In [ ]:
```
# This project is about the Data of internet usage [in kb] by graduate student at an indian university.

Answer for the following questions usign Data Analysis.

What is the most frequent internet activity time of the day ?
How often the ip changes ?
How often the device changed.
What is the average usage per hour , per day and per month ?

Hint: internet_session.csv provided to students.
```

In [ ]:
```
# 1 - Importing and Cleaning the Data
```

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
FileName = 'E:\DSPP\Assignments\ML DL Assignment 1/internet_session.csv'
internet_usage = pd.read_csv(FileName, parse_dates=['start_time'])
```

In [5]:
```
internet_usage
```

Out[5]:

	name	start_time	usage_time	IP	MAC	upload	download	total_transfer	seession_break_reason
0	user1	2022-05-10 02:59:32	00:00:36:28	10.55.14.222	48:E7:DA:58:22:E9	15861.76	333168.64	349030.40	Idle-Timeout
1	user1	2022-05-10 18:53:27	00:01:49:56	10.55.2.253	48:E7:DA:58:22:E9	16957.44	212152.32	229109.76	Idle-Timeout
2	user1	2022-05-10 21:20:44	00:01:35:00	10.55.2.253	48:E7:DA:58:22:E9	14080.0	195153.92	209233.92	Idle-Timeout
3	user1	2022-05-11 00:37:42	00:00:26:00	10.55.2.253	48:E7:DA:58:22:E9	5242.88	40806.4	46049.28	Idle-Timeout
4	user1	2022-05-11 02:59:38	00:00:11:52	10.55.2.253	48:E7:DA:58:22:E9	22067.2	10772.48	32839.68	Idle-Timeout
...	...	...	...	...	...	...	...	...	...
4707	user9	2022-11-04 01:11:34	00:06:54:32	10.55.4.189	DA:2F:97:0E:B7:D0	107960.32	2390753.28	2495610.88	Idle-Timeout
4708	user9	2022-11-04 10:26:09	00:00:23:49	10.55.4.59	DA:2F:97:0E:B7:D0	11407.36	209674.24	221081.60	Idle-Timeout
4709	user9	2022-11-04 20:41:42	00:01:24:13	10.55.15.186	DA:2F:97:0E:B7:D0	18995.2	373657.6	392652.80	Idle-Timeout
4710	user9	2022-11-05 00:21:06	00:08:49:43	10.55.4.159	DA:2F:97:0E:B7:D0	46602.24	593766.4	640368.64	Idle-Timeout
4711	user9	2022-11-05 20:55:37	00:01:06:20	10.55.2.33	DA:2F:97:0E:B7:D0	21237.76	298536.96	319774.72	NaN

4712 rows × 9 columns

In [3]:
```
internet_usage.shape
```

Out[3]:
```
(4712, 9)
```

In [6]:
```
internet_usage.columns
```

Out[6]:
```
Index(['name', 'start_time', 'usage_time', 'IP', 'MAC', 'upload', 'download',
       'total_transfer', 'seession_break_reason'],
      dtype='object')
```

In [61]:
```
# Let's make the names of the columns in lower case so as to make them easier to work with
```

In [7]:
```
internet_usage.columns = internet_usage.columns.str.lower()
internet_usage.columns
```

Out[7]:
```
Index(['name', 'start_time', 'usage_time', 'ip', 'mac', 'upload', 'download',
       'total_transfer', 'seession_break_reason'],
      dtype='object')
```

In [ ]:
```
# Then we weill check the data type of the columns of dataset
```

In [8]:
```
internet_usage.dtypes
```

Out[8]:
```
name                          object
start_time              datetime64[ns]
usage_time                    object
ip                            object
mac                           object
upload                        object
download                      object
total_transfer               float64
seession_break_reason         object
dtype: object
```

In [ ]:
```
# We need to fix the wrong data type for those columns  and then we will check the null values and drop them if necessary
```

In [9]:
```
internet_usage.isna().sum()
```

Out[9]:
```
name                    0
start_time              0
usage_time              0
ip                      0
mac                     0
upload                  0
download                0
total_transfer          0
```

```
seession_break_reason    9
dtype: int64
```

In [ ]:
```
# Column "seession_break_reason" has some null values since they are very less, we can delete them as it won't affect the analysis
```

In [10]:
```
internet_usage = internet_usage.dropna().copy()
internet_usage.isna().sum()
```

Out[10]:
```
name                    0
start_time              0
usage_time              0
ip                      0
mac                     0
upload                  0
download                0
total_transfer          0
seession_break_reason   0
dtype: int64
```

In [ ]:
```
# Then check if the dataset contains duplicates, if any then drop them
```

In [11]:
```
internet_usage.duplicated().sum()
```

Out[11]:
```
0
```

In [ ]:
```
# Since there No duplicates, so now we will start converting the data types of columns from strings to numeric
```

In [12]:
```
internet_usage['usage_time'] = internet_usage['usage_time'].str.replace('00:', '', 1)
internet_usage['usage_time'] = pd.to_datetime(internet_usage['usage_time'])

internet_usage['upload'] = internet_usage['upload'].str.extract('(\d+)', expand=False)
internet_usage.upload = internet_usage.upload.astype(float)

internet_usage['download'] = internet_usage['download'].str.extract('(\d+)', expand=False)
internet_usage.download = internet_usage.download.astype(float)

device = []
basename = 'device'
mac = internet_usage['mac'][0]
device_number = 1
for i in internet_usage['mac']:
    if i == mac:
        device.append(basename + str(device_number))
    else:
        device_number += 1
        device.append(basename + str(device_number))
        mac = i
```

```
internet_usage['device'] = device

internet_usage.dtypes
```

Out[12]:
```
name                          object
start_time              datetime64[ns]
usage_time              datetime64[ns]
ip                              object
mac                             object
upload                         float64
download                       float64
total_transfer                 float64
seession_break_reason           object
device                          object
dtype: object
```

In [ ]:
```
# Now since all the 9 columns are in the right data types, so we can proceed to the exploratory data analysis
```

In [ ]:
```
# 2 - Exploratory Data Analysis
```

In [13]:
```
internet_usage.describe(include='all', datetime_is_numeric=True)
```

Out[13]:

	name	start_time	usage_time	ip	mac	upload	download	total_transfer	seession_break_reason	devi
count	4703	4703	4703	4703	4703	4.703000e+03	4.703000e+03	4.703000e+03	4703	47
unique	9	NaN	NaN	1299	33	NaN	NaN	NaN	5	12
top	user4	NaN	NaN	10.55.0.89	48:E7:DA:58:22:E9	NaN	NaN	NaN	Idle-Timeout	device12
freq	725	NaN	NaN	80	1235	NaN	NaN	NaN	4350	1
mean	NaN	2022-08-08 09:35:44.875184896	2023-02-01 02:10:05.038485760	NaN	NaN	3.378702e+04	3.966645e+05	4.304372e+05	NaN	N
min	NaN	2022-05-09 22:52:41	2023-02-01 00:00:01	NaN	NaN	2.000000e+00	9.000000e+00	1.120000e+00	NaN	N
25%	NaN	2022-06-14 18:33:06.500000	2023-02-01 00:31:42	NaN	NaN	6.082000e+03	5.199800e+04	6.187008e+04	NaN	N
50%	NaN	2022-08-19 13:56:28	2023-02-01 01:19:40	NaN	NaN	1.531900e+04	1.782680e+05	2.027930e+05	NaN	N
75%	NaN	2022-09-24 22:30:58.500000	2023-02-01 02:49:02	NaN	NaN	3.399600e+04	4.593660e+05	4.993997e+05	NaN	N
max	NaN	2022-11-05 18:41:14	2023-02-01 22:00:07	NaN	NaN	2.841640e+06	2.790261e+07	2.855272e+07	NaN	N

	name	start_time	usage_time	ip	mac	upload	download	total_transfer	seession_break_reason	devi
std	NaN	NaN	NaN	NaN	NaN	9.493243e+04	9.657778e+05	9.960848e+05	NaN	N

In [ ]:
```
# Now we start calculating some descriptive statistics, like how many users are there  and what is their Record counts
```

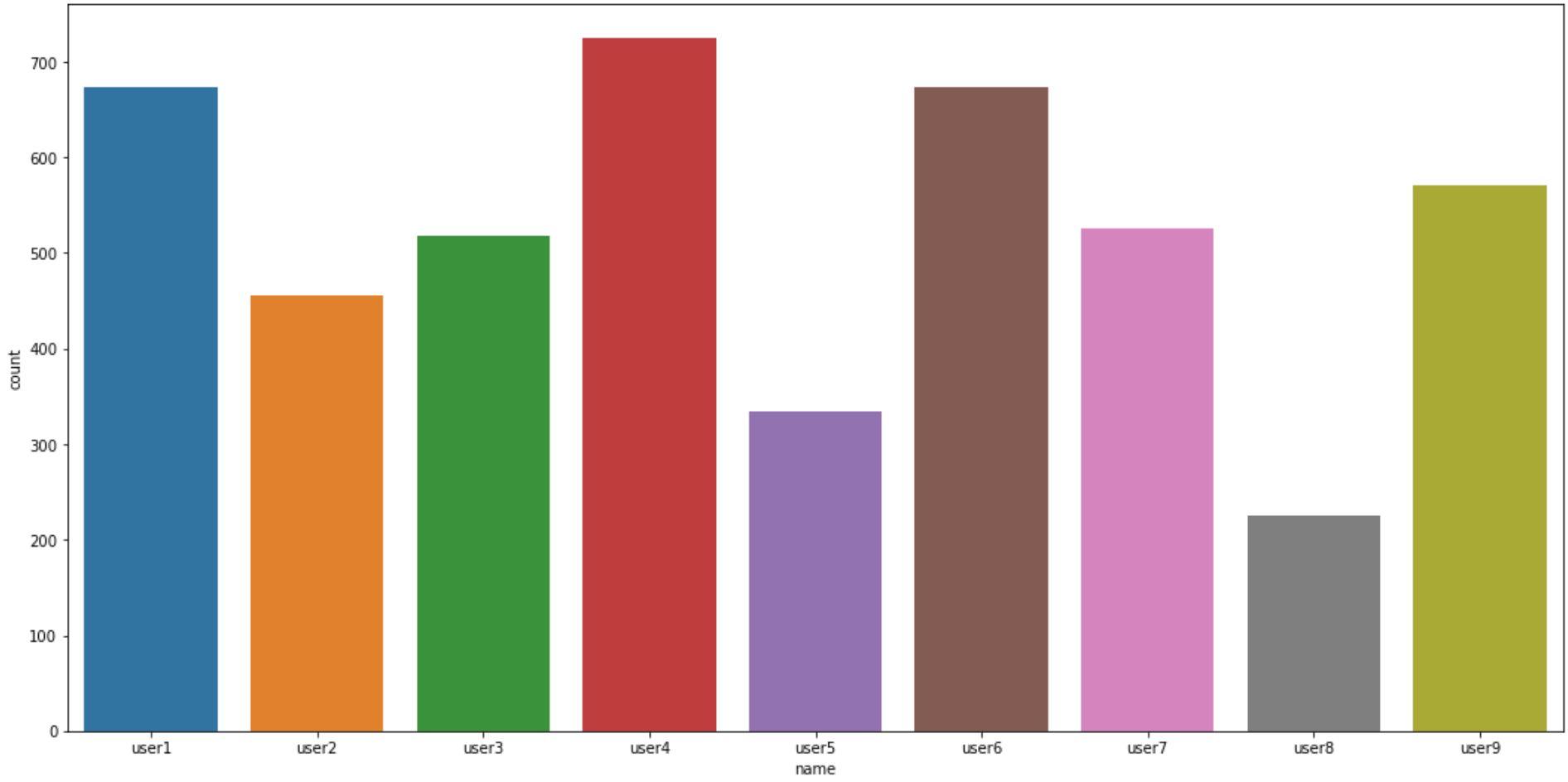
In [14]:
```
internet_usage.name.value_counts()
```

Out[14]:
```
user4    725
user6    674
user1    673
user9    571
user7    526
user3    518
user2    456
user5    335
user8    225
Name: name, dtype: int64
```

In [ ]:
```
# User4 has the most records with count of 725, while user8 has the least with a record count of 225
```

In [15]:
```
plt.figure(figsize=(18, 9))
ax = sns.countplot(x='name', data=internet_usage)
ax.bar_label(ax.containers[0])
plt.title("User Count")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-15-bce6e8c2fcb8> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.countplot(x='name', data=internet_usage)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("User Count")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [16]:
```
print('The earliest start time is:')
print(internet_usage.start_time.min())
print('The latest start time is:')
print(internet_usage.start_time.max())
```

```
The earliest start time is:
2022-05-09 22:52:41
The latest start time is:
2022-11-05 18:41:14
```

In [17]:
```
print('The minimum usage time is:')
print(internet_usage.usage_time.min())
print('The maximum usage time is:')
print(internet_usage.usage_time.max())
print('The average usage time is:')
print(internet_usage.usage_time.mean())
```

```
The minimum usage time is:
2023-02-01 00:00:01
The maximum usage time is:
2023-02-01 22:00:07
The average usage time is:
2023-02-01 02:10:05.038485760
```

In [16]:
```
print('The minimum usage time per user:')
usage_time_minimum = internet_usage.groupby('name').usage_time.min()
usage_time_minimum
```

```
The minimum usage time per user:
```

Out[16]:
```
name
user1   2023-01-07 00:00:18
user2   2023-01-07 00:00:08
user3   2023-01-07 00:00:01
user4   2023-01-07 00:00:45
user5   2023-01-07 00:01:07
user6   2023-01-07 00:00:18
user7   2023-01-07 00:00:20
user8   2023-01-07 00:00:20
user9   2023-01-07 00:00:09
Name: usage_time, dtype: datetime64[ns]
```

In [18]:
```
plt.figure(figsize=(18, 9))
usage_time_minimum.plot(kind='bar', logy=True)
plt.title("Minimum usage time per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-18-3b07873a052c> in <module>
      1 plt.figure(figsize=(18, 9))
----> 2 usage_time_minimum.plot(kind='bar', logy=True)
      3 plt.title("Minimum usage time per user")
      4 plt.show()
      5 plt.clf()

NameError: name 'usage_time_minimum' is not defined
<Figure size 1296x648 with 0 Axes>
```

In [ ]:
```
# User3 has the least amount of minimum usage time of 1 second, while user5 has the highest time of 1 minute and 7 seconds.
```

In [19]:
```
print('The maximum usage time per user:')
usage_time_maximum = internet_usage.groupby('name').usage_time.max()
usage_time_maximum
```

```
The maximum usage time per user:
```

Out[19]:
```
name
user1   2023-02-01 19:35:11
user2   2023-02-01 20:39:52
user3   2023-02-01 17:01:28
user4   2023-02-01 18:11:43
user5   2023-02-01 06:36:11
user6   2023-02-01 19:35:11
user7   2023-02-01 22:00:07
user8   2023-02-01 17:24:26
user9   2023-02-01 19:26:09
Name: usage_time, dtype: datetime64[ns]
```

In [20]:
```
plt.figure(figsize=(18, 9))
usage_time_maximum.plot(kind='bar', logy=True)
plt.title("The maximum usage time per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
OverflowError                             Traceback (most recent call last)
<ipython-input-20-644d668e7b55> in <module>
      2 usage_time_maximum.plot(kind='bar', logy=True)
      3 plt.title("The maximum usage time per user")
----> 4 plt.show()
      5 plt.clf()

~\Anaconda3\lib\site-packages\matplotlib\pyplot.py in show(*args, **kwargs)
    351     """
    352     _warn_if_gui_out_of_main_thread()
--> 353     return _backend_mod.show(*args, **kwargs)
    354
    355

~\Anaconda3\lib\site-packages\ipykernel\pylab\backend_inline.py in show(close, block)
     41             display(
     42                 figure_manager.canvas.figure,
---> 43                 metadata=_fetch_figure_metadata(figure_manager.canvas.figure)
     44             )
     45     finally:

~\Anaconda3\lib\site-packages\ipykernel\pylab\backend_inline.py in _fetch_figure_metadata(fig)
    178     if _is_transparent(fig.get_facecolor()):
    179         # the background is transparent
--> 180         ticksLight = _is_light([label.get_color()
    181                                 for axes in fig.axes
    182                                 for axis in (axes.xaxis, axes.yaxis)

~\Anaconda3\lib\site-packages\ipykernel\pylab\backend_inline.py in <listcomp>(.0)
    181                                 for axes in fig.axes
    182                                 for axis in (axes.xaxis, axes.yaxis)
```

```
--> 183                                 for label in axis.get_ticklabels()])
    184         if ticksLight.size and (ticksLight == ticksLight[0]).all():
    185             # there are one or more tick labels, all with the same lightness

~\Anaconda3\lib\site-packages\matplotlib\axis.py in get_ticklabels(self, minor, which)
   1253         if minor:
   1254             return self.get_minorticklabels()
-> 1255         return self.get_majorticklabels()
   1256
   1257     def get_majorticklines(self):

~\Anaconda3\lib\site-packages\matplotlib\axis.py in get_majorticklabels(self)
   1205     def get_majorticklabels(self):
   1206         """Return this Axis' major tick labels, as a list of `~.text.Text`."""
-> 1207         ticks = self.get_major_ticks()
   1208         labels1 = [tick.label1 for tick in ticks if tick.label1.get_visible()]
   1209         labels2 = [tick.label2 for tick in ticks if tick.label2.get_visible()]

~\Anaconda3\lib\site-packages\matplotlib\axis.py in get_major_ticks(self, numticks)
   1376         r"""Return the list of major `.Tick`\s."""
   1377         if numticks is None:
-> 1378             numticks = len(self.get_majorticklocs())
   1379
   1380         while len(self.majorTicks) < numticks:

~\Anaconda3\lib\site-packages\matplotlib\axis.py in get_majorticklocs(self)
   1281     def get_majorticklocs(self):
   1282         """Return this Axis' major tick locations in data coordinates."""
-> 1283         return self.major.locator()
   1284
   1285     def get_minorticklocs(self):

~\Anaconda3\lib\site-packages\matplotlib\dates.py in __call__(self)
   1333     def __call__(self):
   1334         # docstring inherited
-> 1335         dmin, dmax = self.viewlim_to_dt()
   1336         locator = self.get_locator(dmin, dmax)
   1337         return locator()

~\Anaconda3\lib\site-packages\matplotlib\dates.py in viewlim_to_dt(self)
   1088         if vmin > vmax:
   1089             vmin, vmax = vmax, vmin
-> 1090         return num2date(vmin, self.tz), num2date(vmax, self.tz)
   1091
   1092     def _get_unit(self):

~\Anaconda3\lib\site-packages\matplotlib\dates.py in num2date(x, tz)
    519     if tz is None:
    520         tz = _get_rc_timezone()
--> 521     return _from_ordinalf_np_vectorized(x, tz).tolist()
    522
```

```
    523

~\Anaconda3\lib\site-packages\numpy\lib\function_base.py in __call__(self, *args, **kwargs)
   2106             vargs.extend([kwargs[_n] for _n in names])
   2107
-> 2108         return self._vectorize_call(func=func, args=vargs)
   2109
   2110     def _get_ufunc_and_otypes(self, func, args):

~\Anaconda3\lib\site-packages\numpy\lib\function_base.py in _vectorize_call(self, func, args)
   2190                         for a in args]
   2191
-> 2192             outputs = ufunc(*inputs)
   2193
   2194             if ufunc.nout == 1:

~\Anaconda3\lib\site-packages\matplotlib\dates.py in _from_ordinalf(x, tz)
    348
    349     dt = (np.datetime64(get_epoch()) +
--> 350           np.timedelta64(int(np.round(x * MUSECONDS_PER_DAY)), 'us'))
    351     if dt < np.datetime64('0001-01-01') or dt >= np.datetime64('10000-01-01'):
    352         raise ValueError(f'Date ordinal {x} converts to {dt} (using '

OverflowError: int too big to convert
```

In [21]:

```
print('The average usage time per user:')
usage_time_average = internet_usage.groupby('name').usage_time.mean()
usage_time_average
```

```
The average usage time per user:
---------------------------------------------------------------------------
DataError                                 Traceback (most recent call last)
<ipython-input-21-512ffca6f479> in <module>
      1 print('The average usage time per user:')
----> 2 usage_time_average = internet_usage.groupby('name').usage_time.mean()
      3 usage_time_average

~\Anaconda3\lib\site-packages\pandas\core\groupby\groupby.py in mean(self, numeric_only)
   1391         Name: B, dtype: float64
   1392         """
-> 1393         return self._cython_agg_general(
   1394             "mean",
   1395             alt=lambda x, axis: Series(x).mean(numeric_only=numeric_only),

~\Anaconda3\lib\site-packages\pandas\core\groupby\groupby.py in _cython_agg_general(self, how, alt, numeric_only, min_count)
   1049
   1050         if len(output) == 0:
-> 1051             raise DataError("No numeric types to aggregate")
   1052
   1053         return self._wrap_aggregated_output(output, index=self.grouper.result_index)
```

```
DataError: No numeric types to aggregate
```

In [ ]:
```
# user7 has the highest maximum time usage of 22 hours and 7 seconds, while user5 has the least time usage of 6 hours 36 minutes and
```

In [62]:
```
print('The average usage time per user:')
usage_time_average = internet_usage.groupby('name').usage_time.mean()
usage_time_average
```

```
The average usage time per user:
---------------------------------------------------------------------------
DataError                                 Traceback (most recent call last)
<ipython-input-62-512ffca6f479> in <module>
      1 print('The average usage time per user:')
----> 2 usage_time_average = internet_usage.groupby('name').usage_time.mean()
      3 usage_time_average

~\Anaconda3\lib\site-packages\pandas\core\groupby\groupby.py in mean(self, numeric_only)
   1391         Name: B, dtype: float64
   1392         """
-> 1393         return self._cython_agg_general(
   1394             "mean",
   1395             alt=lambda x, axis: Series(x).mean(numeric_only=numeric_only),

~\Anaconda3\lib\site-packages\pandas\core\groupby\groupby.py in _cython_agg_general(self, how, alt, numeric_only, min_count)
   1049
   1050         if len(output) == 0:
-> 1051             raise DataError("No numeric types to aggregate")
   1052
   1053         return self._wrap_aggregated_output(output, index=self.grouper.result_index)

DataError: No numeric types to aggregate
```

In [22]:
```
plt.figure(figsize=(18, 9))
usage_time_average.plot(kind='bar', logy=True)
plt.title("The average usage time per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-22-f91b1ff533cd> in <module>
      1 plt.figure(figsize=(18, 9))
----> 2 usage_time_average.plot(kind='bar', logy=True)
      3 plt.title("The average usage time per user")
      4 plt.show()
      5 plt.clf()

NameError: name 'usage_time_average' is not defined
<Figure size 1296x648 with 0 Axes>
```

In [ ]:
```
# We can see that on an average, User8 has the most usage with 4 hours 3 minutes and 14 seconds,
# while User5 has the least amount of average time usage with one hour 20 minutes and 11 seconds
```

In [24]:
```
internet_usage.ip.value_counts()
```

Out[24]:
```
10.55.0.89      80
10.55.14.148    64
10.55.15.221    55
10.55.1.50      48
10.55.0.248     44
                ..
10.55.11.49      1
10.55.15.44      1
10.55.8.19       1
10.55.6.95       1
10.55.6.209      1
Name: ip, Length: 1299, dtype: int64
```

In [ ]:
```
# The most used IP Adress is 10:55:0:89
```

In [25]:
```
internet_usage.device.value_counts()
```

Out[25]:
```
device1206    194
device835     137
device11      137
device1212    132
device312     113
             ...
device641       1
device1137      1
device866       1
device639       1
device1082      1
Name: device, Length: 1224, dtype: int64
```

In [ ]:
```
# The most used device is device1206 with 194 times
```

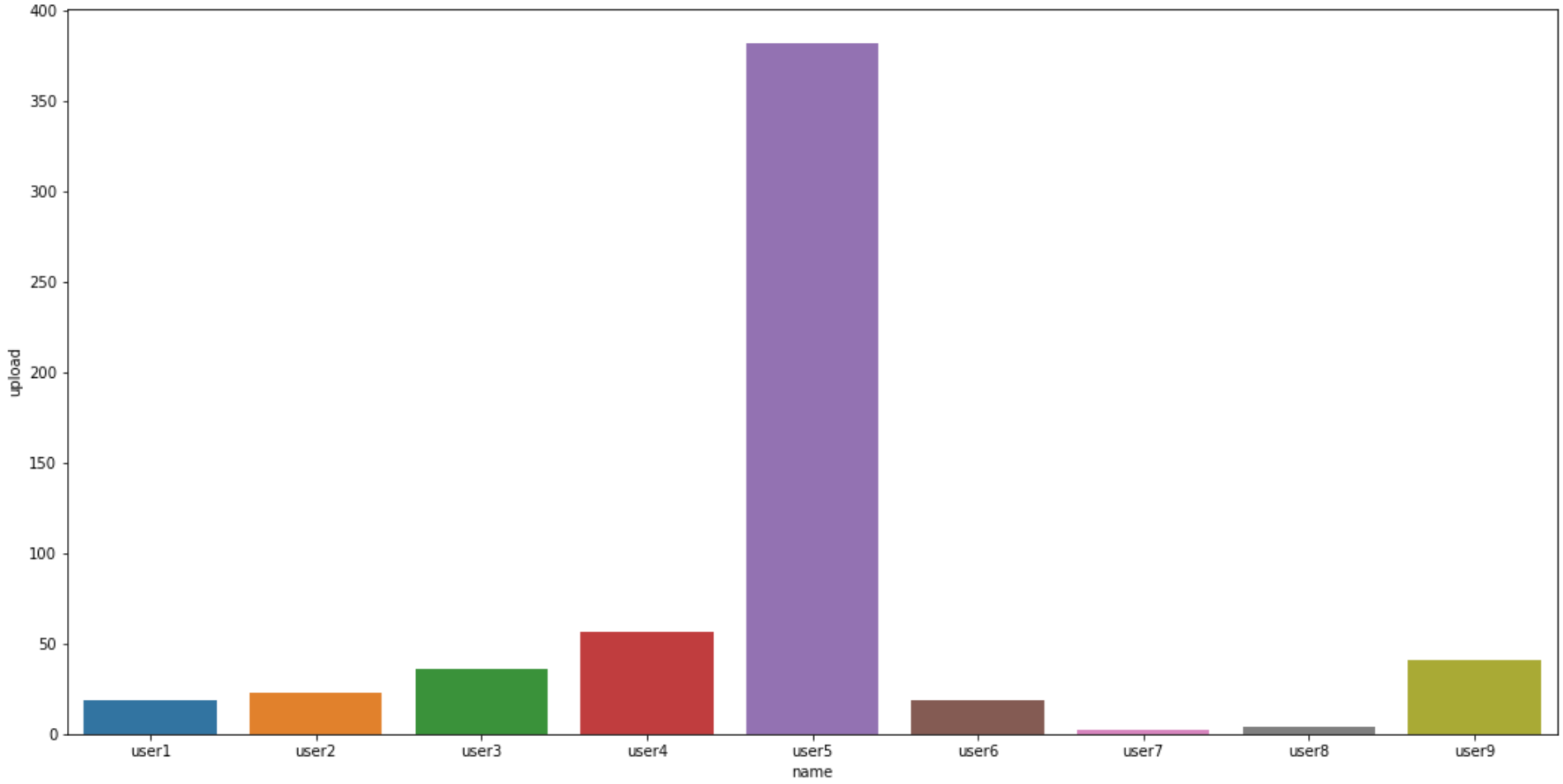
In [26]:
```
print('The minimum upload per user:')
internet_usage.groupby('name').upload.min()
```

```
The minimum upload per user:
```

Out[26]:
```
name
user1     19.0
user2     23.0
user3     36.0
user4     56.0
user5    382.0
user6     19.0
user7      2.0
user8      4.0
user9     41.0
Name: upload, dtype: float64
```

In [27]:
```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='upload' , data=internet_usage, ci=None, estimator=np.min)
ax.bar_label(ax.containers[0])
plt.title("Minimum upload per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-27-4d69f97fd0b6> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='upload' , data=internet_usage, ci=None, estimator=np.min)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Minimum upload per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [28]:
```
# user 2 has the lowest minimum upload with 2Kb while user5 has the highest minimum upload with 382Kb
```

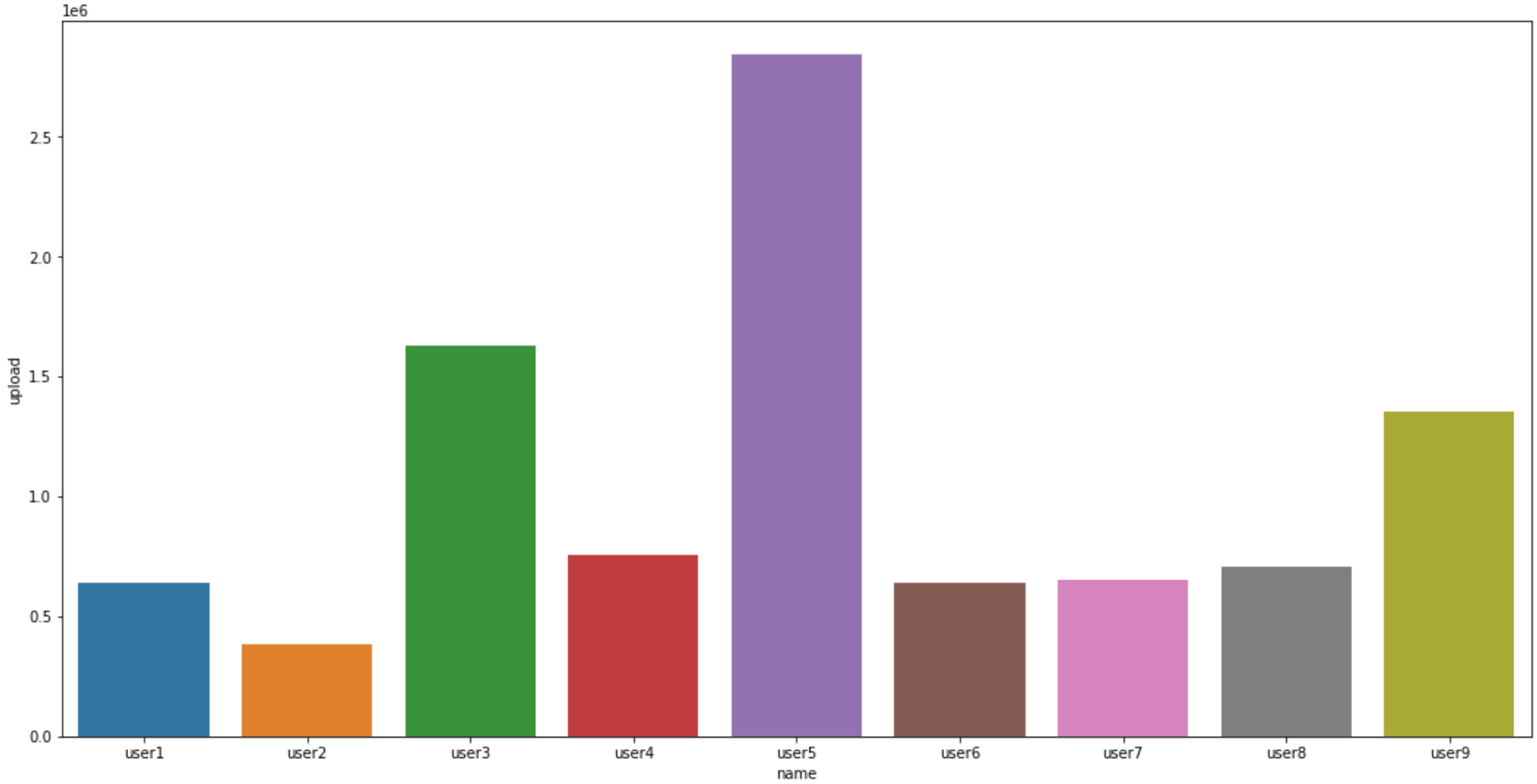
In [29]:
```
print('The maximum upload per user:')
internet_usage.groupby('name').upload.max()
```

```
The maximum upload per user:
```

Out[29]:
```
name
user1     638566.0
user2     379955.0
user3    1625292.0
user4     754462.0
user5    2841640.0
user6     638566.0
user7     653731.0
user8     709058.0
user9    1352663.0
Name: upload, dtype: float64
```

In [30]:
```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='upload' , data=internet_usage, ci=None, estimator=np.max)
ax.bar_label(ax.containers[0])
plt.title("Maximum upload per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-30-21f95715dd26> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='upload' , data=internet_usage, ci=None, estimator=np.max)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Maximum upload per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
# User 5 has the highest maximum upload with 2841640Kb with user 2 having the lowest of maximum upload with 379955Kb
```

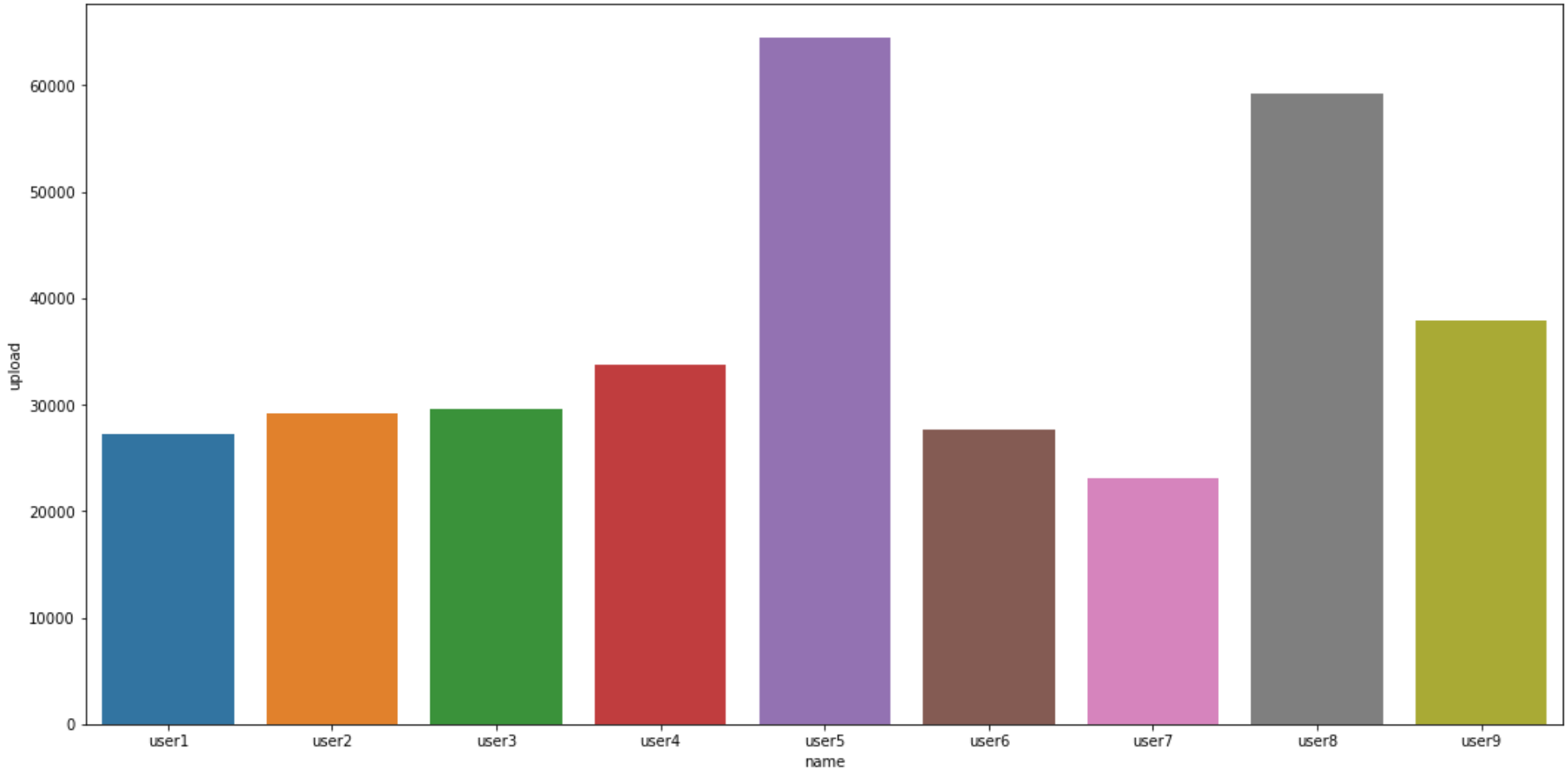
In [31]:
```
print('The average upload per user:')
round(internet_usage.groupby('name').upload.mean(), 2)
```

```
The average upload per user:
```

Out[31]:
```
name
user1    27291.34
user2    29188.79
user3    29594.88
user4    33783.74
user5    64500.35
user6    27700.73
user7    23075.54
user8    59190.12
user9    37944.66
Name: upload, dtype: float64
```

In [32]:
```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='upload' , data=internet_usage, ci=None, estimator=np.mean)
ax.bar_label(ax.containers[0])
plt.title("Average upload per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-32-93276443d33c> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='upload' , data=internet_usage, ci=None, estimator=np.mean)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Average upload per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
# User5 has the highest average upload with 64500.35Kb while User 7 has the lower average with 23075.54Kb

Now we will do the same but with the download, so calculating minimum, maximum and average total, and after that per user
```

In [33]:
```
print('The minimum download is: ' + str(internet_usage.download.min()) + 'Kb')
print('The maximum download is: ' + str(internet_usage.download.max()) + 'Kb')
print('The average download is: ' + str(round(internet_usage.download.mean(), 2)) + 'Kb')
```

```
The minimum download is: 9.0Kb
The maximum download is: 27902607.0Kb
The average download is: 396664.52Kb
```

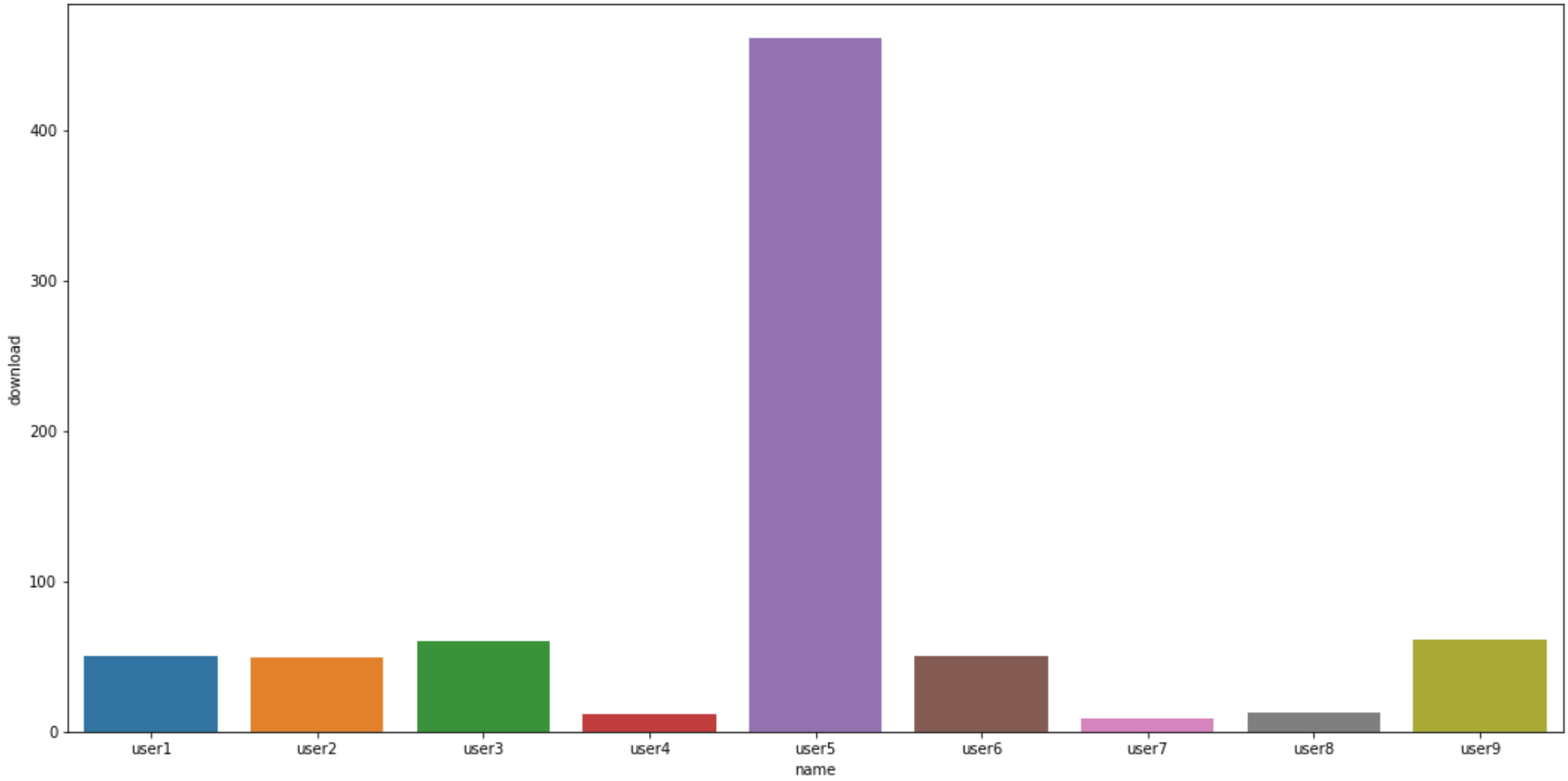
In [34]:
```
print('The minimum download per user:')
internet_usage.groupby('name').download.min()
```

```
The minimum download per user:
```

Out[34]:
```
name
user1     50.0
user2     49.0
user3     60.0
user4     12.0
user5    461.0
user6     50.0
user7      9.0
user8     13.0
user9     61.0
Name: download, dtype: float64
```

In [35]:
```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='download' , data=internet_usage, ci=None, estimator=np.min)
ax.bar_label(ax.containers[0])
plt.title("Minimum download per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-35-09d2a192287d> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='download' , data=internet_usage, ci=None, estimator=np.min)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Minimum download per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
User 5 has the highest minimum download with 461Kb while User 7 has the lowest with 9Kb
```

In [36]:
```
print('The maximum download per user:')
internet_usage.groupby('name').download.max()
```

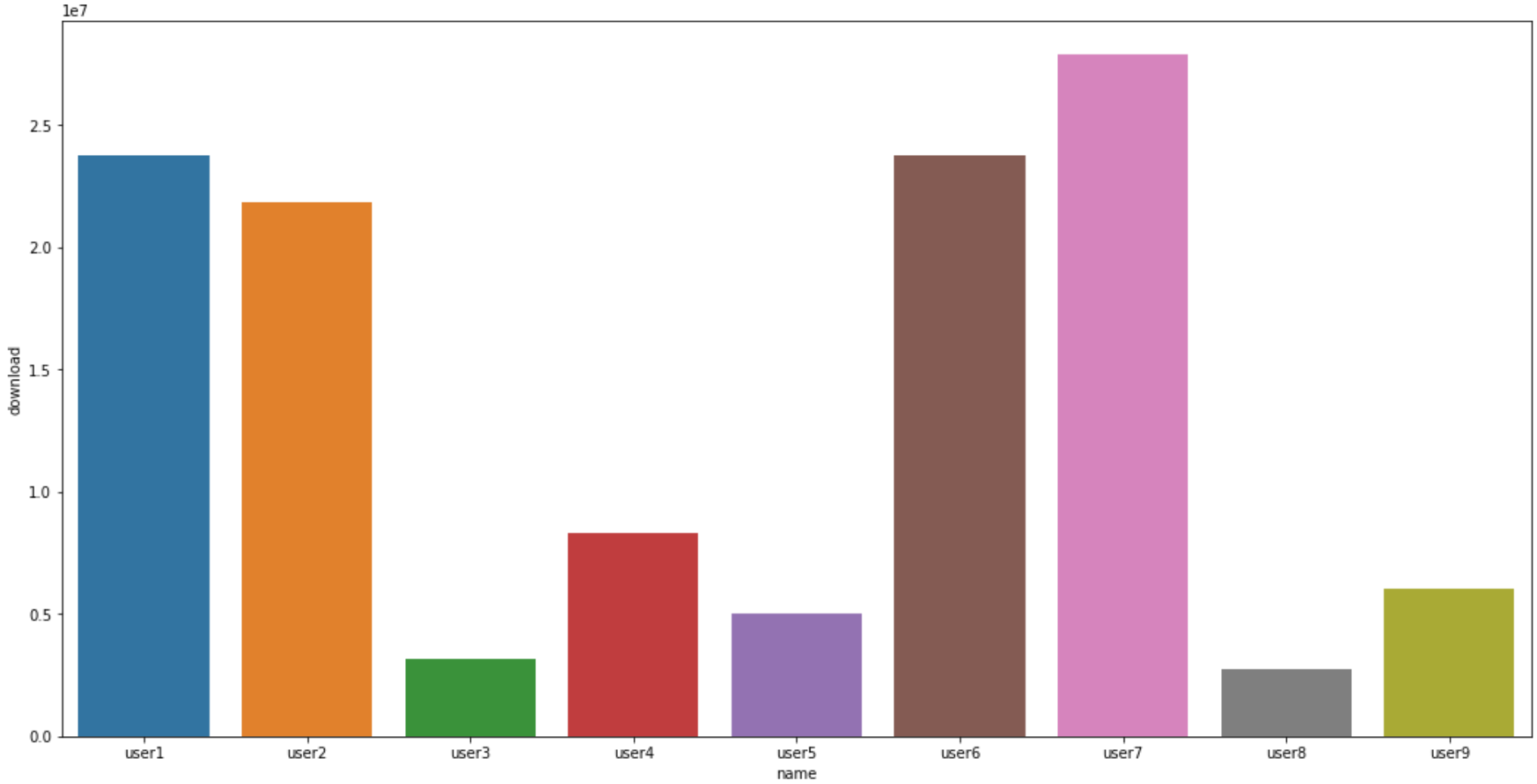
```
The maximum download per user:
```

Out[36]:
```
name
user1    23760732.0
user2    21831352.0
user3     3145728.0
user4     8325693.0
user5     5033164.0
user6    23760732.0
user7    27902607.0
user8     2747269.0
user9     6008340.0
Name: download, dtype: float64
```

In [37]:

```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='download' , data=internet_usage, ci=None, estimator=np.max)
ax.bar_label(ax.containers[0])
plt.title("Maximum download per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-37-0dce36657fd5> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='download' , data=internet_usage, ci=None, estimator=np.max)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Maximum download per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
User 7 has the highest maximum download with 27902607Kb while user 8 has the lowest with 2747269Kb
```

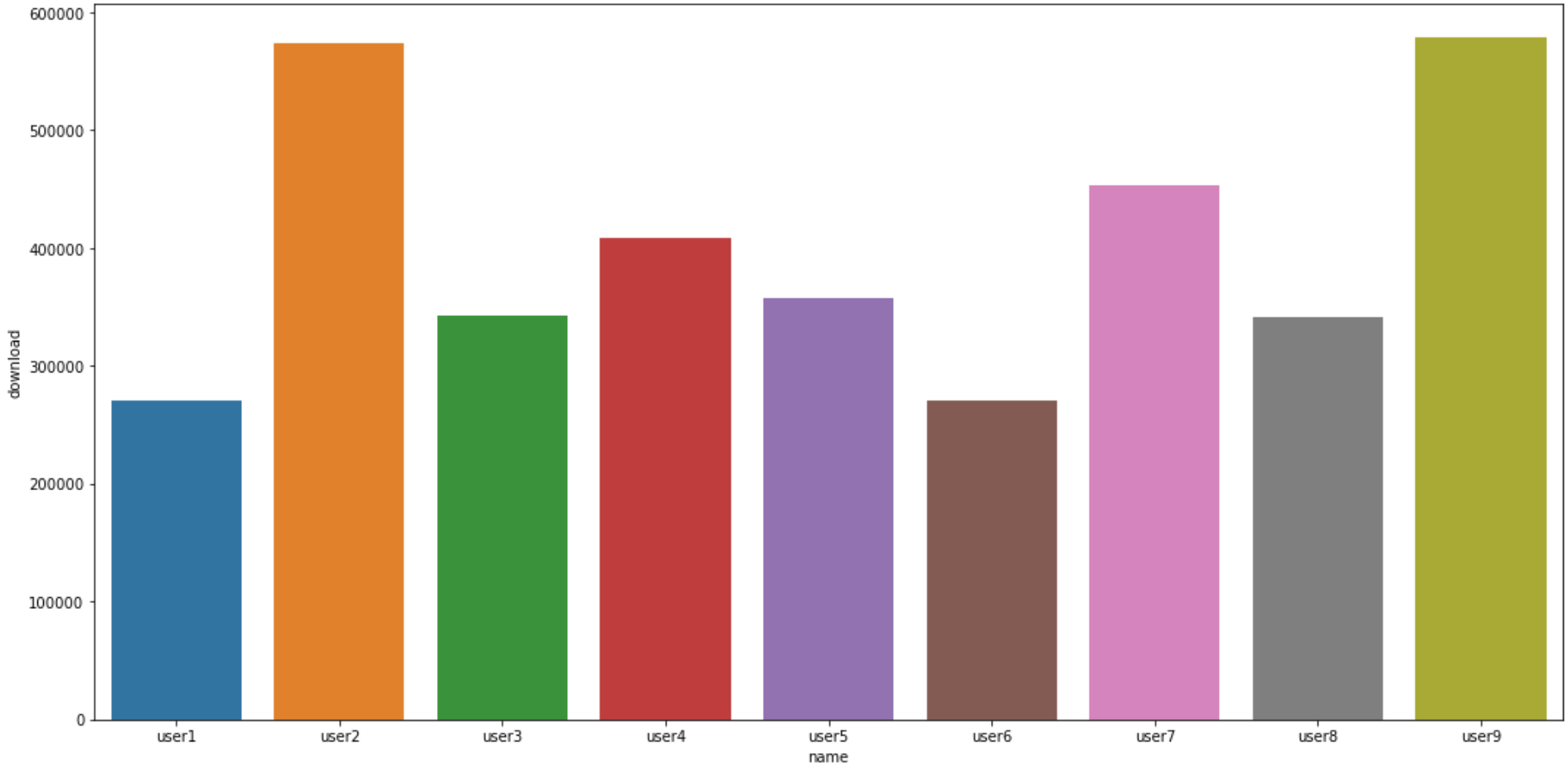
In [38]:
```
print('The average download per user:')
round(internet_usage.groupby('name').download.mean(), 2)
```

```
The average download per user:
```

Out[38]:
```
name
user1    270725.96
user2    573798.02
user3    342230.37
user4    408580.26
user5    357278.08
user6    270545.18
user7    453828.61
user8    341417.12
user9    578981.51
Name: download, dtype: float64
```

In [39]:
```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='download' , data=internet_usage, ci=None, estimator=np.mean)
ax.bar_label(ax.containers[0])
plt.title("Average download per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-39-0aa4753351c6> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='download' , data=internet_usage, ci=None, estimator=np.mean)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Average download per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
User 9 has the highest average download with 578982.51Kb while user 6 has the lowest with 270545.18Kb
```

In [41]:
```
# We will repeat all that with the total transfer column, minimum, maximum and average and than doing the same with each user

print('The minimum total transfer is: ' + str(internet_usage.total_transfer.min()) + 'Kb')
print('The maximum total transfer is: ' + str(internet_usage.total_transfer.max()) + 'Kb')
print('The average total transfer is: ' + str(round(internet_usage.total_transfer.mean(), 2)) + 'Kb')
```

```
The minimum total transfer is: 1.12Kb
The maximum total transfer is: 28552724.48Kb
The average total transfer is: 430437.21Kb
```

In [42]:
```
print('The minimum total transfer per user:')
internet_usage.groupby('name').total_transfer.min()
```

```
The minimum total transfer per user:
```

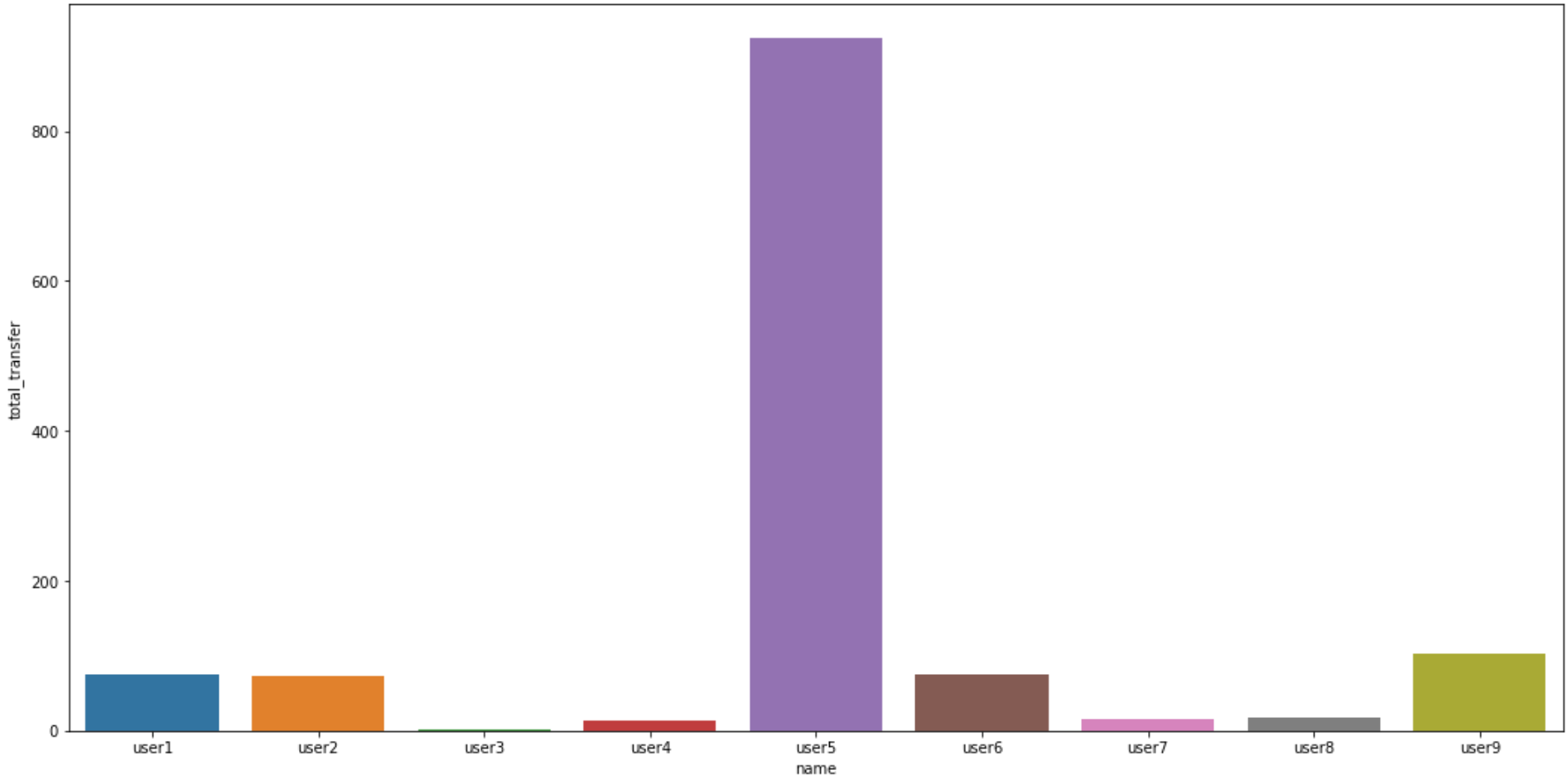
Out[42]:
```
name
user1     75.34
```

```
user2     73.13
user3      1.12
user4     13.45
user5    924.40
user6     75.34
user7     15.25
user8     18.08
user9    102.64
Name: total_transfer, dtype: float64
```

In [43]:

```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='total_transfer' , data=internet_usage, ci=None, estimator=np.min)
ax.bar_label(ax.containers[0])
plt.title("Minimum total transfer per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-43-134147240bb4> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='total_transfer' , data=internet_usage, ci=None, estimator=np.min)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Minimum total transfer per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
# User 5 has the highest minimum total transfer with 924.4Kb while user 3 has the lowest with 1.12Kb
```

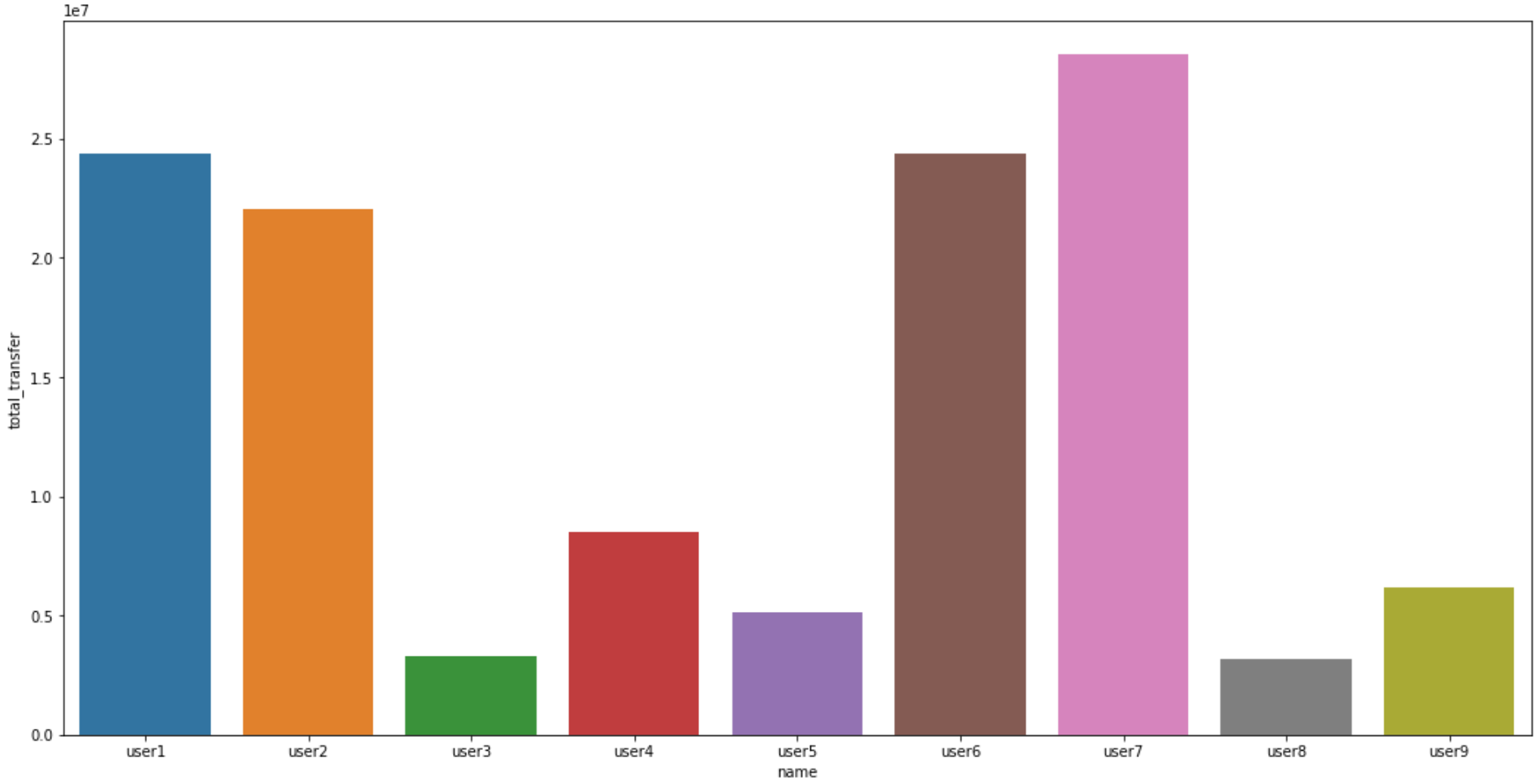
In [44]:
```
print('The maximum total transfer per user:')
internet_usage.groupby('name').total_transfer.max()
```

```
The maximum total transfer per user:
```

Out[44]:
```
name
user1    24389877.76
user2    22051553.28
user3     3282042.88
user4     8524922.88
user5     5158993.92
user6    24389877.76
user7    28552724.48
user8     3166699.52
user9     6155141.12
Name: total_transfer, dtype: float64
```

In [45]:
```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='total_transfer' , data=internet_usage, ci=None, estimator=np.max)
ax.bar_label(ax.containers[0])
plt.title("Maximum total transfer per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-45-6c8bb17d684f> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='total_transfer' , data=internet_usage, ci=None, estimator=np.max)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Maximum total transfer per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
User 7 has the highest maximum total transfer with 28552724.48Kb while user 8 has the lowest with 3166699.52Kb
```

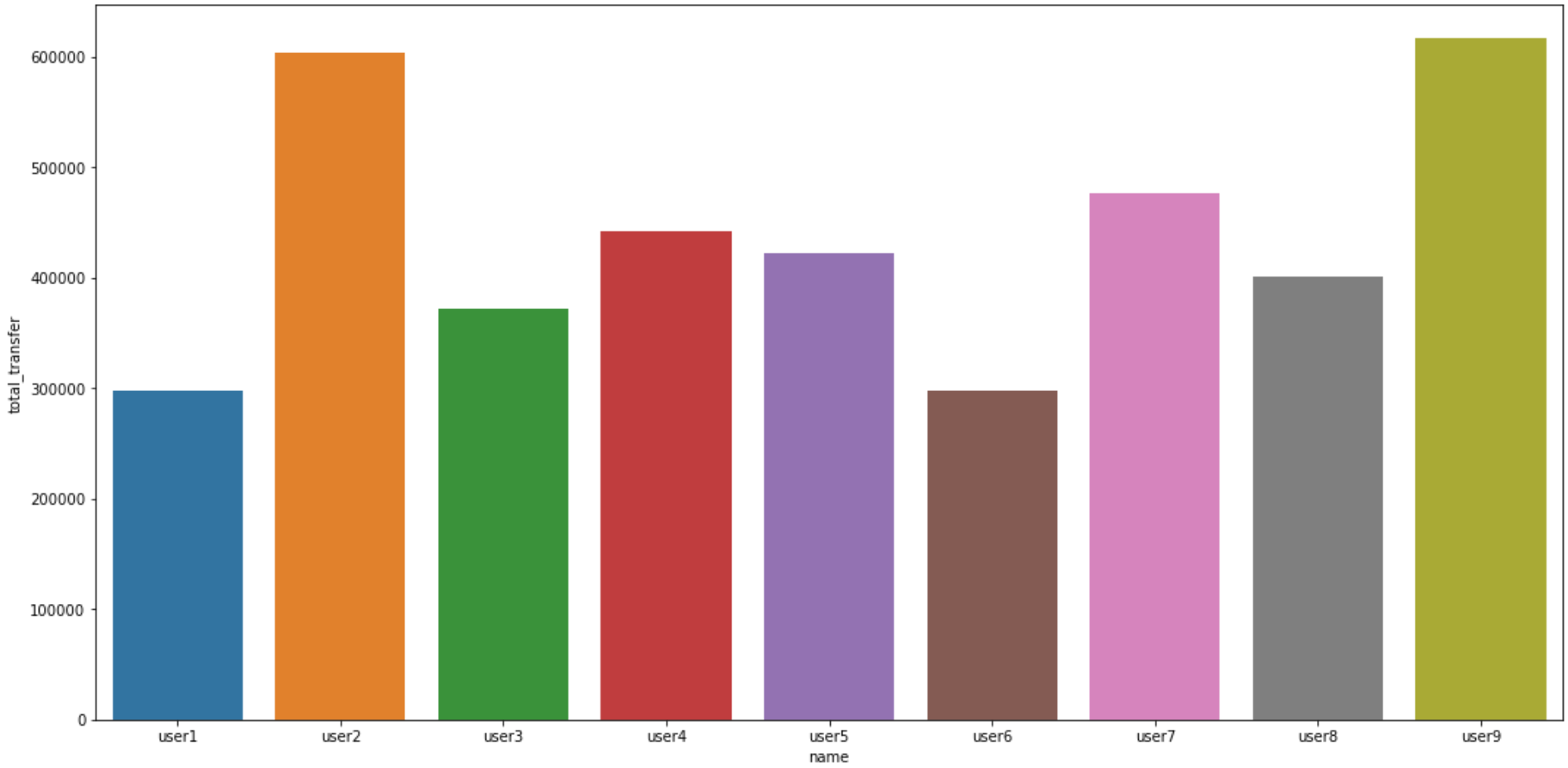
In [46]:
```
print('The average total transfer per user:')
round(internet_usage.groupby('name').total_transfer.mean(), 2)
```

```
The average total transfer per user:
```

Out[46]:
```
name
user1    297971.21
user2    602904.19
user3    371826.53
user4    442413.51
user5    421772.04
user6    298199.88
user7    476923.04
user8    400682.28
user9    616875.57
Name: total_transfer, dtype: float64
```

In [47]:
```
plt.figure(figsize=(18, 9))
ax = sns.barplot(x='name', y='total_transfer' , data=internet_usage, ci=None, estimator=np.mean)
ax.bar_label(ax.containers[0])
plt.title("Average total transfer per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-47-680e1a41cca5> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.barplot(x='name', y='total_transfer' , data=internet_usage, ci=None, estimator=np.mean)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Average total transfer per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



In [ ]:
```
User 9 has the highest average total transfer with 616875.57Kb while user 1 has the lowest with 297971.21Kb
```

In [48]:
```
internet_usage.seession_break_reason.value_counts()
```

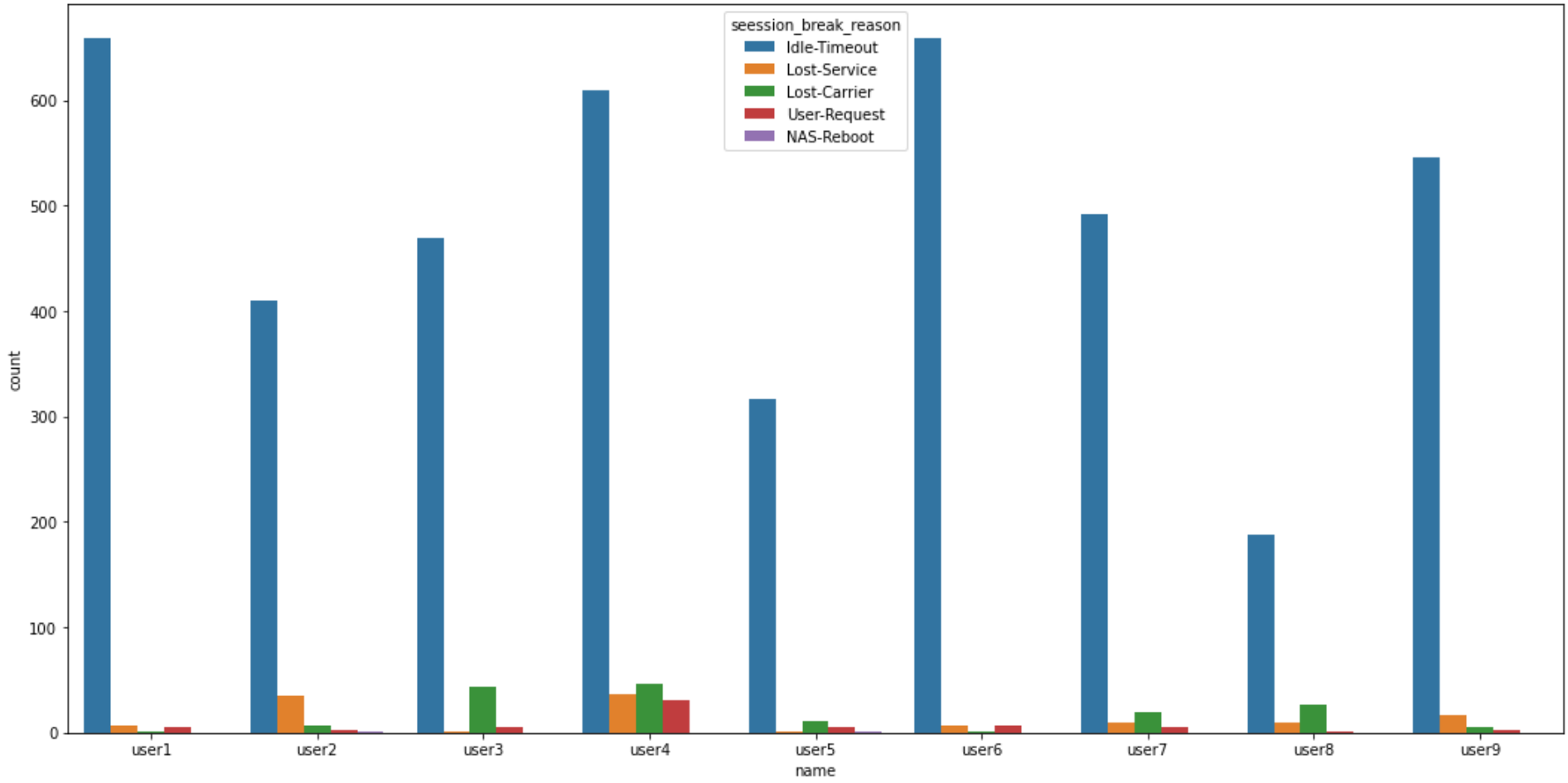
Out[48]:
```
Idle-Timeout    4350
Lost-Carrier     162
Lost-Service     124
User-Request      65
NAS-Reboot         2
Name: seession_break_reason, dtype: int64
```

In [ ]:
```
# The majority of the session break reasons were from "Idle-Timeout" with 4350 times,
while the other reasons have very low occurrences in comparison, with "NAS-Reboot" having the lowest with only 2 occurrences
```

In [49]:
```
plt.figure(figsize=(18, 9))
ax = sns.countplot(x='name', hue = 'seession_break_reason' , data=internet_usage)
ax.bar_label(ax.containers[0])
```

```
plt.title("Session break reason Count per user")
plt.show()
plt.clf()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-49-067d4d3006e4> in <module>
      1 plt.figure(figsize=(18, 9))
      2 ax = sns.countplot(x='name', hue = 'seession_break_reason' , data=internet_usage)
----> 3 ax.bar_label(ax.containers[0])
      4 plt.title("Session break reason Count per user")
      5 plt.show()

AttributeError: 'AxesSubplot' object has no attribute 'bar_label'
```



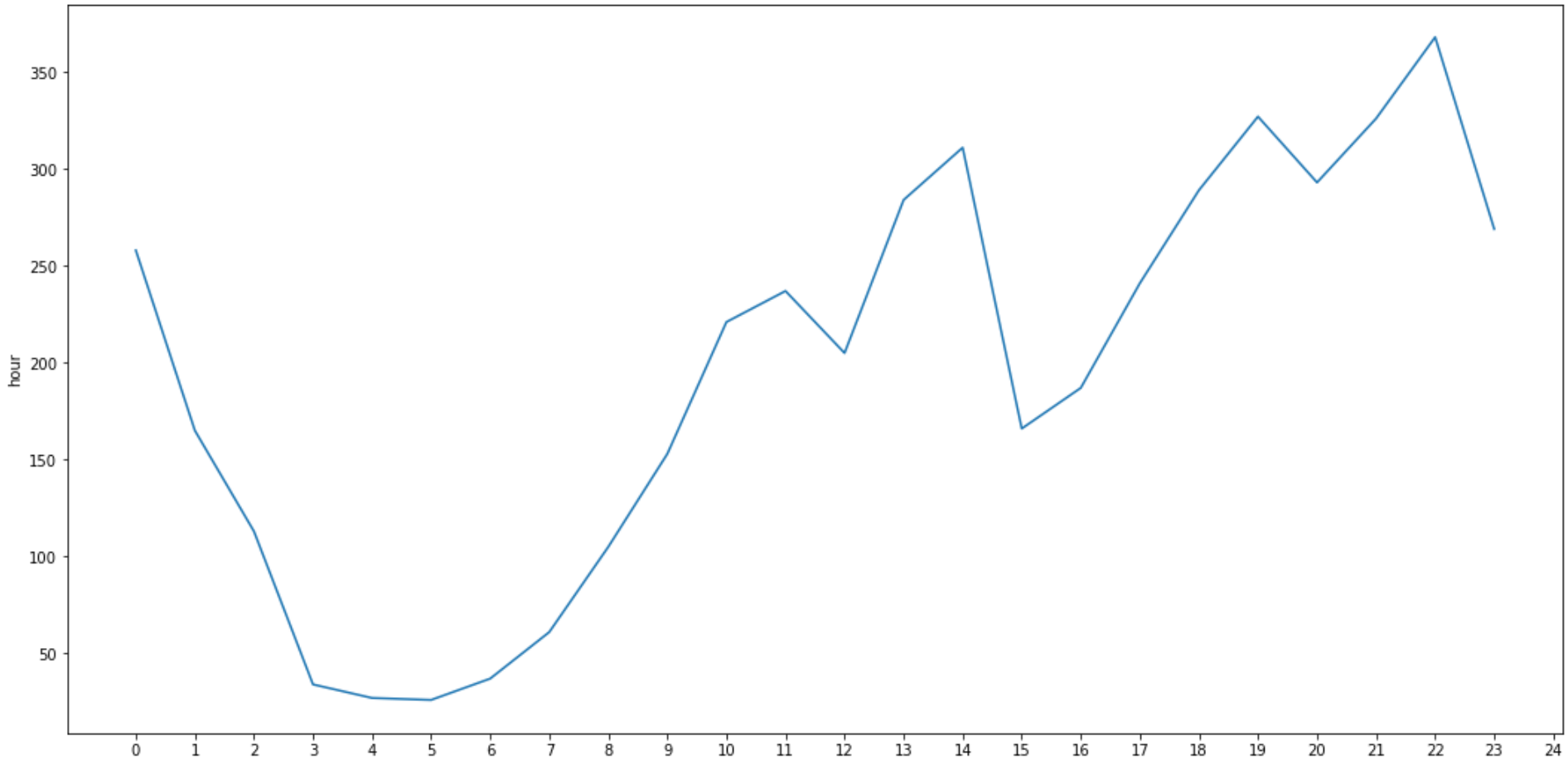
In [ ]:

If we check the session break reasons count per user, it's the same thing,
the majority are from "Idle-Timeout" while the other reasons have very low occurrences

In [ ]:
```
# 3 - Deeper Analysis
```

In [ ]:
```
# Now that we finished with the EDA, we can go deeper into our analysis and answer the questions asked earlier
We will start with this question:
What is the most frequent internet activity time of the day ?
```

In [50]:
```
internet_usage['hour'] = pd.to_datetime(internet_usage['start_time']).dt.hour
frequent_activity_time_of_day = internet_usage['hour'].value_counts().sort_index()
plt.figure(figsize=(18, 9))
sns.lineplot(data=frequent_activity_time_of_day)
plt.xticks(np.linspace(start=0, stop=24, num=25))
plt.show()
plt.clf()
```




```
<Figure size 432x288 with 0 Axes>
```

In [ ]:
```
The most frequent internet activity time of the day is 22h or 10pm
Now we answer this question:
How often the ip changes ?
```

In [51]:
```
base_ip = '48:E7:DA:58:22:E9'
ip_count = 0
for i in range(1, internet_usage.shape[0]):
    if internet_usage.iloc[i]['ip'] != base_ip:
        ip_count +=1
        base_ip = internet_usage.iloc[i]['ip']

print('The IP Adress changed ' + str(ip_count) + ' times')
```

```
The IP Adress changed 2303 times
```

In [52]:
```
base_device = 'device1'
device_count = 0
for i in range(1, internet_usage.shape[0]):
    if internet_usage.iloc[i]['device'] != base_device:
        device_count +=1
        base_device = internet_usage.iloc[i]['device']

print('The device changed ' + str(device_count) + ' times')
```

```
The device changed 1223 times
```

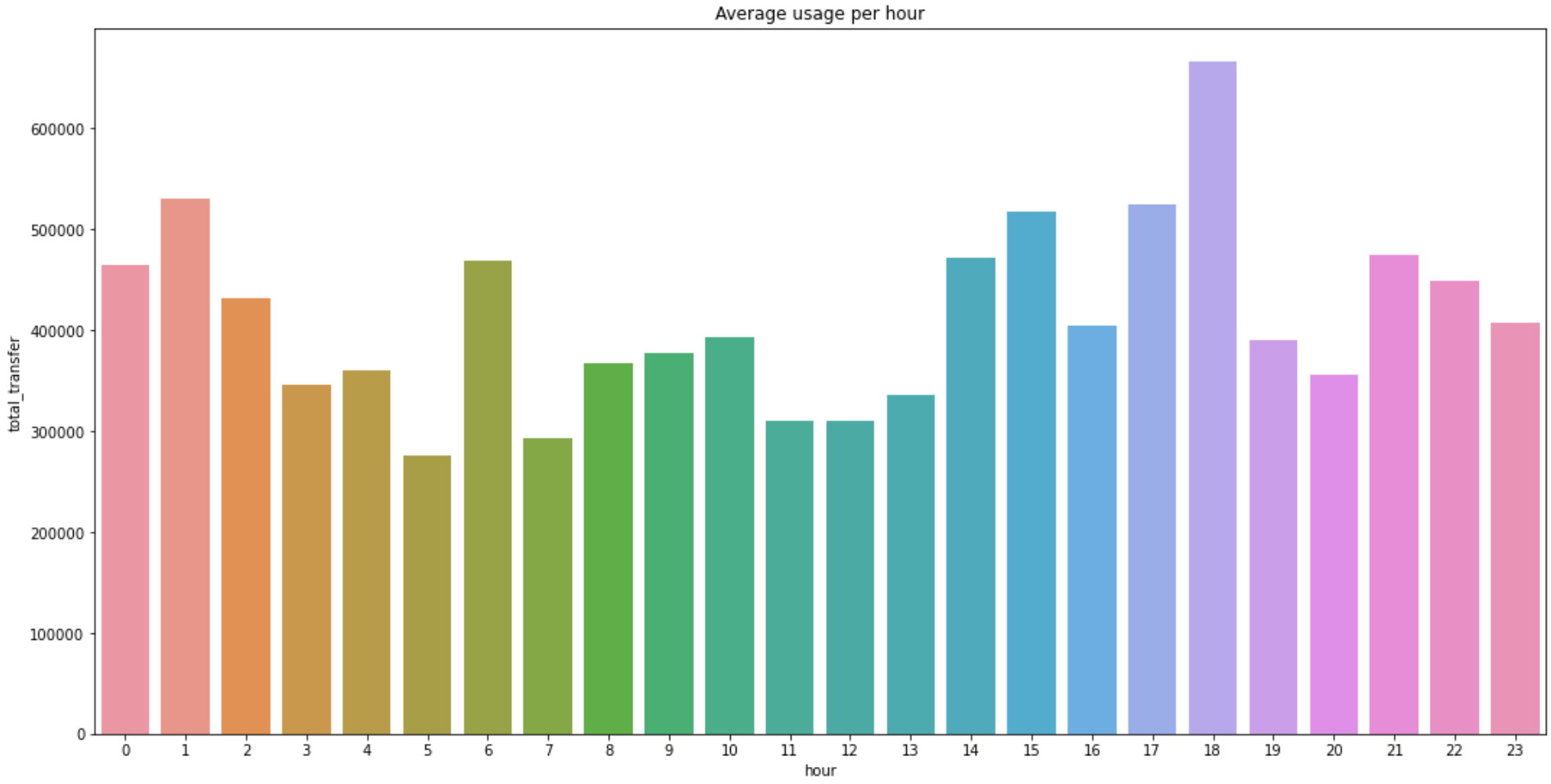
In [53]:
```
internet_usage.reset_index(inplace=True)

internet_usage['day'] = internet_usage['start_time'].dt.day
internet_usage['month'] = internet_usage['start_time'].dt.month

hourly_average = internet_usage.groupby('hour').total_transfer.mean()
print('The Average usage per hour is:\n ' +  str(round(hourly_average, 2)))
```

```
The Average usage per hour is:
 hour
0     464530.44
1     530880.86
2     431576.11
3     345303.34
4     359809.44
5     275960.91
6     468959.59
7     292886.83
8     366681.92
9     377480.64
10    393259.12
11    309492.45
12    310137.98
```

```
13    335270.58
14    472403.71
15    517005.11
16    403919.40
17    525423.69
18    666590.76
19    389841.79
20    355862.80
21    474038.34
22    449600.50
23    407785.08
Name: total_transfer, dtype: float64
```

In [54]:

```
plt.figure(figsize=(18, 9))
sns.barplot(x='hour', y='total_transfer' , data=internet_usage, ci=None, estimator=np.mean)
plt.title("Average usage per hour")
plt.show()
plt.clf()
```



```
<Figure size 432x288 with 0 Axes>
```

In [ ]:
```
And now we see the average usage per day
```

In [55]:
```
daily_average = internet_usage.groupby('day').total_transfer.mean()
print('The Average usage per day is:\n ' +  str(round(daily_average, 2)))
```

```
The Average usage per day is:
 day
1     396705.04
2     494496.48
3     445865.63
4     676332.03
5     652195.66
6     396261.75
7     402259.89
8     301859.57
9     393521.97
10    350665.02
11    729857.65
12    346695.95
13    501906.70
14    352701.10
15    521520.51
16    426719.39
17    475795.71
18    337490.93
19    301941.32
20    365130.12
21    462211.69
22    486595.37
23    383153.93
24    320598.94
25    443689.47
26    463432.02
27    324318.12
28    494576.34
29    363645.61
30    361418.88
31    369118.01
Name: total_transfer, dtype: float64
```

In [ ]:
```
# And now we see the average usage per day
```

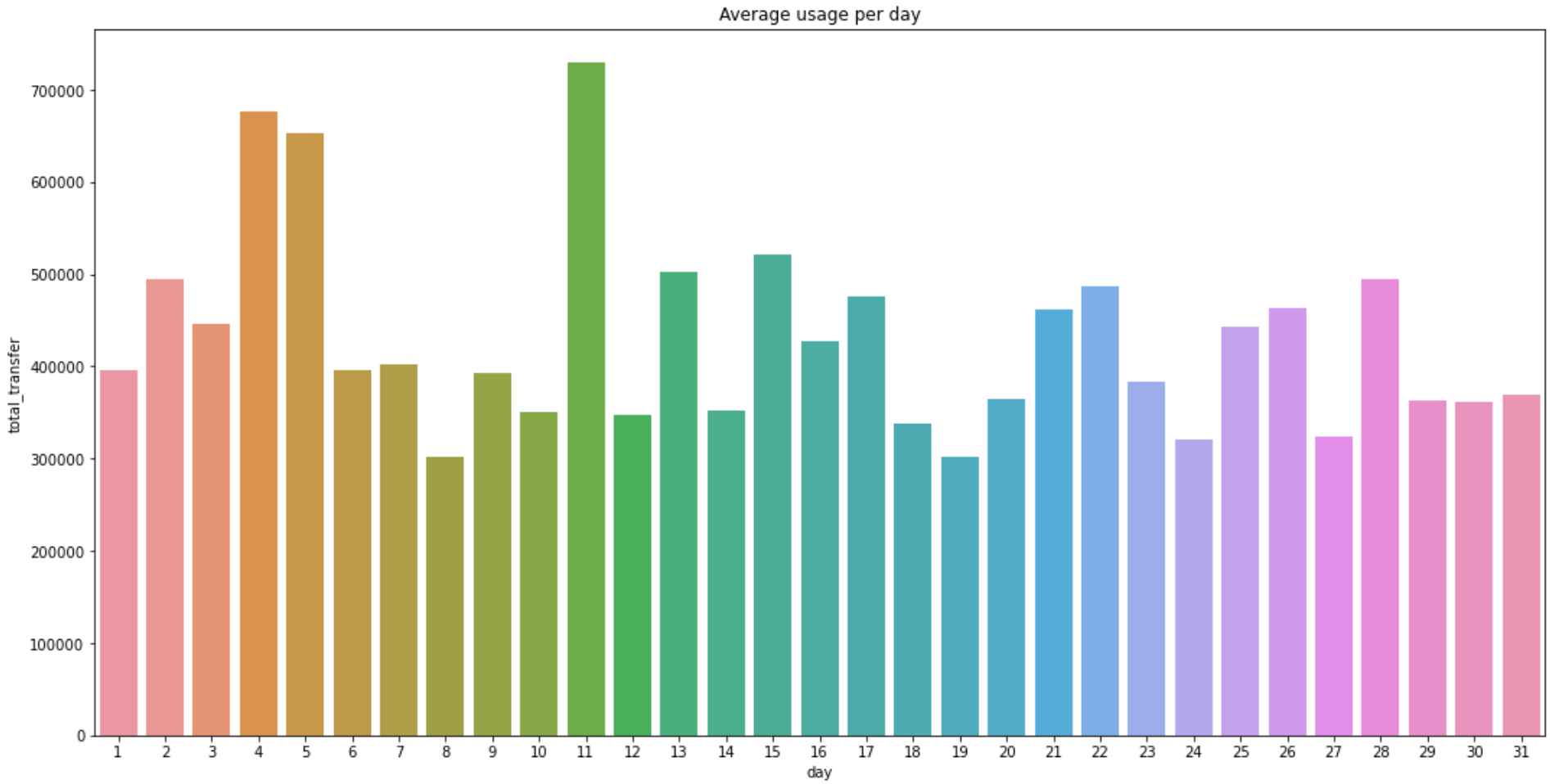
In [56]:
```
daily_average = internet_usage.groupby('day').total_transfer.mean()
print('The Average usage per day is:\n ' +  str(round(daily_average, 2)))
```

```
The Average usage per day is:
 day
1     396705.04
```

```
2     494496.48
3     445865.63
4     676332.03
5     652195.66
6     396261.75
7     402259.89
8     301859.57
9     393521.97
10    350665.02
11    729857.65
12    346695.95
13    501906.70
14    352701.10
15    521520.51
16    426719.39
17    475795.71
18    337490.93
19    301941.32
20    365130.12
21    462211.69
22    486595.37
23    383153.93
24    320598.94
25    443689.47
26    463432.02
27    324318.12
28    494576.34
29    363645.61
30    361418.88
31    369118.01
Name: total_transfer, dtype: float64
```

In [57]:

```
plt.figure(figsize=(18, 9))
sns.barplot(x='day', y='total_transfer' , data=internet_usage, ci=None, estimator=np.mean)
plt.title("Average usage per day")
plt.show()
plt.clf()
```



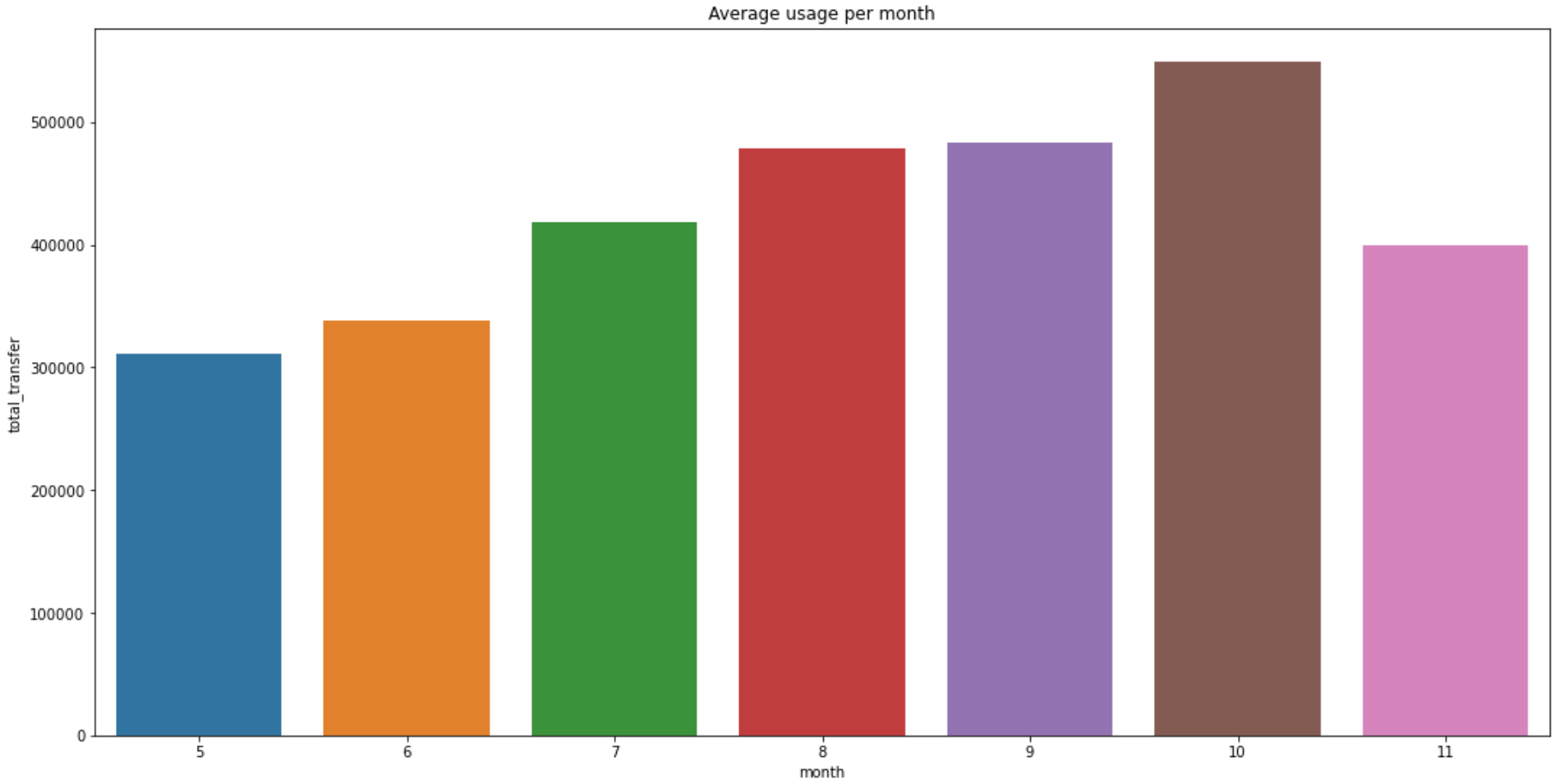

```
<Figure size 432x288 with 0 Axes>
```

In [ ]:
```
# And now the average usage per month
```

In [58]:
```
monthly_average = internet_usage.groupby('month').total_transfer.mean()
print('The Average usage per month is:\n ' +  str(round(monthly_average, 2)))
```

```
The Average usage per month is:
 month
5     311177.16
6     338418.08
7     418583.99
8     479042.44
9     482955.52
10    549467.63
11    399804.11
Name: total_transfer, dtype: float64
```

In [59]:
```
plt.figure(figsize=(18, 9))
sns.barplot(x='month', y='total_transfer' , data=internet_usage, ci=None, estimator=np.mean)
plt.title("Average usage per month")
plt.show()
plt.clf()
```



<Figure size 432x288 with 0 Axes>

In [ ]:
```
# 3 - Conclusion
```

In [ ]:
```
In this project we had a dataset about the internet usage [in kb] by graduate students at an indian university.
We imported the data, cleaned it, analyzed it and answered the questions asked
-The dataset contains 9 users that used 1224 difference devices to connect to the internet while uploading 2841640.0Kb
and downloading 27902607.0Kb with a total transfer of 28552724.48Kb during a period of 7 months
-The most frequent internet activity time of the day is 22h or 10pm
-The IP Adress changed 2303 times while the devices used changed 1223 times
-The highest average usage per hour was 666590.76Kb around18h or 6pm, the highest average usage per day was 729857.65Kb
```

```
around the 11th day of the month, while the highest average usage per month was during the month of October with 549467.63Kb
total transfer of data
```