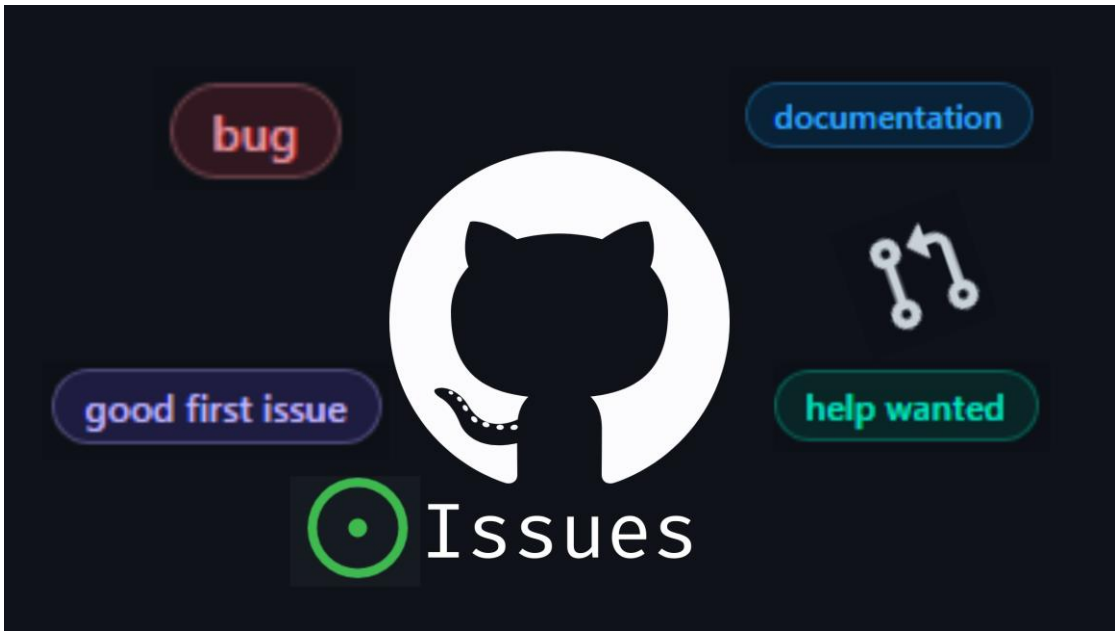


Exploratory Data Analysis

GITHUB TICKET ISSUE

Mansoor Bukhari | Artificial Intelligence | 8/21/2024

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


### Load dataset

df = pd.read_csv('archive/github_issues_tickets.csv',
low_memory=False)
```

Check number of rows and columns

```
df.shape
```

```
(15955, 132)
```

Show Column

```
print('These are the columns which are present in our dataset')
i = 1
for column in df.columns:
    print(i,'.', column)
    i += 1
del i
```

```
These are the columns which are present in our dataset
1 . answers_0_author
2 . answers_0_body
3 . answers_0_creation_time
4 . answers_1_author
5 . answers_1_body
6 . answers_1_creation_time
7 . answers_2_author
8 . answers_2_body
9 . answers_2_creation_time
10 . answers_3_author
11 . answers_3_body
12 . answers_3_creation_time
13 . answers_4_author
14 . answers_4_body
15 . answers_4_creation_time
16 . answers_5_author
17 . answers_5_body
18 . answers_5_creation_time
19 . answers_6_author
```

```
20 . answers_6_body
21 . answers_6_creation_time
22 . answers_7_author
23 . answers_7_body
24 . answers_7_creation_time
25 . answers_8_author
26 . answers_8_body
27 . answers_8_creation_time
28 . answers_9_author
29 . answers_9_body
30 . answers_9_creation_time
31 . assignee
32 . assignee_id
33 . assignee_login
34 . assignee_type
35 . assignee_url
36 . body
37 . closed_at
38 . comments
39 . comments_url
40 . created_at
41 . html_url
42 . id
43 . labels_0_color
44 . labels_0_default
45 . labels_0_description
46 . labels_0_id
47 . labels_0_name
48 . labels_0_url
49 . labels_10_color
50 . labels_10_default
51 . labels_10_description
52 . labels_10_id
53 . labels_10_name
54 . labels_10_url
55 . labels_1_color
56 . labels_1_default
57 . labels_1_description
58 . labels_1_id
59 . labels_1_name
60 . labels_1_url
61 . labels_2_color
62 . labels_2_default
63 . labels_2_description
64 . labels_2_id
65 . labels_2_name
66 . labels_2_url
67 . labels_3_color
68 . labels_3_default
```

```
69 . labels_3_description
70 . labels_3_id
71 . labels_3_name
72 . labels_3_url
73 . labels_4_color
74 . labels_4_default
75 . labels_4_description
76 . labels_4_id
77 . labels_4_name
78 . labels_4_url
79 . labels_5_color
80 . labels_5_default
81 . labels_5_description
82 . labels_5_id
83 . labels_5_name
84 . labels_5_url
85 . labels_6_color
86 . labels_6_default
87 . labels_6_description
88 . labels_6_id
89 . labels_6_name
90 . labels_6_url
91 . labels_7_color
92 . labels_7_default
93 . labels_7_description
94 . labels_7_id
95 . labels_7_name
96 . labels_7_url
97 . labels_8_color
98 . labels_8_default
99 . labels_8_description
100 . labels_8_id
101 . labels_8_name
102 . labels_8_url
103 . labels_9_color
104 . labels_9_default
105 . labels_9_description
106 . labels_9_id
107 . labels_9_name
108 . labels_9_url
109 . milestone
110 . milestone_description
111 . milestone_due_on
112 . milestone_title
113 . reactions_confused
114 . reactions_eyes
115 . reactions_heart
116 . reactions_hooray
117 . reactions_laugh
```

```
118 . reactions_minus_1
119 . reactions_plus_1
120 . reactions_rocket
121 . reactions_total_count
122 . reactions_url
123 . repo_name
124 . state
125 . state_reason
126 . title
127 . updated_at
128 . url
129 . user_id
130 . user_login
131 . user_type
132 . user_url
```

Check for null values

```
df.isna().sum()
```

```
answers_0_author               0
answers_0_body                 0
answers_0_creation_time        0
answers_1_author            3447
answers_1_body              3447
                              ...
url                            0
user_id                        0
user_login                     0
user_type                      0
user_url                       0
Length: 132, dtype: int64
```

Print all Columns

```
print(df.isna().sum().to_string())
```

```
answers_0_author                0
answers_0_body                  0
answers_0_creation_time         0
answers_1_author             3447
answers_1_body               3447
answers_1_creation_time      3447
answers_2_author             6949
answers_2_body               6949
answers_2_creation_time      6949
```

```
answers_3_author              9520
answers_3_body                9520
answers_3_creation_time       9520
answers_4_author             11442
answers_4_body               11442
answers_4_creation_time      11442
answers_5_author             12841
answers_5_body               12841
answers_5_creation_time      12841
answers_6_author             13808
answers_6_body               13808
answers_6_creation_time      13808
answers_7_author             14616
answers_7_body               14616
answers_7_creation_time      14616
answers_8_author             15200
answers_8_body               15200
answers_8_creation_time      15200
answers_9_author             15636
answers_9_body               15636
answers_9_creation_time      15636
assignee                     15955
assignee_id                  12497
assignee_login               12497
assignee_type                12497
assignee_url                 12497
body                             0
closed_at                        0
comments                         0
comments_url                     0
created_at                       0
html_url                         0
id                               0
labels_0_color                   9
labels_0_default                 9
labels_0_description          8692
labels_0_id                      9
labels_0_name                    9
labels_0_url                     9
labels_10_color              15951
labels_10_default            15951
labels_10_description        15955
labels_10_id                 15951
labels_10_name               15951
labels_10_url                15951
labels_1_color                7517
labels_1_default              7517
labels_1_description         12014
labels_1_id                   7517
```

```
labels_1_name                7517
labels_1_url                 7517
labels_2_color              12172
labels_2_default            12172
labels_2_description        13987
labels_2_id                 12172
labels_2_name               12172
labels_2_url                12172
labels_3_color              14409
labels_3_default            14409
labels_3_description        15066
labels_3_id                 14409
labels_3_name               14409
labels_3_url                14409
labels_4_color              15293
labels_4_default            15293
labels_4_description        15526
labels_4_id                 15293
labels_4_name               15293
labels_4_url                15293
labels_5_color              15654
labels_5_default            15654
labels_5_description        15741
labels_5_id                 15654
labels_5_name               15654
labels_5_url                15654
labels_6_color              15811
labels_6_default            15811
labels_6_description        15844
labels_6_id                 15811
labels_6_name               15811
labels_6_url                15811
labels_7_color              15871
labels_7_default            15871
labels_7_description        15889
labels_7_id                 15871
labels_7_name               15871
labels_7_url                15871
labels_8_color              15911
labels_8_default            15911
labels_8_description        15931
labels_8_id                 15911
labels_8_name               15911
labels_8_url                15911
labels_9_color              15936
labels_9_default            15936
labels_9_description        15947
labels_9_id                 15936
labels_9_name               15936
```

```
labels_9_url                15936
milestone                   15955
milestone_description       15261
milestone_due_on            15100
milestone_title             13546
reactions_confused              0
reactions_eyes                  0
reactions_heart                 0
reactions_hooray                0
reactions_laugh                 0
reactions_minus_1               0
reactions_plus_1                0
reactions_rocket                0
reactions_total_count           0
reactions_url                   0
repo_name                       0
state                           0
state_reason                    0
title                           0
updated_at                      0
url                             0
user_id                         0
user_login                      0
user_type                       0
user_url                        0
```

Print Only Columns which have null values

```
# assign null values into missing_values variable
missing_values = df.isna().sum()
columns_with_missing = missing_values[missing_values > 0] # Filter
columns with missing values

print(columns_with_missing.to_string())
```

```
answers_1_author             3447
answers_1_body               3447
answers_1_creation_time      3447
answers_2_author             6949
answers_2_body               6949
answers_2_creation_time      6949
answers_3_author             9520
answers_3_body               9520
answers_3_creation_time      9520
answers_4_author            11442
answers_4_body              11442
answers_4_creation_time     11442
answers_5_author            12841
```

```
answers_5_body              12841
answers_5_creation_time     12841
answers_6_author            13808
answers_6_body              13808
answers_6_creation_time     13808
answers_7_author            14616
answers_7_body              14616
answers_7_creation_time     14616
answers_8_author            15200
answers_8_body              15200
answers_8_creation_time     15200
answers_9_author            15636
answers_9_body              15636
answers_9_creation_time     15636
assignee                    15955
assignee_id                 12497
assignee_login              12497
assignee_type               12497
assignee_url                12497
labels_0_color                  9
labels_0_default                9
labels_0_description         8692
labels_0_id                     9
labels_0_name                   9
labels_0_url                    9
labels_10_color             15951
labels_10_default           15951
labels_10_description       15955
labels_10_id                15951
labels_10_name              15951
labels_10_url               15951
labels_1_color               7517
labels_1_default             7517
labels_1_description        12014
labels_1_id                  7517
labels_1_name                7517
labels_1_url                 7517
labels_2_color              12172
labels_2_default            12172
labels_2_description        13987
labels_2_id                 12172
labels_2_name               12172
labels_2_url                12172
labels_3_color              14409
labels_3_default            14409
labels_3_description        15066
labels_3_id                 14409
labels_3_name               14409
labels_3_url                14409
```

```
labels_4_color              15293
labels_4_default            15293
labels_4_description        15526
labels_4_id                 15293
labels_4_name               15293
labels_4_url                15293
labels_5_color              15654
labels_5_default            15654
labels_5_description        15741
labels_5_id                 15654
labels_5_name               15654
labels_5_url                15654
labels_6_color              15811
labels_6_default            15811
labels_6_description        15844
labels_6_id                 15811
labels_6_name               15811
labels_6_url                15811
labels_7_color              15871
labels_7_default            15871
labels_7_description        15889
labels_7_id                 15871
labels_7_name               15871
labels_7_url                15871
labels_8_color              15911
labels_8_default            15911
labels_8_description        15931
labels_8_id                 15911
labels_8_name               15911
labels_8_url                15911
labels_9_color              15936
labels_9_default            15936
labels_9_description        15947
labels_9_id                 15936
labels_9_name               15936
labels_9_url                15936
milestone                   15955
milestone_description       15261
milestone_due_on            15100
milestone_title             13546
```

Note: `Most of the columns have large number of missing values`

```
```

Check info of dataset

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15955 entries, 0 to 15954
Columns: 132 entries, answers_0_author to user_url
dtypes: float64(15), int64(12), object(105)
memory usage: 16.1+ MB
```

Show reaction columns name

```
for i in range(112, 122):
    print(df.columns[i], end=', ')
```

```
reactions_confused, reactions_eyes, reactions_heart, reactions_hooray,
reactions_laugh, reactions_minus_1, reactions_plus_1,
reactions_rocket, reactions_total_count, reactions_url,
```

Let's Store Reactions columns in list

```
reaction_columns = df.columns[112:121]
```

Show Maximum value of reaction

```
i = 1
for reaction in reaction_columns:
    print(f'{i}. {reaction} {df[reaction].max()}')
    i += 1
del i
```

```
1. reactions_confused 2
2. reactions_eyes 8
3. reactions_heart 5
4. reactions_hooray 3
5. reactions_laugh 3
6. reactions_minus_1 3
7. reactions_plus_1 46
8. reactions_rocket 4
9. reactions_total_count 46
```

Show Minimum value of reaction

```
i = 1
for reaction in reaction_columns:
    print(f'{i}. {reaction} {df[reaction].min()}')
```

```
    i += 1
del i
```

```
1. reactions_confused 0
2. reactions_eyes 0
3. reactions_heart 0
4. reactions_hooray 0
5. reactions_laugh 0
6. reactions_minus_1 0
7. reactions_plus_1 0
8. reactions_rocket 0
9. reactions_total_count 0
```

Sort Dataset on the basis of reaction

```
df.sort_values(by='reactions_total_count', inplace=True)
```

Show Top 5 Values

```
df['reactions_total_count'].tail()
```

```
8423     24
10661    25
7569     25
5401     28
2371     46
Name: reactions_total_count, dtype: int64
```

Show Low reaction

```
df['reactions_total_count'].head()
```

```
0        0
10218    0
10219    0
10220    0
10221    0
Name: reactions_total_count, dtype: int64
```

Convert Date Into Date

```
filtered_df = df[df['milestone_due_on'].notnull()]
print('Values which are not null in milestone_due_on are')
i = 1
```

```
for row in filtered_df['milestone_due_on']:
    print(f'{i}. {row}')
    if i == 5:
        break
    i += 1
del i
```

```
Values which are not null in milestone_due_on are
1. 2021-10-29T07:00:00Z
2. 2020-10-31T07:00:00Z
3. 2023-03-13T07:00:00Z
4. 2023-01-07T08:00:00Z
5. 2023-05-16T07:00:00Z
```

```
df['milestone_due_on'].info()
```

```
<class 'pandas.core.series.Series'>
Index: 15955 entries, 0 to 2371
Series name: milestone_due_on
Non-Null Count  Dtype
--------------  -----
855 non-null    object
dtypes: object(1)
memory usage: 249.3+ KB
```

```
df['milestone_due_on'] = pd.to_datetime(df['milestone_due_on'],
format='%Y-%m-%dT%H:%M:%SZ', errors='coerce')
```

```
df['milestone_due_on'].info()
```

```
<class 'pandas.core.series.Series'>
Index: 15955 entries, 0 to 2371
Series name: milestone_due_on
Non-Null Count  Dtype
--------------  -----
854 non-null    datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 249.3 KB
```

Noticed that their are more than one columns containg date

Note: One things is common b/w these columns that they have word time in them

```
# Select columns containing "time" in their names
time_columns = df.filter(like='time').columns

print(f'These are columns contating dates')
i = 1
```

```
for t in time_columns:
    print(f'{i}. {t}')
    i += 1

del i
```

```
These are columns contating dates
1. answers_0_creation_time
2. answers_1_creation_time
3. answers_2_creation_time
4. answers_3_creation_time
5. answers_4_creation_time
6. answers_5_creation_time
7. answers_6_creation_time
8. answers_7_creation_time
9. answers_8_creation_time
10. answers_9_creation_time
```

```
df[time_columns].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 15955 entries, 0 to 2371
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   answers_0_creation_time  15955 non-null  object
 1   answers_1_creation_time  12508 non-null  object
 2   answers_2_creation_time  9006 non-null   object
 3   answers_3_creation_time  6435 non-null   object
 4   answers_4_creation_time  4513 non-null   object
 5   answers_5_creation_time  3114 non-null   object
 6   answers_6_creation_time  2147 non-null   object
 7   answers_7_creation_time  1339 non-null   object
 8   answers_8_creation_time  755 non-null    object
 9   answers_9_creation_time  319 non-null    object
dtypes: object(10)
memory usage: 1.3+ MB
```

```
df[time_columns].head(4)
```

```
         answers_0_creation_time    answers_1_creation_time  \
0      2023-05-05T18:09:31+00:00  2023-05-05T18:19:32+00:00
10218  2020-12-06T15:06:41+00:00  2020-12-06T15:17:31+00:00
10219  2024-08-04T14:28:59+00:00  2024-08-05T20:59:55+00:00
10220  2024-03-31T20:53:54+00:00  2024-05-13T19:12:58+00:00

         answers_2_creation_time    answers_3_creation_time  \
0      2023-05-16T21:51:45+00:00  2023-06-19T00:09:04+00:00
10218  2020-12-06T15:22:07+00:00  2020-12-06T15:28:18+00:00
10219  2024-08-05T22:46:02+00:00                        NaN
10220                        NaN                        NaN
```

```
         answers_4_creation_time    answers_5_creation_time  \
0                            NaN                       NaN
10218  2020-12-06T15:31:09+00:00  2020-12-06T15:43:03+00:00
10219                        NaN                       NaN
10220                        NaN                       NaN

         answers_6_creation_time answers_7_creation_time  \
0                            NaN                    NaN
10218  2021-07-12T11:40:22+00:00                    NaN
10219                        NaN                    NaN
10220                        NaN                    NaN

      answers_8_creation_time answers_9_creation_time
0                         NaN                    NaN
10218                     NaN                    NaN
10219                     NaN                    NaN
10220                     NaN                    NaN
```

Convert their data type into datetime, as till now thay have object

```
for time_col in time_columns:
    df[time_col] = pd.to_datetime(df[time_col])

df[time_columns].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 15955 entries, 0 to 2371
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   answers_0_creation_time  15955 non-null  datetime64[ns, UTC]
 1   answers_1_creation_time  12508 non-null  datetime64[ns, UTC]
 2   answers_2_creation_time  9006 non-null   datetime64[ns, UTC]
 3   answers_3_creation_time  6435 non-null   datetime64[ns, UTC]
 4   answers_4_creation_time  4513 non-null   datetime64[ns, UTC]
 5   answers_5_creation_time  3114 non-null   datetime64[ns, UTC]
 6   answers_6_creation_time  2147 non-null   datetime64[ns, UTC]
 7   answers_7_creation_time  1339 non-null   datetime64[ns, UTC]
 8   answers_8_creation_time  755 non-null    datetime64[ns, UTC]
 9   answers_9_creation_time  319 non-null    datetime64[ns, UTC]
dtypes: datetime64[ns, UTC](10)
memory usage: 1.3 MB
```

```
df[time_columns].head(4)
```

```
        answers_0_creation_time   answers_1_creation_time  \
0     2023-05-05 18:09:31+00:00 2023-05-05 18:19:32+00:00
10218 2020-12-06 15:06:41+00:00 2020-12-06 15:17:31+00:00
10219 2024-08-04 14:28:59+00:00 2024-08-05 20:59:55+00:00
10220 2024-03-31 20:53:54+00:00 2024-05-13 19:12:58+00:00
```

```
        answers_2_creation_time   answers_3_creation_time  \
0     2023-05-16 21:51:45+00:00 2023-06-19 00:09:04+00:00
10218 2020-12-06 15:22:07+00:00 2020-12-06 15:28:18+00:00
10219 2024-08-05 22:46:02+00:00                       NaT
10220                       NaT                       NaT

        answers_4_creation_time   answers_5_creation_time  \
0                           NaT                       NaT
10218 2020-12-06 15:31:09+00:00 2020-12-06 15:43:03+00:00
10219                       NaT                       NaT
10220                       NaT                       NaT

        answers_6_creation_time answers_7_creation_time  \
0                           NaT                     NaT
10218 2021-07-12 11:40:22+00:00                     NaT
10219                       NaT                     NaT
10220                       NaT                     NaT

      answers_8_creation_time answers_9_creation_time
0                         NaT                    NaT
10218                     NaT                    NaT
10219                     NaT                    NaT
10220                     NaT                    NaT
```

Some Date Columns also have word at, `Let's Identify Them`

```
# Select columns containing "_at" in their names
at_columns = df.filter(like='_at').columns

at_columns

Index(['closed_at', 'created_at', 'updated_at'], dtype='object')

df[at_columns]

                  closed_at            created_at
updated_at
0      2023-05-19T08:06:42Z  2023-05-05T15:54:28Z  2023-06-
19T00:09:05Z
10218  2021-07-12T12:18:12Z  2020-12-06T15:00:07Z  2021-07-
12T12:18:12Z
10219  2024-08-13T05:19:29Z  2024-08-03T20:17:41Z  2024-08-
13T05:19:29Z
10220  2024-06-27T17:56:07Z  2024-03-17T06:02:24Z  2024-06-
27T17:56:07Z
10221  2024-05-17T21:00:02Z  2024-05-12T11:26:23Z  2024-05-
17T21:00:03Z
...                     ...                   ...                   ..
```

```
.
8423   2024-08-03T16:37:54Z  2024-07-12T12:41:22Z  2024-08-
03T16:37:55Z
10661  2024-07-24T15:38:33Z  2024-07-11T12:13:59Z  2024-07-
24T18:39:53Z
7569   2024-07-31T11:34:41Z  2022-02-02T22:25:59Z  2024-08-
15T00:24:55Z
5401   2024-03-27T10:23:50Z  2022-04-21T09:23:13Z  2024-03-
27T10:23:50Z
2371   2017-07-20T08:20:06Z  2017-06-21T04:03:35Z  2017-07-
20T08:20:06Z

[15955 rows x 3 columns]
```

```
df[at_columns].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 15955 entries, 0 to 2371
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   closed_at   15955 non-null  object
 1   created_at  15955 non-null  object
 2   updated_at  15955 non-null  object
dtypes: object(3)
memory usage: 498.6+ KB
```

Convert it into datetime

```
for at_col in at_columns:
    df[at_col] = pd.to_datetime(df[at_col])

df[at_columns].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 15955 entries, 0 to 2371
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype              
---  ------      --------------  -----              
 0   closed_at   15955 non-null  datetime64[ns, UTC]
 1   created_at  15955 non-null  datetime64[ns, UTC]
 2   updated_at  15955 non-null  datetime64[ns, UTC]
dtypes: datetime64[ns, UTC](3)
memory usage: 498.6 KB
```

```
print('These are the columns which are present in our dataset')
for column in df.columns:
    print(column, end=' ')
```

```
These are the columns which are present in our dataset
answers_0_author answers_0_body answers_0_creation_time
answers_1_author answers_1_body answers_1_creation_time
answers_2_author answers_2_body answers_2_creation_time
answers_3_author answers_3_body answers_3_creation_time
answers_4_author answers_4_body answers_4_creation_time
answers_5_author answers_5_body answers_5_creation_time
answers_6_author answers_6_body answers_6_creation_time
answers_7_author answers_7_body answers_7_creation_time
answers_8_author answers_8_body answers_8_creation_time
answers_9_author answers_9_body answers_9_creation_time assignee
assignee_id assignee_login assignee_type assignee_url body closed_at
comments comments_url created_at html_url id labels_0_color
labels_0_default labels_0_description labels_0_id labels_0_name
labels_0_url labels_10_color labels_10_default labels_10_description
labels_10_id labels_10_name labels_10_url labels_1_color
labels_1_default labels_1_description labels_1_id labels_1_name
labels_1_url labels_2_color labels_2_default labels_2_description
labels_2_id labels_2_name labels_2_url labels_3_color labels_3_default
labels_3_description labels_3_id labels_3_name labels_3_url
labels_4_color labels_4_default labels_4_description labels_4_id
labels_4_name labels_4_url labels_5_color labels_5_default
labels_5_description labels_5_id labels_5_name labels_5_url
labels_6_color labels_6_default labels_6_description labels_6_id
labels_6_name labels_6_url labels_7_color labels_7_default
labels_7_description labels_7_id labels_7_name labels_7_url
labels_8_color labels_8_default labels_8_description labels_8_id
labels_8_name labels_8_url labels_9_color labels_9_default
labels_9_description labels_9_id labels_9_name labels_9_url milestone
milestone_description milestone_due_on milestone_title
reactions_confused reactions_eyes reactions_heart reactions_hooray
reactions_laugh reactions_minus_1 reactions_plus_1 reactions_rocket
reactions_total_count reactions_url repo_name state state_reason title
updated_at url user_id user_login user_type user_url
```

Get Neumeric Columns

```
# Select columns containing "_id" in their names
id_columns = df.filter(like='_id').columns

df[id_columns].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 15955 entries, 0 to 2371
```

```
Data columns (total 13 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   assignee_id    3458 non-null   float64
 1   labels_0_id    15946 non-null  float64
 2   labels_10_id   4 non-null      float64
 3   labels_1_id    8438 non-null   float64
 4   labels_2_id    3783 non-null   float64
 5   labels_3_id    1546 non-null   float64
 6   labels_4_id    662 non-null    float64
 7   labels_5_id    301 non-null    float64
 8   labels_6_id    144 non-null    float64
 9   labels_7_id    84 non-null     float64
 10  labels_8_id    44 non-null     float64
 11  labels_9_id    19 non-null     float64
 12  user_id        15955 non-null  int64  
dtypes: float64(12), int64(1)
memory usage: 1.7 MB
```

Descriptive Statistics

```
# Descriptive statistics for numerical columns
numerical_columns = df.select_dtypes(include=['number'])

numerical_columns.describe()
```

```
       assignee   assignee_id      comments            id  
labels_0_id  \
count       0.0  3.458000e+03  15955.000000  1.595500e+04  
1.594600e+04  
mean        NaN  1.749557e+07      3.515763  1.260217e+09  
5.939866e+08  
std         NaN  2.539853e+07      2.350171  7.164309e+08  
7.441277e+08  
min         NaN  1.880000e+02      0.000000  4.569830e+05  
4.691000e+03  
25%         NaN  1.035718e+06      2.000000  6.383548e+08  
9.818045e+07  
50%         NaN  6.304622e+06      3.000000  1.158454e+09  
3.181933e+08  
75%         NaN  2.321990e+07      5.000000  1.943650e+09  
8.235093e+08  
max         NaN  1.740995e+08     10.000000  2.471639e+09  
7.076707e+09  

       labels_10_description  labels_10_id   labels_1_id   labels_2_id
```

```
\
count                      0.0  4.000000e+00  8.438000e+03  3.783000e+03

mean                       NaN  1.321237e+09  1.445116e+09  1.745514e+09

std                        NaN  7.381434e+08  1.474591e+09  1.627694e+09

min                        NaN  9.444607e+08  1.502700e+04  2.061490e+05

25%                        NaN  9.531392e+08  3.740270e+08  5.673587e+08

50%                        NaN  9.560321e+08  9.092565e+08  1.178448e+09

75%                        NaN  1.324129e+09  1.891218e+09  2.386125e+09

max                        NaN  2.428421e+09  7.321408e+09  7.278782e+09


        labels_3_id  ...  reactions_confused  reactions_eyes
reactions_heart  \
count  1.546000e+03  ...        15955.000000    15955.000000
15955.000000
mean   2.044768e+09  ...            0.002256        0.016170
0.006518
std    1.681983e+09  ...            0.054805        0.176646
0.108935
min    3.736123e+07  ...            0.000000        0.000000
0.000000
25%    8.143317e+08  ...            0.000000        0.000000
0.000000
50%    1.369687e+09  ...            0.000000        0.000000
0.000000
75%    2.644694e+09  ...            0.000000        0.000000
0.000000
max    7.156337e+09  ...            2.000000        8.000000
5.000000

       reactions_hooray  reactions_laugh  reactions_minus_1
reactions_plus_1  \
count      15955.000000     15955.000000       15955.000000
15955.000000
mean           0.000940         0.001692           0.000752
0.305171
std            0.041128         0.055394           0.033581
1.301563
min            0.000000         0.000000           0.000000
0.000000
25%            0.000000         0.000000           0.000000
0.000000
50%            0.000000         0.000000           0.000000
```

```
0.000000
75%             0.000000          0.000000            0.000000
0.000000
max             3.000000          3.000000            3.000000
46.000000

       reactions_rocket  reactions_total_count       user_id
count      15955.000000           15955.000000  1.595500e+04
mean           0.001379               0.334879  2.634162e+07
std            0.050053               1.361916  3.477796e+07
min            0.000000               0.000000  1.510000e+02
25%            0.000000               0.000000  1.878454e+06
50%            0.000000               0.000000  1.089753e+07
75%            0.000000               0.000000  3.875948e+07
max            4.000000              46.000000  1.784310e+08

[8 rows x 27 columns]
```

Handle Missing Values

```
# Check for missing data
missing_data = df.isna().sum()

df.shape
```

```
(15955, 132)
```

```
# Handle missing data
df.dropna(subset=['created_at', 'title'], inplace=True)

df.shape
```

```
(15955, 132)
```

Note: No values is missing in essential columns

```
# Get Categorical Columns
categorical_columns = df.select_dtypes(include=['object'])
```

Time Analysis

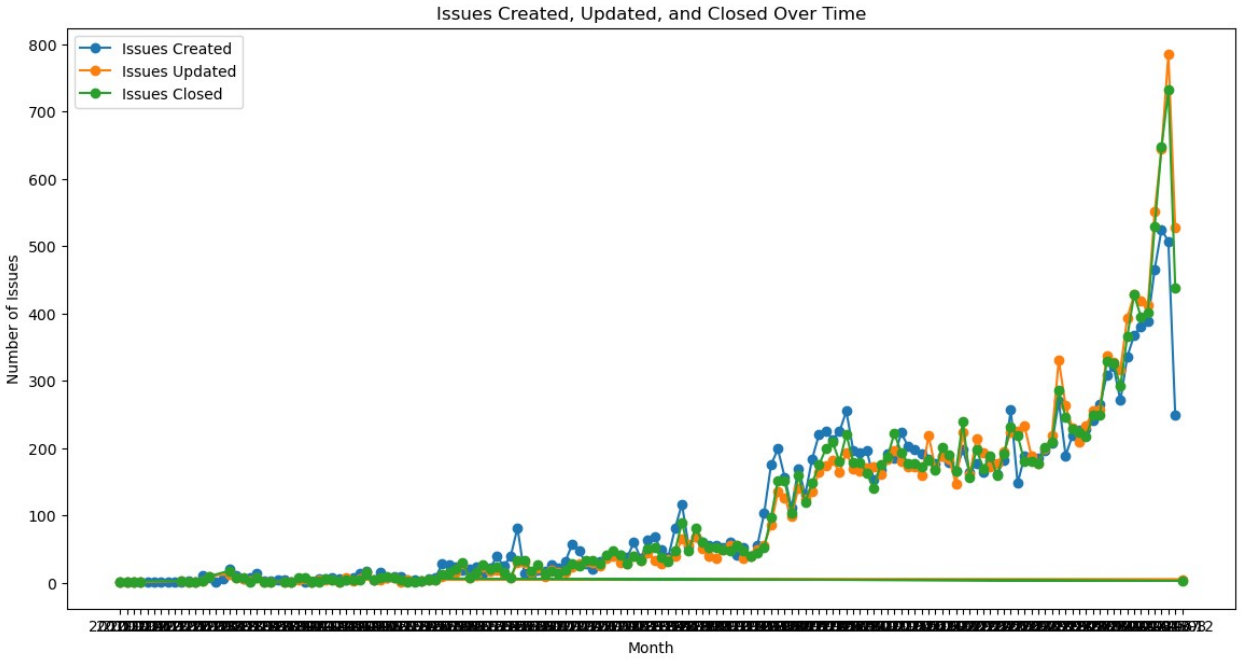
Remove Time Zone Information From Dates

```
for at_col in at_columns:
    df[at_col] = df[at_col].dt.tz_localize(None)

df[at_columns]
```

```
                closed_at          created_at          updated_at
0     2023-05-19 08:06:42 2023-05-05 15:54:28 2023-06-19 00:09:05
10218 2021-07-12 12:18:12 2020-12-06 15:00:07 2021-07-12 12:18:12
10219 2024-08-13 05:19:29 2024-08-03 20:17:41 2024-08-13 05:19:29
10220 2024-06-27 17:56:07 2024-03-17 06:02:24 2024-06-27 17:56:07
10221 2024-05-17 21:00:02 2024-05-12 11:26:23 2024-05-17 21:00:03
...                   ...                 ...                 ...
8423  2024-08-03 16:37:54 2024-07-12 12:41:22 2024-08-03 16:37:55
10661 2024-07-24 15:38:33 2024-07-11 12:13:59 2024-07-24 18:39:53
7569  2024-07-31 11:34:41 2022-02-02 22:25:59 2024-08-15 00:24:55
5401  2024-03-27 10:23:50 2022-04-21 09:23:13 2024-03-27 10:23:50
2371  2017-07-20 08:20:06 2017-06-21 04:03:35 2017-07-20 08:20:06

[15955 rows x 3 columns]
```

```
# Group by month and count the number of issues created, updated, and
closed
monthly_created =
df.groupby(df['created_at'].dt.to_period('M')).size()
monthly_updated =
df.groupby(df['updated_at'].dt.to_period('M')).size()
monthly_closed = df.groupby(df['closed_at'].dt.to_period('M')).size()

# Convert PeriodIndex to string
monthly_created.index = monthly_created.index.astype(str)
monthly_updated.index = monthly_updated.index.astype(str)
monthly_closed.index = monthly_closed.index.astype(str)

# Plotting the time series
plt.figure(figsize=(14, 7))
plt.plot(monthly_created, label='Issues Created', marker='o')
plt.plot(monthly_updated, label='Issues Updated', marker='o')
plt.plot(monthly_closed, label='Issues Closed', marker='o')
plt.title('Issues Created, Updated, and Closed Over Time')
plt.xlabel('Month')
plt.ylabel('Number of Issues')
```

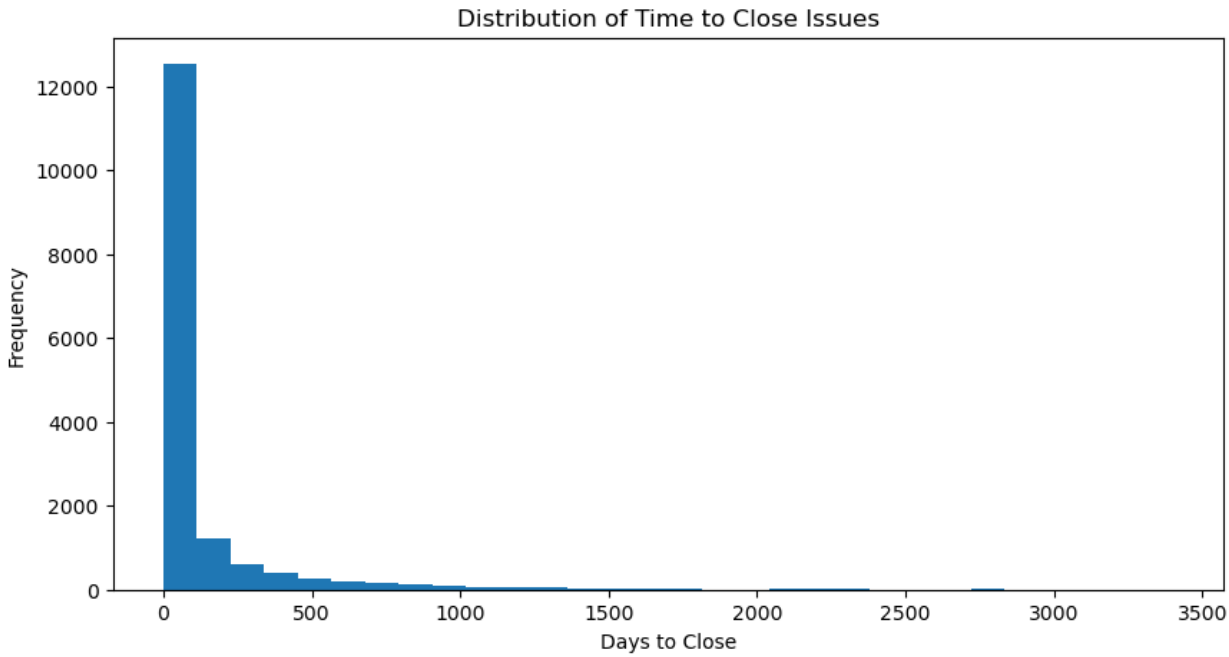
```
plt.legend()
plt.show()
```




```
# Calculate time to close
df['time_to_close'] = (df['closed_at'] - df['created_at']).dt.days

# Descriptive statistics of time to close
print(df['time_to_close'].describe())
```

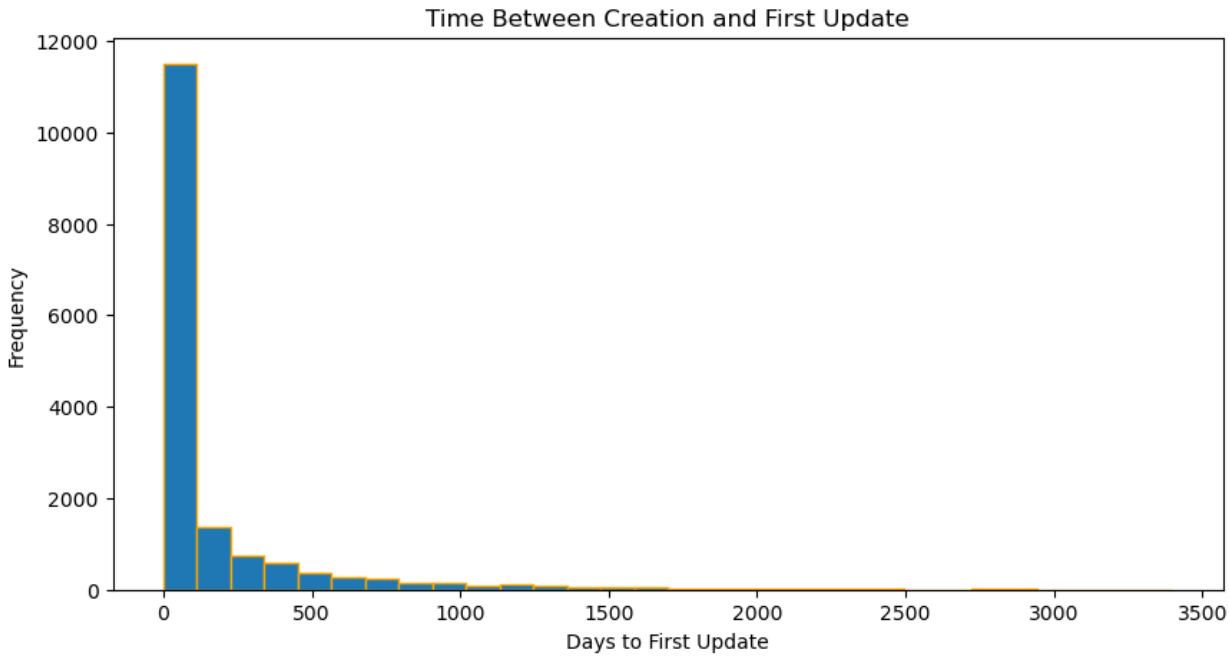
```
count    15955.000000
mean       115.006393
std        276.364167
min          0.000000
25%          1.000000
50%         12.000000
75%         81.000000
max       3399.000000
Name: time_to_close, dtype: float64
```

```
# Plotting the distribution
plt.figure(figsize=(10, 5))
df['time_to_close'].dropna().plot(kind='hist', bins=30,
title='Distribution of Time to Close Issues')
plt.xlabel('Days to Close')
plt.show()
```



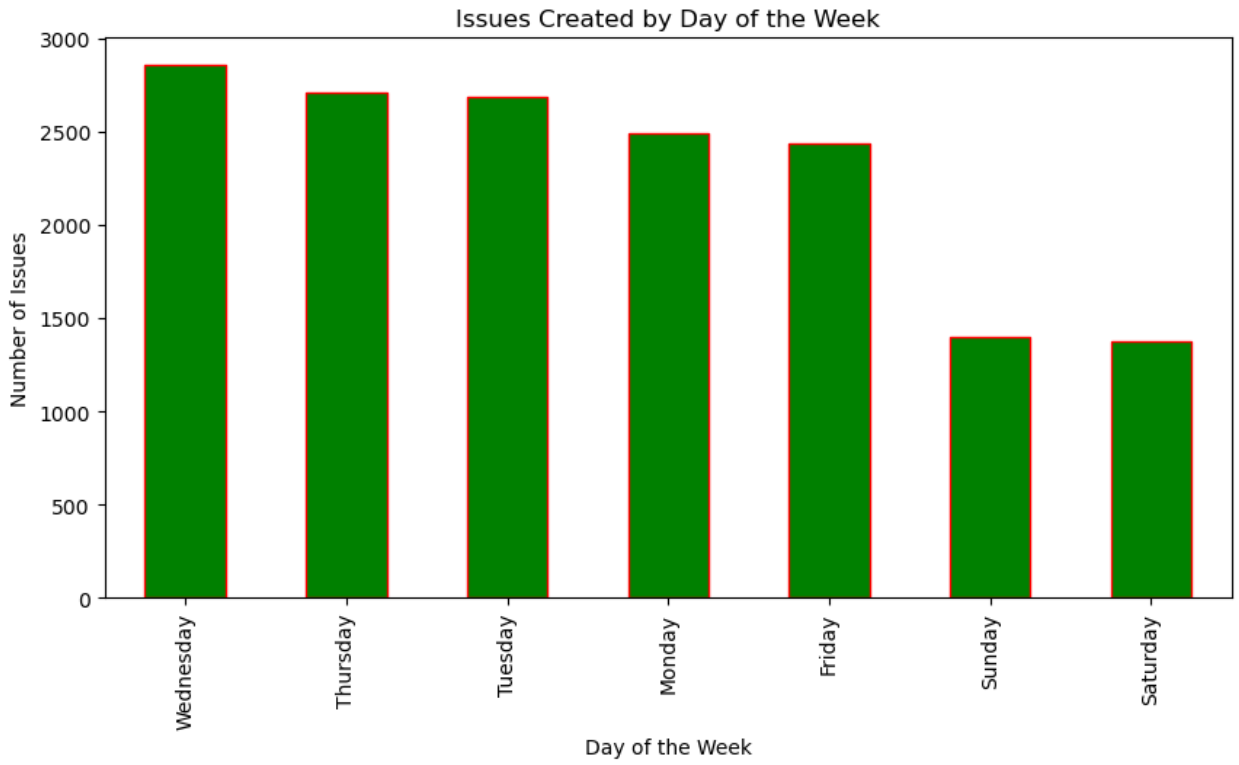
Times Between Events

```
# Calculate time between creation and first update
df['time_to_first_update'] = (df['updated_at'] -
df['created_at']).dt.days

# Descriptive statistics
print(df['time_to_first_update'].describe())

count    15955.000000
mean       160.433469
std        333.551460
min          0.000000
25%          2.000000
50%         24.000000
75%        143.000000
max       3399.000000
Name: time_to_first_update, dtype: float64

# Plotting the distribution
plt.figure(figsize=(10, 5))
df['time_to_first_update'].dropna().plot(kind='hist', bins=30,
title='Time Between Creation and First Update', edgecolor='orange')
plt.xlabel('Days to First Update')
plt.show()
```

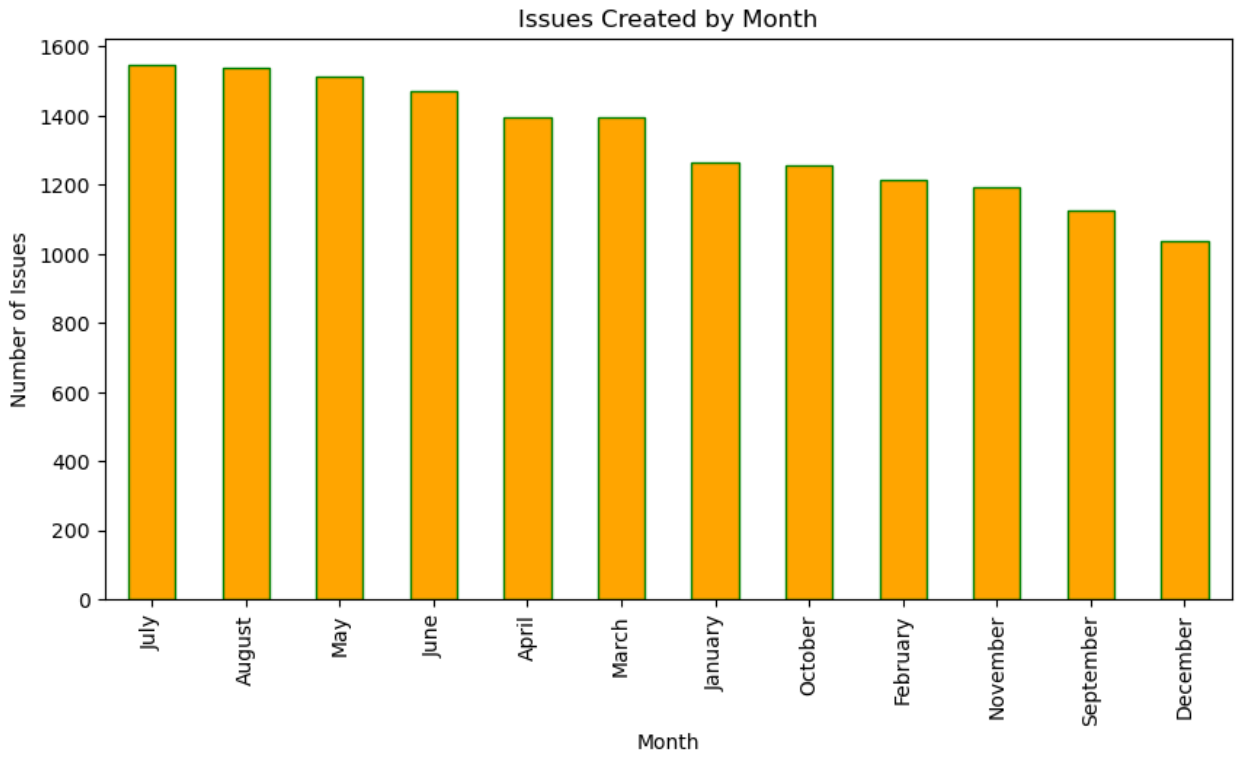


Sesanolity Analysis

```
# Analyzing the day of the week
df['day_of_week'] = df['created_at'].dt.day_name()
day_of_week_counts = df['day_of_week'].value_counts()


plt.figure(figsize=(10, 5))
day_of_week_counts.plot(kind='bar', title='Issues Created by Day of
the Week', color='green',edgecolor='red')
plt.xlabel('Day of the Week')
plt.ylabel('Number of Issues')
plt.show()
```

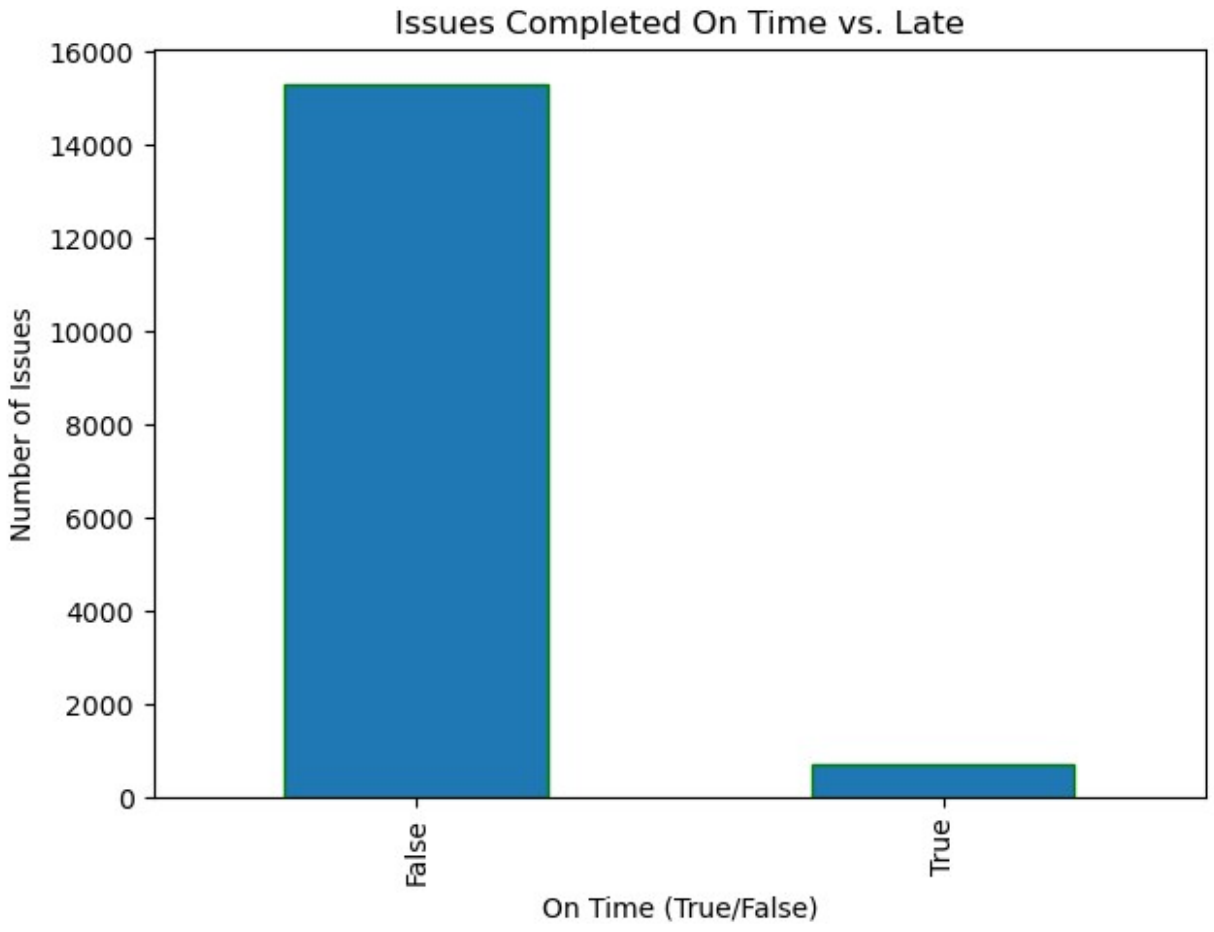



```
# Analyzing the month of the year
df['month'] = df['created_at'].dt.month_name()
month_counts = df['month'].value_counts()

plt.figure(figsize=(10, 5))
month_counts.plot(kind='bar', title='Issues Created by
Month',color='orange', edgecolor='green')
plt.xlabel('Month')
plt.ylabel('Number of Issues')
plt.show()
```



Milestone Analysis

```
df['milestone_due_on'] = pd.to_datetime(df['milestone_due_on'],
errors='coerce')

# Ensure timezone consistency
df['milestone_due_on'] = df['milestone_due_on'].dt.tz_localize(None)

# Calculate if issues were closed before the milestone due date
df['closed_before_due'] = df['closed_at'] < df['milestone_due_on']

# Count of issues completed on time vs. late
on_time_counts = df['closed_before_due'].value_counts()

plt.figure(figsize=(7, 5))
on_time_counts.plot(kind='bar', title='Issues Completed On Time vs.
Late', edgecolor='green')
plt.xlabel('On Time (True/False)')
plt.ylabel('Number of Issues')
plt.show()
```

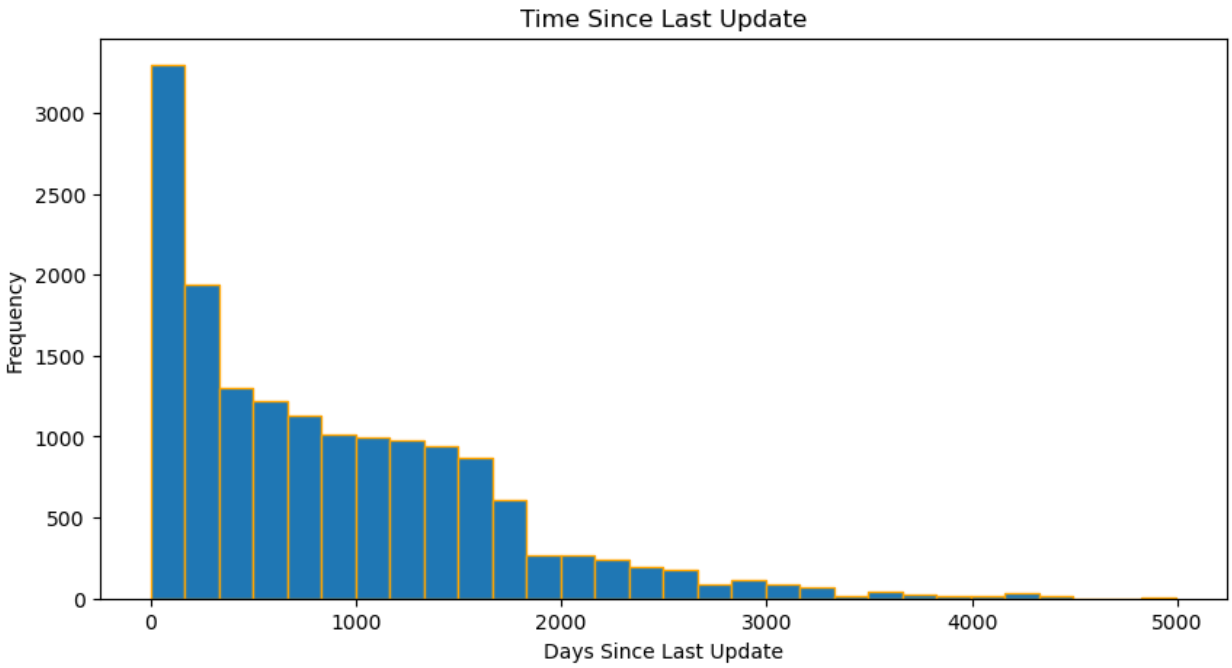


Time Since Last Activity

```
# Remove timezone information from 'updated_at'

# Calculate time since last update
df['time_since_last_update'] = (pd.Timestamp.now() -
df['updated_at']).dt.days

# Descriptive statistics
print(df['time_since_last_update'].describe())
```

```
count    15955.000000
mean       877.525917
std        784.217220
min          3.000000
25%        220.000000
50%        697.000000
75%       1352.000000
```

```
max       4991.000000
Name: time_since_last_update, dtype: float64

# Plotting the distribution
plt.figure(figsize=(10, 5))
df['time_since_last_update'].dropna().plot(kind='hist', bins=30,
title='Time Since Last Update', edgecolor='orange')
plt.xlabel('Days Since Last Update')
plt.show()
```



Reaction Columns

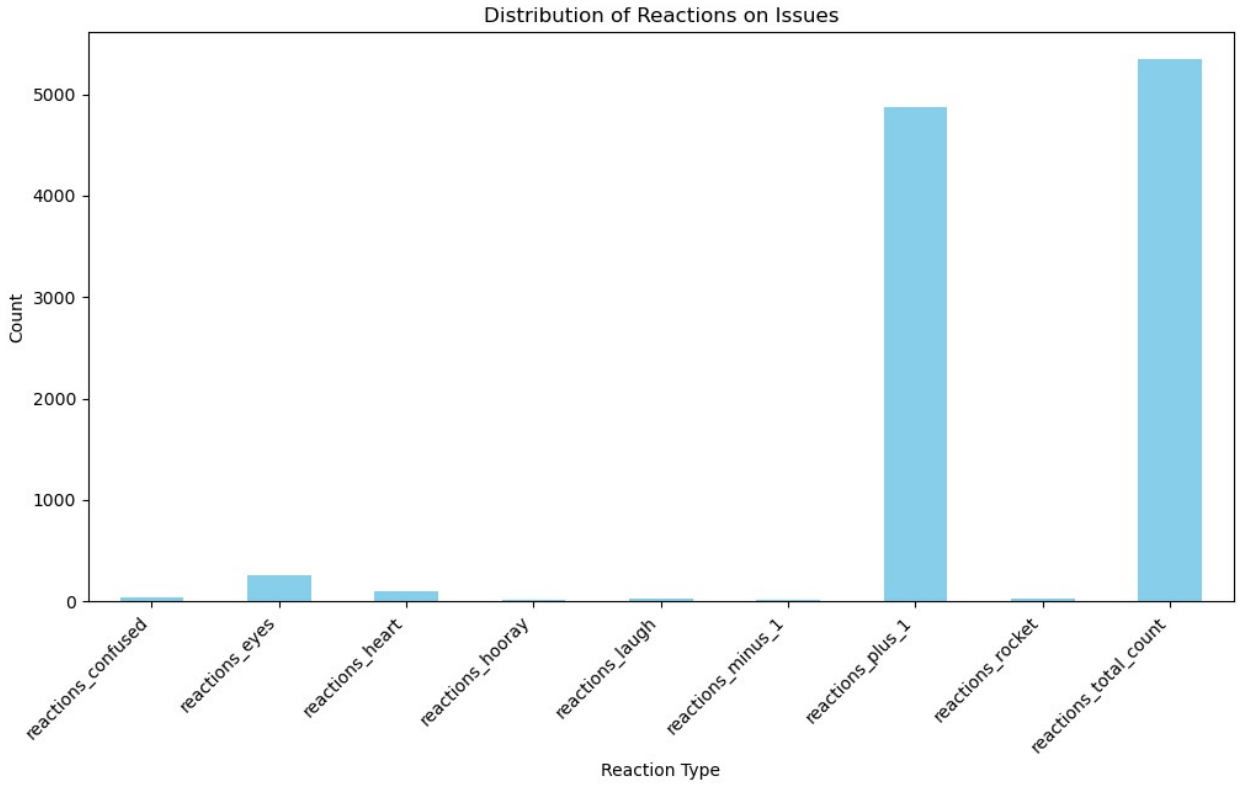
```
# Create a new DataFrame to sum reactions
reactions_summary = df[reaction_columns].sum()

reactions_summary
```

```
reactions_confused          36
reactions_eyes             258
reactions_heart            104
reactions_hooray            15
reactions_laugh             27
reactions_minus_1           12
reactions_plus_1          4869
reactions_rocket            22
```

```
reactions_total_count    5343
dtype: int64
```

Plotting the reaction distribution

```
plt.figure(figsize=(12, 6))
reactions_summary.plot(kind='bar', color='skyblue')
plt.title('Distribution of Reactions on Issues')
plt.xlabel('Reaction Type')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.show()
```

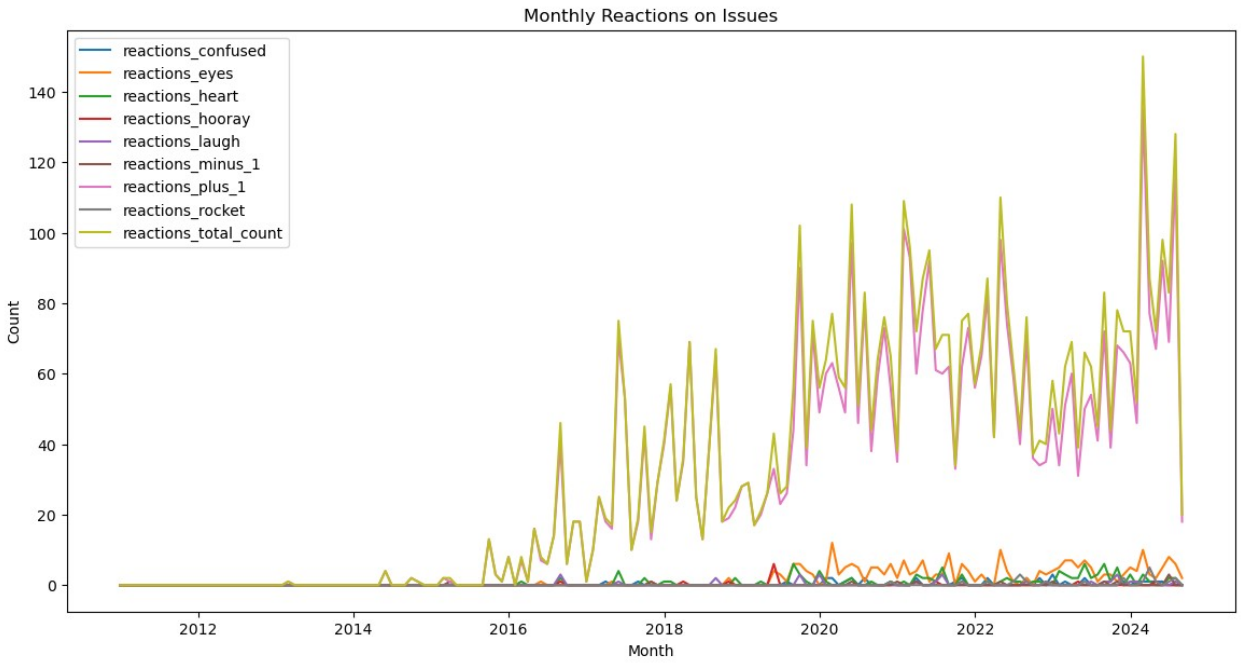


Time Series Of Reaction

Monthly Reaction

```
# Set 'created_at' as the index
df.set_index('created_at', inplace=True)
```

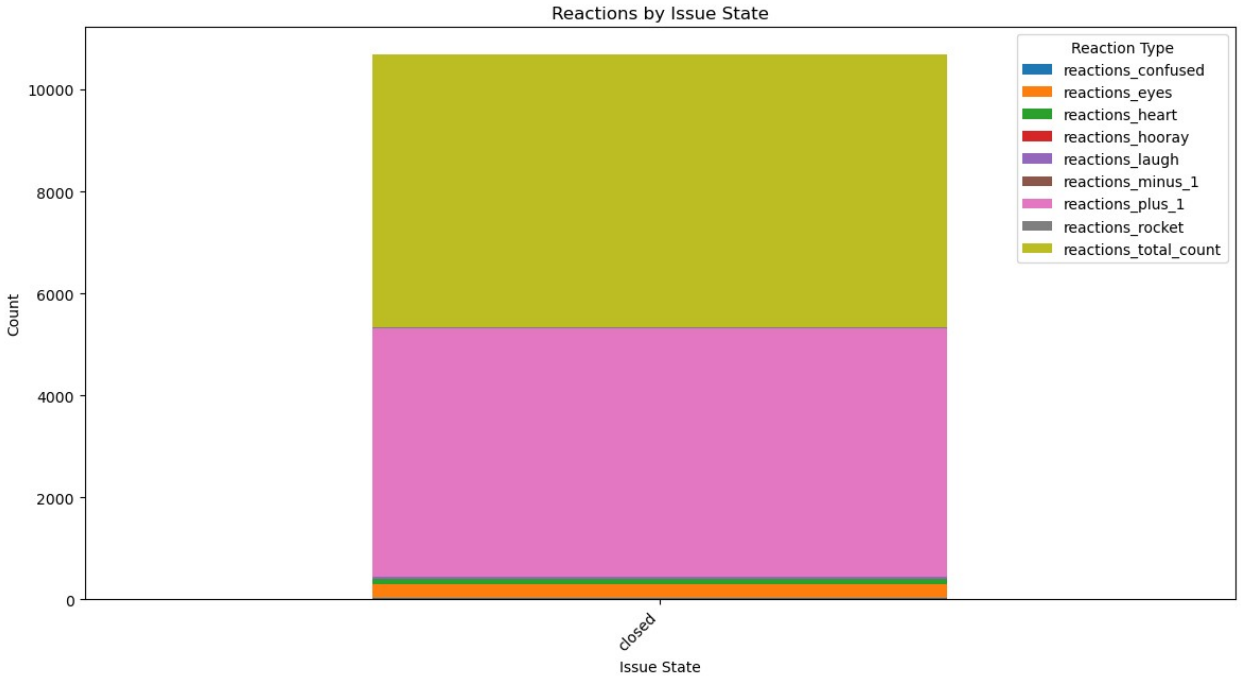
```
# Resampling by month and summing reactions
monthly_reactions = df[reaction_columns].resample('ME').sum()

# Plotting the time series of reactions
plt.figure(figsize=(14, 7))
for reaction in reaction_columns:
    plt.plot(monthly_reactions.index, monthly_reactions[reaction],
label=reaction)

plt.title('Monthly Reactions on Issues')
plt.xlabel('Month')
plt.ylabel('Count')
plt.legend()
plt.show()
```



Reaction Counts By Issues State

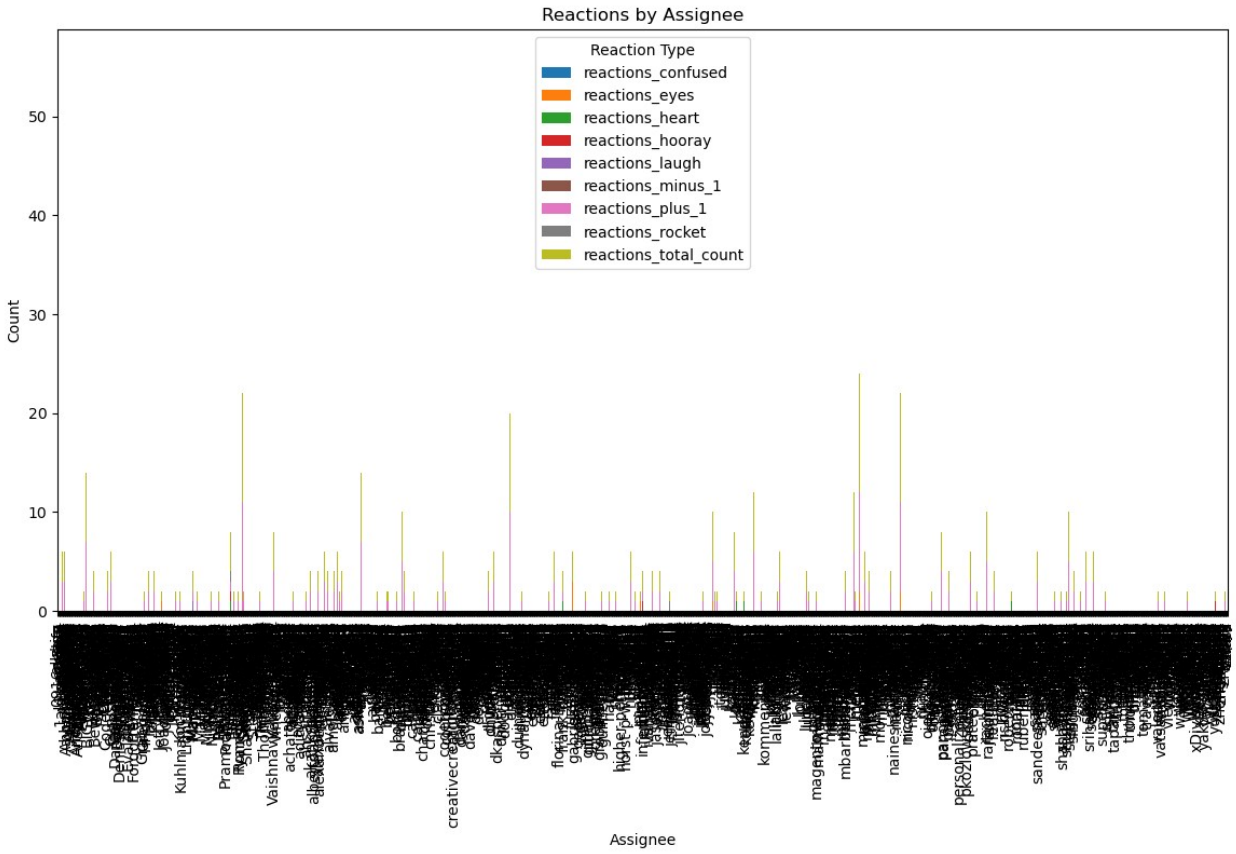
```
# Aggregating reactions by issue state
reaction_state_summary = df.groupby('state')[reaction_columns].sum()

# Plotting reactions by issue state
reaction_state_summary.plot(kind='bar', stacked=True, figsize=(14, 7))
plt.title('Reactions by Issue State')
plt.xlabel('Issue State')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
```

```
plt.legend(title='Reaction Type')
plt.show()
```



Reaction By Issue Assignee

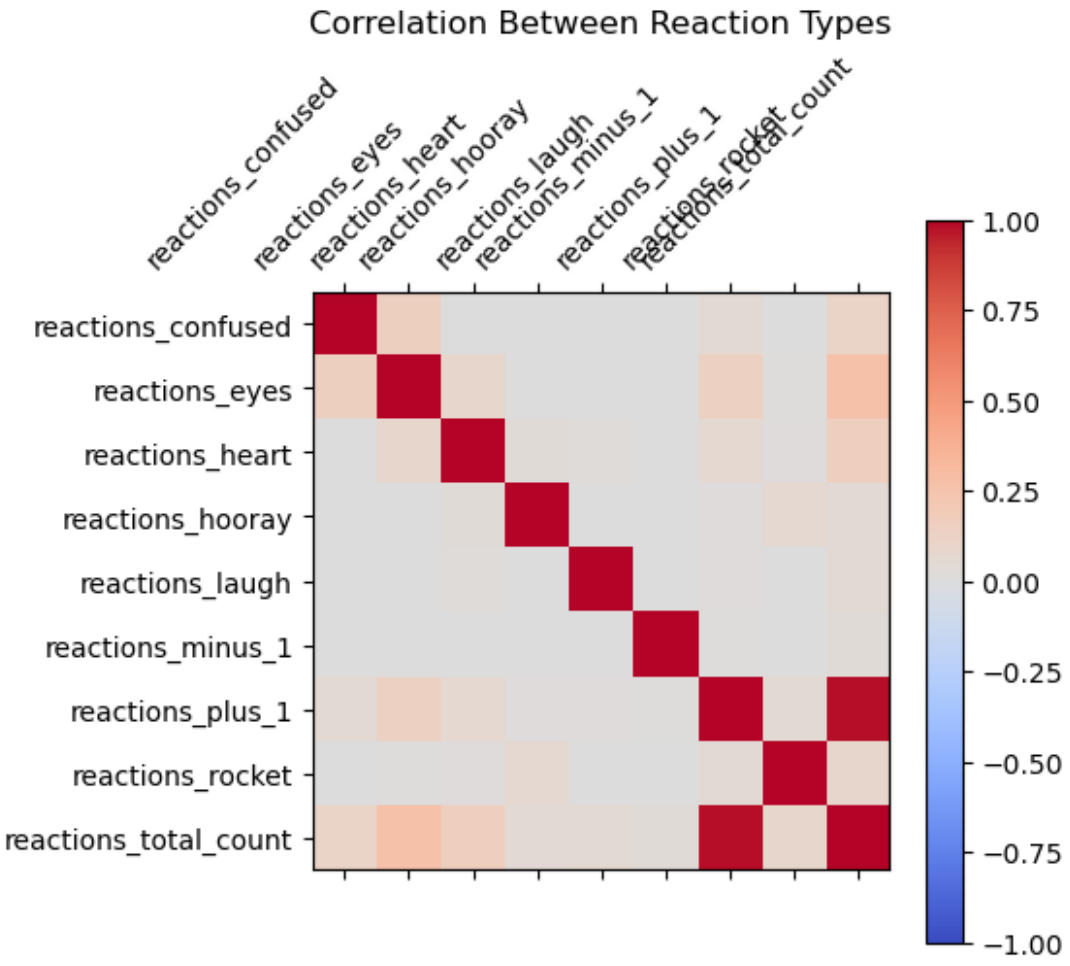
```
# Aggregating reactions by assignee
reaction_assignee_summary = df.groupby('assignee_login')
[reaction_columns].sum()

# Plotting the reactions by assignee
reaction_assignee_summary.plot(kind='bar', stacked=True, figsize=(14,
7))
plt.title('Reactions by Assignee')
plt.xlabel('Assignee')
plt.ylabel('Count')
plt.xticks(rotation=90, ha='center')
plt.legend(title='Reaction Type')
plt.show()
```



Reaction Co Relation

```
# Calculate correlation matrix
reaction_correlation = df[reaction_columns].corr()

# Plotting the correlation matrix
plt.figure(figsize=(12, 10))
cax = plt.matshow(reaction_correlation, cmap='coolwarm', vmin=-1,
vmax=1)
plt.colorbar(cax)
plt.xticks(range(len(reaction_columns)), reaction_columns,
rotation=45, ha='right')
plt.yticks(range(len(reaction_columns)), reaction_columns)
plt.title('Correlation Between Reaction Types')
plt.show()
```

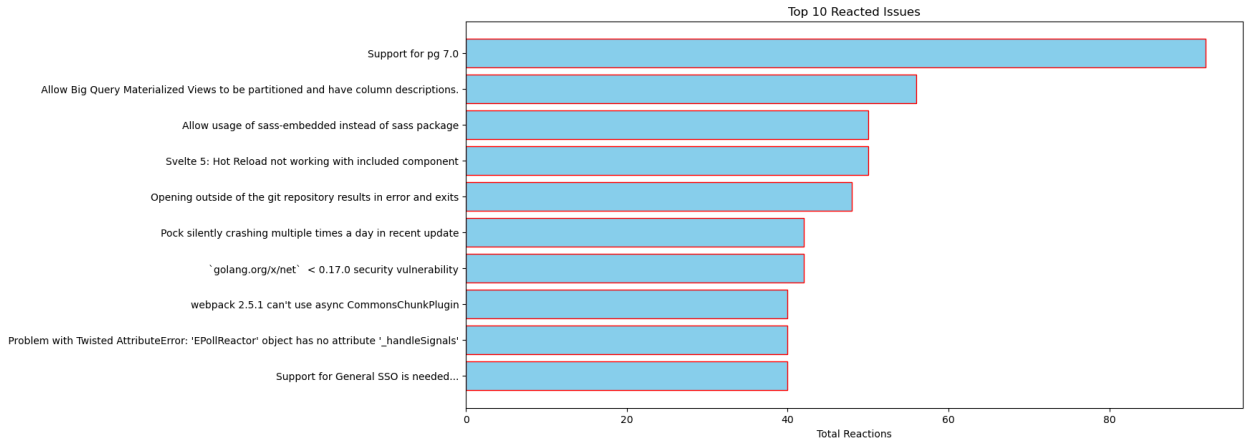
```
<Figure size 1200x1000 with 0 Axes>
```



Top Reacted Issue

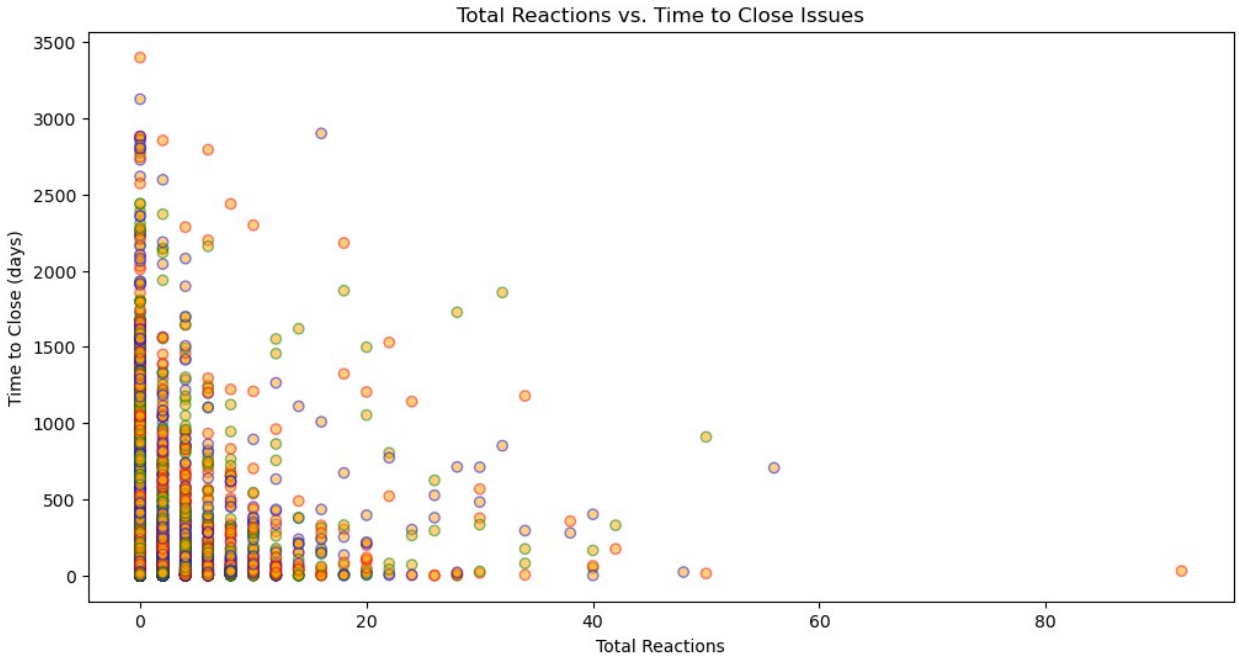
```
# Calculate total reactions per issue
df['total_reactions'] = df[reaction_columns].sum(axis=1)

# Sort by total reactions and get the top issues
top_reacted_issues = df[['title',
'total_reactions']].sort_values(by='total_reactions',
ascending=False).head(10)

# Plotting the top reacted issues
plt.figure(figsize=(14, 7))
plt.barh(top_reacted_issues['title'],
top_reacted_issues['total_reactions'], color='skyblue',
edgecolor='red')
plt.xlabel('Total Reactions')
plt.title('Top 10 Reacted Issues')
```

```
plt.gca().invert_yaxis()
plt.show()
```



Impact of reaction on issue Closure

```
# Scatter plot of total reactions vs. time to close
plt.figure(figsize=(12, 6))
plt.scatter(df['total_reactions'], df['time_to_close'], alpha=0.5,
c='orange', edgecolors=['red','green','blue'])
plt.xlabel('Total Reactions')
plt.ylabel('Time to Close (days)')
plt.title('Total Reactions vs. Time to Close Issues')
plt.show()
```


Total Reactions vs. Time to Close Issues
Time to Close (days)
Total Reactions
0
500
1000
1500
2000
2500
3000
3500
0
20
40
60
80