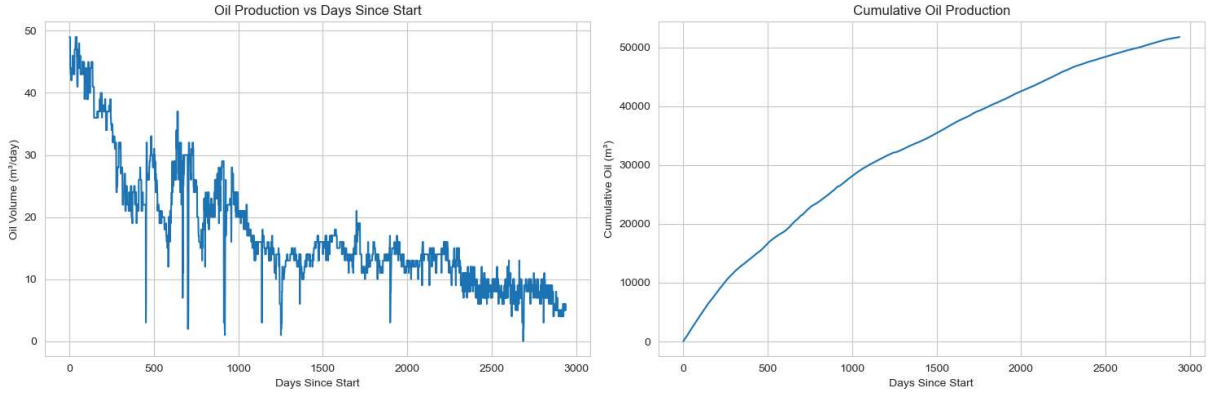

```
INSIGHT:

The oil production trend for Well No. 807 shows a clear long-term decline from

approximately 50 m³/day in 2013 to below 10 m³/day by 2021, indicating progressive

reservoir depletion and potential increases in water cut or operational inefficiencies.

While short-term fluctuations and brief recoveries are visible, likely due to well
interventions

or equipment adjustments,

the overall 7-day moving average trend confirms a steady reduction in productivity.
```

This visualization provides an essential overview of the well's performance behavior over time,

supporting effective monitoring and informed decision-making on production optimization or

decline management strategies.

In [28]:
```
# production decline analysis
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 5))
ax1.plot(df['Days_Since_Start'], df['Oil volume (m3/day)'])
ax1.set_title('Oil Production vs Days Since Start')
ax1.set_xlabel('Days Since Start')
ax1.set_ylabel('Oil Volume (m³/day)')
ax2.plot(df['Days_Since_Start'], df['Cumulative_Oil'])
ax2.set_title('Cumulative Oil Production')
ax2.set_xlabel('Days Since Start')
ax2.set_ylabel('Cumulative Oil (m³)')
plt.tight_layout()
plt.show()
```



INSIGHT:

The first plot illustrates a gradual decline in daily oil production over time,

starting from around 45–50 m³/day and dropping below 10 m³/day by the end of the observed period.

This downward trend reflects reservoir pressure depletion and possible production inefficiencies as the well matures.

The second plot, showing cumulative oil production, rises steadily but begins to flatten toward the end,

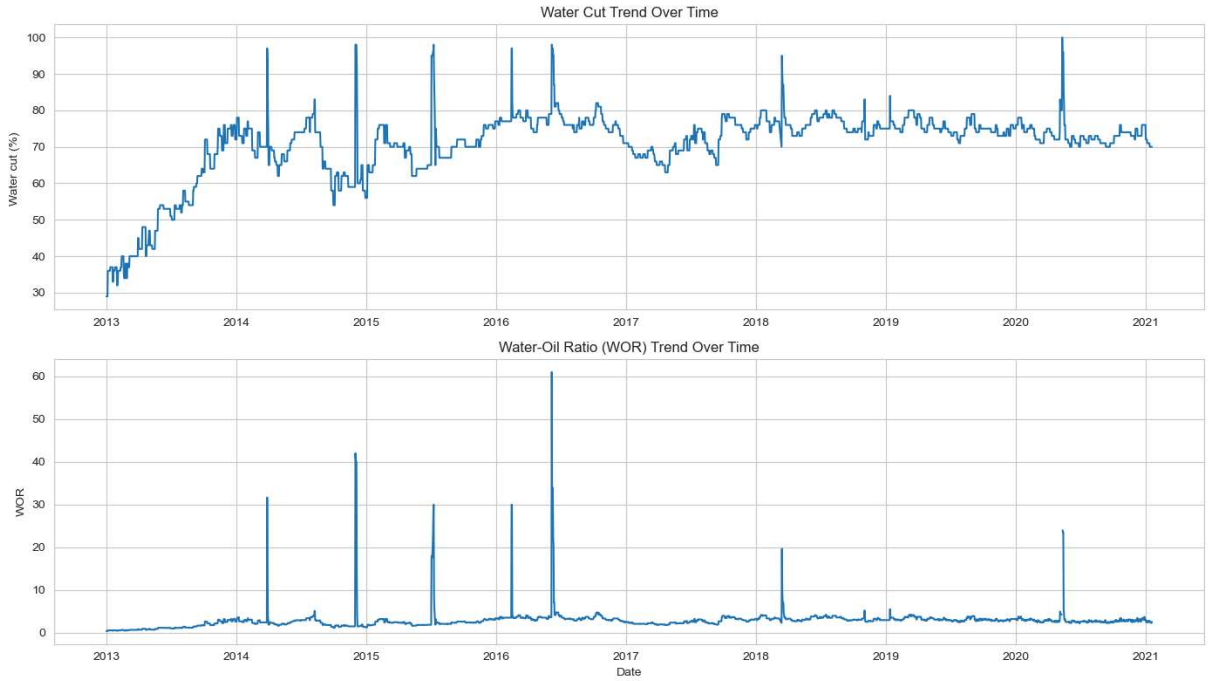
indicating slower production rates despite ongoing operation.

Together, these graphs effectively visualize production behavior and highlight the well's

transition from a high-yield to a declining phase, supporting timely decision-making for

reservoir management and production optimization.

In [29]:
```
# water cut and WOR analysis
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(14, 8))
ax1.plot(df['Date'], df['Water cut (%) '])
ax1.set_title('Water Cut Trend Over Time')
ax1.set_ylabel('Water cut (%) ')
ax2.plot(df['Date'], df['WOR'])
ax2.set_title('Water-Oil Ratio (WOR) Trend Over Time')
ax2.set_ylabel('WOR')
ax2.set_xlabel('Date')
plt.tight_layout()
plt.show()
```



INSIGHT:

The plots show the evolution of water production performance in Well 807 from 2013 to 2021.

The top plot reveals a steady rise in water cut from about 30% in 2013 to around 70–80% by 2016,

after which it stabilizes with minor fluctuations. This indicates progressive water breakthrough

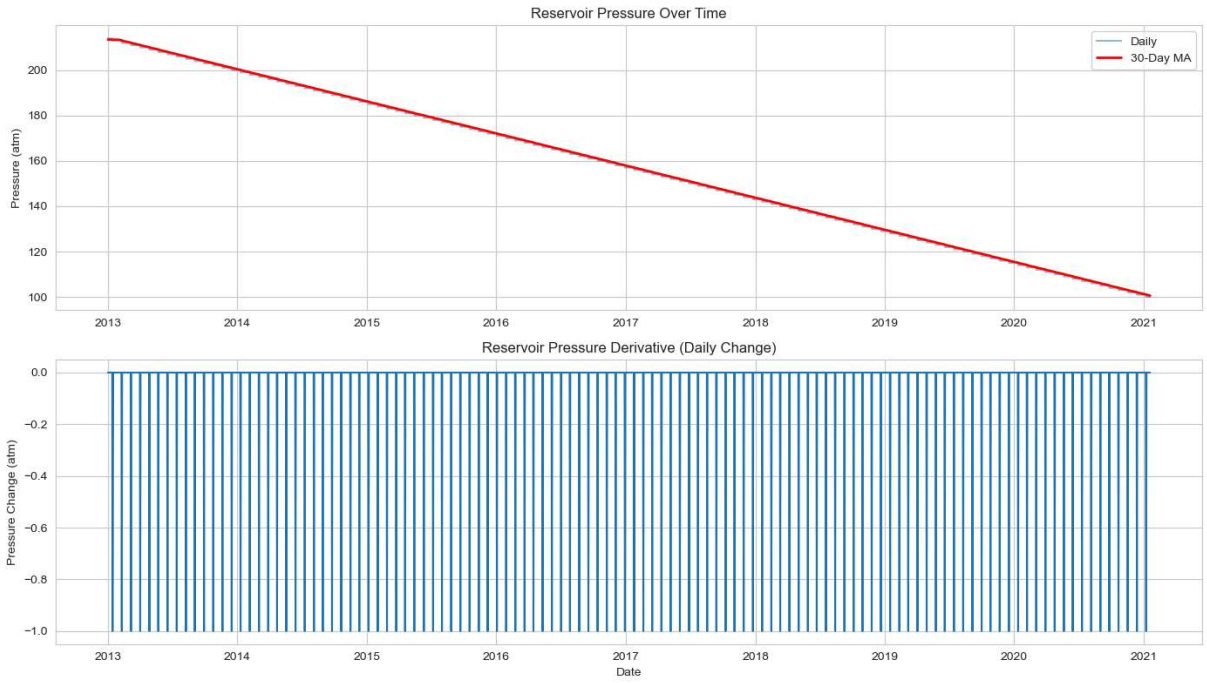
and maturity of the reservoir. The bottom plot shows corresponding spikes in the

Water–Oil Ratio (WOR), particularly between 2014 and 2017, reflecting periods of excessive water production.

Overall, the sustained high water cut and periodic WOR peaks suggest increasing water

encroachment and declining oil efficiency over time, signaling the need for water control or reservoir management interventions.

In [30]:
```
# reservoir pressure behavior
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(14, 8))
ax1.plot(df['Date'], df['Reservoir pressure (atm)'], alpha=0.5, label='Daily')
ax1.plot(df['Date'], df['Pressure_MA_30'], linewidth=2, label='30-Day MA', color='red')
ax1.set_title('Reservoir Pressure Over Time')
ax1.set_ylabel('Pressure (atm)')
ax1.legend()
ax2.plot(df['Date'], df['Pressure_Derivative'])
ax2.set_title('Reservoir Pressure Derivative (Daily Change)')
ax2.set_ylabel('Pressure Change (atm)')
ax2.set_xlabel('Date')
plt.tight_layout()
plt.show()
```



INSIGHT:

These plots illustrate a consistent decline in reservoir pressure from around 220 atm in 2013 to just above 100 atm by 2021,

as shown by both the daily values and the smoothed 30-day moving average.

This steady downward trend reflects continuous reservoir depletion, likely due to sustained production without sufficient

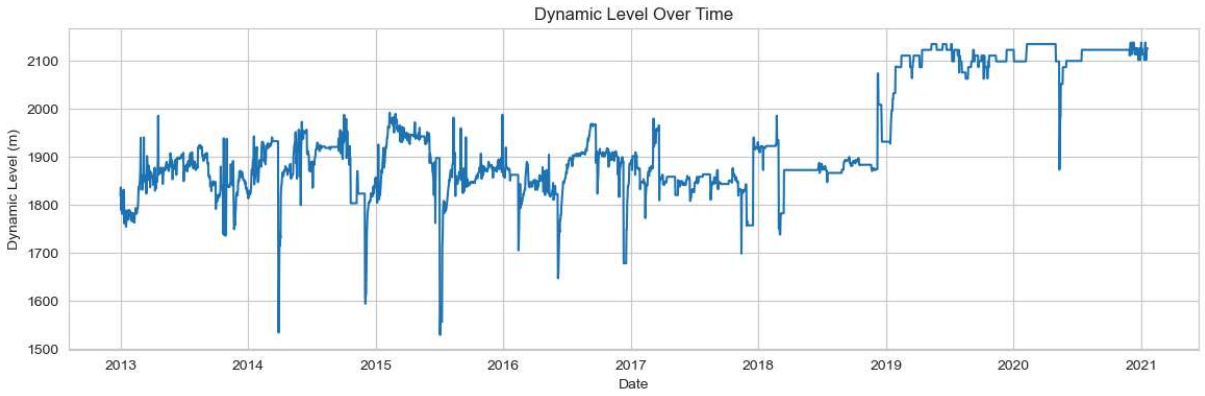
pressure support from natural drive or secondary recovery methods.

The derivative plot below confirms a nearly constant rate of pressure decline throughout

the monitoring period, indicating stable but ongoing energy loss within the reservoir system.

This suggests that reservoir pressure maintenance or enhanced recovery measures may be needed to sustain production efficiency.

In [31]:
```
# dynamic level behavior
plt.figure(figsize=(14, 4))
plt.plot(df['Date'], df['Dynamic level (m)'])
plt.title('Dynamic Level Over Time')
plt.xlabel('Date')
plt.ylabel('Dynamic Level (m)')
plt.show()
```



INSIGHT:

We can see that from 2013 to around 2018, the dynamic level showed

significant variability with frequent dips, which may indicate operational inconsistencies

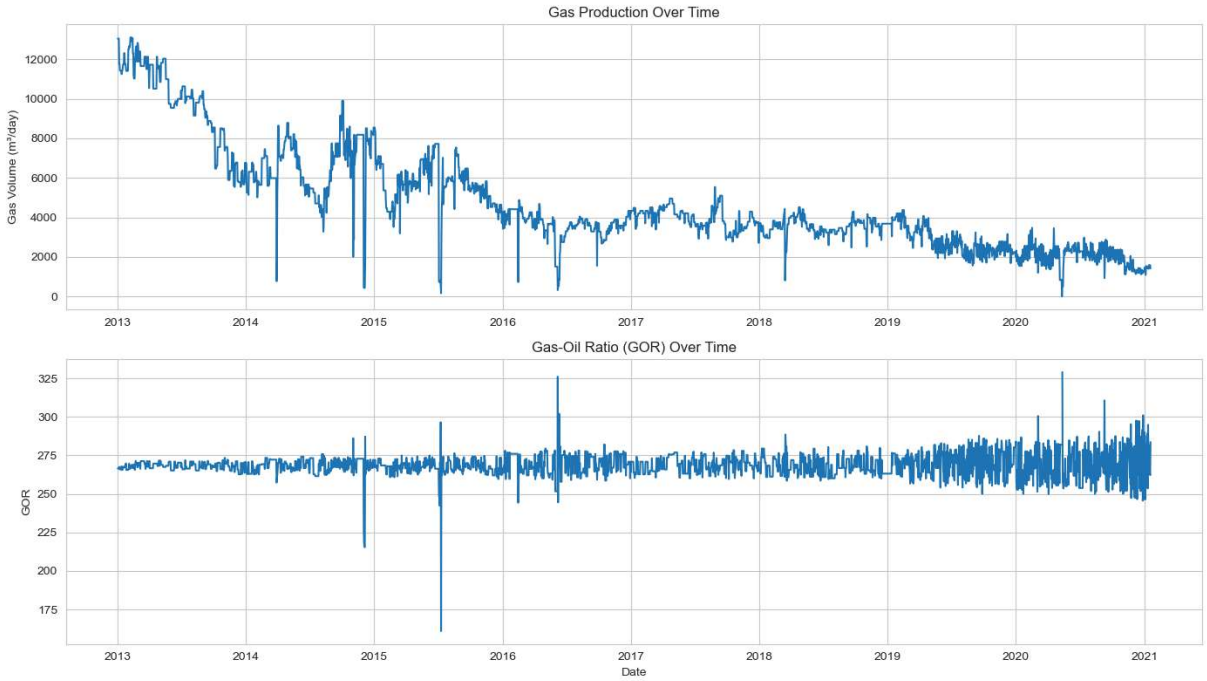
or external influences affecting performance. However, starting in 2018, there’s a clear improvement

the levels rise and then stabilize at a much higher range through 2019 to 2021.

This suggests that the changes or interventions made around that time were effective, leading to a more stable and sustained

performance in the system.

In [32]:
```
# gas production analysis
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(14, 8))
ax1.plot(df['Date'], df['Gas volume (m3/day)'])
ax1.set_title('Gas Production Over Time')
ax1.set_ylabel('Gas Volume (m³/day)')
ax2.plot(df['Date'], df['GOR'])
ax2.set_title('Gas-Oil Ratio (GOR) Over Time')
ax2.set_ylabel('GOR')
ax2.set_xlabel('Date')
plt.tight_layout()
plt.show()
```



INSIGHT:

I can see that gas production has shown a clear and steady decline from 2013 to 2021, dropping from

around 12,000 m³/day to below 2,000 m³/day. This indicates a consistent reduction in reservoir pressure

or resource availability over time. Meanwhile, the Gas-Oil Ratio (GOR) remains relatively

stable through most of the period, with some minor fluctuations, but shows a gradual increase and

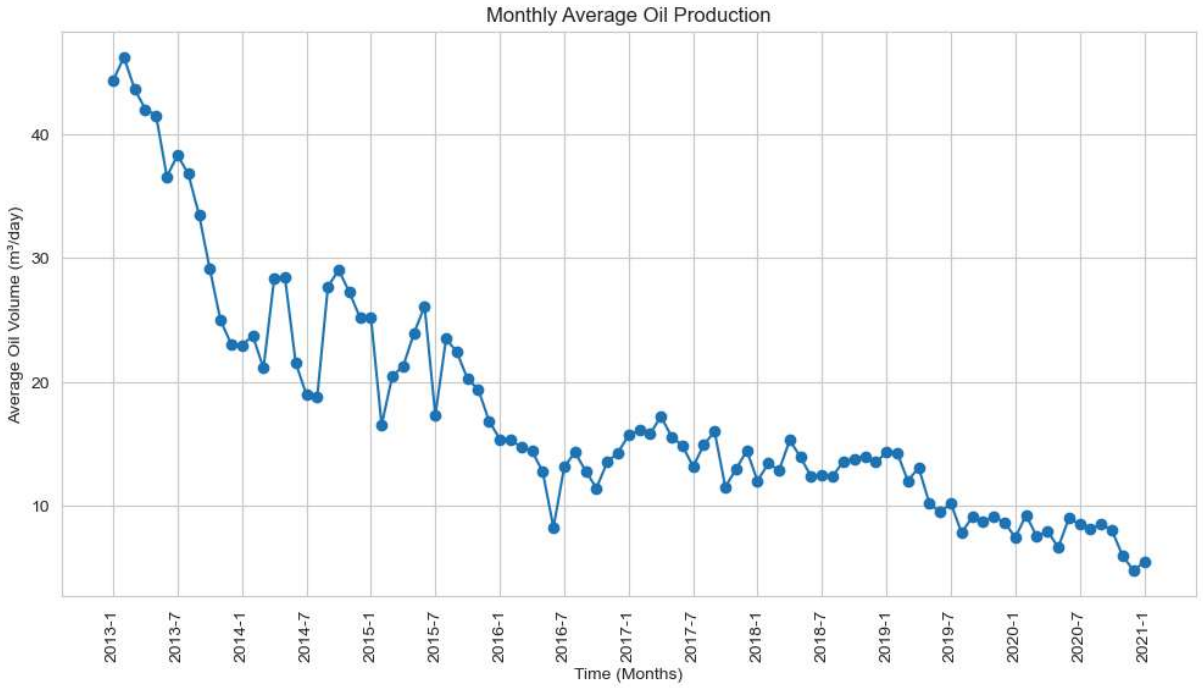
more variability after 2018. This suggests that as gas production decreases, the reservoir may be

transitioning, potentially producing proportionally more gas relative to oil toward the later years.

Overall, this trend highlights a maturing field that may require optimization strategies or

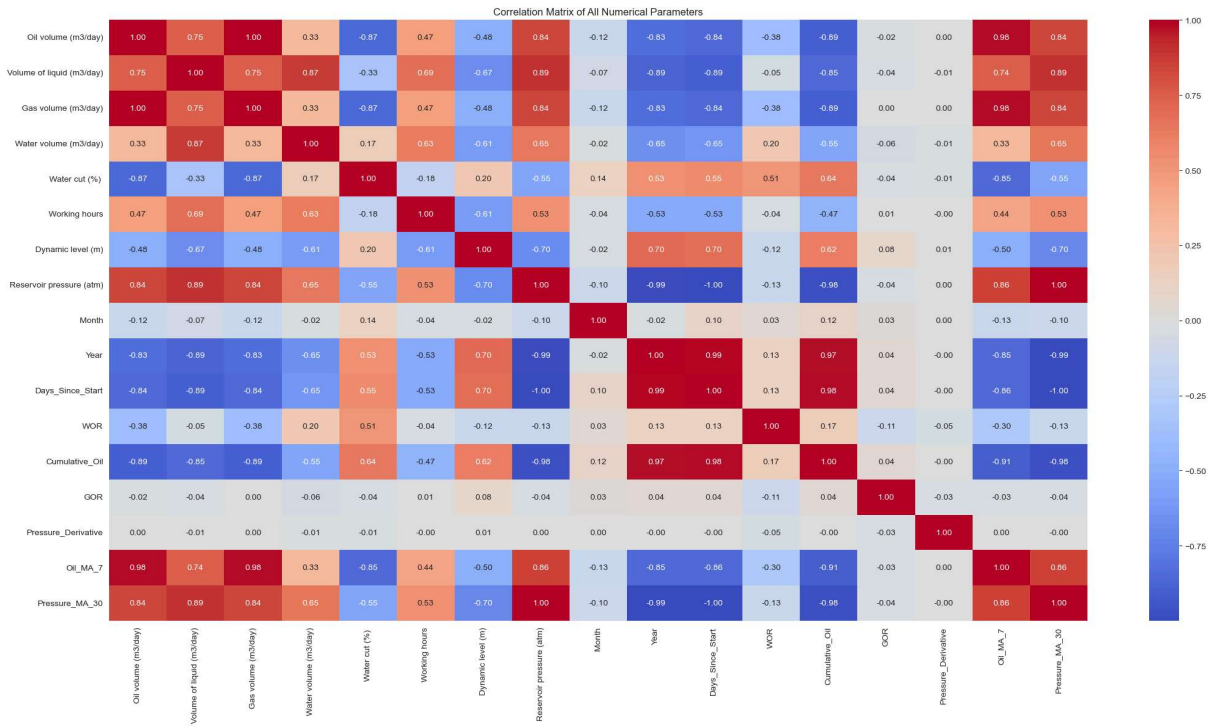
interventions to maintain production efficiency.

In [33]:
```
# Seasonal/Monthly Production Patterns
plt.figure(figsize=(12, 6))
monthly_avg = df.groupby(['Year', 'Month'])['Oil volume (m3/day)'].mean().reset_index()
plt.plot(monthly_avg.index, monthly_avg['Oil volume (m3/day)'], marker='o')
plt.title('Monthly Average Oil Production')
plt.xlabel('Time (Months)')
plt.ylabel('Average Oil Volume (m³/day)')
plt.xticks(ticks=range(0, len(monthly_avg), 6), rotation = 90,
           labels=monthly_avg[['Year','Month']].iloc[::6].apply(lambda x: f"{x[0]}-{x[1]
plt.show()
```




```
INSIGHT:

This chart shows the monthly average oil production trend from 2013 to 2021.

The data clearly indicates a continuous decline in oil output over the years

starting above 40 m³/day in early 2013 and dropping steadily to below 10 m³/day by
2021.

While there are short periods of minor recovery or fluctuation, the overall trajectory
reflects a consistent depletion

in reservoir performance or production capacity. This long term decline highlights the
maturity

of the field and suggests the need for enhanced recovery strategies or operational
interventions to sustain production levels.
```

In [34]:
```
# Correlation Analysis
plt.figure(figsize=(22, 12))
numeric_cols = df.select_dtypes(include=[np.number]).columns
numeric_cols = numeric_cols.drop('Date') if 'Date' in numeric_cols else numeric_cols
corr_matrix = df[numeric_cols].corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', center=0, fmt='.2f')
plt.title('Correlation Matrix of All Numerical Parameters')
plt.tight_layout()
plt.show()
```



INSIGHT:

This correlation matrix provides valuable insights into the key factors influencing oil production.

Oil volume shows a very strong positive correlation with gas volume (0.98), reservoir pressure (0.84),

and cumulative oil (0.89) indicating that higher reservoir pressure and gas production are closely

associated with increased oil output. There’s also a strong correlation with volume of liquid (0.75),

suggesting that total fluid production impacts oil rates.

Conversely, oil volume has a strong negative correlation with water cut (-0.87) and Days_Since_Start (-0.94),

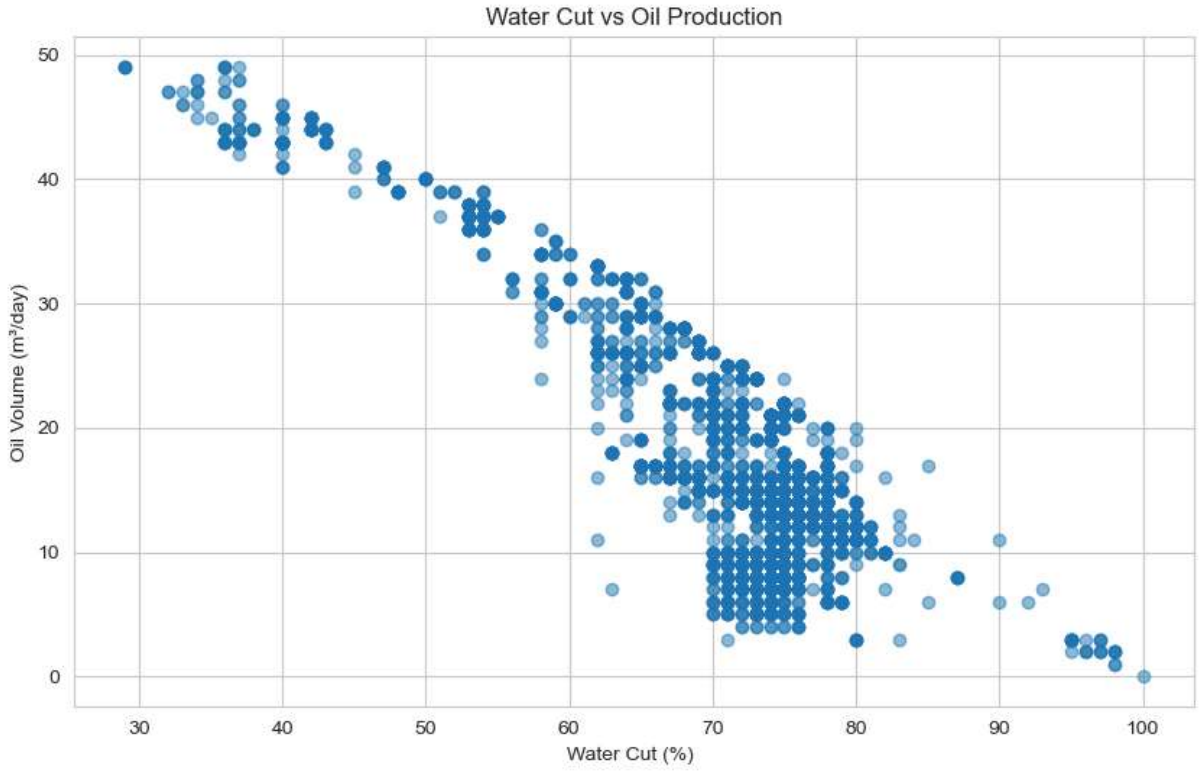
meaning that as water production increases and the well matures, oil output declines significantly.

Additionally, oil production decreases as dynamic level (-0.45) drops, reinforcing the relationship

between reservoir performance and production rates. Overall, maintaining reservoir pressure and

managing water production appear to be the most critical factors in sustaining oil production over time.

In [35]:
```
# Water Cut vs Oil Production
plt.figure(figsize=(10, 6))
plt.scatter(df['Water cut (%) '], df['Oil volume (m3/day)'], alpha=0.5)
plt.title('Water Cut vs Oil Production')
plt.xlabel('Water Cut (%) ')
plt.ylabel('Oil Volume (m³/day)')
plt.show()
```

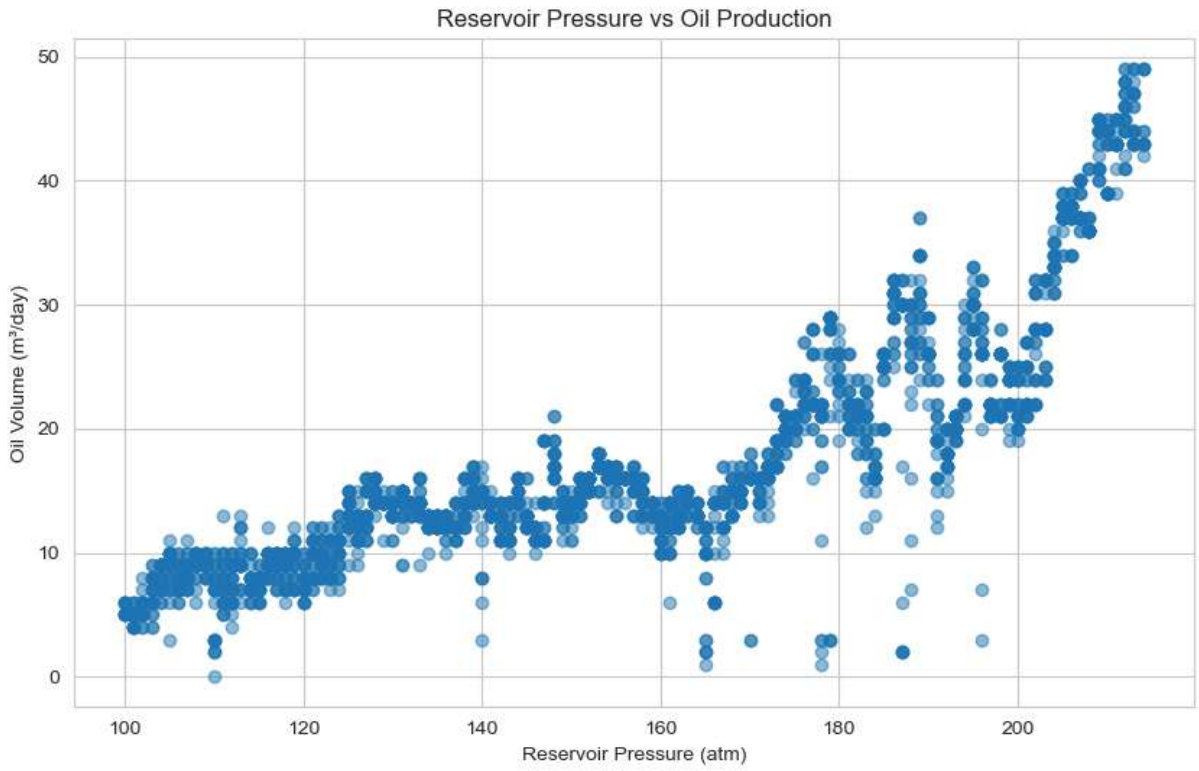


INSIGHT:

Water cut % shows a strong negative correlation with the oip production indicating that, as the oil production increases,

the water portion or % decreases in the well

In [36]:
```
# Reservoir Pressure vs Production
plt.figure(figsize=(10, 6))
plt.scatter(df['Reservoir pressure (atm)'], df['Oil volume (m3/day)'], alpha=0.5)
plt.title('Reservoir Pressure vs Oil Production')
plt.xlabel('Reservoir Pressure (atm)')
plt.ylabel('Oil Volume (m³/day)')
plt.show()
```




```
INSIGHT:

This plot shows a strong correlation between the oil produced vs the reservoir
pressure.

This shows that, the higher the reservoir pressure, the higher the oil production and
vice versa.

Thus the reservoirs pressure must be maintained or increased to ensure stable oil
production.
```