WATER QUALITY ANALYSIS

INTRODUCTION

Access to safe drinking-water is essential to health, a basic human right and a component of effective policy for health protection. This is important as a health and development issue at a national, regional and local level. In some regions, it has been shown that investments in water supply and sanitation can yield a net economic benefit, since the reductions in adverse health effects and health care costs outweigh the costs of undertaking the interventions.

HARDWARE AND SOFTWARE REQUIREMENT:-

HARDWARE:-

Processor- Intel® Core™ i5 11th gen

Graphic card- NVIDIA® GeForce® GTX 1650

RAM- 8 GB, DDR4.

-Hardisk-512 GB SSD

SOFTWARE

- Jupyter Notebook

- Python

Libraries like:-

- matplot

- seaborn

- pandas

- numpy

In [1]:
```
import numpy as np # linear algebra
import matplotlib.pyplot as plt # library for data visualization contains- graphs, charts etc
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns # like matplot, less complex and more features
```

In [2]:
```
data = pd.read_csv("water_potability.csv") # read the csv data
```

In [3]:
```
data.head() #print the first five elements from the data set
```

Out[3]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0

In [4]:
```
data.tail() #print the last five elements from the data set
```

Out[4]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
3271	4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.894419	66.687695	4.435821	1
3272	7.808856	193.553212	17329.802160	8.061362	NaN	392.449580	19.903225	NaN	2.798243	1
3273	9.419510	175.762646	33155.578218	7.350233	NaN	432.044783	11.039070	69.845400	3.298875	1
3274	5.126763	230.603758	11983.869376	6.303357	NaN	402.883113	11.168946	77.488213	4.708658	1
3275	7.874671	195.102299	17404.177061	7.509306	NaN	327.459760	16.140368	78.698446	2.309149	1

MOTIVATION

Water pollution can have some tremendously-adverse effect on the health of any and every life form living in the vicinity of the polluted water body or using water that has been polluted to some extent. At a certain level polluted water can be detrimental to crops and reduce the fertility of soil thus harming the overall agricultural sector and the country as well. When sea water is polluted it can also impact oceanic life in a bad way. The most fundamental effect of water pollution is however on the quality of the water, consuming which can lead to several ailments. In the urban areas water is used for both industrial and domestic purposes from waterbodies such as rivers, lakes, streams, wells, and ponds. Worst still, 80% of the water that we use for our domestic purposes is passed out in the form of wastewater. In most of the cases, this water is not treated properly and as such it leads to tremendous pollution of surface-level freshwater.In fact as far as India is concerned polluted water is one of the major factors behind the general low levels of health in India, especially in the rural areas. Polluted water can lead to diseases such as cholera, tuberculosis, dysentery, jaundice, diarrhoea, etc. In fact, around 80% stomach ailments in India happen because of consuming polluted water.

OBJECTIVE

This dataset contains all the factors that determines the potability of water. Using data visualization can help us better understand the objective of dataset. We will plot graphs and charts using various libraries like matplot, seaborn, plotly. We will compare various parameters and understand how it effects the potability of water.

1. pH value:

PH is an important parameter in evaluating the acid–base balance of water. It is also the indicator of acidic or alkaline condition of water status. WHO has recommended maximum permissible limit of pH from 6.5 to 8.5. The current investigation ranges were 6.52–6.83 which are in the range of WHO standards.

2. Hardness:

Hardness is mainly caused by calcium and magnesium salts. These salts are dissolved from geologic deposits through which water travels. The length of time water is in contact with hardness producing material helps determine how much hardness there is in raw water. Hardness was originally defined as the capacity of water to precipitate soap caused by Calcium and Magnesium.

3. Solids (Total dissolved solids - TDS)

Water has the ability to dissolve a wide range of inorganic and some organic minerals or salts such as potassium, calcium, sodium, bicarbonates, chlorides, magnesium, sulfates etc. These minerals produced un-wanted taste and diluted color in appearance of water. This is the important parameter for the use of water. The water with high TDS value indicates that water is highly mineralized. Desirable limit for TDS is 500 mg/l and maximum limit is 1000 mg/l which prescribed for drinking purpose.

4. Chloramines:

Chlorine and chloramine are the major disinfectants used in public water systems. Chloramines are most commonly formed when ammonia is added to chlorine to treat drinking water. Chlorine levels up to 4 milligrams per liter (mg/L or 4 parts per million (ppm)) are considered safe in drinking water.

5. Sulfate:

Sulfates are naturally occurring substances that are found in minerals, soil, and rocks. They are present in ambient air, groundwater, plants, and food. The principal commercial use of sulfate is in the chemical industry. Sulfate concentration in seawater is about 2,700 milligrams per liter (mg/L). It ranges from 3 to 30 mg/L in most freshwater supplies, although much higher concentrations (1000 mg/L) are found in some geographic locations.

6. Conductivity:

Pure water is not a good conductor of electric current rather’s a good insulator. Increase in ions concentration enhances the electrical conductivity of water. Generally, the amount of dissolved solids in water determines the electrical conductivity. Electrical conductivity (EC) actually measures the ionic process of a solution that enables it to transmit current. According to WHO standards, EC value should not exceeded 400 μS/cm.

7. Organic_carbon:

Total Organic Carbon (TOC) in source waters comes from decaying natural organic matter (NOM) as well as synthetic sources. TOC is a measure of the total amount of carbon in organic compounds in pure water. According to US EPA < 2 mg/L as TOC in treated / drinking water, and < 4 mg/Lit in source water which is use for treatment.

8. Trihalomethanes:

THMs are chemicals which may be found in water treated with chlorine. The concentration of THMs in drinking water varies according to the level of organic material in the water, the amount of chlorine required to treat the water, and the temperature of the water that is being treated. THM levels up to 80 ppm is considered safe in drinking water.

9. Turbidity:

The turbidity of water depends on the quantity of solid matter present in the suspended state. It is a measure of light emitting properties of water and the test is used to indicate the quality of waste discharge with respect to colloidal matter. The mean turbidity value obtained for Wondo Genet Campus (0.98 NTU) is lower than the WHO recommended value of 5.00 NTU.

10. Potability:

Indicates if water is safe for human consumption where 1 means Potable and 0 means Not potable.

In [5]:
```
data.isnull() #print all the null values
```

Out[5]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	True	False	False	False	False	False	False	False	False	False
1	False	False	False	False	True	False	False	False	False	False
2	False	False	False	False	True	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
3271	False	False	False	False	False	False	False	False	False	False
3272	False	False	False	False	True	False	False	True	False	False
3273	False	False	False	False	True	False	False	False	False	False
3274	False	False	False	False	True	False	False	False	False	False
3275	False	False	False	False	True	False	False	False	False	False

3276 rows × 10 columns

In [6]:
```
data.isnull().sum()  #total sum of all the values
```

Out[6]:
```
ph                 491
Hardness             0
Solids               0
Chloramines          0
Sulfate            781
Conductivity         0
Organic_carbon       0
Trihalomethanes    162
Turbidity            0
Potability           0
dtype: int64
```

In [7]:
```
data.shape #total number of rows and columns
```

Out[7]:
```
(3276, 10)
```

In [8]:
```
data['ph'] #shows all data the from the ph column
```

Out[8]:
```
0            NaN
1       3.716080
2       8.099124
3       8.316766
4       9.092223
          ...
3271    4.668102
3272    7.808856
3273    9.419510
3274    5.126763
3275    7.874671
Name: ph, Length: 3276, dtype: float64
```

In [9]:
```
data['ph'].mean() #mean of all the values
```

Out[9]:
```
7.080794504276819
```

MEAN

A mean is the simple mathematical average of a set of two or more numbers. The mean for a given set of numbers can be computed in more than one way, including the arithmetic mean method, which uses the sum of the numbers in the series, and the geometric mean method, which is the average of a set of product

In [10]:
```
data['ph'].fillna(data['ph'].mean(), inplace=True) #We fill the empty cells with the mean of all data
```

In [11]:
```
data.isnull()
```

Out[11]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	True	False	False	False	False	False
2	False	False	False	False	True	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
3271	False	False	False	False	False	False	False	False	False	False
3272	False	False	False	False	True	False	False	True	False	False
3273	False	False	False	False	True	False	False	False	False	False
3274	False	False	False	False	True	False	False	False	False	False
3275	False	False	False	False	True	False	False	False	False	False

3276 rows × 10 columns

This helps to show sum of cells with null the data

In [12]:
```
data.isnull().sum()
```

Out[12]:
```
ph                   0
Hardness             0
Solids               0
Chloramines          0
Sulfate            781
Conductivity         0
Organic_carbon       0
Trihalomethanes    162
Turbidity            0
Potability           0
dtype: int64
```

Same process is followed for the other columns as well.

In [13]:
```
data['Sulfate'].mean()
```

Out[13]:
```
333.7757766108134
```

In [14]:
```
data['Sulfate'].fillna(data['Sulfate'].mean(),inplace= True)
```

In [15]:
```
data['Sulfate'].isnull()
```

Out[15]:
```
0       False
1       False
2       False
3       False
4       False
        ...
3271    False
3272    False
3273    False
3274    False
3275    False
Name: Sulfate, Length: 3276, dtype: bool
```

In [16]:
```
data.isnull().sum()
```

Out[16]:
```
ph                   0
Hardness             0
Solids               0
Chloramines          0
Sulfate              0
Conductivity         0
Organic_carbon       0
Trihalomethanes    162
Turbidity            0
Potability           0
dtype: int64
```

In [17]:
```
data['Trihalomethanes'].fillna(data['Trihalomethanes'].mean(), inplace=True)
```

In [18]:
```
data.isnull().sum()
```

Out[18]:
```
ph                 0
Hardness           0
Solids             0
Chloramines        0
Sulfate            0
Conductivity       0
Organic_carbon     0
Trihalomethanes    0
Turbidity          0
Potability         0
dtype: int64
```

This shows data type and description of data

In [19]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3276 entries, 0 to 3275
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ph               3276 non-null   float64
 1   Hardness         3276 non-null   float64
 2   Solids           3276 non-null   float64
 3   Chloramines      3276 non-null   float64
 4   Sulfate          3276 non-null   float64
 5   Conductivity     3276 non-null   float64
 6   Organic_carbon   3276 non-null   float64
 7   Trihalomethanes  3276 non-null   float64
 8   Turbidity        3276 non-null   float64
 9   Potability       3276 non-null   int64  
dtypes: float64(9), int64(1)
memory usage: 256.1 KB
```

In [20]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3276 entries, 0 to 3275
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ph               3276 non-null   float64
 1   Hardness         3276 non-null   float64
 2   Solids           3276 non-null   float64
 3   Chloramines      3276 non-null   float64
 4   Sulfate          3276 non-null   float64
 5   Conductivity     3276 non-null   float64
 6   Organic_carbon   3276 non-null   float64
 7   Trihalomethanes  3276 non-null   float64
 8   Turbidity        3276 non-null   float64
 9   Potability       3276 non-null   int64  
dtypes: float64(9), int64(1)
memory usage: 256.1 KB
```

In [21]:
```
data.describe()
```

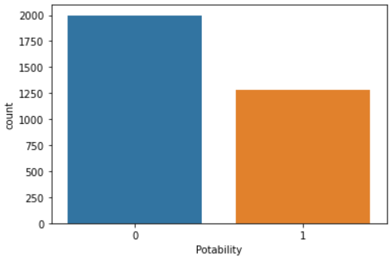
Out[21]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
count	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000
mean	7.080795	196.369496	22014.092526	7.122277	333.775777	426.205111	14.284970	66.396293	3.966786	0.390110
std	1.469956	32.879761	8768.570828	1.583085	36.142612	80.824064	3.308162	15.769881	0.780382	0.487849
min	0.000000	47.432000	320.942611	0.352000	129.000000	181.483754	2.200000	0.738000	1.450000	0.000000
25%	6.277673	176.850538	15666.690297	6.127421	317.094638	365.734414	12.065801	56.647656	3.439711	0.000000
50%	7.080795	196.967627	20927.833607	7.130299	333.775777	421.884968	14.218338	66.396293	3.955028	0.000000
75%	7.870050	216.667456	27332.762127	8.114887	350.385756	481.792304	16.557652	76.666609	4.500320	1.000000
max	14.000000	323.124000	61227.196008	13.127000	481.030642	753.342620	28.300000	124.000000	6.739000	1.000000

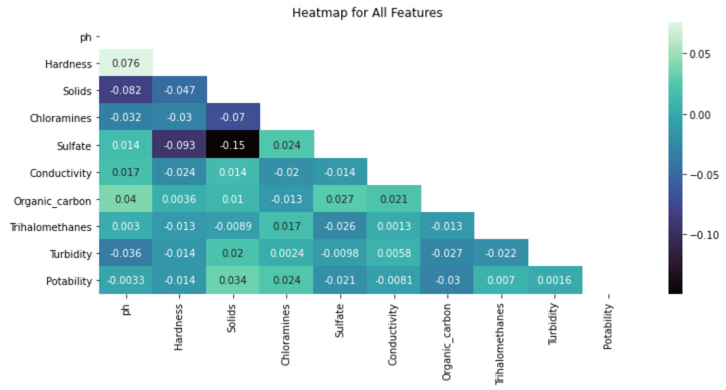
EDA (exploratory data analysis)

In [22]:
```
sns.countplot(x="Potability",data=data)
```

Out[22]: <AxesSubplot:xlabel='Potability', ylabel='count'>



In [23]:
```
plt.figure(figsize=(12, 5))
mask = np.triu(np.ones_like(data.corr(), dtype= bool))
sns.heatmap(data.corr(), mask=mask,annot=True,cmap='mako')
plt.title('Heatmap for All Features');
```



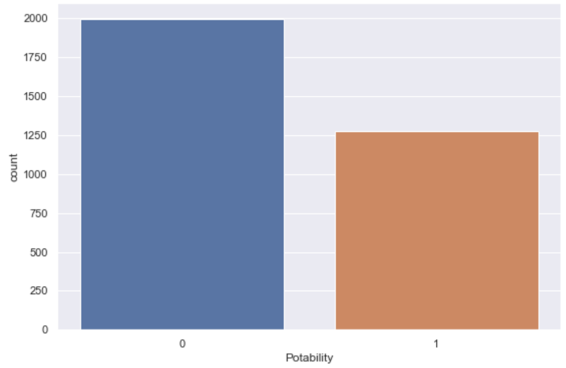
See the correlation of all the feature variables with the target variable

In [24]:
```
data['Potability'].value_counts()
```

Out[24]:
```
0    1998
1    1278
Name: Potability, dtype: int64
```

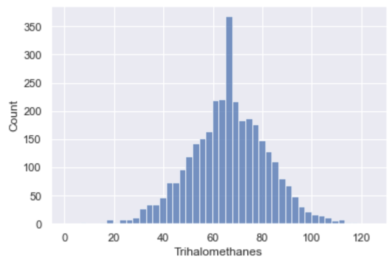
This will help to count the number of potable and non potable water

In [25]:
```
plt.figure(figsize=(9,6))
sns.set_theme(style="darkgrid")
sns.countplot(x="Potability", data=data)#countplot is barplot for seaborn
plt.plot();
```



In [26]:
```
sns.histplot(data=data, x='Trihalomethanes')
```

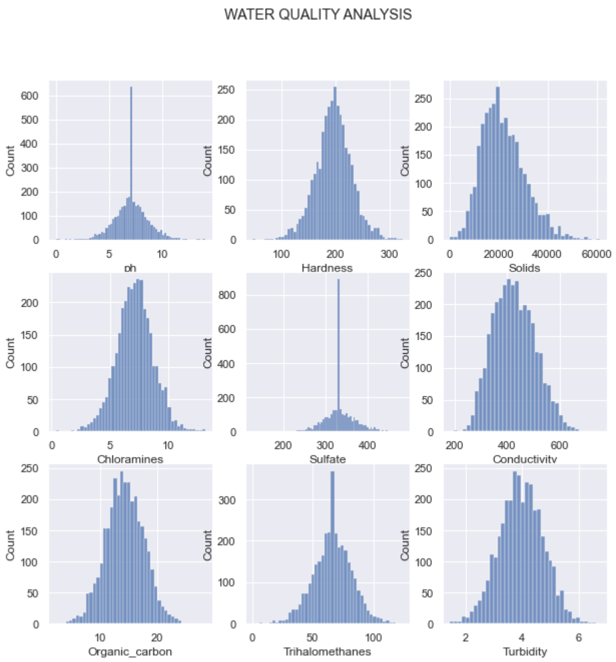
Out[26]: `<AxesSubplot:xlabel='Trihalomethanes', ylabel='Count'>`



SUBPLOTS

These are used to plot graphs for individual columns seperately

In [27]:
```
fig, axes = plt.subplots(3,3, figsize=(10,10))
fig.suptitle("WATER QUALITY ANALYSIS")
sns.histplot(ax= axes[0,0], data=data , x='ph')
sns.histplot(ax= axes[0,1], data=data , x='Hardness')
sns.histplot(ax= axes[0,2], data=data , x='Solids')
sns.histplot(ax= axes[1,0], data=data , x='Chloramines')
sns.histplot(ax= axes[1,1], data=data , x='Sulfate')
sns.histplot(ax= axes[1,2], data=data , x='Conductivity')
sns.histplot(ax= axes[2,0], data=data , x='Organic_carbon')
sns.histplot(ax= axes[2,1], data=data , x='Trihalomethanes')
sns.histplot(ax= axes[2,2], data=data , x='Turbidity');
```



In [28]:
```
non_potable = data[data["Potability"]== 0] #we are assigning variable data with potability 0
potable = data[data["Potability"] == 1]#we are assigning the variable data with potability 1
non_potable
```

Out[28]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	7.080795	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	129.422921	18630.057858	6.635246	333.775777	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.541732	9.275884	333.775777	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0
...	...	...	...	...	...	...	...	...	...	...
3112	6.616731	195.096968	34277.760400	7.632639	333.775777	417.465080	13.432557	47.945936	3.622379	0
3113	7.734569	230.919506	21776.594455	6.908591	333.775777	395.114961	15.033557	92.697369	3.821456	0
3114	6.971577	185.906938	27959.987873	7.214510	349.743879	414.067354	19.882917	36.179003	3.226349	0
3115	4.709187	179.141018	22291.418577	6.774276	407.417977	371.264843	18.186801	86.528627	3.860084	0
3116	5.230003	176.714023	27971.891806	7.597981	413.914001	440.355374	14.423614	72.837370	3.045612	0

1998 rows × 10 columns

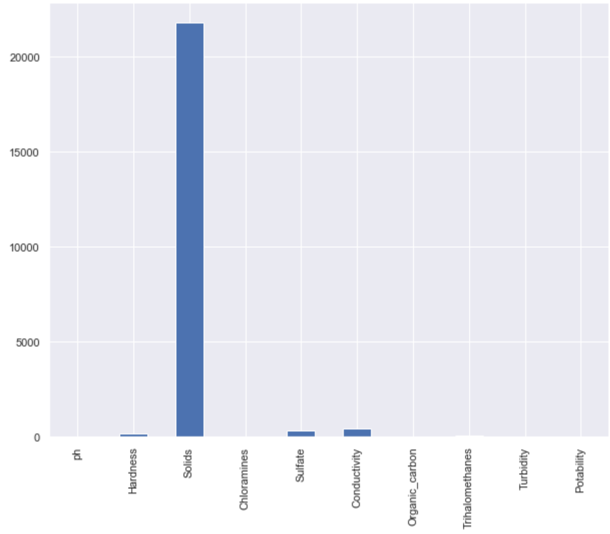
this is the mean of all paramneters with potability 0

In [29]:
```
agg_non_potable = non_potable.mean()
agg_non_potable
```

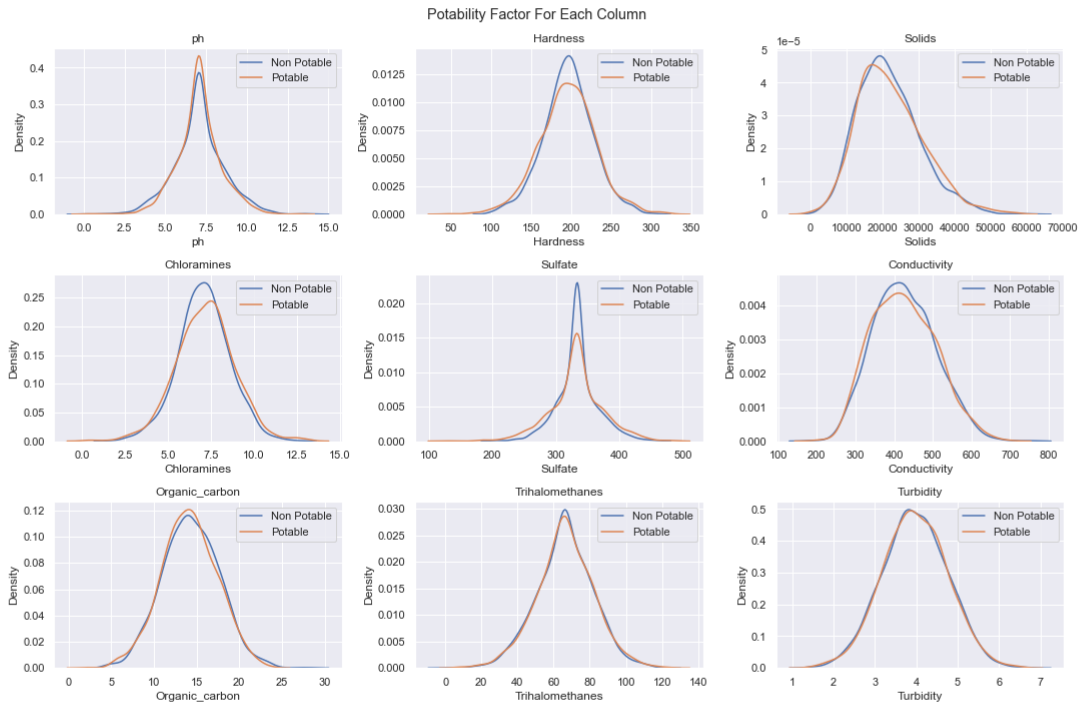
Out[29]:
```
ph                      7.084658
Hardness              196.733292
Solids              21777.490788
Chloramines             7.092175
Sulfate               334.371700
Conductivity          426.730454
Organic_carbon         14.364335
Trihalomethanes        66.308522
Turbidity               3.965800
Potability              0.000000
dtype: float64
```

In [30]:
```
plt.figure(figsize=(10,8))
agg_non_potable.plot(kind='bar')
```

Out[30]: <AxesSubplot:>

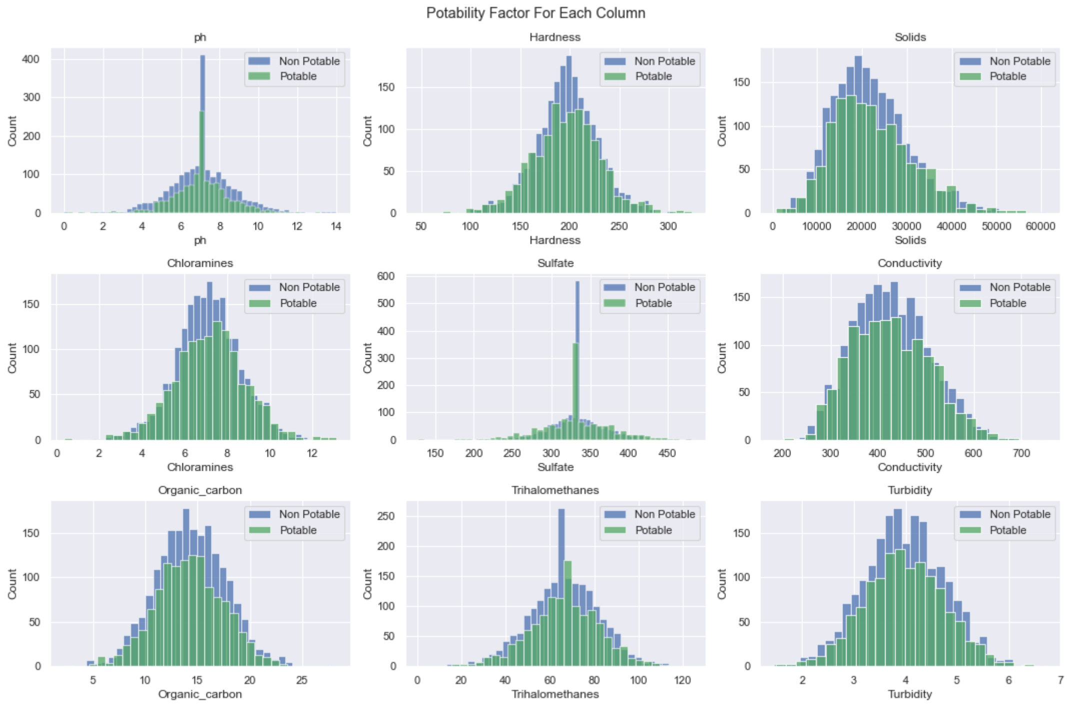


In [31]:
```
plt.figure(figsize=(15,10))
for ax, col in enumerate(data.columns[:9]):
    plt.subplot(3,3, ax+1)
    plt.suptitle("Potability Factor For Each Column")
    plt.title(col)
    sns.kdeplot(x=non_potable[col], label= "Non Potable")
    sns.kdeplot(x=potable[col], label="Potable")
    plt.legend()
plt.tight_layout()
```



This plot shows how the various factors effects the density of water

In [32]:
```
plt.figure(figsize=(15,10))
for ax, col in enumerate(data.columns[:9]):
    plt.subplot(3,3, ax+1)
    plt.suptitle("Potability Factor For Each Column")
    plt.title(col)
    sns.histplot(x=non_potable[col], label= "Non Potable")
    sns.histplot(x=potable[col], label="Potable",color="g")
    plt.legend()
plt.tight_layout()
```



Factors effecting the potability of water

In [33]:
```
data["Hardness"]
```

Out[33]:
```
0       204.890455
1       129.422921
2       224.236259
3       214.373394
4       181.101509
           ...
3271    193.681735
3272    193.553212
3273    175.762646
3274    230.603758
3275    195.102299
Name: Hardness, Length: 3276, dtype: float64
```

This functions helps to determine hardness of water and catergories it into soft, slightly hard, moderately hard, hard, very hard

In [34]:
```
def hardeness(x):
    if x<17.1:
        x="Soft"
    elif 17.1<x<60:
        x="Slightly Hard"
    elif 60 <x <120:
        x="Moderately Hard"
    elif 120 <x <180:
        x="Hard"
    elif x > 180:
        x="Very Hard"
    return x
```

This functions helps to determine nature of water by calculating its pH

In [35]:
```
def phs(x):
    if (x > 9):
        x = "Alkaline water"
    elif (x <= 9 and x > 8):
        x = "Bottled waters labeled as alkaline"
    elif (x <= 8 and x > 7.5 ):
        x = "Ocean water"
    elif(x == 7.5 ):
        x = "Tap water"
    elif(x < 7.5 and x >=6.5):
        x = "Water Bottles"
    elif(x < 6.5 and x >=5.5):
        x = "Distilled osmosis water"
    else:
        x = "Acidic water"
    return x
```

In [36]:
```
data["ph_Scale"] = data["ph"].apply(phs)
```

In [37]:
```
data["Hard"] = data["Hardness"].apply(hardeness)
```

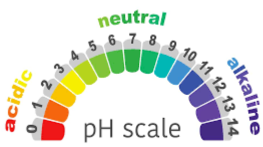
In [38]:
```
data["Hard"].value_counts() #general count of amount of water samples that are categorised
```

Out[38]:
```
Very Hard          2341
Hard                883
Moderately Hard      51
Slightly Hard         1
Name: Hard, dtype: int64
```

WATER HARDNESS SCALE					
ppm as $CaCO_3$	Grains/Gallon	German degrees	Clark degrees	French degrees	Classification
<60	<3.5	<3.4	<4.2	<6.0	Soft
61 - 120	3.51 – 6.96	3.41 – 6.72	4.21 – 8.40	6.1 – 12.0	Moderately Hard
121 - 180	6.97 – 10.44	6.73 -10.08	8.40 – 12.60	12.1 – 18.0	Hard
>180	>10.44	>10.08	>12.60	>18.0	Very Hard

In [39]:
```
data["ph_Scale"].value_counts() #general count of amount of water samples that are categorised
```

Out[39]:
```
Water Bottles                         1249
Distilled osmosis water                554
Bottled waters labeled as alkaline     424
Acidic water                           414
Ocean water                            328
Alkaline water                         307
Name: ph_Scale, dtype: int64
```



In [40]:
```
data
```

Out[40]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability	ph_Scale	Hard
0	7.080795	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0	Water Bottles	Very Hard
1	3.716080	129.422921	18630.057858	6.635246	333.775777	592.885359	15.180013	56.329076	4.500656	0	Acidic water	Hard
2	8.099124	224.236259	19909.541732	9.275884	333.775777	418.606213	16.868637	66.420093	3.055934	0	Bottled waters labeled as alkaline	Very Hard
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0	Bottled waters labeled as alkaline	Very Hard
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0	Alkaline water	Very Hard
...	...	...	...	...	...	...	...	...	...	...	...	...
3271	4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.894419	66.687695	4.435821	1	Acidic water	Very Hard
3272	7.808856	193.553212	17329.802160	8.061362	333.775777	392.449580	19.903225	66.396293	2.798243	1	Ocean water	Very Hard
3273	9.419510	175.762646	33155.578218	7.350233	333.775777	432.044783	11.039070	69.845400	3.298875	1	Alkaline water	Hard
3274	5.126763	230.603758	11983.869376	6.303357	333.775777	402.883113	11.168946	77.488213	4.708658	1	Acidic water	Very Hard
3275	7.874671	195.102299	17404.177061	7.509306	333.775777	327.459760	16.140368	78.698446	2.309149	1	Ocean water	Very Hard

3276 rows × 12 columns

In [41]:
```
data = data[['ph','ph_Scale','Hardness','Hard','Solids','Chloramines','Sulfate','Conductivity','Organic_carbon','Trihalomethanes','Turbidity','Potability']]
```

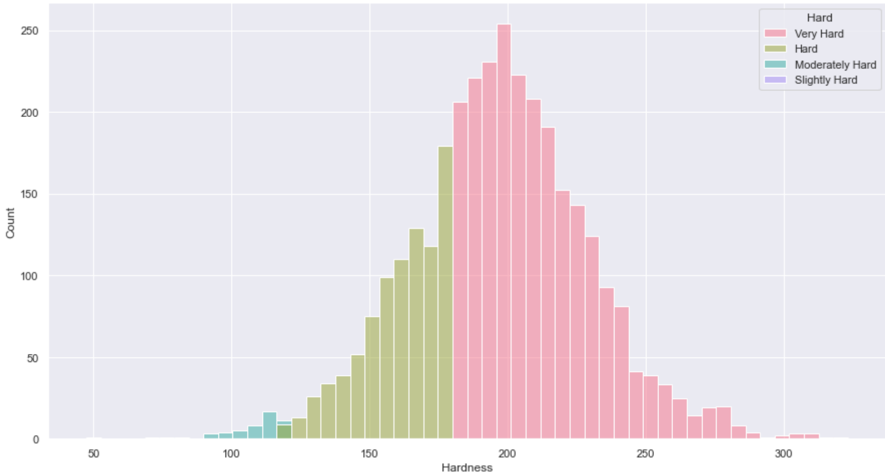
In [42]:
```
data
```

Out[42]:

	ph	ph_Scale	Hardness	Hard	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	7.080795	Water Bottles	204.890455	Very Hard	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	Acidic water	129.422921	Hard	18630.057858	6.635246	333.775777	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	Bottled waters labeled as alkaline	224.236259	Very Hard	19909.541732	9.275884	333.775777	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	Bottled waters labeled as alkaline	214.373394	Very Hard	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
4	9.092223	Alkaline water	181.101509	Very Hard	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0
...	...	...	...	...	...	...	...	...	...	...	...	...
3271	4.668102	Acidic water	193.681735	Very Hard	47580.991603	7.166639	359.948574	526.424171	13.894419	66.687695	4.435821	1
3272	7.808856	Ocean water	193.553212	Very Hard	17329.802160	8.061362	333.775777	392.449580	19.903225	66.396293	2.798243	1
3273	9.419510	Alkaline water	175.762646	Hard	33155.578218	7.350233	333.775777	432.044783	11.039070	69.845400	3.298875	1
3274	5.126763	Acidic water	230.603758	Very Hard	11983.869376	6.303357	333.775777	402.883113	11.168946	77.488213	4.708658	1
3275	7.874671	Ocean water	195.102299	Very Hard	17404.177061	7.509306	333.775777	327.459760	16.140368	78.698446	2.309149	1

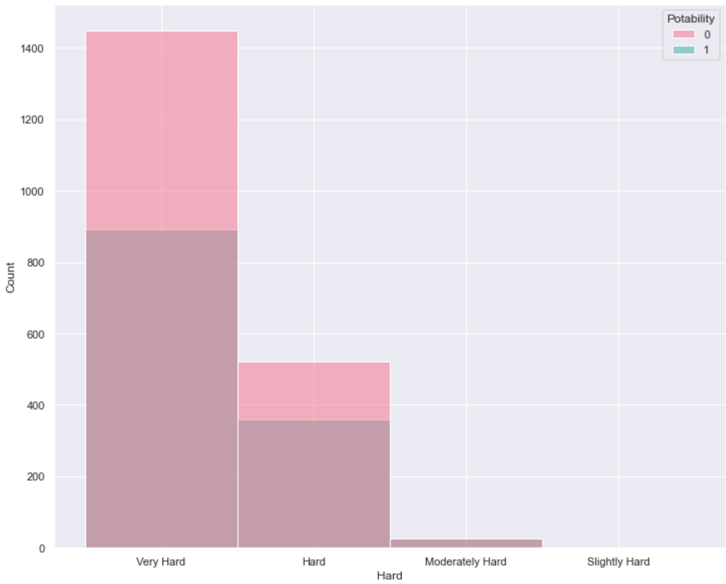
3276 rows × 12 columns

In [43]:
```
plt.figure(figsize=(15,8))
sns.histplot(x="Hardness", hue="Hard", data=data, palette="husl");
```



Categorising water according to its hardness

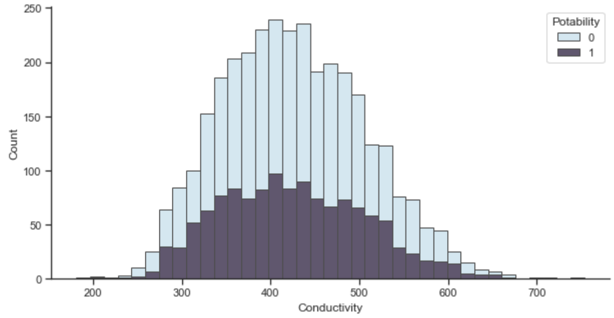
In [44]:
```
plt.figure(figsize=(12,10))
sns.histplot(x="Hard",hue="Potability", data=data,palette="husl");
```



Potability of water according to its hardness

In [45]:
```
sns.set_theme(style="ticks")
f, ax = plt.subplots(figsize=(10,5))
sns.despine(f)

sns.histplot(data,x="Conductivity", hue="Potability",multiple="stack",palette="ch:s=.25,rot=-.25",edgecolor=".3");
```



Graph determines the conductivity of water according to its potability

CONCLUSION

After considering all the parameters and comparing the hardness and pH level of water we can finally represent the potability of water in the form of pie chart categorise the water in potable and non-potable form. As we can see that the percentage of non potable water is more than that potable which is alarming and we should be working on its conservation.

The best way to solve these issues is to prevent them. The first major solution in this context is conservation of soil. Soil erosion can contribute to water pollution. So, if soil can be conserved we can prevent water pollution too. We can follow measures such as planting more trees, managing erosion in a better way, and use farming methods that are better for the soil. In the same vein it is also important to follow the right methods in disposing toxic waste. For starters, we can use products that have lesser amounts of volatile organic compounds in them. Even in cases where toxic material like paints, cleaning supplies, and stain removers are used, they need to be disposed off in the right way. It is also important to look into oil leaks in one's cars and machines.

It is said that leaked oil – even from cars and machines – is one of the principal contributors to water pollution. Hence, it is important to look at cars and machines, which run on oil, on a regular basis, to check them for any possible oil leak. It is important after work – especially in factories and production units where oil is used – to clean up the wasted oil and either dispose it properly or keep it for later use. Following are some other ways in which this problem can be addressed adequately:

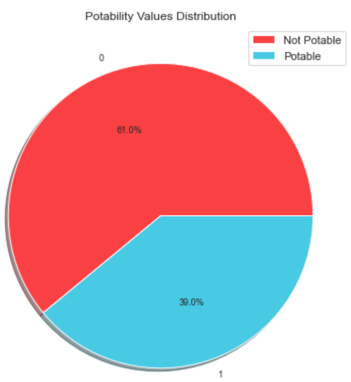
-Cleaning up waterways and beaches

-Avoiding the usage of non-biodegradable material like plastic

-Being more involved in various measures pertaining to preventing water pollution.

In [46]:
```
colors=['#f94144', '#48cae4']
labels=['Not Potable','Potable']
pieplot = data.groupby('Potability').size()
pieplot.plot(kind='pie', colors=colors, subplots=True,shadow=True, figsize=(7, 7), fontsize=9, autopct='%1.1f%%')
plt.title("Potability Values Distribution")
plt.legend(labels)
plt.ylabel("")
```

Out[46]: `Text(0, 0.5, '')`



Finally this graphs determine the amount of potable and non-potable water sample

FUTURE WORK

With help of the data set that we have worked on we can further enhance this project by adding machine learning models which help us to determine the potability of water with help of trained datasets. By adding more columns to the data we can have other factors to decide the quality of water as well, which will make our analysis much more precise and informative.