

In [1]:

```
import pandas as pd
```

In [2]:

```
df = pd.read_csv('train_peptides.csv')
```

In [3]:

```
df.head()
```

Out[3]:

	visit_id	visit_month	patient_id	UniProt	Peptide_I
0	55_0	0	55	O00391	NEQEQLGQWHL
1	55_0	0	55	O00533	GNPEPTFSWTK
2	55_0	0	55	O00533	IEIPSSVQQVPTI
3	55_0	0	55	O00533	KPQSAVYSTGSNGILLC(UniMod_4)EAEGEPQPTIK
4	55_0	0	55	O00533	SMEQNPGPLEYR

◀ ▶

In [4]:

```
df.tail()
```

Out[4]:

	visit_id	visit_month	patient_id	UniProt	Peptide_F
981829	58648_108	108	58648	Q9UHG2	ILAGSADSEGVAAPR
981830	58648_108	108	58648	Q9UKV8	SGNIPAGTTVDTK
981831	58648_108	108	58648	Q9Y646	LALLVDTVGPR
981832	58648_108	108	58648	Q9Y6R7	AGC(UniMod_4)VAESTAVC(UniMod_4)R
981833	58648_108	108	58648	Q9Y6R7	GATTSPGVYELSSR

◀ ▶

In [5]:

```
df.shape
```

Out[5]:

(981834, 6)

In [6]:

```
df.columns
```

Out[6]:

```
Index(['visit_id', 'visit_month', 'patient_id', 'UniProt', 'Peptide',
       'PeptideAbundance'],
      dtype='object')
```

In [7]:

```
df.duplicated().sum()
```

Out[7]:

```
0
```

In [8]:

```
df.isnull().sum()
```

Out[8]:

```
visit_id      0
visit_month    0
patient_id     0
UniProt        0
Peptide        0
PeptideAbundance  0
dtype: int64
```

In [9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 981834 entries, 0 to 981833
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype  
 ---  --  
 0   visit_id         981834 non-null   object 
 1   visit_month      981834 non-null   int64  
 2   patient_id       981834 non-null   int64  
 3   UniProt          981834 non-null   object 
 4   Peptide          981834 non-null   object 
 5   PeptideAbundance 981834 non-null   float64
dtypes: float64(1), int64(2), object(3)
memory usage: 44.9+ MB
```

In [10]:

```
df.describe()
```

Out[10]:

	visit_month	patient_id	PeptideAbundance
count	981834.000000	981834.000000	9.818340e+05
mean	26.105061	32603.465361	6.428902e+05
std	22.913897	18605.934422	3.377989e+06
min	0.000000	55.000000	1.099850e+01
25%	6.000000	16566.000000	2.817425e+04
50%	24.000000	29313.000000	7.430830e+04
75%	48.000000	49995.000000	2.213388e+05
max	108.000000	65043.000000	1.787520e+08

In [11]:

```
df.nunique()
```

Out[11]:

```
visit_id           1113
visit_month        15
patient_id         248
UniProt            227
Peptide             968
PeptideAbundance    738931
dtype: int64
```

In [12]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [13]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [14]:

```
df['visit_month'].unique()
```

Out[14]:

```
array([  0,   3,   6,  12,  18,  24,  30,  36,  48,  54,  60,  72,  84,
       96, 108], dtype=int64)
```

In [15]:

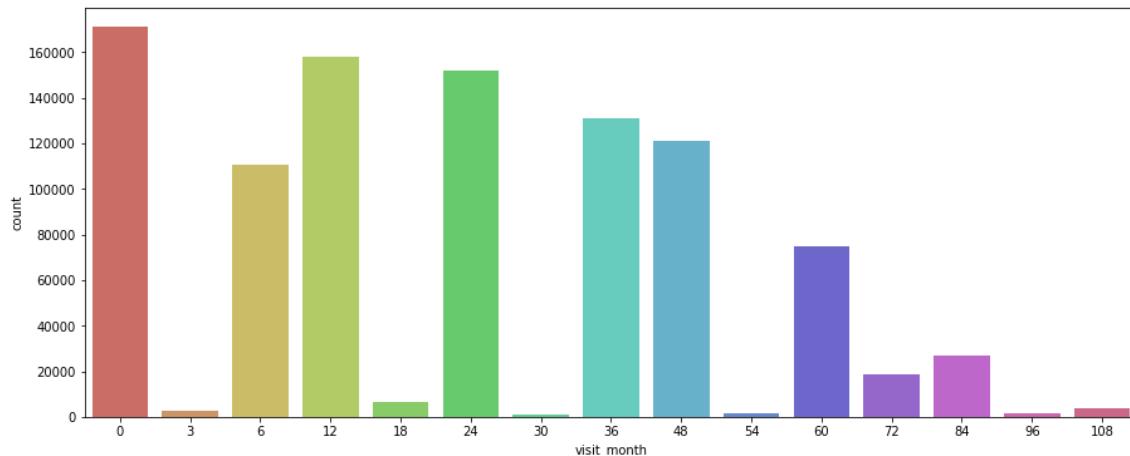
```
df['visit_month'].value_counts()
```

Out[15]:

```
0      171048  
12     158114  
24     152036  
36     130903  
48     121343  
6      110733  
60     74562  
84     26729  
72     18642  
18     6822  
108    3631  
3      2742  
96     1815  
54     1801  
30     913  
Name: visit_month, dtype: int64
```

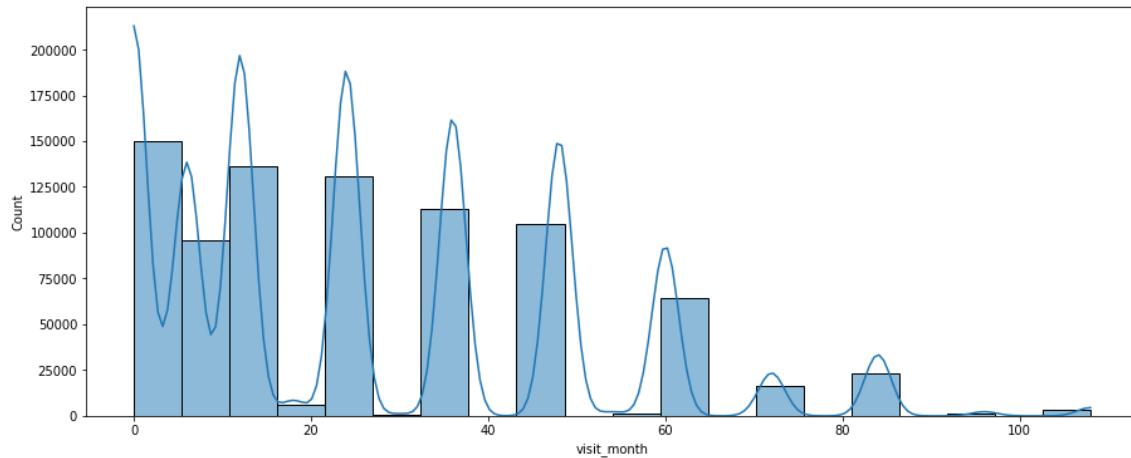
In [16]:

```
plt.figure(figsize=(15,6))  
sns.countplot('visit_month', data = df, palette = 'hls')  
plt.show()
```



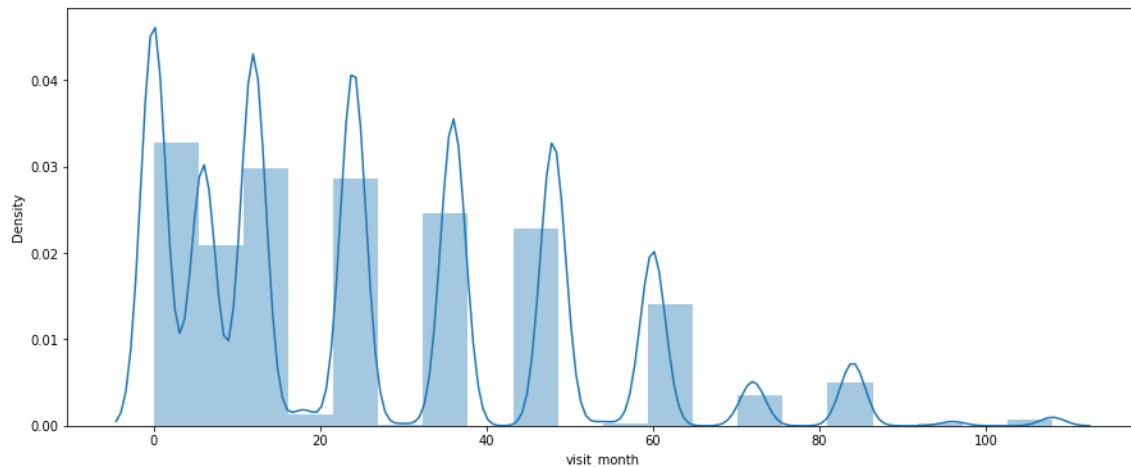
In [270]:

```
plt.figure(figsize=(15,6))
sns.histplot(df['visit_month'], kde = True, bins = 20, palette = 'hls')
plt.show()
```



In [272]:

```
plt.figure(figsize=(15,6))
sns.distplot(df['visit_month'], kde = True, bins = 20)
plt.show()
```



In [17]:

```
df['patient_id'].unique()
```

Out[17]:

```
array([ 55, 1517, 1923, 2660, 3636, 3863, 4161, 4172, 5027,
       5178, 5645, 5742, 6054, 6211, 7051, 7117, 7568, 7832,
     8699, 10053, 10174, 10541, 10715, 10718, 11459, 11686, 11928,
   12516, 12636, 12703, 12755, 13368, 13618, 13968, 14035, 14124,
  14242, 14450, 14811, 15009, 15504, 15590, 16238, 16347, 16566,
  16574, 17154, 17414, 17727, 18183, 18204, 18553, 18560, 19088,
 20212, 20216, 20352, 20404, 20664, 20791, 21126, 21537, 21729,
 22126, 22623, 23192, 23244, 23391, 23636, 24278, 24690, 24820,
 24911, 25562, 25739, 25750, 25827, 25911, 26005, 26104, 26210,
 26809, 27079, 27300, 27464, 27468, 27715, 27971, 28327, 28342,
 28818, 29313, 29417, 30119, 30155, 30416, 30894, 30951, 31121,
 31154, 31693, 33548, 33558, 34182, 35231, 35465, 35477, 35675,
 35696, 36797, 37312, 37566, 38419, 39144, 40022, 40200, 40340,
 40751, 40874, 40967, 41444, 41617, 41628, 41871, 41883, 41930,
 42003, 42385, 42579, 44001, 44154, 44682, 45161, 45181, 46837,
 47103, 47513, 47881, 48780, 48928, 49239, 49672, 49683, 49995,
 50611, 50907, 51243, 51879, 51893, 51899, 52266, 53103, 54095,
 54406, 55240, 55256, 55302, 55662, 56073, 56075, 56119, 56317,
 56675, 56691, 56727, 57009, 57321, 57416, 57478, 57507, 57646,
 58189, 58270, 58648, 58653, 58823, 59550, 59574, 60170, 60326,
 60443, 60803, 61503, 61974, 62329, 62437, 62723, 62732, 62792,
 63875, 63889, 64669, 64674, 65043, 23175, 56538, 57468, 942,
 5036, 7151, 8344, 13360, 13852, 14270, 15245, 16778, 16931,
 20581, 20707, 24818, 31270, 33108, 37220, 40798, 44789, 47171,
 52119, 55096, 57108, 58674, 60788, 4923, 6420, 7265, 7886,
 10138, 13804, 14344, 17201, 20460, 20792, 27607, 27872, 27893,
 27987, 39719, 40650, 40876, 40980, 42086, 45662, 51708, 51846,
 52998, 54979, 58597, 7508, 12931], dtype=int64)
```

In [18]:

```
df['patient_id'].value_counts()
```

Out[18]:

20404	8272
23391	8147
62723	8111
26210	8076
18183	7965
	...
55240	1810
52119	1804
25911	1782
42086	1781
41930	876

Name: patient_id, Length: 248, dtype: int64

In [19]:

```
df['UniProt'].unique()
```

Out[19]:

```
array(['000391', '000533', '000584', '014498', '014773', '014791',
       '015240', '015394', '043505', '060888', '075144', '075326',
       '094919', 'P00441', 'P00450', 'P00734', 'P00736', 'P00738',
       'P00746', 'P00747', 'P00748', 'P00751', 'P01008', 'P01009',
       'P01011', 'P01019', 'P01023', 'P01024', 'P01031', 'P01033',
       'P01034', 'P01042', 'P01344', 'P01591', 'P01608', 'P01621',
       'P01717', 'P01780', 'P01833', 'P01834', 'P01857', 'P01859',
       'P01860', 'P01861', 'P01876', 'P01877', 'P02452', 'P02647',
       'P02649', 'P02652', 'P02655', 'P02656', 'P02671', 'P02675',
       'P02679', 'P02747', 'P02748', 'P02749', 'P02750', 'P02751',
       'P02753', 'P02760', 'P02763', 'P02765', 'P02766', 'P02768',
       'P02774', 'P02787', 'P02790', 'P04004', 'P04075', 'P04156',
       'P04180', 'P04196', 'P04207', 'P04211', 'P04216', 'P04217',
       'P04275', 'P04406', 'P04433', 'P05060', 'P05067', 'P05090',
       'P05155', 'P05156', 'P05408', 'P05452', 'P05546', 'P06310',
       'P06396', 'P06454', 'P06681', 'P06727', 'P07195', 'P07225',
       'P07333', 'P07339', 'P07602', 'P07711', 'P07858', 'P07998',
       'P08123', 'P08133', 'P08253', 'P08294', 'P08493', 'P08571',
       'P08603', 'P08637', 'P08697', 'P09104', 'P09486', 'P09871',
       'P10451', 'P10643', 'P10645', 'P10909', 'P11142', 'P11277',
       'P12109', 'P13473', 'P13521', 'P13591', 'P13611', 'P13671',
       'P13987', 'P14174', 'P14314', 'P14618', 'P16035', 'P16070',
       'P16152', 'P16870', 'P17174', 'P17936', 'P18065', 'P19021',
       'P19652', 'P19823', 'P19827', 'P20774', 'P20933', 'P23083',
       'P23142', 'P24592', 'P25311', 'P27169', 'P30086', 'P31997',
       'P35542', 'P36222', 'P36955', 'P36980', 'P39060', 'P40925',
       'P41222', 'P43121', 'P43251', 'P43652', 'P49588', 'P49908',
       'P51884', 'P54289', 'P55290', 'P61278', 'P61626', 'P61769',
       'P61916', 'P80748', 'P98160', 'Q02818', 'Q06481', 'Q08380',
       'Q12805', 'Q12841', 'Q12907', 'Q13283', 'Q13332', 'Q13451',
       'Q13740', 'Q14118', 'Q14508', 'Q14515', 'Q14624', 'Q15904',
       'Q16270', 'Q16610', 'Q562R1', 'Q6UX71', 'Q6UXB8', 'Q6UXD5',
       'Q7Z3B1', 'Q7Z5P9', 'Q8IWV7', 'Q8N2S1', 'Q8NBJ4', 'Q8NE71',
       'Q92520', 'Q92823', 'Q92876', 'Q96BZ4', 'Q96KN2', 'Q96PD5',
       'Q96S96', 'Q99435', 'Q99674', 'Q99832', 'Q99969', 'Q9BY67',
       'Q9HDC9', 'Q9NQ79', 'Q9NYU2', 'Q9UBR2', 'Q9UBX5', 'Q9UHG2',
       'Q9UNU6', 'Q9Y646', 'Q9Y6R7', 'P01594', 'P02792', 'P32754',
       'P60174', 'Q13449', 'Q99683', 'Q99829', 'Q9UKV8'], dtype=object)
```

In [20]:

```
df['UniProt'].value_counts()
```

Out[20]:

```
P02787      51916
P02768      40900
P01024      35253
P02751      27370
P02649      23082
...
Q6UX71      661
P01780      654
Q562R1      616
Q99832      606
Q99829      489
Name: UniProt, Length: 227, dtype: int64
```

In [21]:

```
df['Peptide'].unique()
```

Out[21]:

```
array(['NEQEQQPLGQWHLS', 'GNPEPTFSWTK', 'IEIPSSVQQVPTIIK',
       'KPQSAVYSTGSNGILLC(UniMod_4)EAEQEPQPTIK', 'SMEQNGPGLEYR',
       'TLKIENVSYQDKGNYR', 'VIAVNEVGR', 'VMTPAVYAPYDVK',
       'VNGSPVDNHPFAGDVVFPR', 'ELDLNSVLLK', 'HGTC(UniMod_4)AAQVDALNSQK
K',
       'ALPGTPVASSQPR', 'LFGGNFAHQASVAR', 'LYQQHGAGLFDVTR',
       'VTEPISAESGEQVER', 'AYQGVAAPFPK', 'QQETAAAETETR', 'THLGEALAPLS
K',
       'ASGSPEPAISWFR', 'NIINSDGGPYVC(UniMod_4)R', 'TALASGGVLDASGDYR',
       'TQSSLVPALTDFVR', 'ALMSPAGMLR', 'GLYDVSVLR',
       'SEGLLAC(UniMod_4)GTNAR', 'ILEVVNQIQDEER', 'QALNTDYLDSDYQR',
       'ADDLGKGNEESTKTGNAGSR', 'TLVVHEKADDLGKGNEESTK', 'ALYLQYTDETF
R',
       'EVGPTNADPVC(UniMod_4)LAK', 'FNKNNEGTYYSPNYPQSR', 'GAYPLSIEPIGV
R',
       'KAEEEHLGILGPQLHADVGDKVK', 'MFTTAPDQVDKEDEDQESNK',
       'MYYSAVDPTKDIFTGLIGPM(UniMod_35)K', 'NNEGTYYSPNYPQSR',
       'OYTDSTFRVPVFR', 'SVPPSASHVAPTFETFYFWTVPK']
```

In [22]:

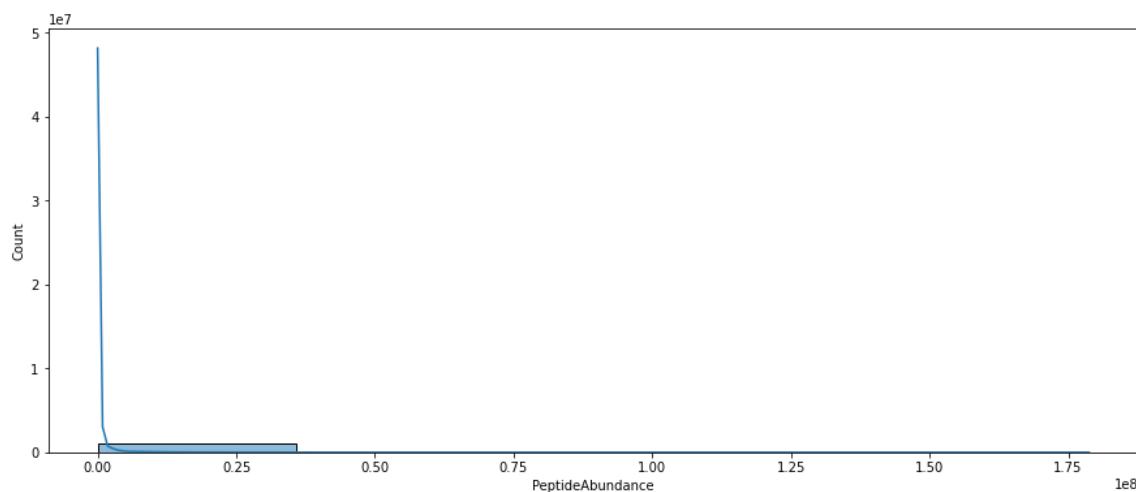
```
df['Peptide'].value_counts()
```

Out[22]:

```
TLLSNLEEAK      1113  
IPTTFENGR      1113  
NILTSNNIDVK     1113  
KYLYEiar        1113  
AIGYLNTGYQR     1113  
...  
HYEGSTVPEK       605  
SLEDQVEMLR      599  
TPSGLYLGTC(UniMod_4)ER 590  
EPQVYTLPPSRDELTK 563  
QALPQVR          489  
Name: Peptide, Length: 968, dtype: int64
```

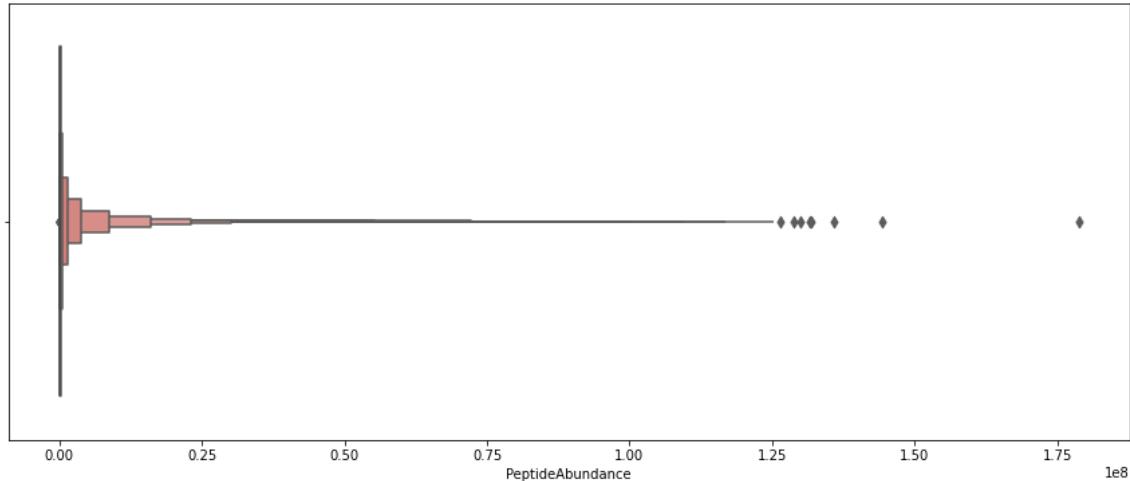
In [23]:

```
plt.figure(figsize=(15,6))  
sns.histplot(df['PeptideAbundance'], bins = 5, kde = True, palette = 'hls')  
plt.show()
```



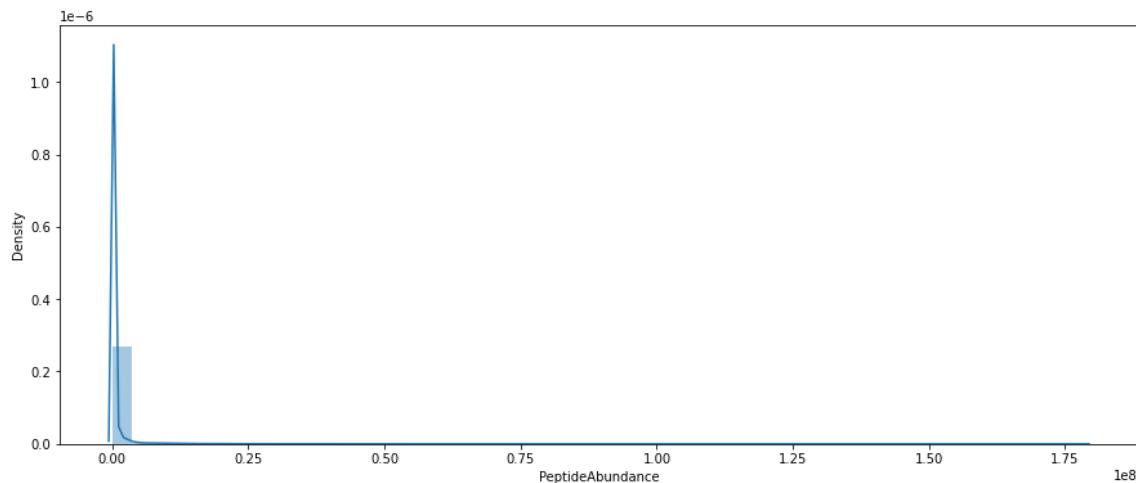
In [24]:

```
plt.figure(figsize=(15,6))
sns.boxenplot(df['PeptideAbundance'], palette = 'hls')
plt.show()
```



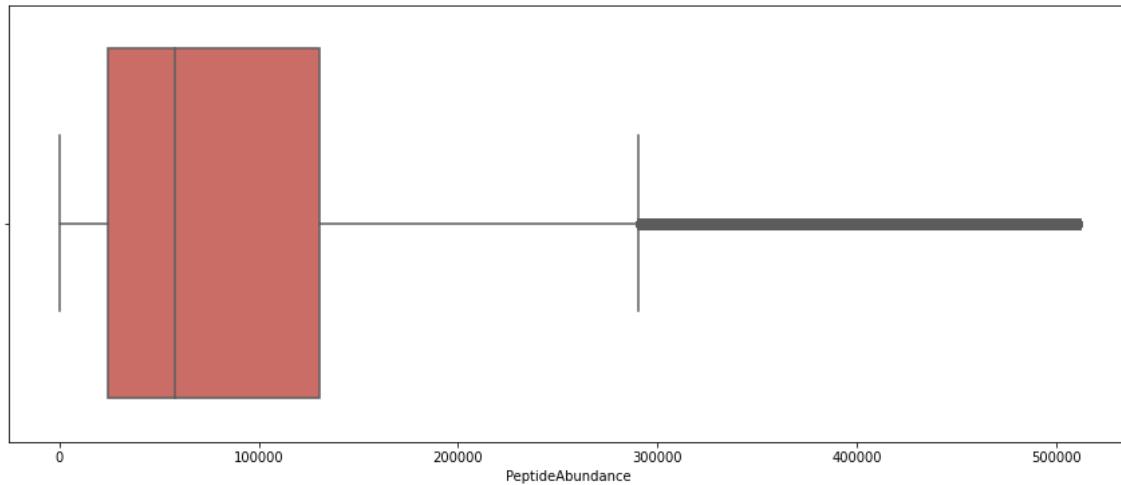
In [25]:

```
plt.figure(figsize=(15,6))
sns.distplot(df['PeptideAbundance'], kde = True)
plt.show()
```



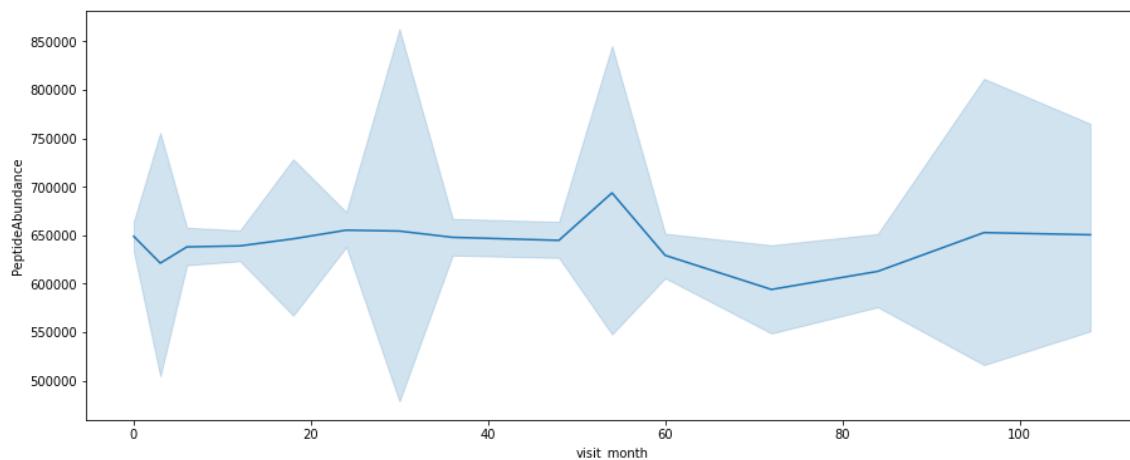
In [273]:

```
plt.figure(figsize=(15,6))
sns.boxplot(df['PeptideAbundance'], palette = 'hls')
plt.show()
```



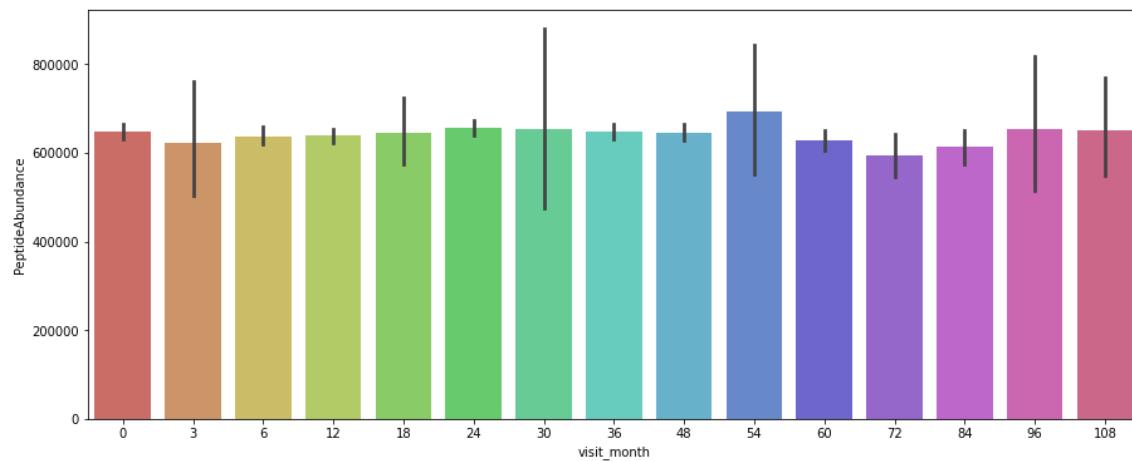
In [26]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df['visit_month'], y = df['PeptideAbundance'], palette = 'hls')
plt.show()
```



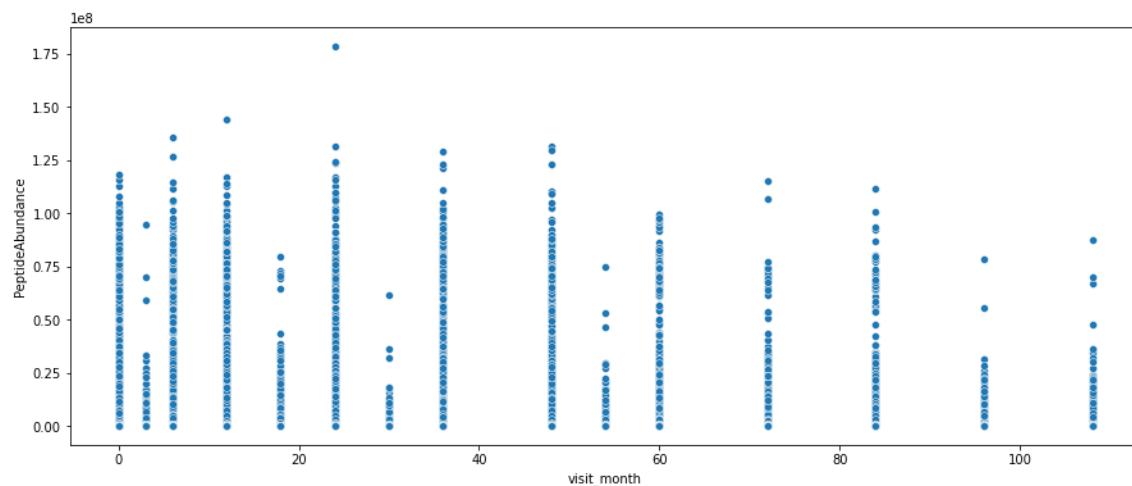
In [27]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df['visit_month'], y = df['PeptideAbundance'], palette = 'hls')
plt.show()
```



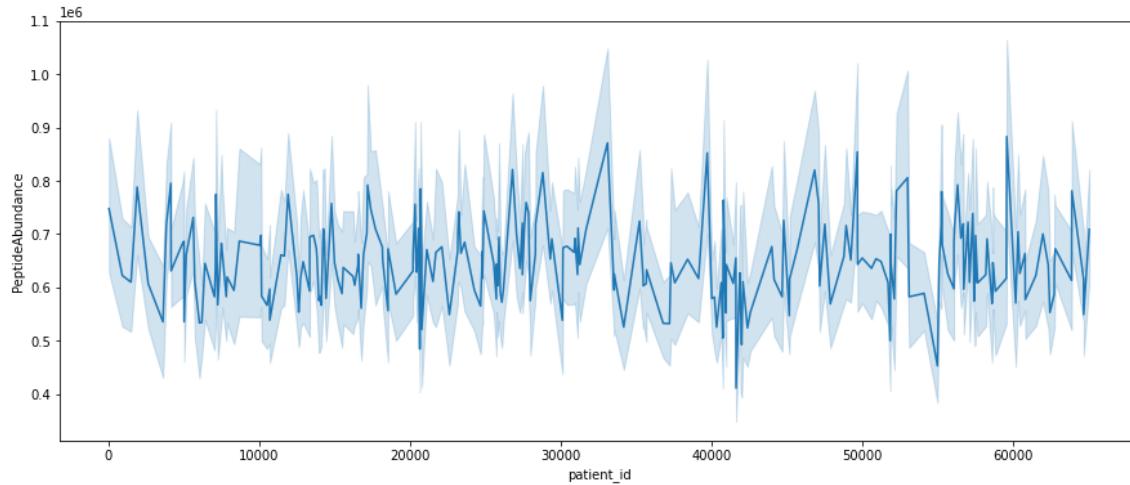
In [28]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df['visit_month'], y = df['PeptideAbundance'], palette = 'hls')
plt.show()
```



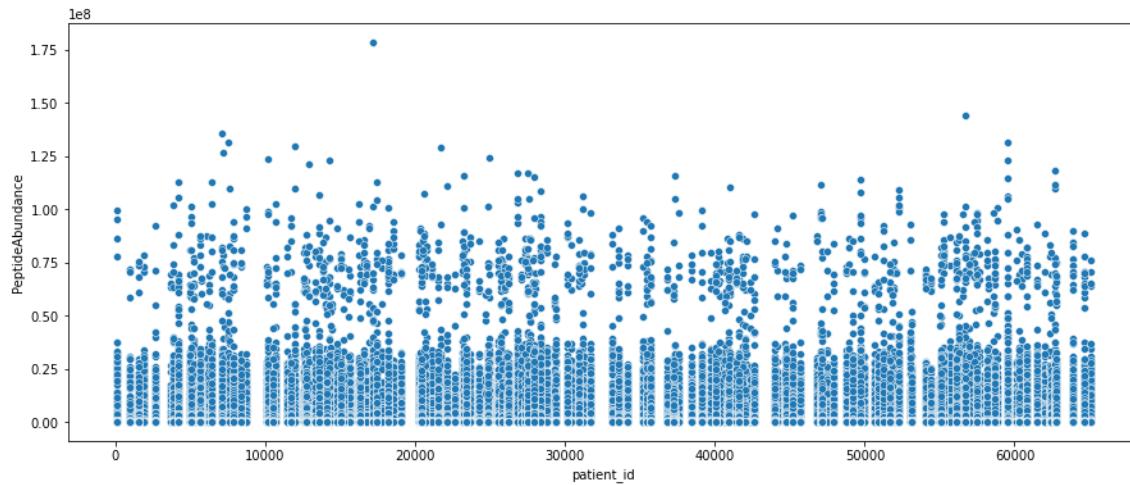
In [29]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df['patient_id'], y = df['PeptideAbundance'], palette = 'hls')
plt.show()
```



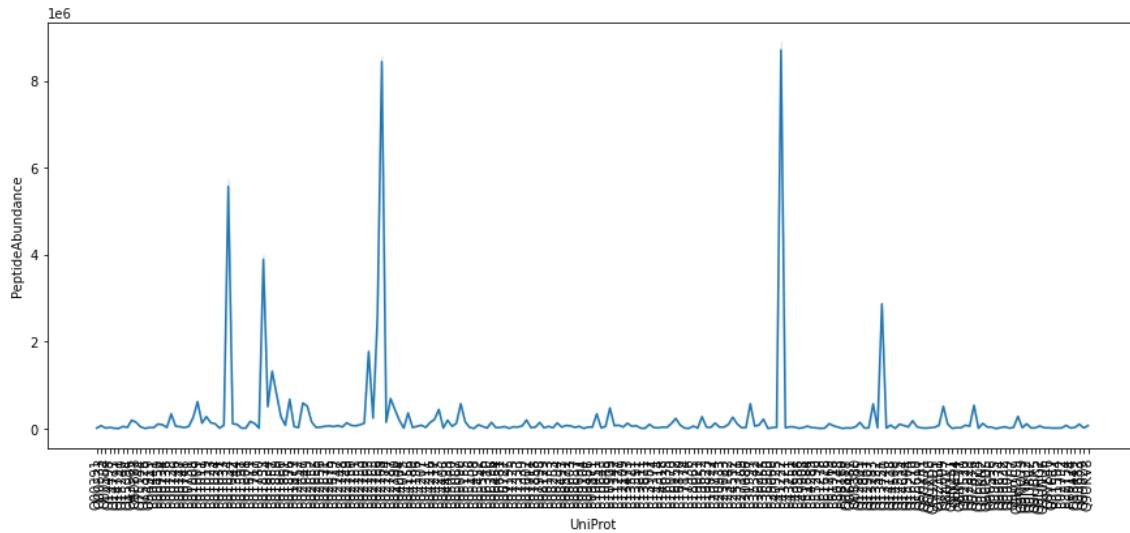
In [30]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df['patient_id'], y = df['PeptideAbundance'], palette = 'hls')
plt.show()
```



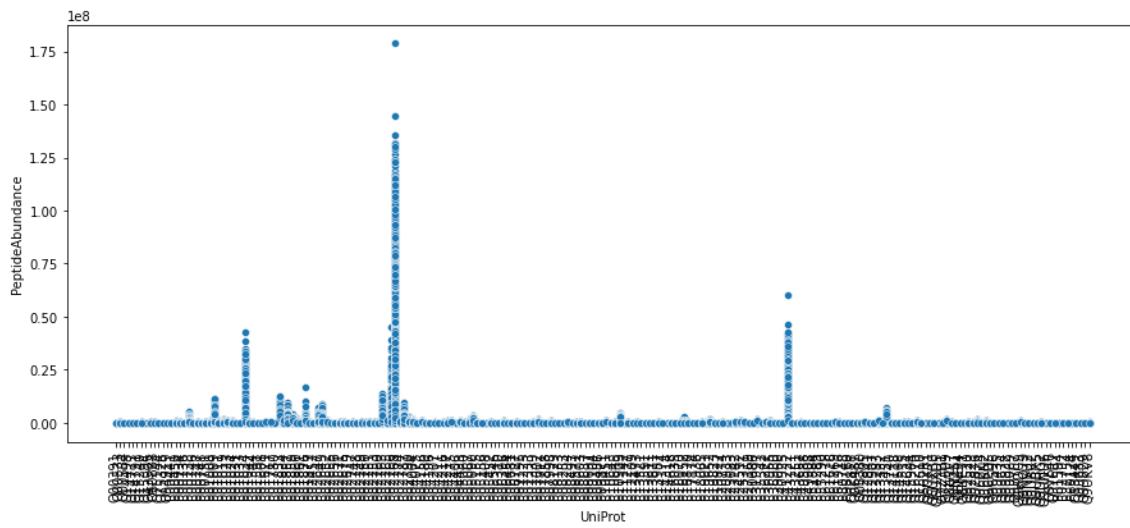
In [31]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df['UniProt'], y = df['PeptideAbundance'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



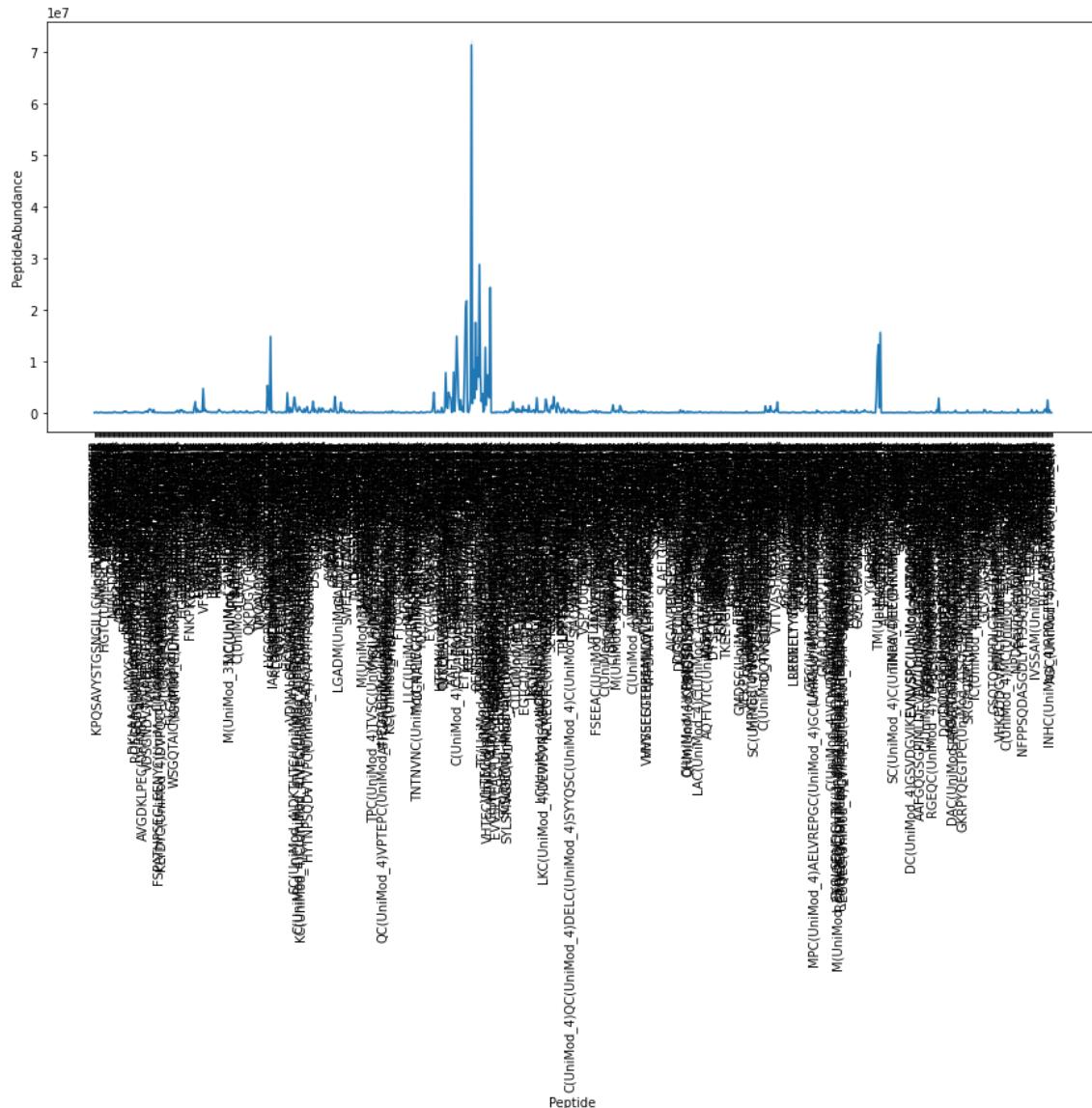
In [32]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df['UniProt'], y = df['PeptideAbundance'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



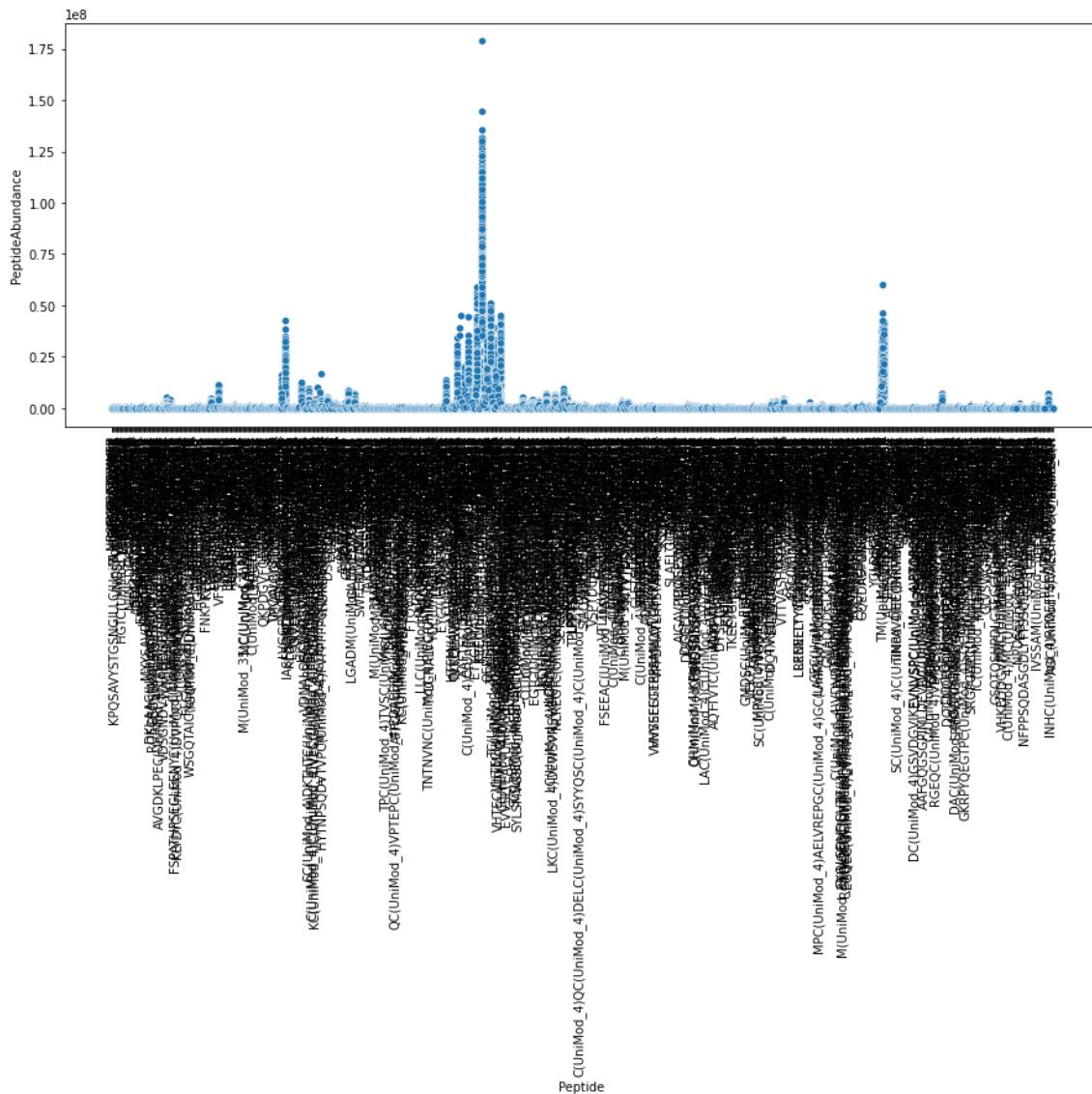
In [33]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df['Peptide'], y = df['PeptideAbundance'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



In [34]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df['Peptide'], y = df['PeptideAbundance'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



In [36]:

```
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3-Q1
```

In [37]:

```
print(IQR)
```

visit_month	42.0
patient_id	33429.0
PeptideAbundance	193164.5
dtype:	float64

In [38]:

```
df = df[~((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).any(axis=1)]
```

In [39]:

```
df.shape
```

Out[39]:

```
(847363, 6)
```

In [40]:

```
df_corr = df.corr()
```

In [41]:

```
df_corr
```

Out[41]:

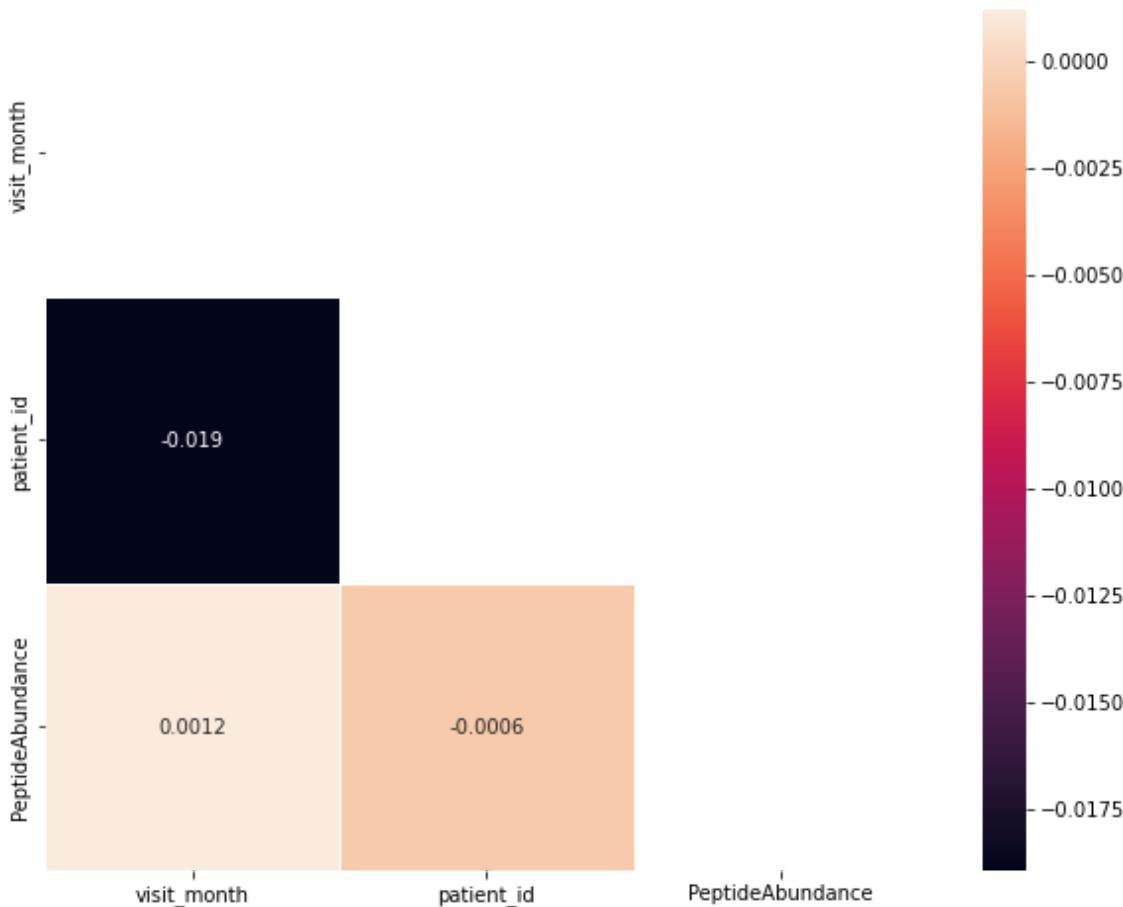
	visit_month	patient_id	PeptideAbundance
visit_month	1.000000	-0.018923	0.001200
patient_id	-0.018923	1.000000	-0.000602
PeptideAbundance	0.001200	-0.000602	1.000000

In [42]:

```
import numpy as np
```

In [43]:

```
plt.figure(figsize=(10, 8))
matrix = np.triu(df_corr)
sns.heatmap(df_corr, annot=True, linewidth=.8, mask=matrix, cmap="rocket");
plt.show()
```



In [45]:

```
df1 = pd.read_csv('train_proteins.csv')
```

In [46]:

```
df1.head()
```

Out[46]:

	visit_id	visit_month	patient_id	UniProt	NPX
0	55_0	0	55	O00391	11254.3
1	55_0	0	55	O00533	732430.0
2	55_0	0	55	O00584	39585.8
3	55_0	0	55	O14498	41526.9
4	55_0	0	55	O14773	31238.0

In [47]:

```
df1.tail()
```

Out[47]:

	visit_id	visit_month	patient_id	UniProt	NPX
232736	58648_108	108	58648	Q9UBX5	27387.8
232737	58648_108	108	58648	Q9UHG2	369437.0
232738	58648_108	108	58648	Q9UKV8	105830.0
232739	58648_108	108	58648	Q9Y646	21257.6
232740	58648_108	108	58648	Q9Y6R7	17953.1

In [48]:

```
df1.shape
```

Out[48]:

```
(232741, 5)
```

In [49]:

```
df1.columns
```

Out[49]:

```
Index(['visit_id', 'visit_month', 'patient_id', 'UniProt', 'NPX'], dtype='object')
```

In [50]:

```
df1.duplicated().sum()
```

Out[50]:

```
0
```

In [51]:

```
df1.isnull().sum()
```

Out[51]:

```
visit_id      0
visit_month   0
patient_id    0
UniProt       0
NPX          0
dtype: int64
```

In [52]:

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 232741 entries, 0 to 232740
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   visit_id    232741 non-null   object  
 1   visit_month 232741 non-null   int64   
 2   patient_id  232741 non-null   int64   
 3   UniProt     232741 non-null   object  
 4   NPX         232741 non-null   float64 
dtypes: float64(1), int64(2), object(2)
memory usage: 8.9+ MB
```

In [53]:

```
df1.describe()
```

Out[53]:

	visit_month	patient_id	NPX
count	232741.000000	232741.000000	2.327410e+05
mean	26.099205	32593.881873	2.712077e+06
std	22.874719	18608.479506	2.224155e+07
min	0.000000	55.000000	8.460820e+01
25%	6.000000	16566.000000	2.946440e+04
50%	24.000000	29313.000000	1.135560e+05
75%	48.000000	49995.000000	5.638940e+05
max	108.000000	65043.000000	6.138510e+08

In [54]:

```
df1.nunique()
```

Out[54]:

```
visit_id      1113
visit_month    15
patient_id    248
UniProt       227
NPX          218795
dtype: int64
```

In [55]:

```
df1['visit_month'].unique()
```

Out[55]:

```
array([ 0,  3,  6, 12, 18, 24, 30, 36, 48, 54, 60, 72, 84,
       96, 108], dtype=int64)
```

In [56]:

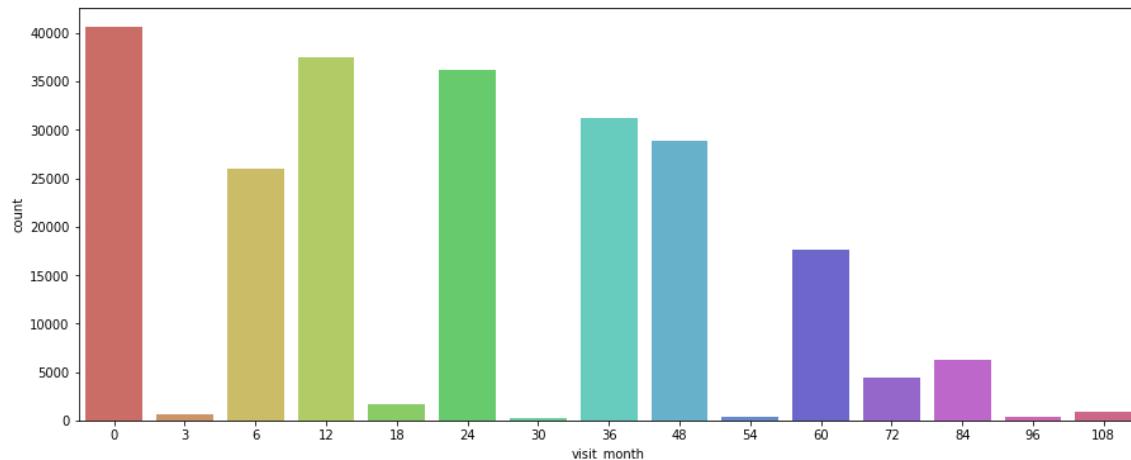
```
df1['visit_month'].value_counts()
```

Out[56]:

```
0      40587  
12     37467  
24     36225  
36     31156  
48     28838  
6      25991  
60     17572  
84     6288  
72     4407  
18     1657  
108    855  
3      641  
96     423  
54     418  
30     216  
Name: visit_month, dtype: int64
```

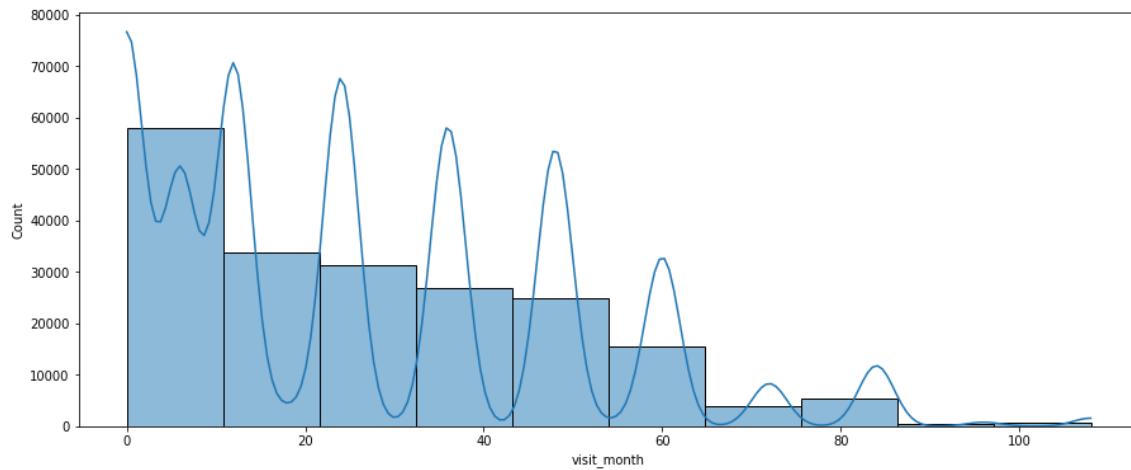
In [57]:

```
plt.figure(figsize=(15,6))  
sns.countplot('visit_month', data = df1, palette = 'hls')  
plt.show()
```



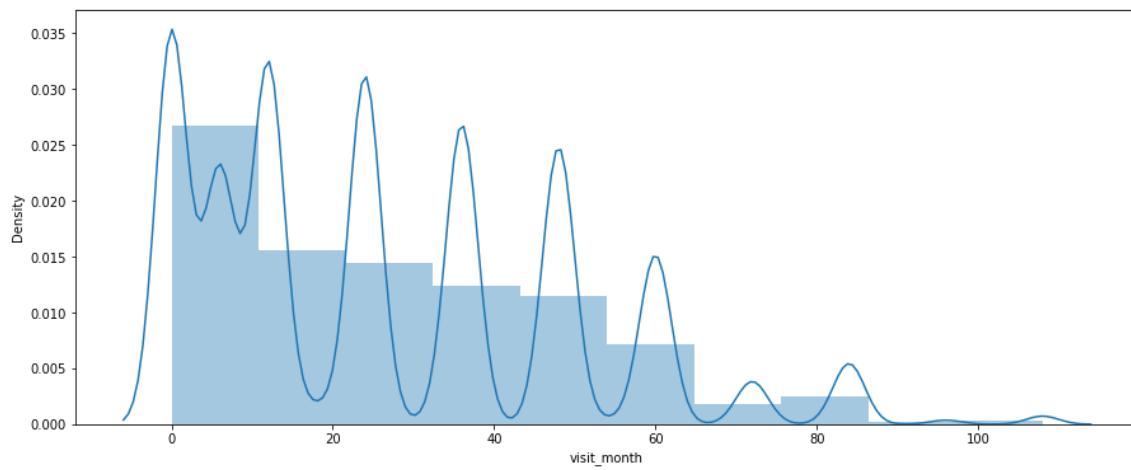
In [274]:

```
plt.figure(figsize=(15,6))
sns.histplot(df1['visit_month'], kde = True, bins = 10, palette = 'hls')
plt.show()
```



In [275]:

```
plt.figure(figsize=(15,6))
sns.distplot(df1['visit_month'], kde = True, bins = 10)
plt.show()
```



In [58]:

```
df1['patient_id'].unique()
```

Out[58]:

```
array([ 55, 1517, 1923, 2660, 3636, 3863, 4161, 4172, 5027,
       5178, 5645, 5742, 6054, 6211, 7051, 7117, 7568, 7832,
       8699, 10053, 10174, 10541, 10715, 10718, 11459, 11686, 11928,
      12516, 12636, 12703, 12755, 13368, 13618, 13968, 14035, 14124,
      14242, 14450, 14811, 15009, 15504, 15590, 16238, 16347, 16566,
      16574, 17154, 17414, 17727, 18183, 18204, 18553, 18560, 19088,
     20212, 20216, 20352, 20404, 20664, 20791, 21126, 21537, 21729,
     22126, 22623, 23192, 23244, 23391, 23636, 24278, 24690, 24820,
     24911, 25562, 25739, 25750, 25827, 25911, 26005, 26104, 26210,
    26809, 27079, 27300, 27464, 27468, 27715, 27971, 28327, 28342,
    28818, 29313, 29417, 30119, 30155, 30416, 30894, 30951, 31121,
   31154, 31693, 33548, 33558, 34182, 35231, 35465, 35477, 35675,
   35696, 36797, 37312, 37566, 38419, 39144, 40022, 40200, 40340,
   40751, 40874, 40967, 41444, 41617, 41628, 41871, 41883, 41930,
   42003, 42385, 42579, 44001, 44154, 44682, 45161, 45181, 46837,
   47103, 47513, 47881, 48780, 48928, 49239, 49672, 49683, 49995,
   50611, 50907, 51243, 51879, 51893, 51899, 52266, 53103, 54095,
   54406, 55240, 55256, 55302, 55662, 56073, 56075, 56119, 56317,
   56675, 56691, 56727, 57009, 57321, 57416, 57478, 57507, 57646,
   58189, 58270, 58648, 58653, 58823, 59550, 59574, 60170, 60326,
   60443, 60803, 61503, 61974, 62329, 62437, 62723, 62732, 62792,
   63875, 63889, 64669, 64674, 65043, 23175, 56538, 57468, 942,
   5036, 7151, 8344, 13360, 13852, 14270, 15245, 16778, 16931,
  20581, 20707, 24818, 31270, 33108, 37220, 40798, 44789, 47171,
  52119, 55096, 57108, 58674, 60788, 4923, 6420, 7265, 7886,
  10138, 13804, 14344, 17201, 20460, 20792, 27607, 27872, 27893,
  27987, 39719, 40650, 40876, 40980, 42086, 45662, 51708, 51846,
  52998, 54979, 58597, 7508, 12931], dtype=int64)
```

In [59]:

```
df1['patient_id'].value_counts()
```

Out[59]:

```
62723    1929
20404    1924
23391    1919
26210    1912
18183    1899
...
55240    429
42086    427
25911    419
52119    417
41930    199
Name: patient_id, Length: 248, dtype: int64
```

In [60]:

```
df1['UniProt'].unique()
```

Out[60]:

```
array(['000391', '000533', '000584', '014498', '014773', '014791',
       '015240', '015394', '043505', '060888', '075144', '075326',
       '094919', 'P00441', 'P00450', 'P00734', 'P00736', 'P00738',
       'P00746', 'P00747', 'P00748', 'P00751', 'P01008', 'P01009',
       'P01011', 'P01019', 'P01023', 'P01024', 'P01031', 'P01033',
       'P01034', 'P01042', 'P01344', 'P01591', 'P01608', 'P01621',
       'P01717', 'P01780', 'P01833', 'P01834', 'P01857', 'P01859',
       'P01860', 'P01861', 'P01876', 'P01877', 'P02452', 'P02647',
       'P02649', 'P02652', 'P02655', 'P02656', 'P02671', 'P02675',
       'P02679', 'P02747', 'P02748', 'P02749', 'P02750', 'P02751',
       'P02753', 'P02760', 'P02763', 'P02765', 'P02766', 'P02768',
       'P02774', 'P02787', 'P02790', 'P04004', 'P04075', 'P04156',
       'P04180', 'P04196', 'P04207', 'P04211', 'P04216', 'P04217',
       'P04275', 'P04406', 'P04433', 'P05060', 'P05067', 'P05090',
       'P05155', 'P05156', 'P05408', 'P05452', 'P05546', 'P06310',
       'P06396', 'P06454', 'P06681', 'P06727', 'P07195', 'P07225',
       'P07333', 'P07339', 'P07602', 'P07711', 'P07858', 'P07998',
       'P08123', 'P08133', 'P08253', 'P08294', 'P08493', 'P08571',
       'P08603', 'P08637', 'P08697', 'P09104', 'P09486', 'P09871',
       'P10451', 'P10643', 'P10645', 'P10909', 'P11142', 'P11277',
       'P12109', 'P13473', 'P13521', 'P13591', 'P13611', 'P13671',
       'P13987', 'P14174', 'P14314', 'P14618', 'P16035', 'P16070',
       'P16152', 'P16870', 'P17174', 'P17936', 'P18065', 'P19021',
       'P19652', 'P19823', 'P19827', 'P20774', 'P20933', 'P23083',
       'P23142', 'P24592', 'P25311', 'P27169', 'P30086', 'P31997',
       'P35542', 'P36222', 'P36955', 'P36980', 'P39060', 'P40925',
       'P41222', 'P43121', 'P43251', 'P43652', 'P49588', 'P49908',
       'P51884', 'P54289', 'P55290', 'P61278', 'P61626', 'P61769',
       'P61916', 'P80748', 'P98160', 'Q02818', 'Q06481', 'Q08380',
       'Q12805', 'Q12841', 'Q12907', 'Q13283', 'Q13332', 'Q13451',
       'Q13740', 'Q14118', 'Q14508', 'Q14515', 'Q14624', 'Q15904',
       'Q16270', 'Q16610', 'Q562R1', 'Q6UX71', 'Q6UXB8', 'Q6UXD5',
       'Q7Z3B1', 'Q7Z5P9', 'Q8IWV7', 'Q8N2S1', 'Q8NBJ4', 'Q8NE71',
       'Q92520', 'Q92823', 'Q92876', 'Q96BZ4', 'Q96KN2', 'Q96PD5',
       'Q96S96', 'Q99435', 'Q99674', 'Q99832', 'Q99969', 'Q9BY67',
       'Q9HDC9', 'Q9NQ79', 'Q9NYU2', 'Q9UBR2', 'Q9UBX5', 'Q9UHG2',
       'Q9UNU6', 'Q9Y646', 'Q9Y6R7', 'P01594', 'P02792', 'P32754',
       'P60174', 'Q13449', 'Q99683', 'Q99829', 'Q9UKV8'], dtype=object)
```

In [61]:

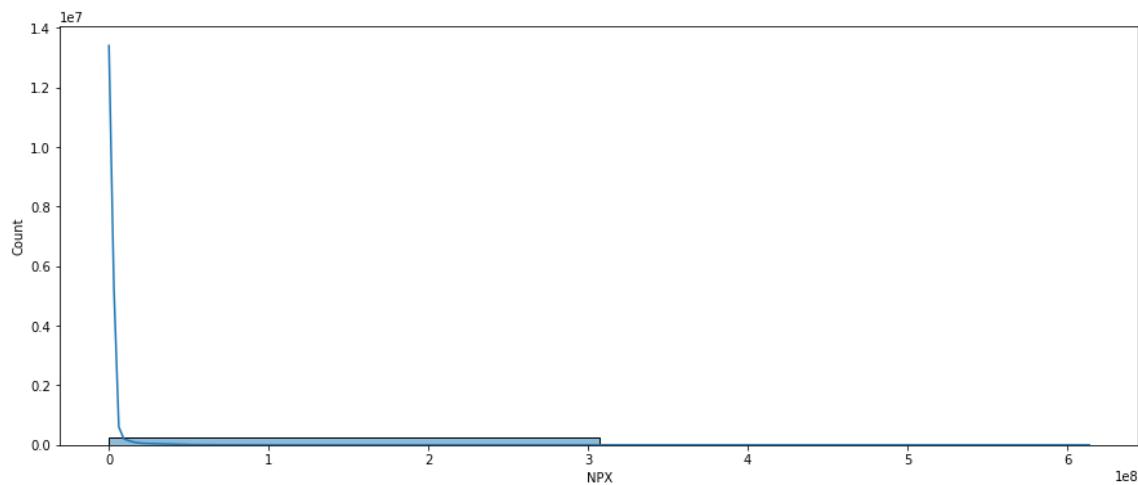
```
df1['UniProt'].value_counts()
```

Out[61]:

```
P01024    1113  
P05090    1113  
P01011    1113  
P01023    1113  
Q92520    1113  
...  
Q6UX71    661  
P01780    654  
Q562R1    616  
Q99832    606  
Q99829    489  
Name: UniProt, Length: 227, dtype: int64
```

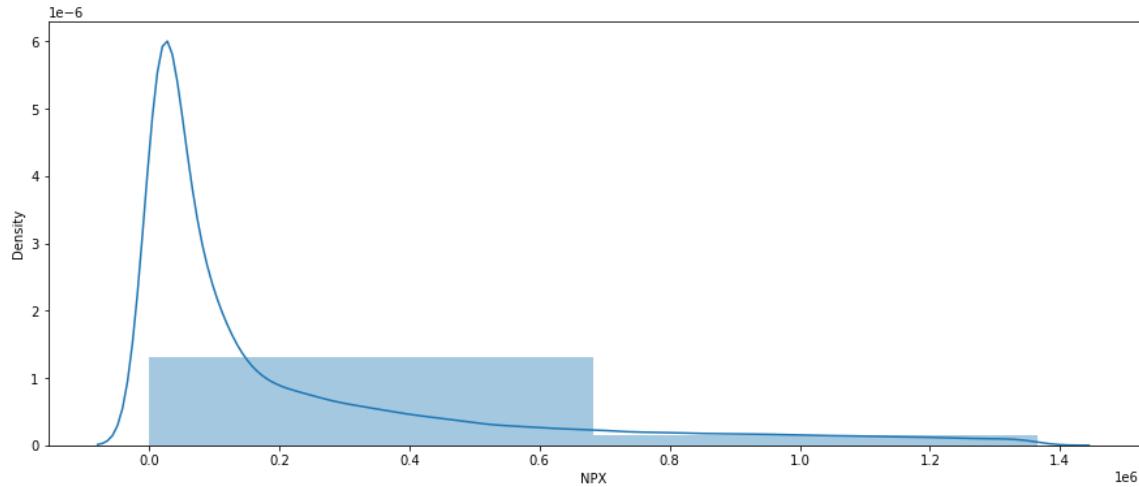
In [68]:

```
plt.figure(figsize=(15,6))  
sns.histplot(df1['NPX'], kde = True, bins = 2, palette = 'hls')  
plt.show()
```



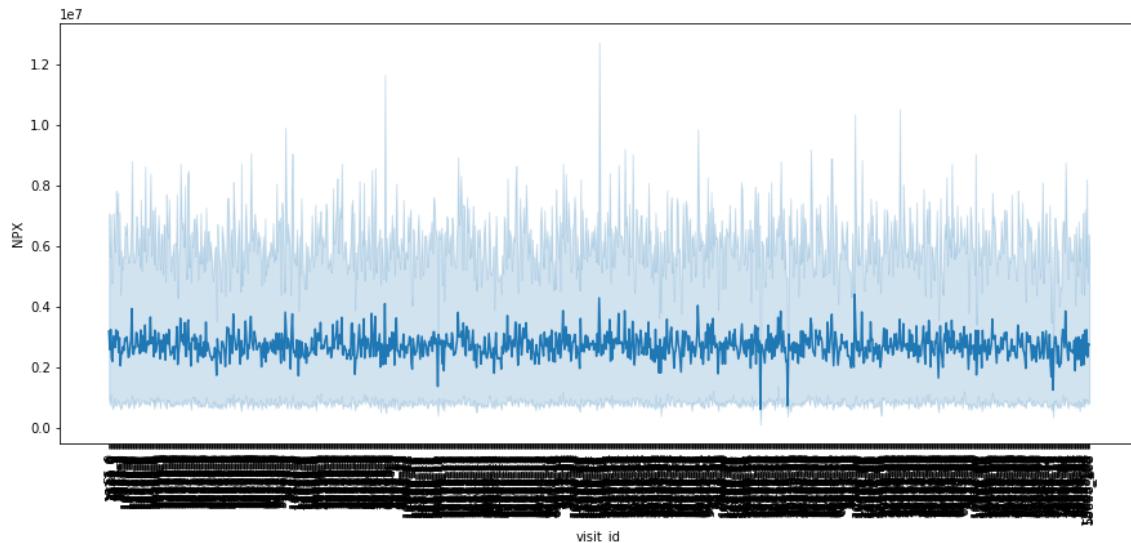
In [276]:

```
plt.figure(figsize=(15,6))
sns.distplot(df1['NPX'], kde = True, bins = 2)
plt.show()
```



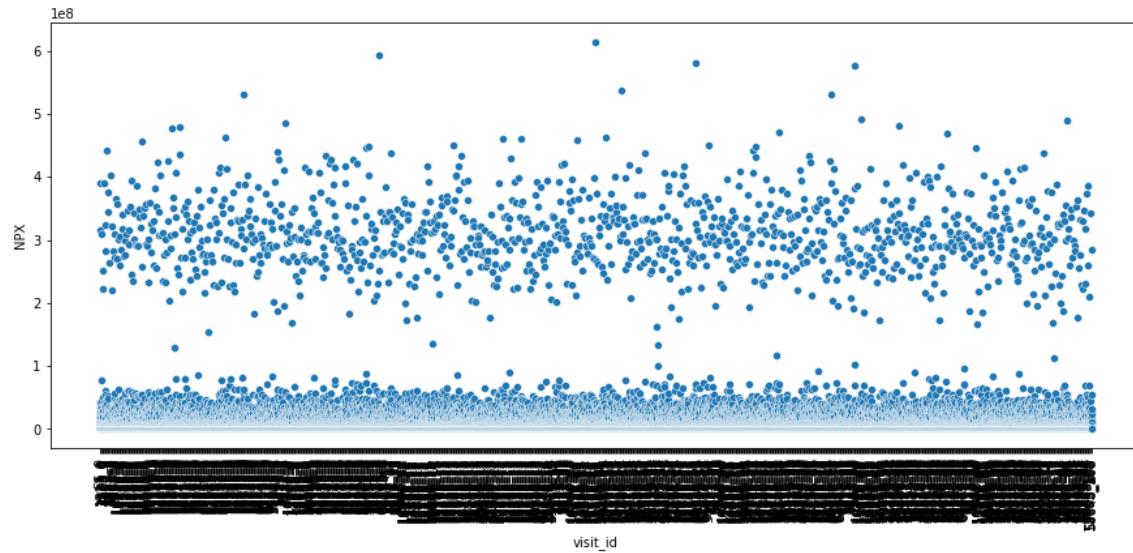
In [70]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df1['visit_id'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



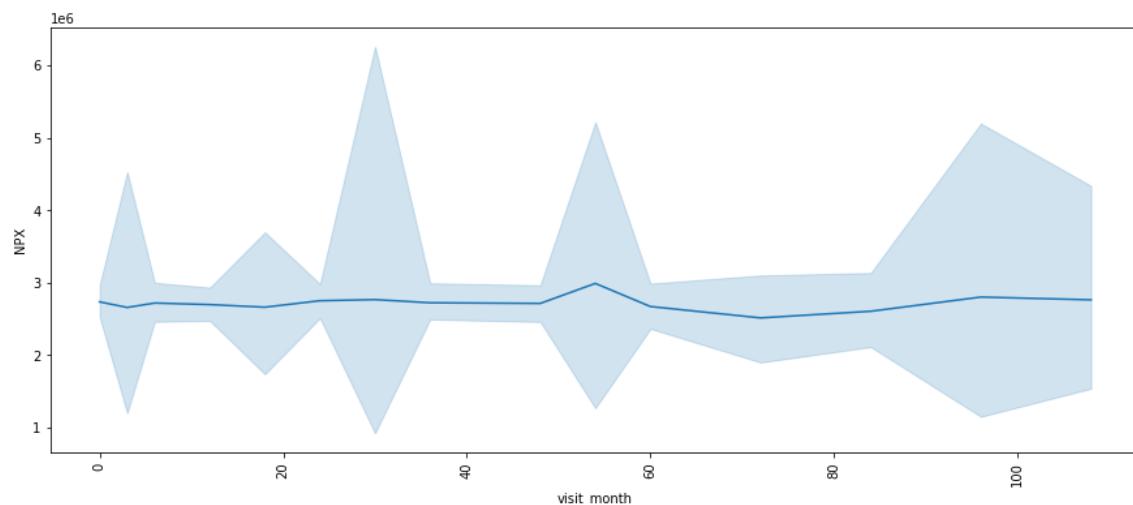
In [71]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df1['visit_id'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



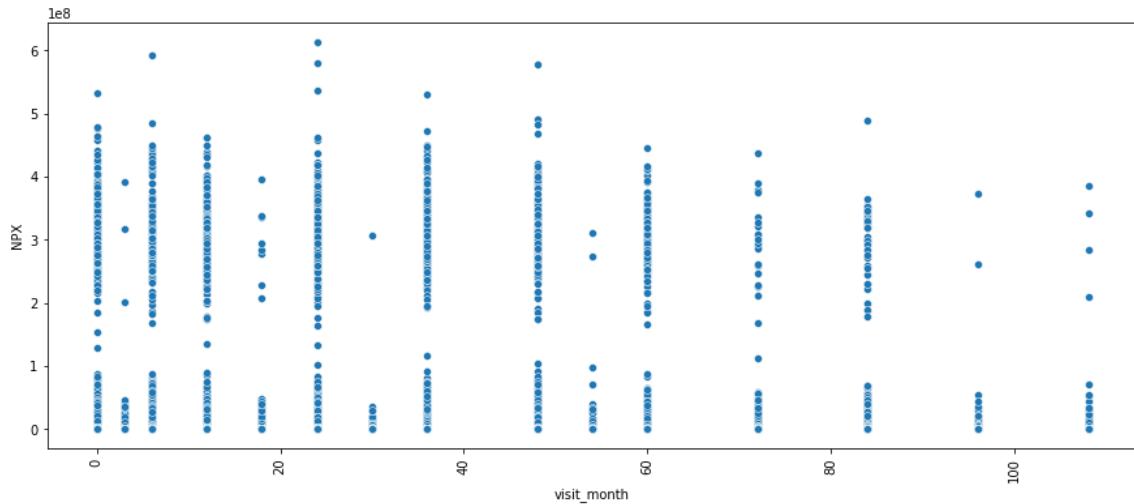
In [72]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df1['visit_month'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



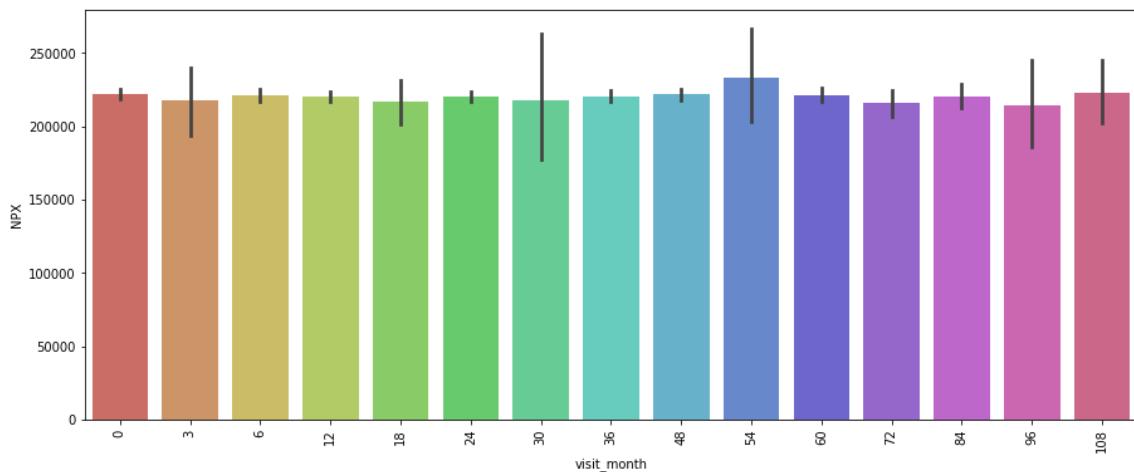
In [73]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df1['visit_month'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



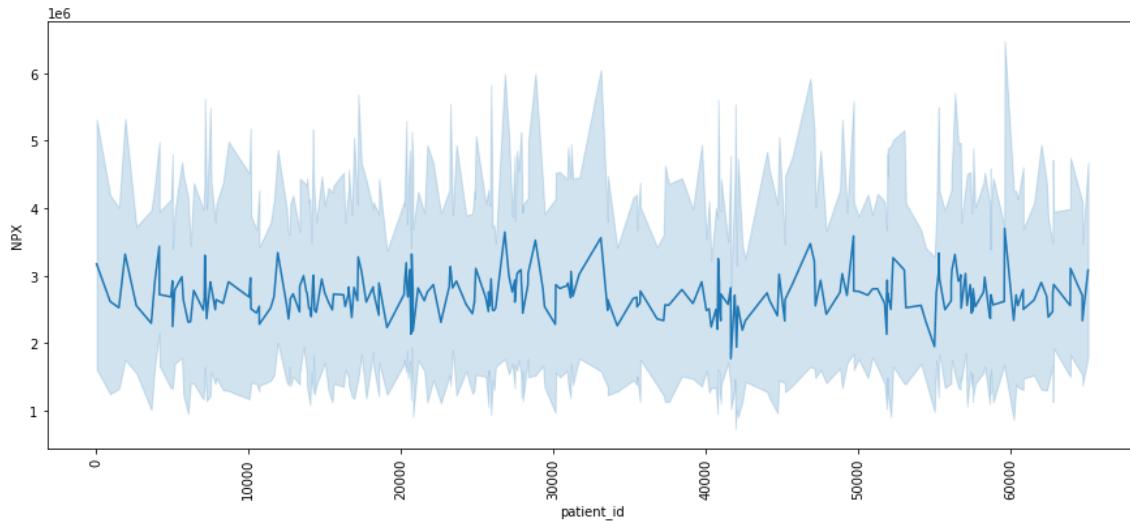
In [277]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df1['visit_month'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



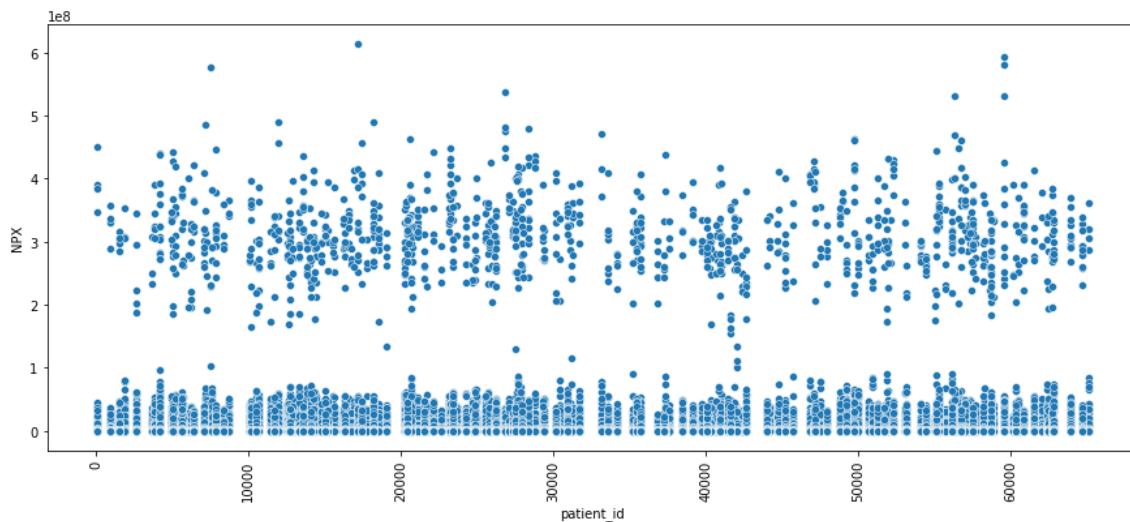
In [74]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df1['patient_id'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



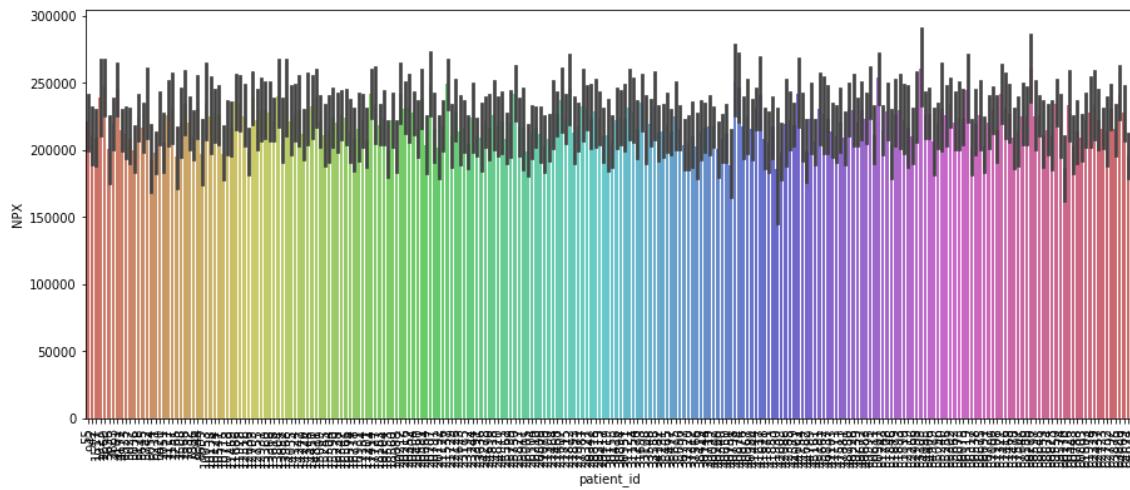
In [75]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df1['patient_id'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



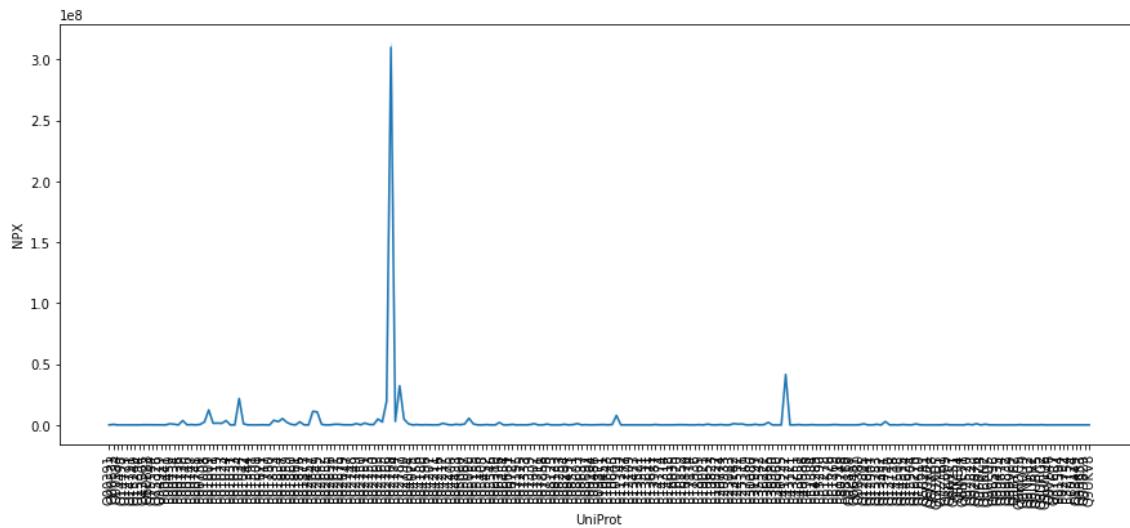
In [278]:

```
plt.figure(figsize=(15,6))
sns.barplot(x = df1['patient_id'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



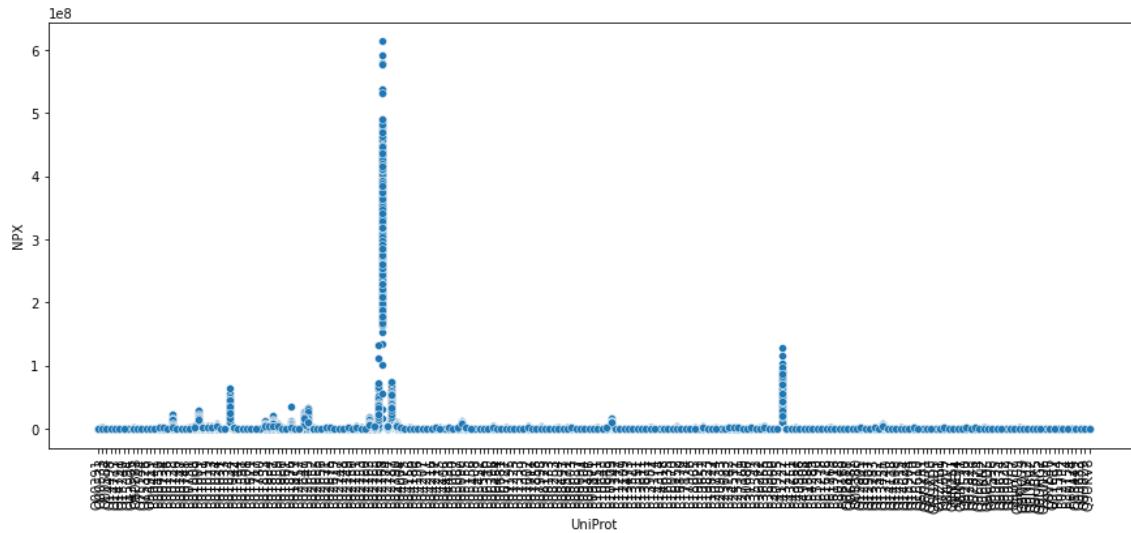
In [76]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df1['UniProt'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



In [77]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df1['UniProt'], y = df1['NPX'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



In [160]:

```
Q1 = df1.quantile(0.25)
Q3 = df1.quantile(0.75)
IQR = Q3-Q1
```

In [161]:

```
print(IQR)
```

visit_month	42.0
patient_id	33429.0
NPX	534429.6
dtype:	float64

In [162]:

```
df1 = df1[~((df1 < (Q1 - 1.5 * IQR)) | (df1 > (Q3 + 1.5 * IQR))).any(axis=1)]
```

In [163]:

```
df1.shape
```

Out[163]:

```
(200981, 5)
```

In [164]:

```
df1_corr = df1.corr()
```

In [165]:

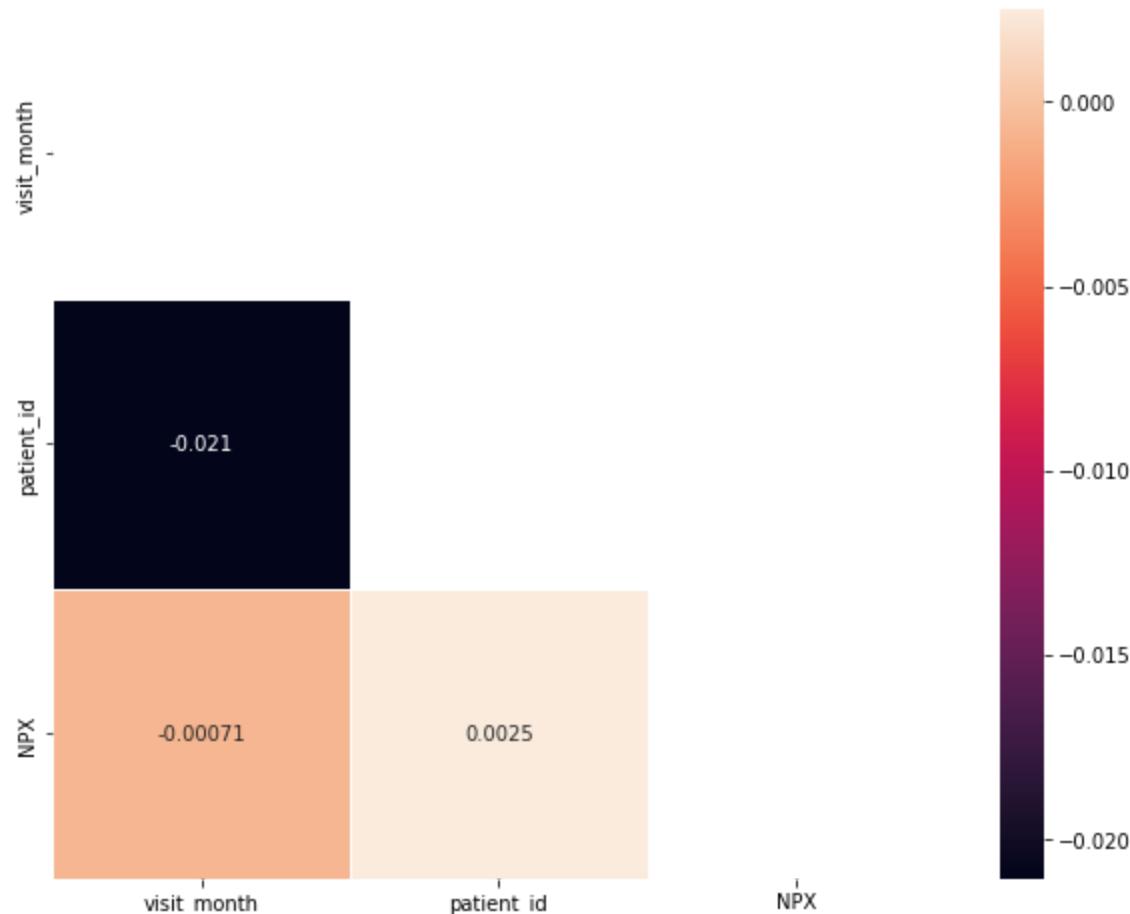
```
df1_corr
```

Out[165]:

	visit_month	patient_id	NPX
visit_month	1.000000	-0.021061	-0.000708
patient_id	-0.021061	1.000000	0.002519
NPX	-0.000708	0.002519	1.000000

In [166]:

```
plt.figure(figsize=(10, 8))
matrix = np.triu(df1_corr)
sns.heatmap(df1_corr, annot=True, linewidth=.8, mask=matrix, cmap="rocket");
plt.show()
```



In [78]:

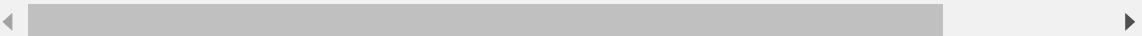
```
df2 = pd.read_csv('train_clinical_data.csv')
```

In [79]:

df2.head()

Out[79]:

	visit_id	patient_id	visit_month	updrs_1	updrs_2	updrs_3	updrs_4	upd23b_clinical_stat
0	55_0	55	0	10.0	6.0	15.0	NaN	
1	55_3	55	3	10.0	7.0	25.0	NaN	
2	55_6	55	6	8.0	10.0	34.0	NaN	
3	55_9	55	9	8.0	9.0	30.0	0.0	
4	55_12	55	12	10.0	10.0	41.0	0.0	



In [80]:

df2.tail()

Out[80]:

	visit_id	patient_id	visit_month	updrs_1	updrs_2	updrs_3	updrs_4	upd23b_clinical_stat
2610	65043_48	65043	48	7.0	6.0	13.0	0.0	
2611	65043_54	65043	54	4.0	8.0	11.0	1.0	
2612	65043_60	65043	60	6.0	6.0	16.0	1.0	
2613	65043_72	65043	72	3.0	9.0	14.0	1.0	
2614	65043_84	65043	84	7.0	9.0	20.0	3.0	



In [81]:

df2.shape

Out[81]:

(2615, 8)

In [82]:

df2.columns

Out[82]:

```
Index(['visit_id', 'patient_id', 'visit_month', 'updrs_1', 'updrs_2',
       'updrs_3', 'updrs_4', 'upd23b_clinical_state_on_medication'],
      dtype='object')
```

In [83]:

df2.duplicated().sum()

Out[83]:

0

In [84]:

```
df2.isnull().sum()
```

Out[84]:

```
visit_id          0  
patient_id        0  
visit_month       0  
updrs_1           1  
updrs_2           2  
updrs_3          25  
updrs_4          1038  
upd23b_clinical_state_on_medication  1327  
dtype: int64
```

In [85]:

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 2615 entries, 0 to 2614  
Data columns (total 8 columns):  
 #   Column           Non-Null Count  Dtype    
---  --    
 0   visit_id         2615 non-null    object   
 1   patient_id      2615 non-null    int64    
 2   visit_month     2615 non-null    int64    
 3   updrs_1          2614 non-null    float64  
 4   updrs_2          2613 non-null    float64  
 5   updrs_3          2590 non-null    float64  
 6   updrs_4          1577 non-null    float64  
 7   upd23b_clinical_state_on_medication 1288 non-null    object   
dtypes: float64(4), int64(2), object(2)  
memory usage: 163.6+ KB
```

In [86]:

```
df2.describe()
```

Out[86]:

	patient_id	visit_month	updrs_1	updrs_2	updrs_3	updrs_4
count	2615.000000	2615.000000	2614.000000	2613.000000	2590.000000	1577.000000
mean	32651.743786	31.190822	7.110559	6.74359	19.421236	1.861763
std	18535.758700	25.199053	5.525955	6.32323	15.000289	3.022112
min	55.000000	0.000000	0.000000	0.00000	0.000000	0.000000
25%	16574.000000	10.500000	3.000000	1.00000	6.000000	0.000000
50%	29417.000000	24.000000	6.000000	5.00000	19.000000	0.000000
75%	50611.000000	48.000000	10.000000	10.00000	29.000000	3.000000
max	65043.000000	108.000000	33.000000	40.00000	86.000000	20.000000

In [87]:

```
df2.nunique()
```

Out[87]:

visit_id	2615
patient_id	248
visit_month	17
updrs_1	32
updrs_2	36
updrs_3	72
updrs_4	19
upd23b_clinical_state_on_medication	2
dtype: int64	

In [89]:

```
mean_value = df2['updrs_1'].mean()
```

In [90]:

```
df2['updrs_1'] = df2['updrs_1'].replace({pd.np.nan: mean_value})
```

In [91]:

```
mean_value = df2['updrs_2'].mean()
```

In [92]:

```
df2['updrs_2'] = df2['updrs_2'].replace({pd.np.nan: mean_value})
```

In [93]:

```
mean_value = df2['updrs_3'].mean()
```

In [94]:

```
df2['updrs_3'] = df2['updrs_3'].replace({pd.np.nan: mean_value})
```

In [95]:

```
mean_value = df2['updrs_4'].mean()
```

In [96]:

```
df2['updrs_4'] = df2['updrs_4'].replace({pd.np.nan: mean_value})
```

In [97]:

```
df2['upd23b_clinical_state_on_medication'].fillna('Not Available', inplace=True)
```

In [98]:

```
df2.isnull().sum()
```

Out[98]:

```
visit_id          0  
patient_id       0  
visit_month      0  
updrs_1          0  
updrs_2          0  
updrs_3          0  
updrs_4          0  
upd23b_clinical_state_on_medication 0  
dtype: int64
```

In [279]:

```
df2['visit_id'].unique()
```

Out[279]:

```
array(['55_0', '55_3', '55_6', ..., '65043_60', '65043_72', '65043_84'],  
      dtype=object)
```

In [100]:

```
df2['patient_id'].unique()
```

Out[100]:

```
array([ 55,   942,  1517,  1923,  2660,  3636,  3863,  4161,  4172,  
     4923,  5027,  5036,  5178,  5645,  5742,  6054,  6211,  6420,  
     7051,  7117,  7151,  7265,  7508,  7568,  7832,  7886,  8344,  
     8699,  10053, 10138, 10174, 10541, 10715, 10718, 11459, 11686,  
    11928, 12516, 12636, 12703, 12755, 12931, 13360, 13368, 13618,  
    13804, 13852, 13968, 14035, 14124, 14242, 14270, 14344, 14450,  
    14811, 15009, 15245, 15504, 15590, 16238, 16347, 16566, 16574,  
    16778, 16931, 17154, 17201, 17414, 17727, 18183, 18204, 18553,  
    18560, 19088, 20212, 20216, 20352, 20404, 20460, 20581, 20664,  
    20707, 20791, 20792, 21126, 21537, 21729, 22126, 22623, 23175,  
    23192, 23244, 23391, 23636, 24278, 24690, 24818, 24820, 24911,  
    25562, 25739, 25750, 25827, 25911, 26005, 26104, 26210, 26809,  
    27079, 27300, 27464, 27468, 27607, 27715, 27872, 27893, 27971,  
    27987, 28327, 28342, 28818, 29313, 29417, 30119, 30155, 30416,  
    30894, 30951, 31121, 31154, 31270, 31693, 33108, 33548, 33558,  
    34182, 35231, 35465, 35477, 35675, 35696, 36797, 37220, 37312,  
    37566, 38419, 39144, 39719, 40022, 40200, 40340, 40650, 40751,  
    40798, 40874, 40876, 40967, 40980, 41444, 41617, 41628, 41871,  
    41883, 41930, 42003, 42086, 42385, 42579, 44001, 44154, 44682,  
    44789, 45161, 45181, 45662, 46837, 47103, 47171, 47513, 47881,  
    48780, 48928, 49239, 49672, 49683, 49995, 50611, 50907, 51243,  
    51708, 51846, 51879, 51893, 51899, 52119, 52266, 52998, 53103,  
    54095, 54406, 54979, 55096, 55240, 55256, 55302, 55662, 56073,  
    56075, 56119, 56317, 56538, 56675, 56691, 56727, 57009, 57108,  
    57321, 57416, 57468, 57478, 57507, 57646, 58189, 58270, 58597,  
    58648, 58653, 58674, 58823, 59550, 59574, 60170, 60326, 60443,  
    60788, 60803, 61503, 61974, 62329, 62437, 62723, 62732, 62792,  
    63875, 63889, 64669, 64674, 65043], dtype=int64)
```

In [101]:

```
df2['patient_id'].value_counts()
```

Out[101]:

```
55096    17  
57108    17  
15009    17  
16778    16  
52266    16  
..  
13968     5  
20792     5  
62732     4  
60443     3  
14450     3  
Name: patient_id, Length: 248, dtype: int64
```

In [102]:

```
df2['visit_month'].unique()
```

Out[102]:

```
array([  0,   3,   6,   9,  12,  18,  24,  30,  36,  42,  48,  54,  60,  
       72,  84,  96, 108], dtype=int64)
```

In [103]:

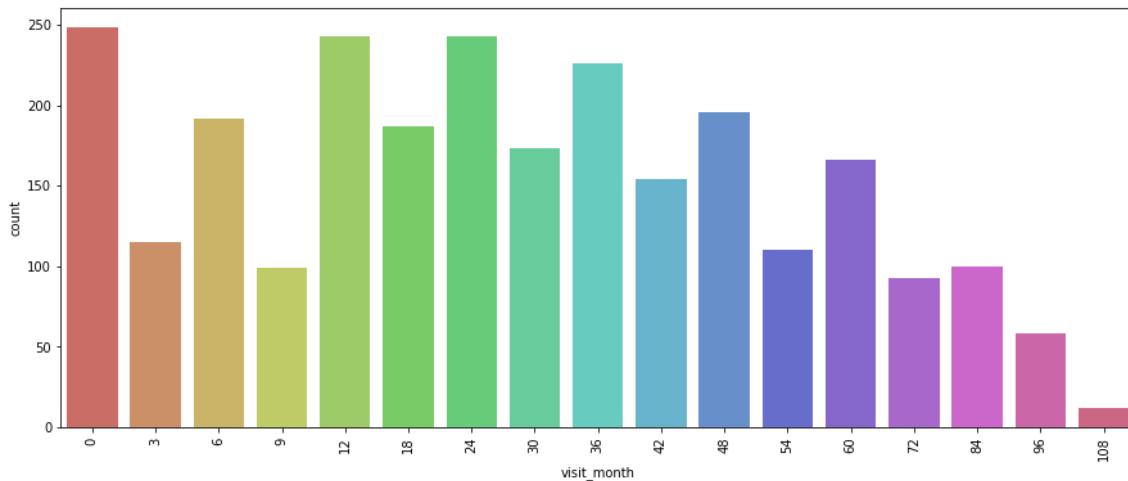
```
df2['visit_month'].value_counts()
```

Out[103]:

```
0      248  
12     243  
24     243  
36     226  
48     196  
6      192  
18     187  
30     173  
60     166  
42     154  
3      115  
54     110  
84     100  
9      99  
72     93  
96     58  
108    12  
Name: visit_month, dtype: int64
```

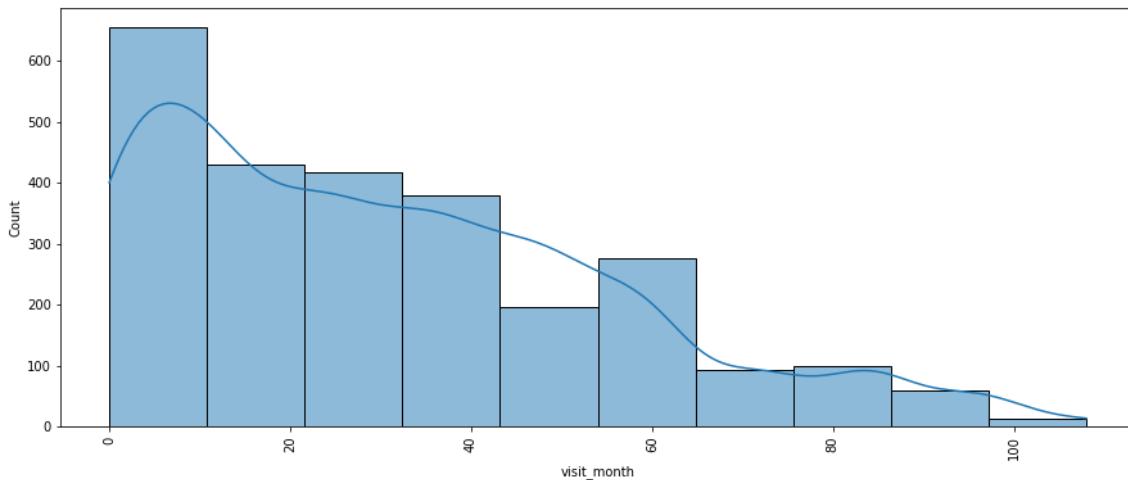
In [104]:

```
plt.figure(figsize=(15,6))
sns.countplot(x = df2['visit_month'], data = df2, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



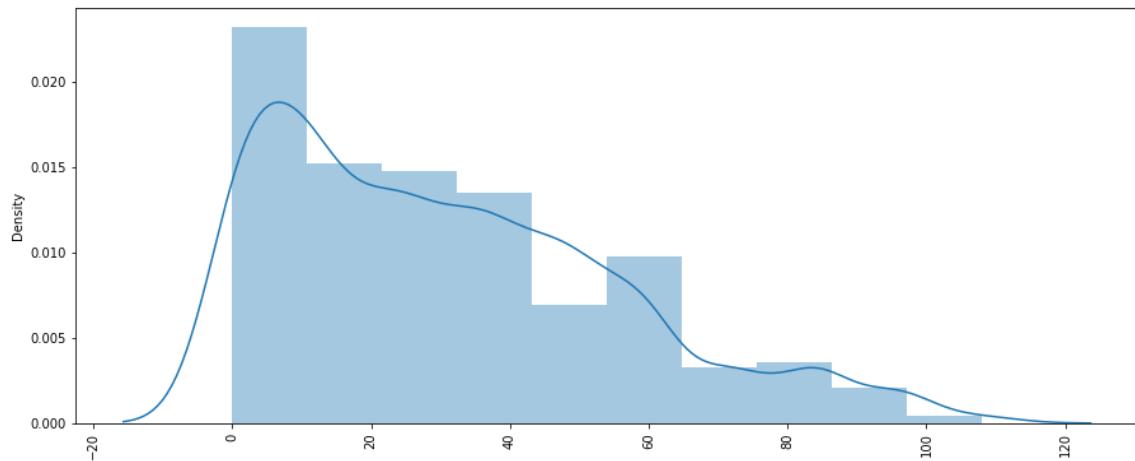
In [280]:

```
plt.figure(figsize=(15,6))
sns.histplot(x = df2['visit_month'], kde = True, bins = 10, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



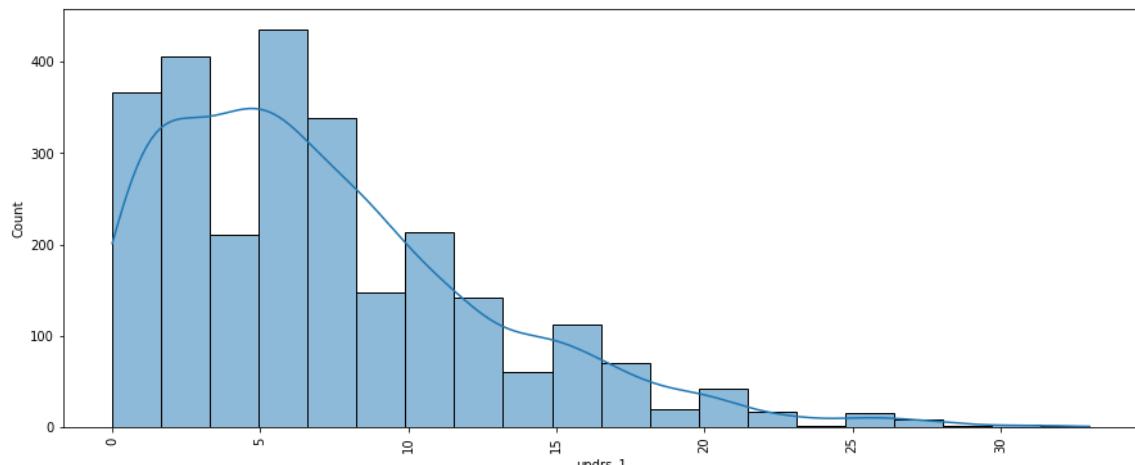
In [281]:

```
plt.figure(figsize=(15,6))
sns.distplot(x = df2['visit_month'], kde = True, bins = 10)
plt.xticks(rotation = 90)
plt.show()
```



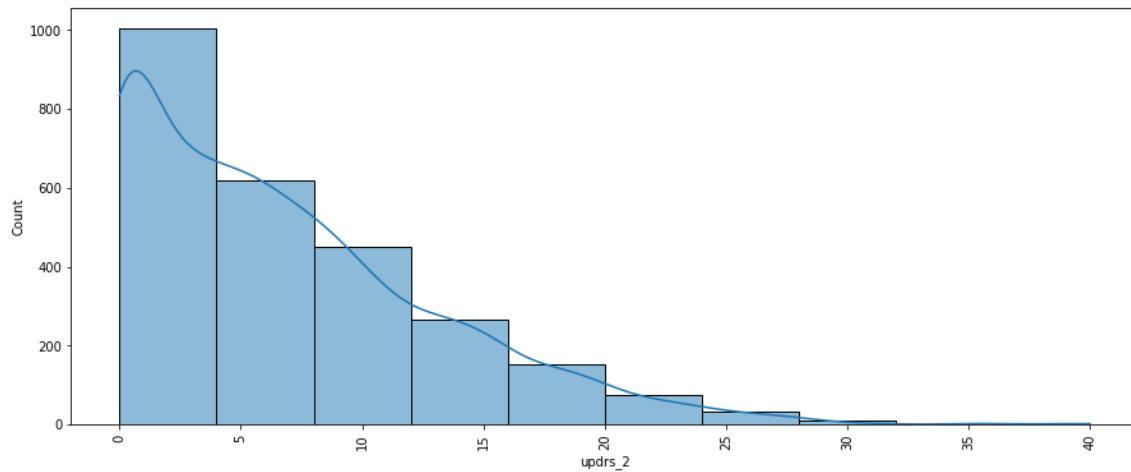
In [282]:

```
plt.figure(figsize=(15,6))
sns.histplot(x = df2['updrs_1'], kde = True, bins = 20, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



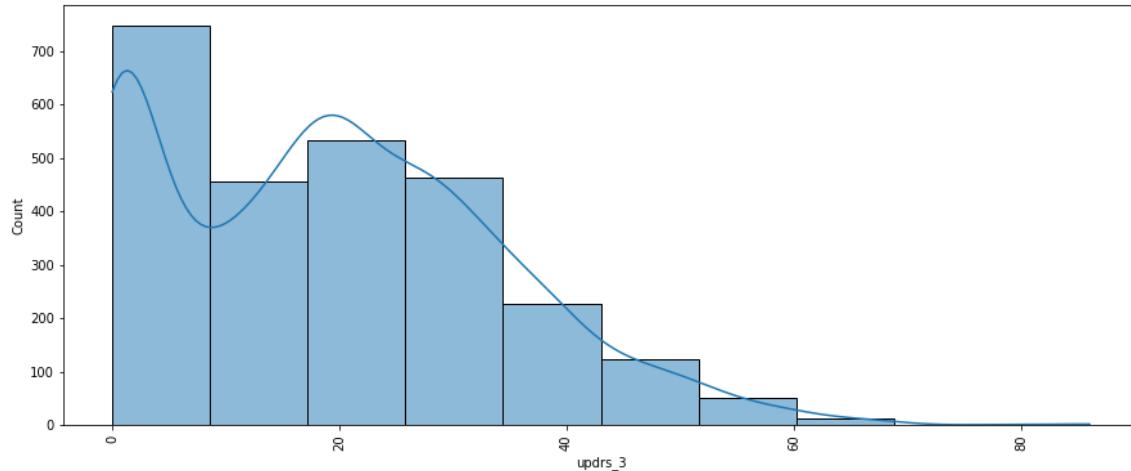
In [106]:

```
plt.figure(figsize=(15,6))
sns.histplot(x = df2['updrs_2'], kde = True, bins = 10, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



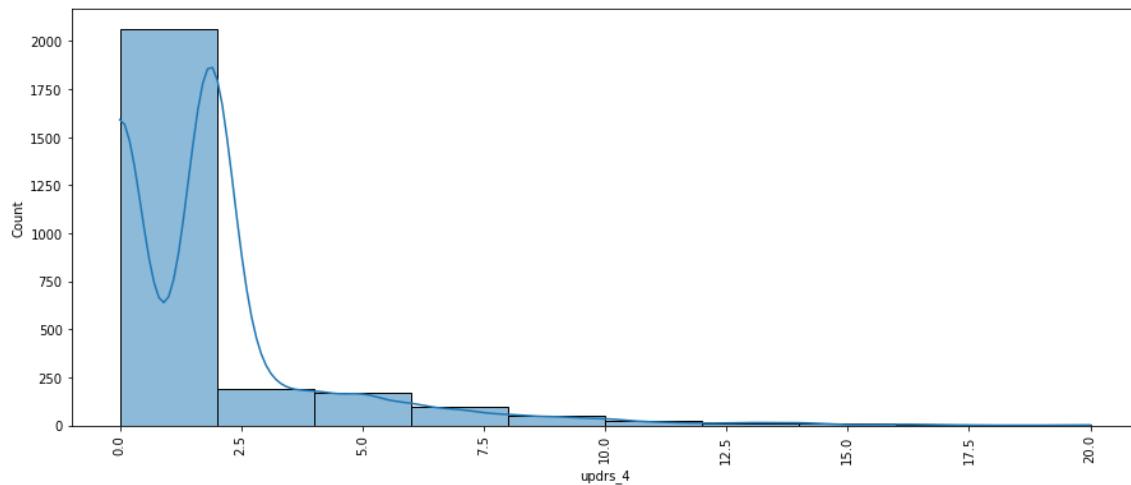
In [107]:

```
plt.figure(figsize=(15,6))
sns.histplot(x = df2['updrs_3'], kde = True, bins = 10, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



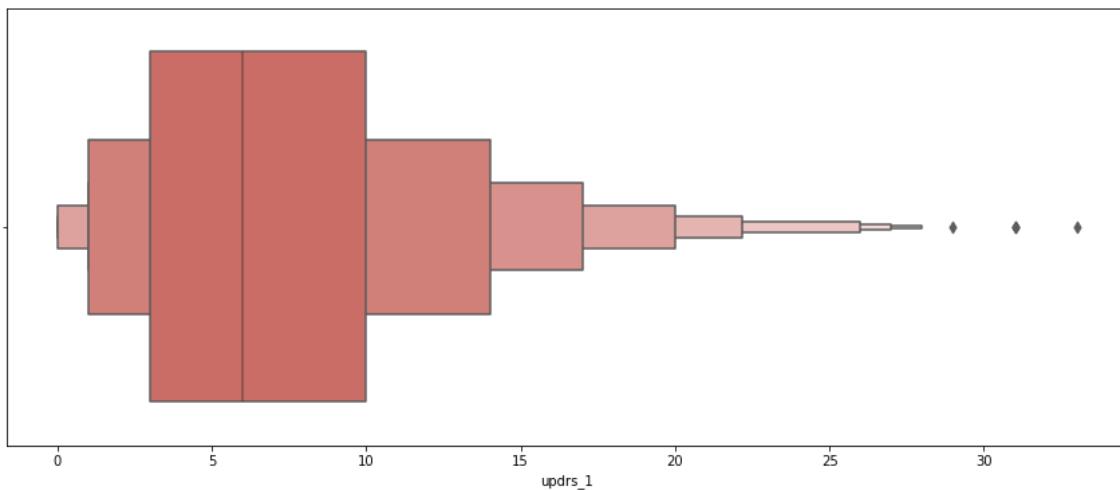
In [108]:

```
plt.figure(figsize=(15,6))
sns.histplot(x = df2['updrs_4'], kde = True, bins = 10, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



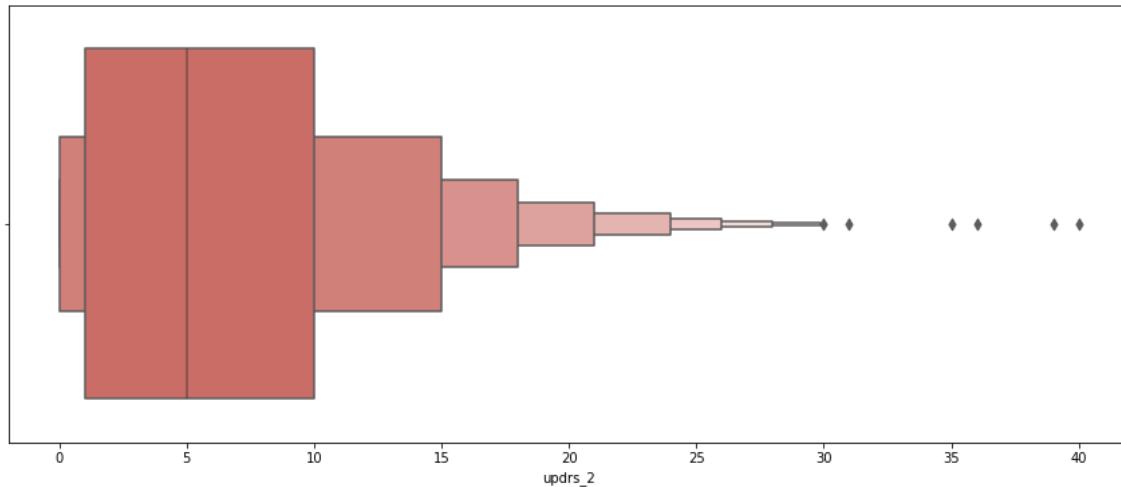
In [156]:

```
plt.figure(figsize=(15,6))
sns.boxenplot(x = df2['updrs_1'], data = df2, palette = 'hls')
plt.show()
```



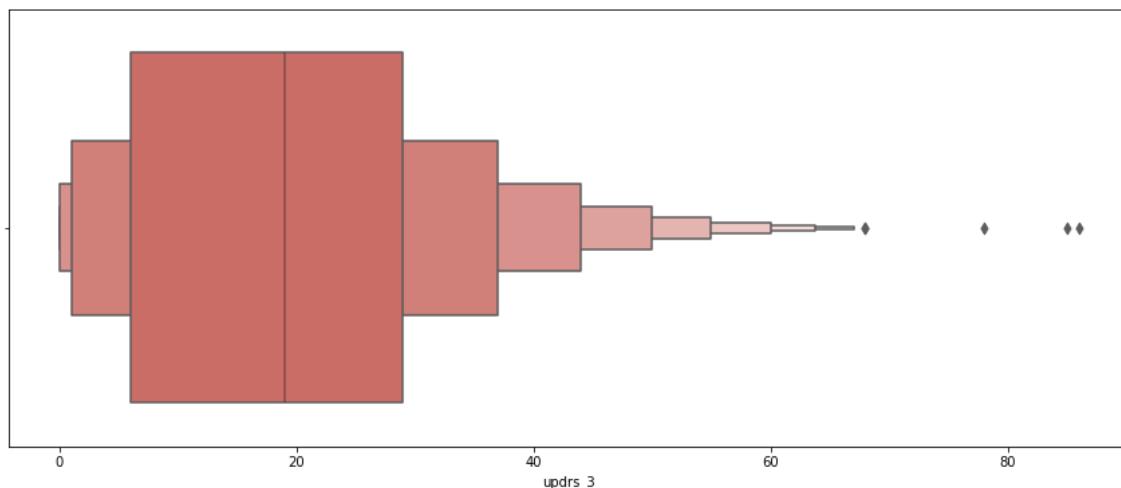
In [157]:

```
plt.figure(figsize=(15,6))
sns.boxenplot(x = df2['updrs_2'], data = df2, palette = 'hls')
plt.show()
```



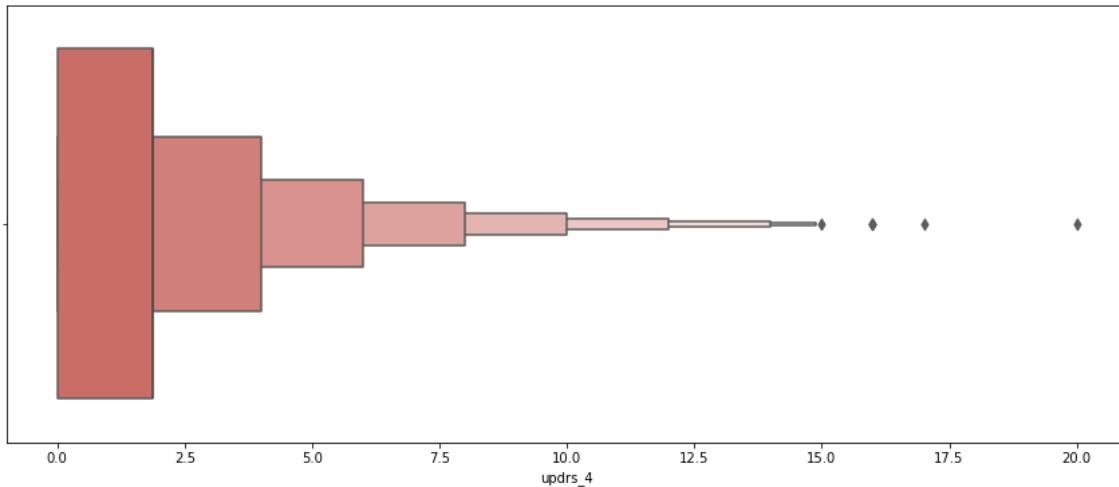
In [158]:

```
plt.figure(figsize=(15,6))
sns.boxenplot(x = df2['updrs_3'], data = df2, palette = 'hls')
plt.show()
```



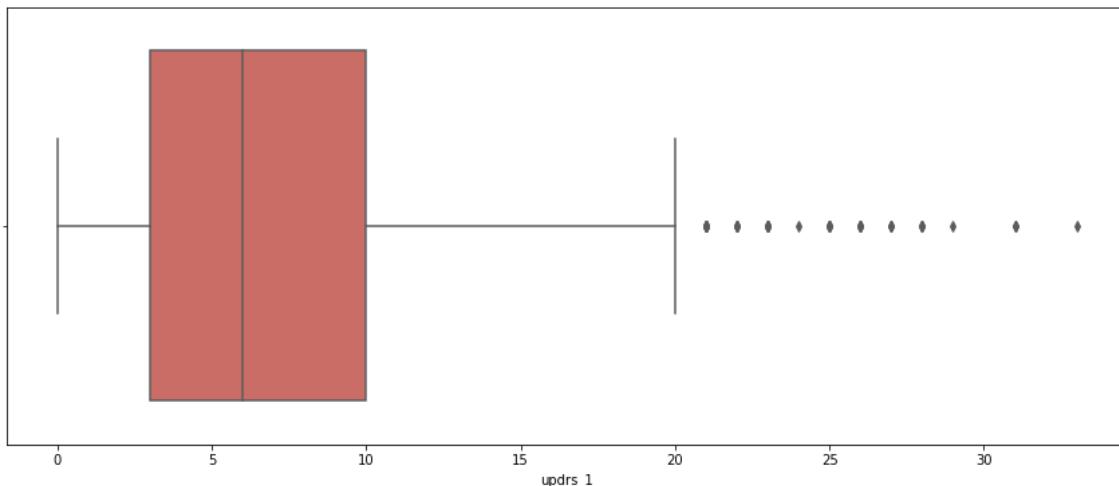
In [159]:

```
plt.figure(figsize=(15,6))
sns.boxenplot(x = df2['updrs_4'], data = df2, palette = 'hls')
plt.show()
```



In [283]:

```
plt.figure(figsize=(15,6))
sns.boxplot(x = df2['updrs_1'], data = df2, palette = 'hls')
plt.show()
```



In [109]:

```
df2['upd23b_clinical_state_on_medication'].unique()
```

Out[109]:

```
array(['Not Available', 'On', 'Off'], dtype=object)
```

In [110]:

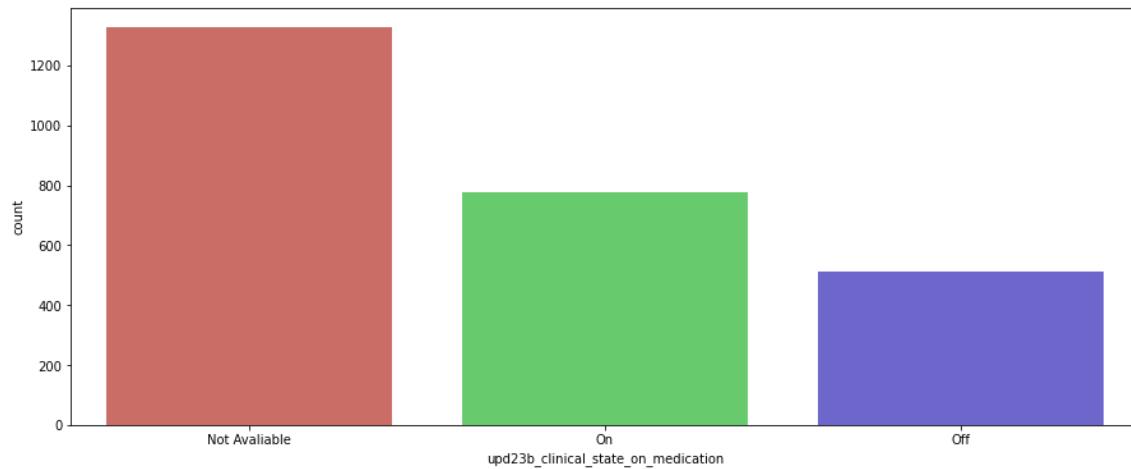
```
df2['upd23b_clinical_state_on_medication'].value_counts()
```

Out[110]:

```
Not Available    1327  
On                775  
Off               513  
Name: upd23b_clinical_state_on_medication, dtype: int64
```

In [112]:

```
plt.figure(figsize=(15,6))  
sns.countplot(x = df2['upd23b_clinical_state_on_medication'], data = df2, palette = 'hls'  
plt.show()
```



In [113]:

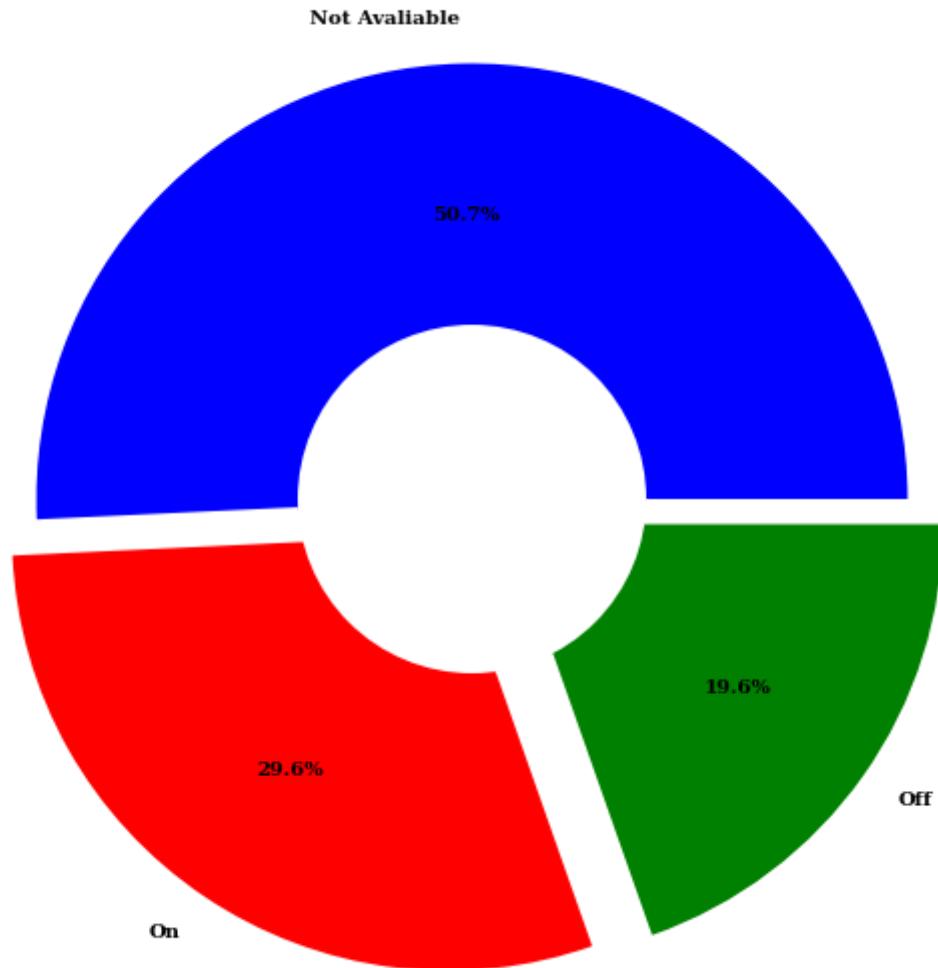
```
label_data = df2['upd23b_clinical_state_on_medication'].value_counts()

explode = (0.0, 0.1, 0.1)
plt.figure(figsize=(20, 10))
patches, texts, pcts = plt.pie(label_data,
                                labels = label_data.index,
                                colors = ['blue', 'red', 'green'],
                                pctdistance = 0.65,
                                shadow = True,
                                startangle = 0,
                                explode = explode,
                                autopct = '%1.1f%%',
                                textprops={ 'fontsize': 10,
                                            'color': 'black',
                                            'weight': 'bold',
                                            'family': 'serif' })
plt.setp(pcts, color='black')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('Clinical_State_on_Medication', size=20, **hfont)

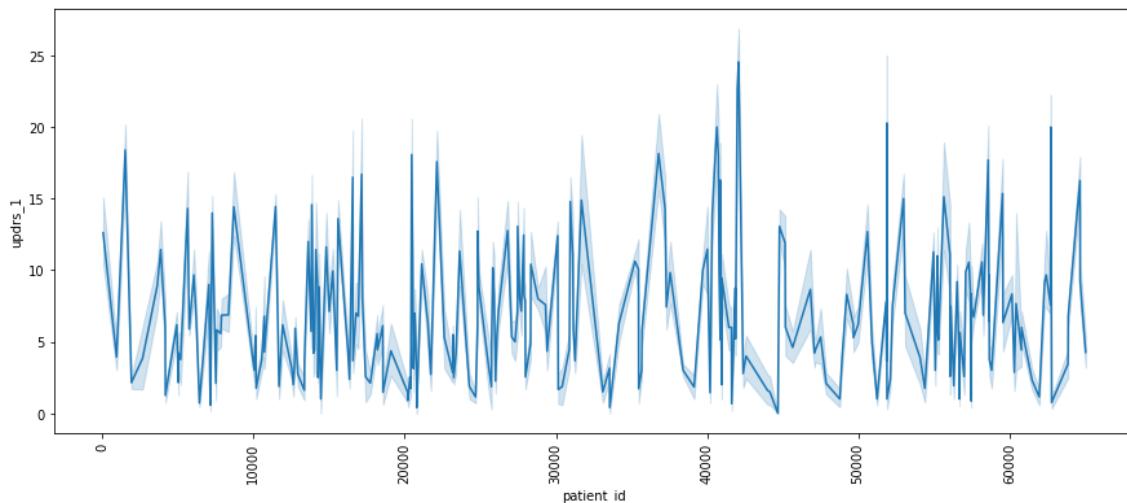
centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```

Clinical_State_on_Medication



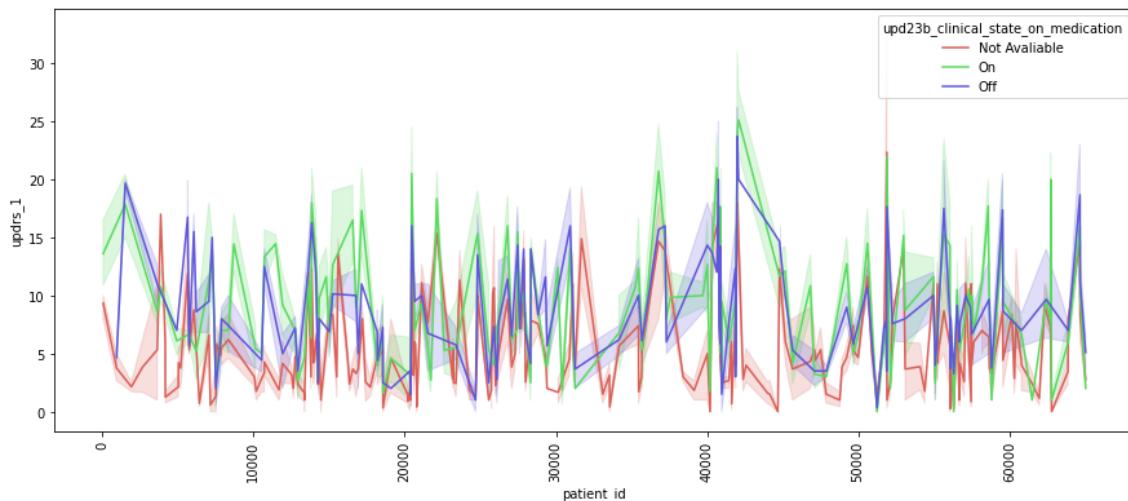
In [114]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['patient_id'], y = df2['updrs_1'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



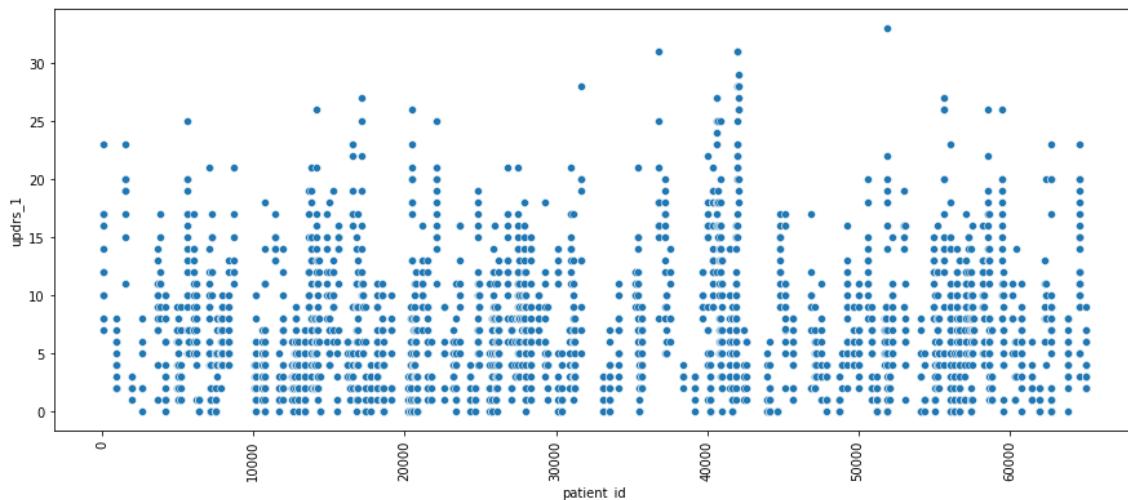
In [285]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['patient_id'], y = df2['updrs_1'], hue = df2['upd23b_clinical_state_on_medication'])
plt.xticks(rotation = 90)
plt.show()
```



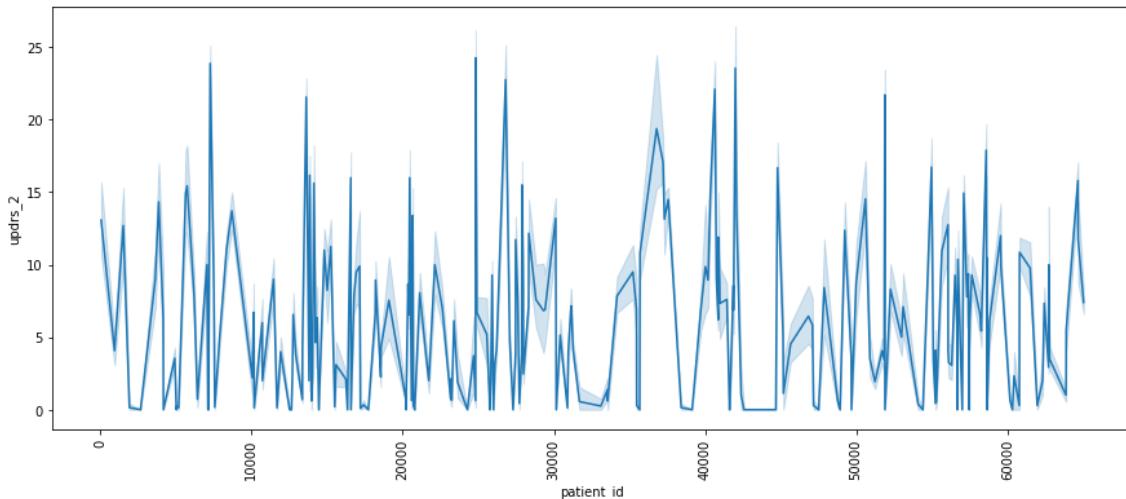
In [115]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['patient_id'], y = df2['updrs_1'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



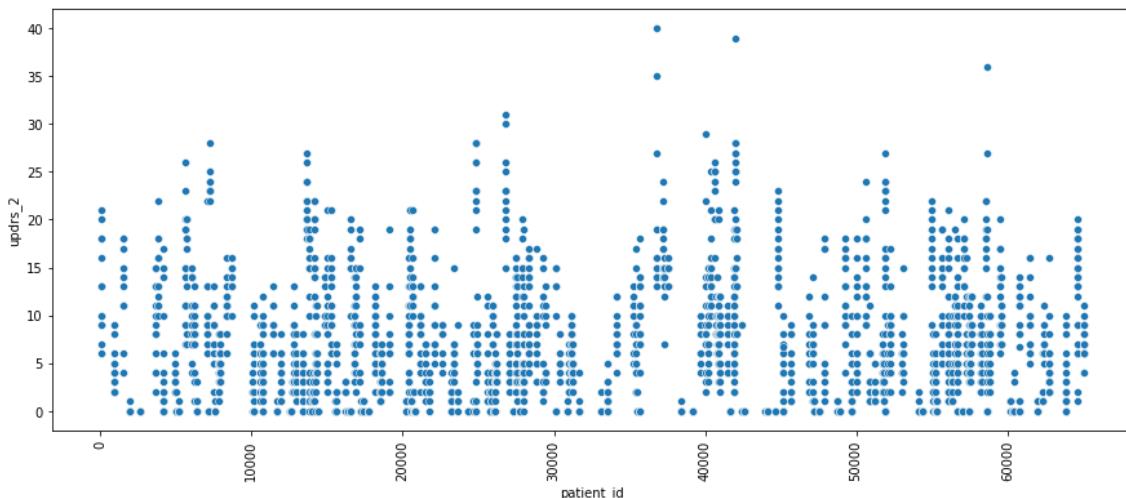
In [116]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['patient_id'], y = df2['updrs_2'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



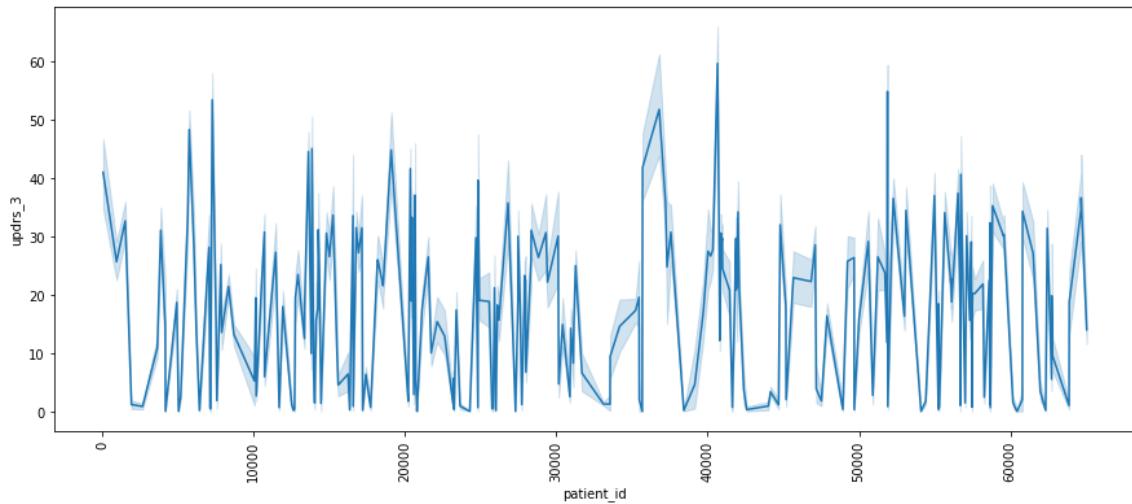
In [117]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['patient_id'], y = df2['updrs_2'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



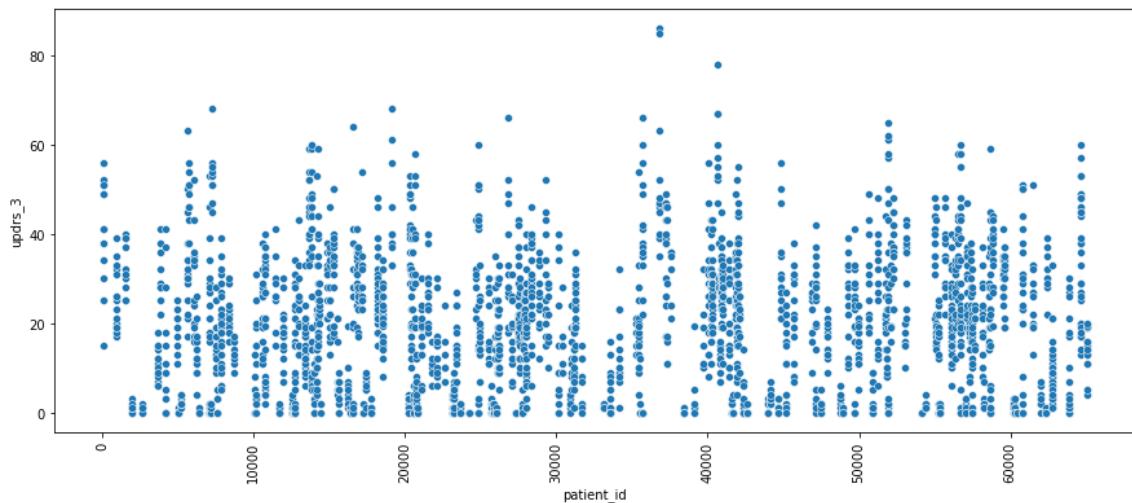
In [118]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['patient_id'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



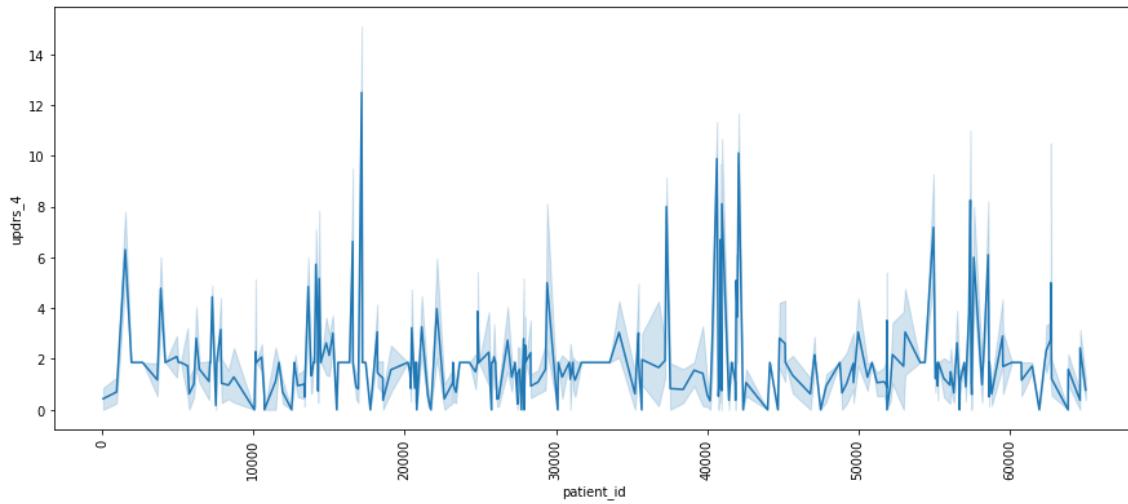
In [119]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['patient_id'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



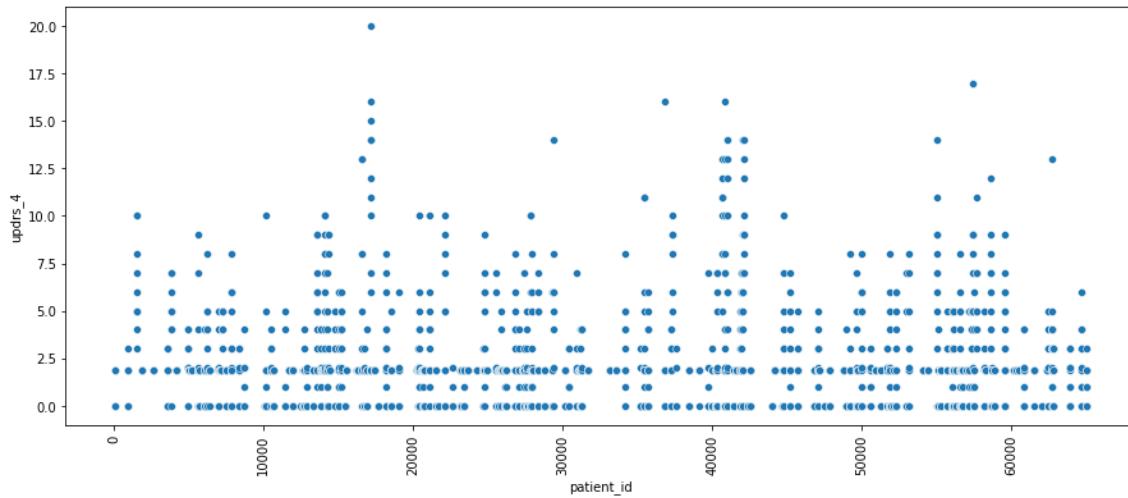
In [120]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['patient_id'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



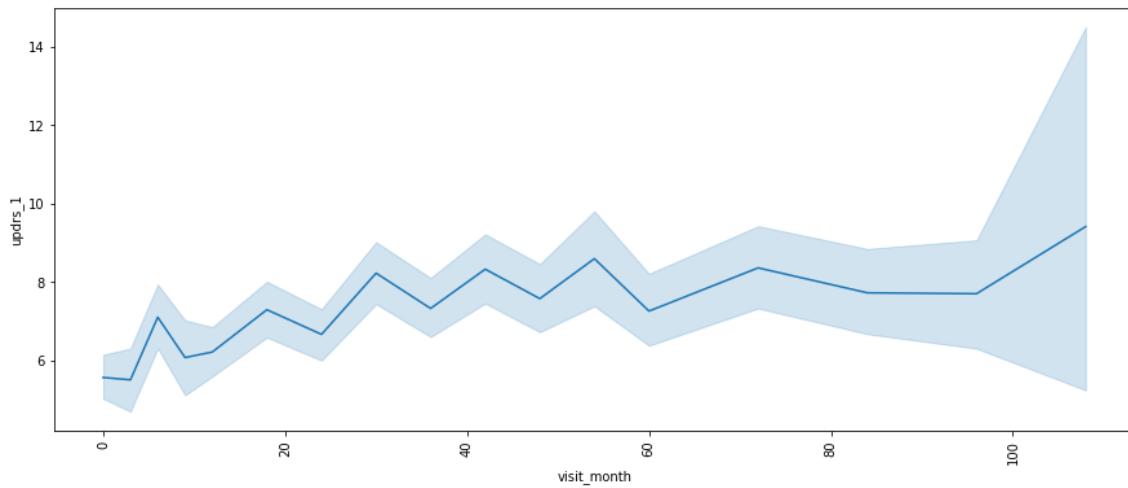
In [121]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['patient_id'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



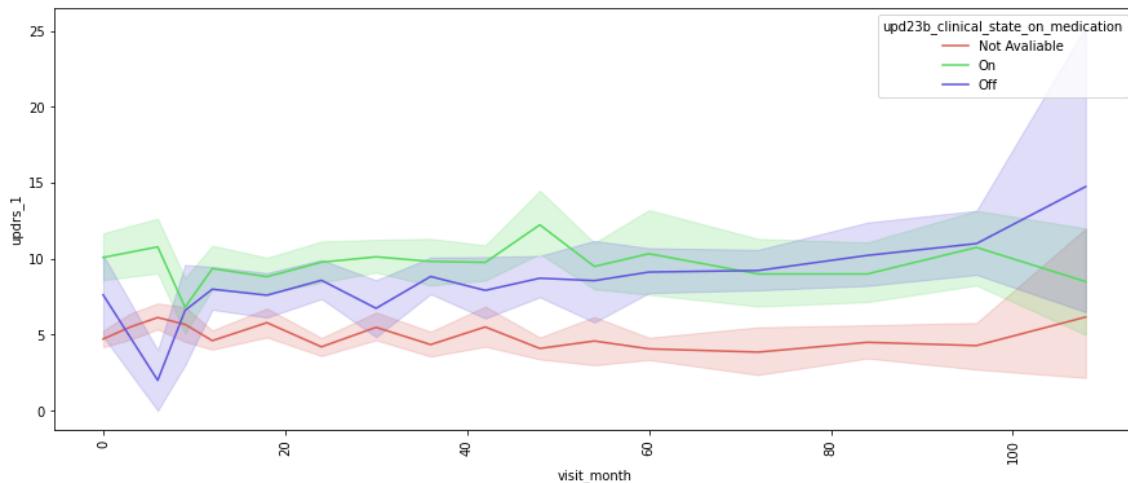
In [124]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['visit_month'], y = df2['updrs_1'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



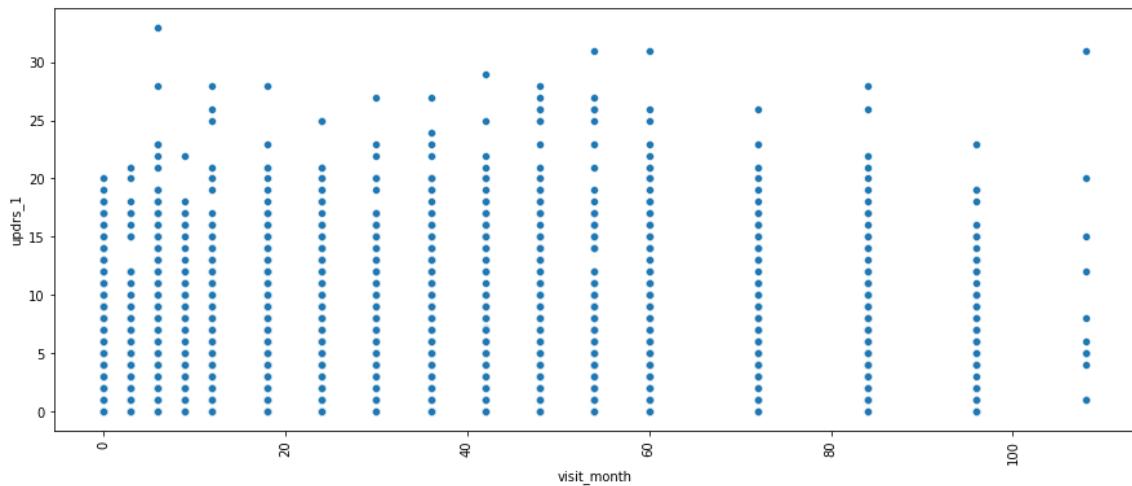
In [286]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['visit_month'], y = df2['updrs_1'], hue = df2['upd23b_clinical_stat']
plt.xticks(rotation = 90)
plt.show()
```



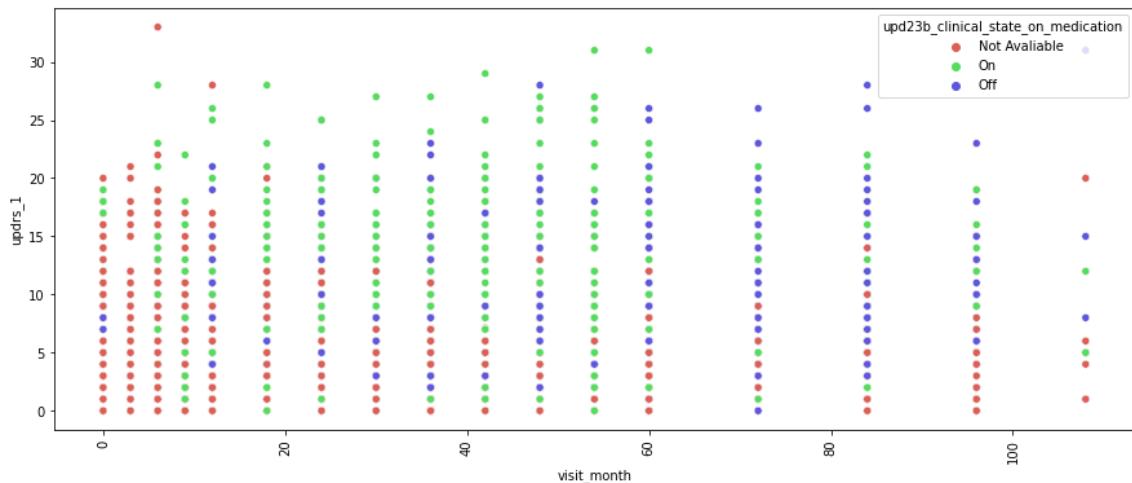
In [125]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['visit_month'], y = df2['updrs_1'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



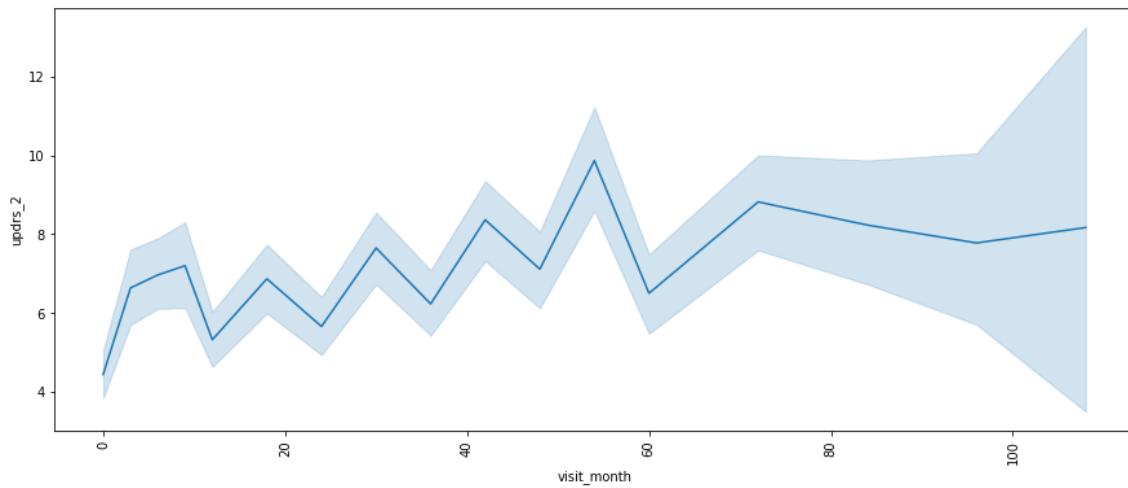
In [287]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['visit_month'], y = df2['updrs_1'], hue = df2['upd23b_clinical_state_on_medicatin'])
plt.xticks(rotation = 90)
plt.show()
```



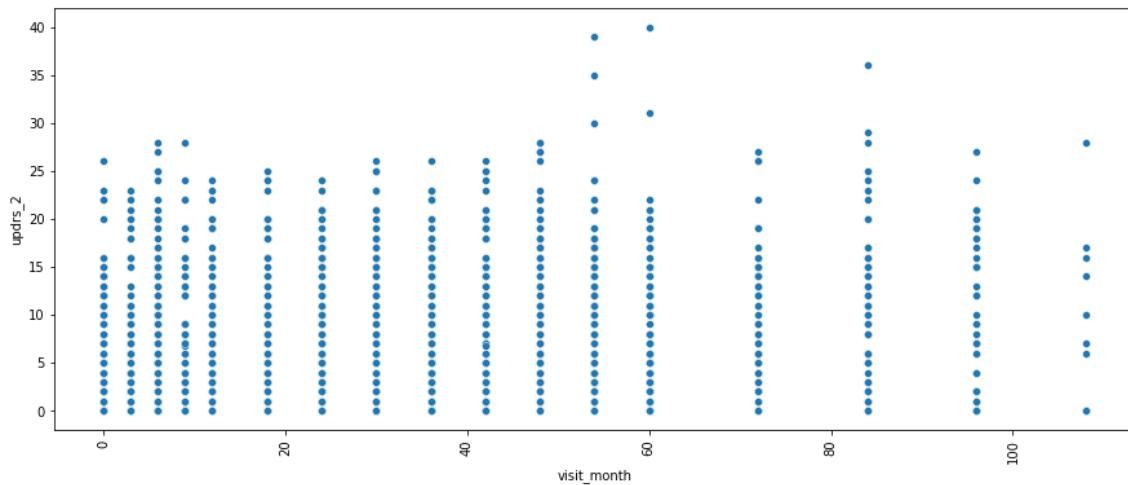
In [126]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['visit_month'], y = df2['updrs_2'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



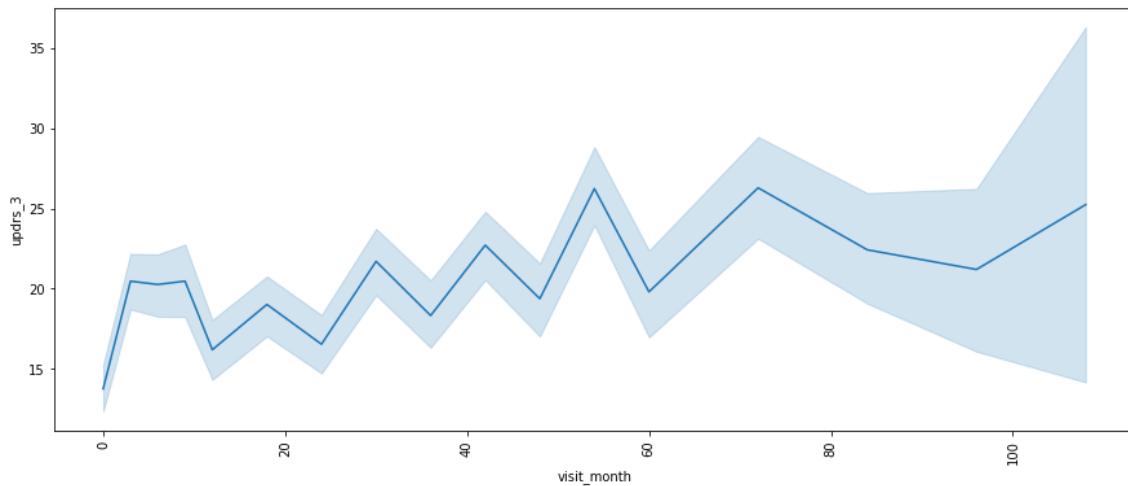
In [127]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['visit_month'], y = df2['updrs_2'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



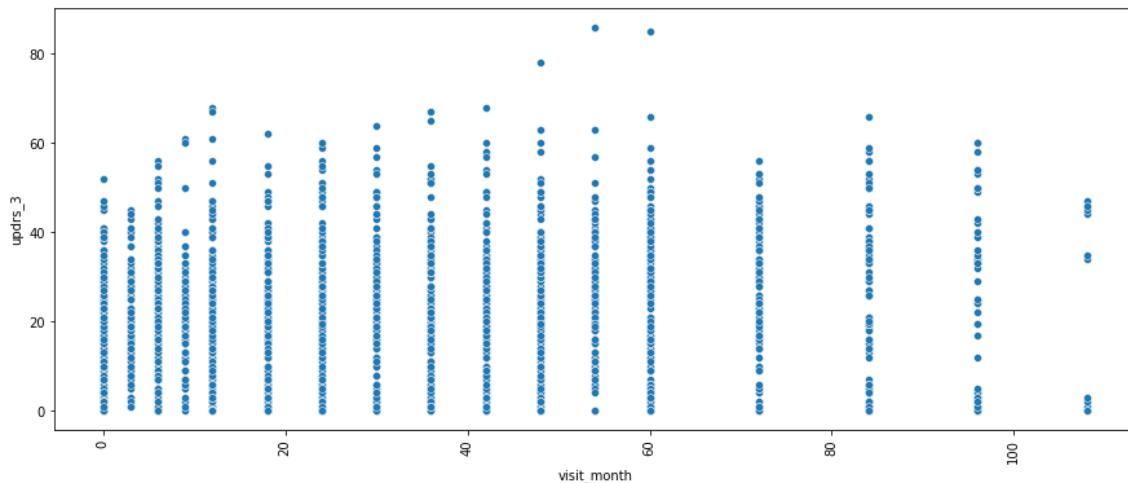
In [128]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['visit_month'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



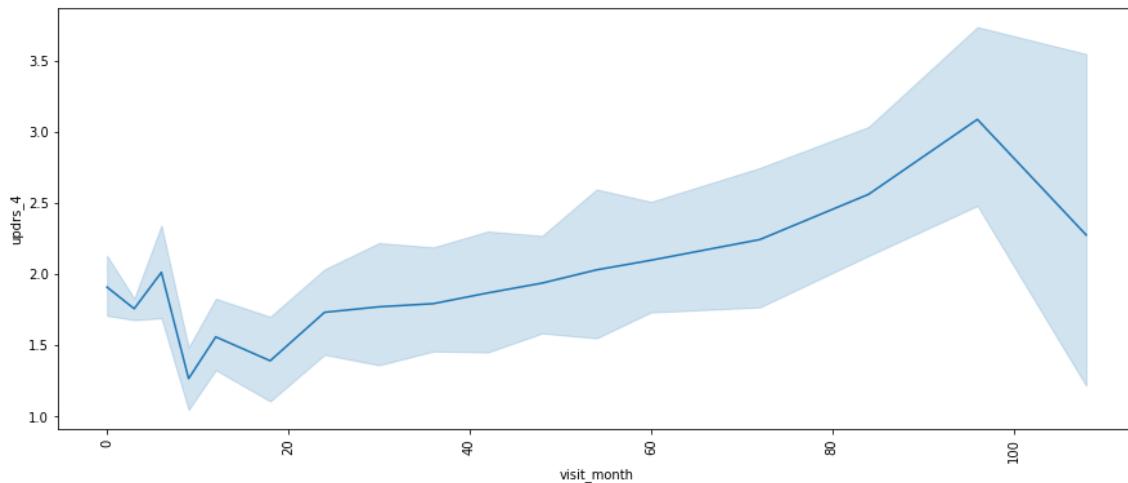
In [130]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['visit_month'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



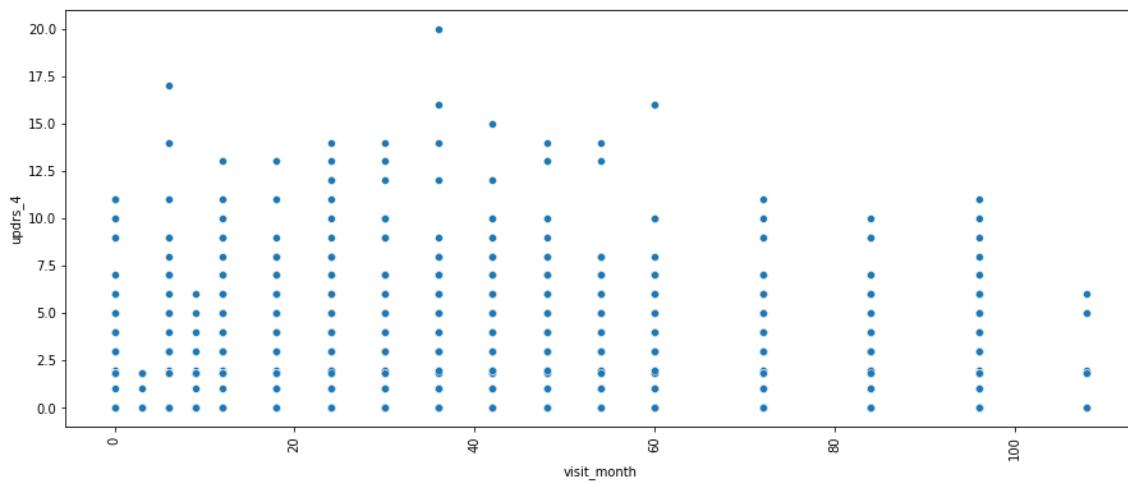
In [131]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['visit_month'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



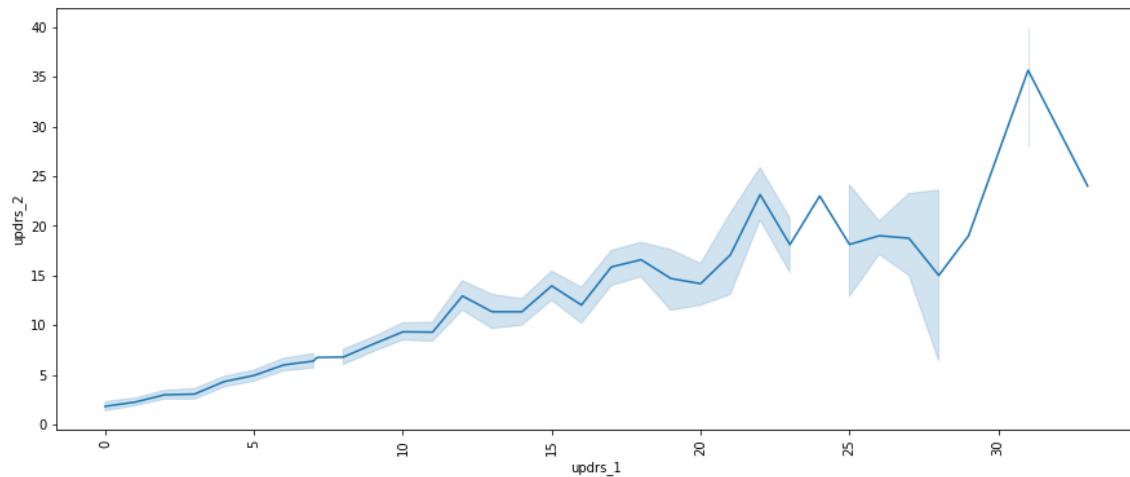
In [132]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['visit_month'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



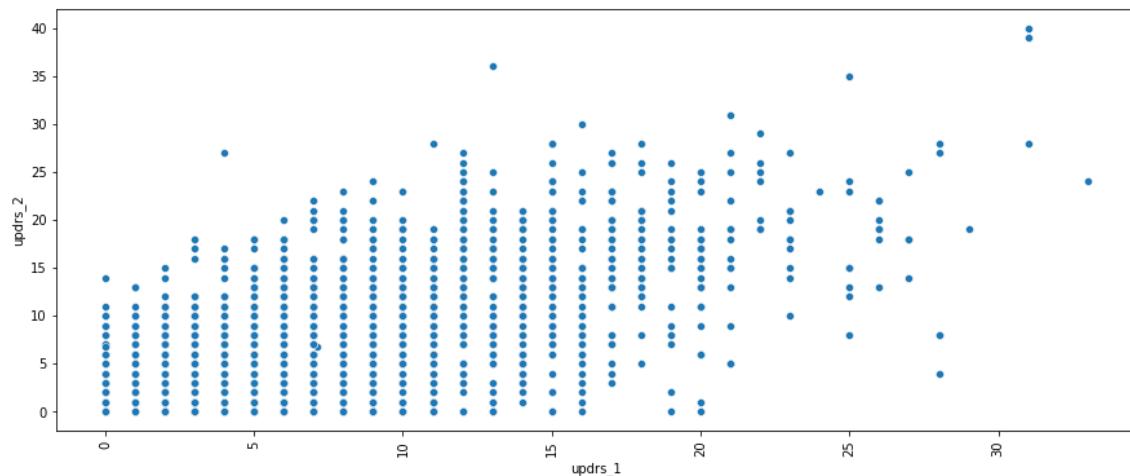
In [133]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['updrs_1'], y = df2['updrs_2'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



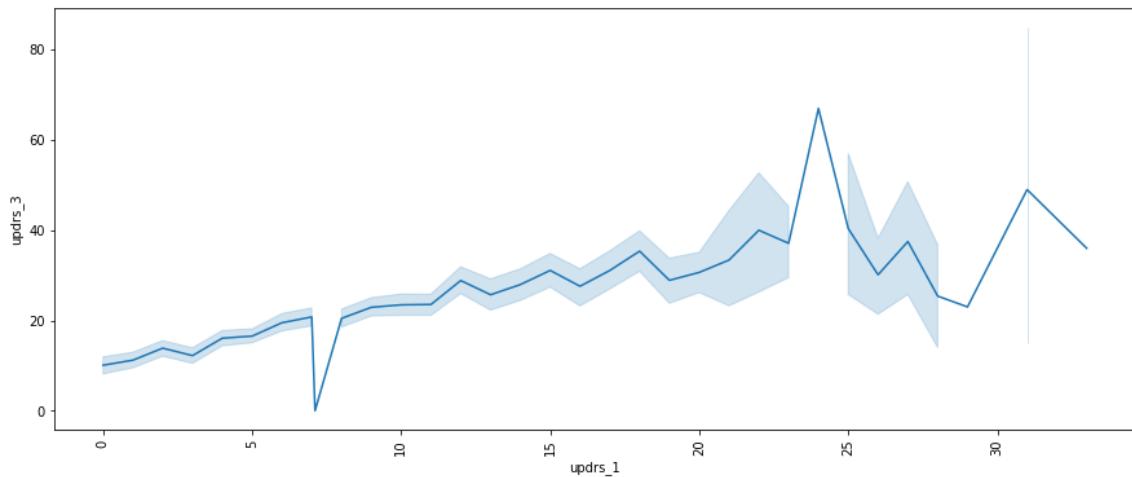
In [135]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['updrs_1'], y = df2['updrs_2'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



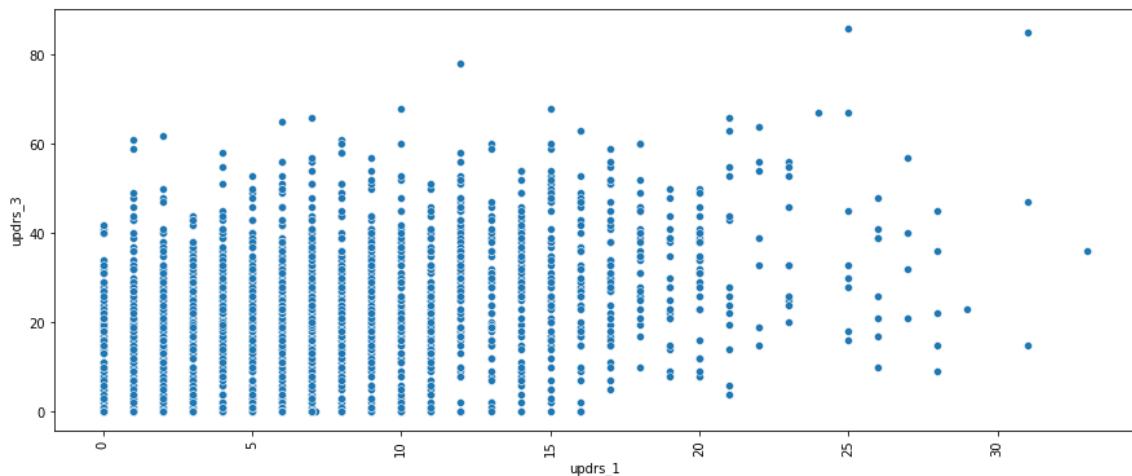
In [134]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['updrs_1'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



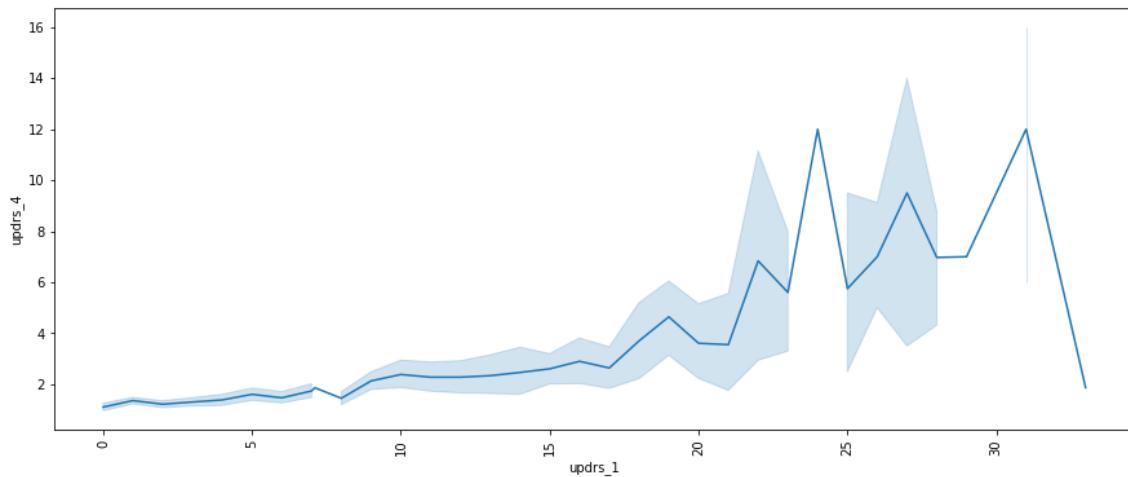
In [137]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['updrs_1'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



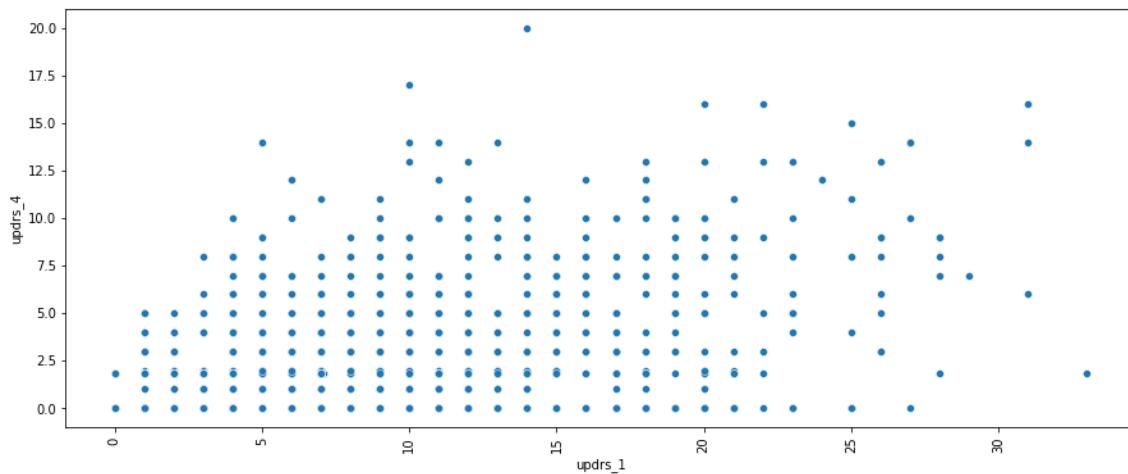
In [138]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['updrs_1'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



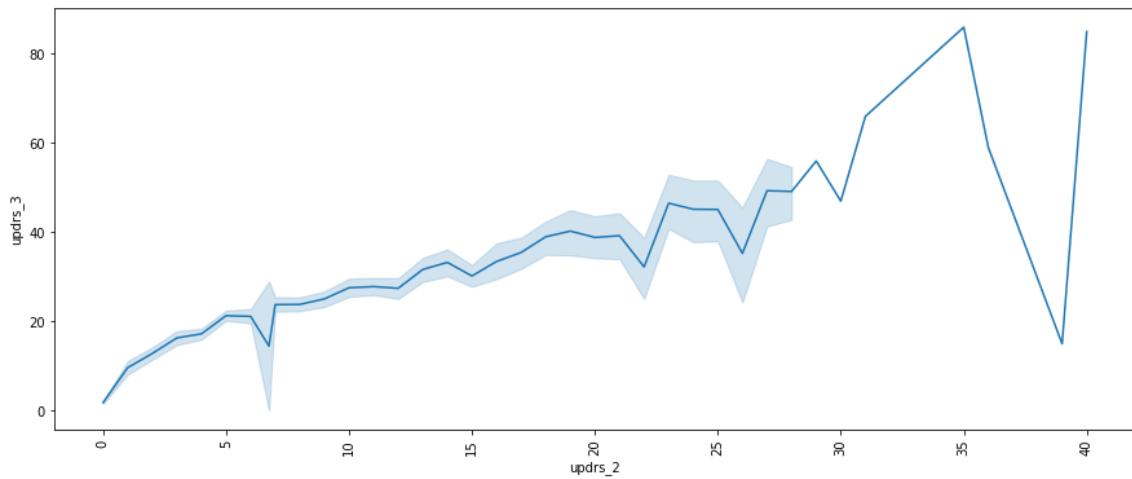
In [139]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['updrs_1'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



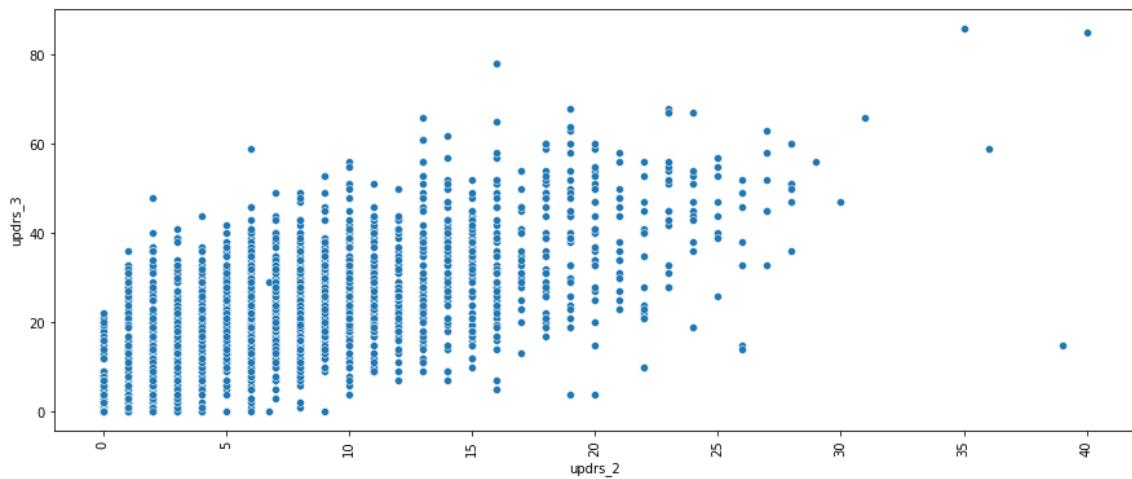
In [140]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['updrs_2'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



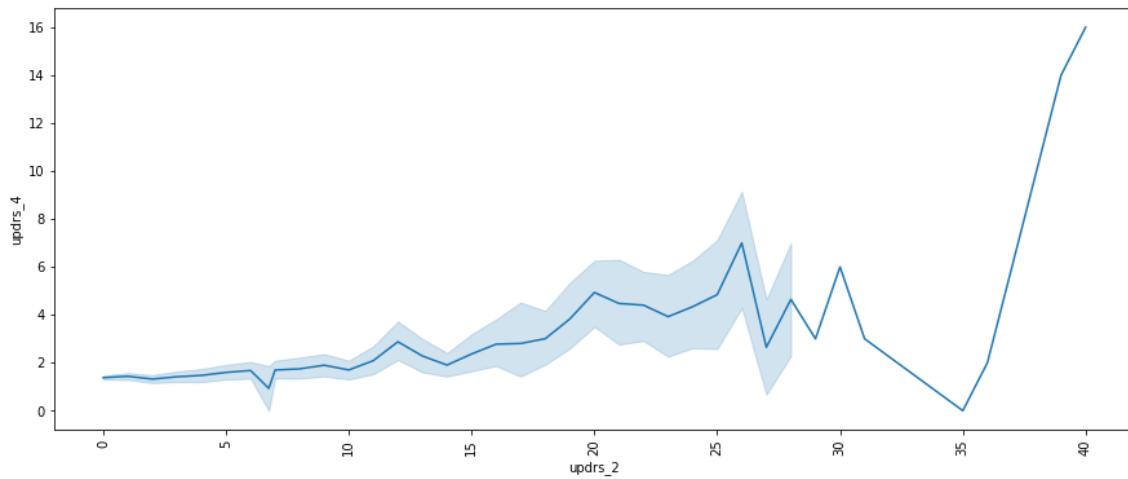
In [141]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['updrs_2'], y = df2['updrs_3'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



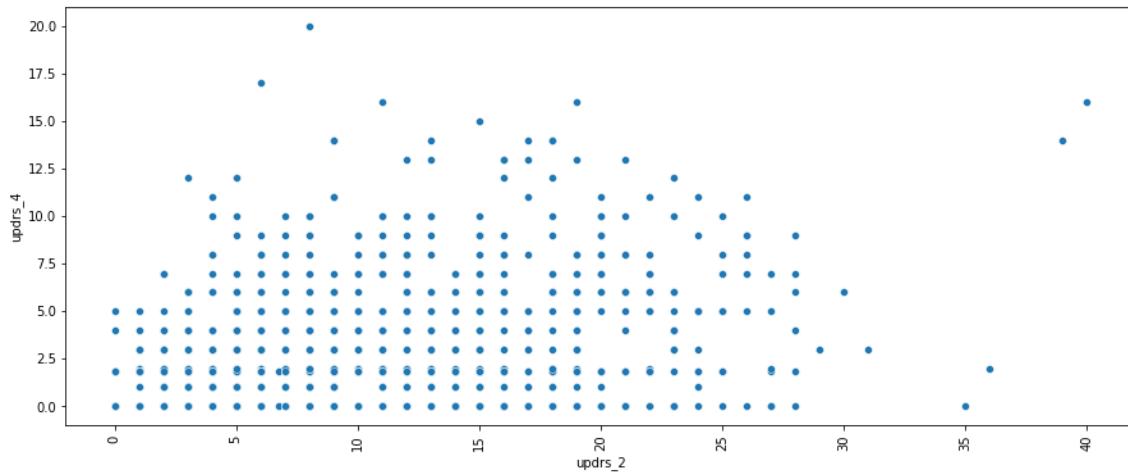
In [142]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['updrs_2'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



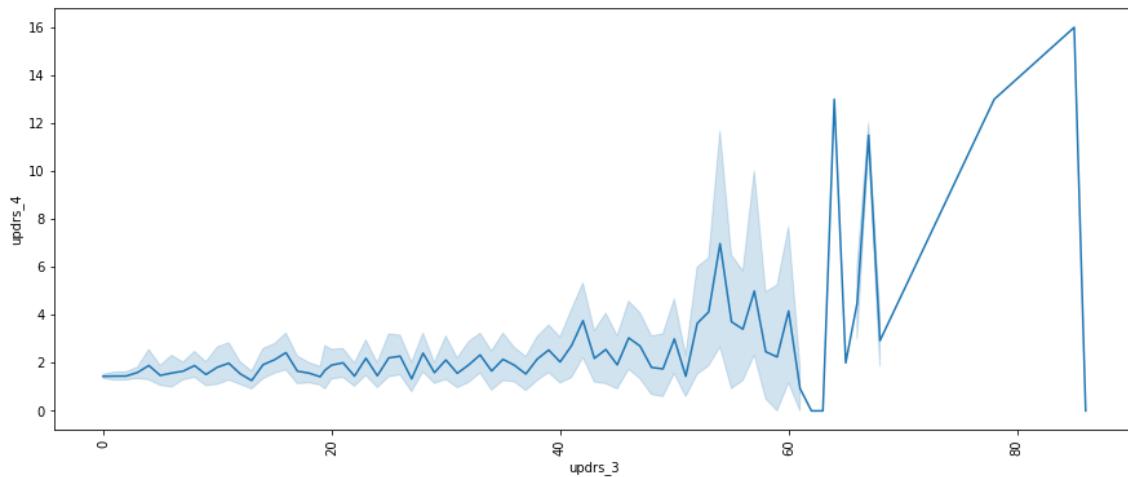
In [143]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['updrs_2'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



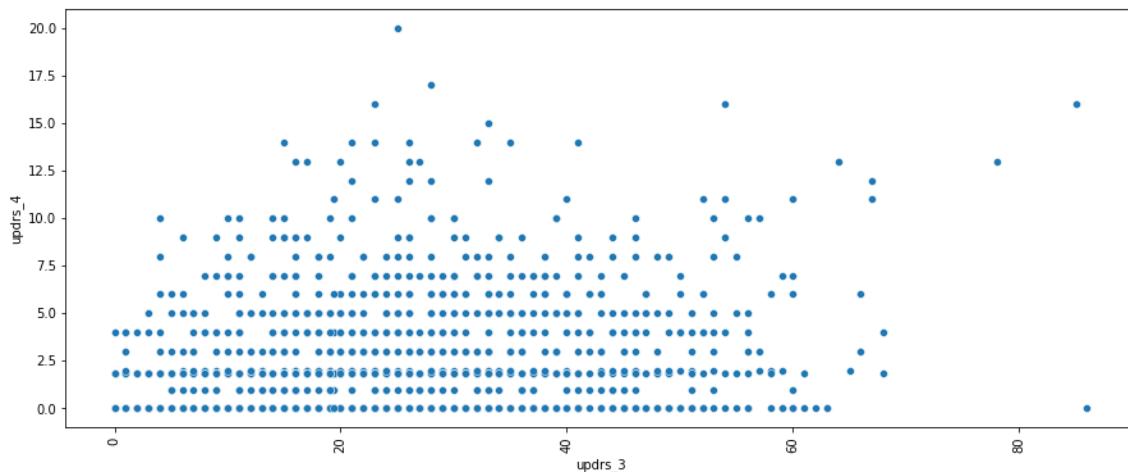
In [144]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df2['updrs_3'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



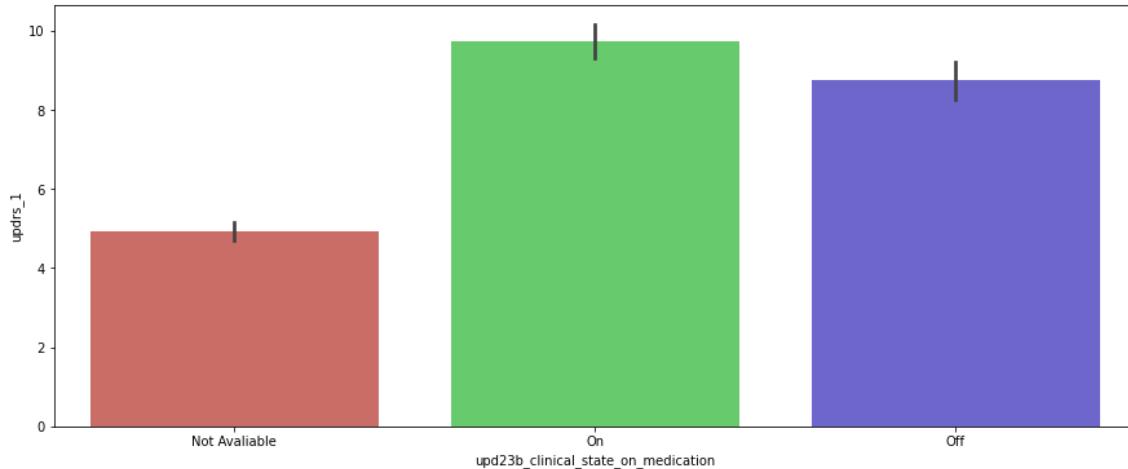
In [145]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x = df2['updrs_3'], y = df2['updrs_4'], palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



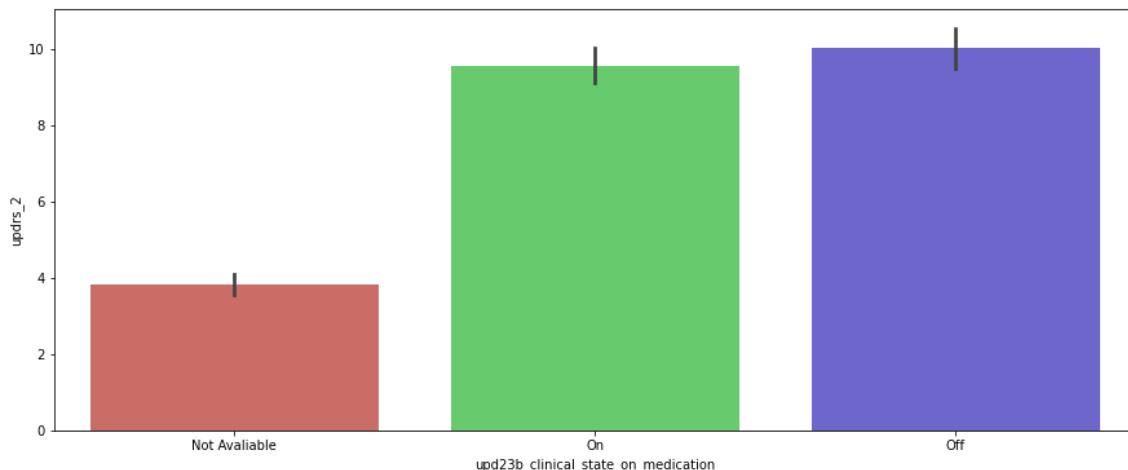
In [147]:

```
plt.figure(figsize=(15,6))
sns.barplot(y = df2['updrs_1'], x = df2['upd23b_clinical_state_on_medication'], palette
plt.show()
```



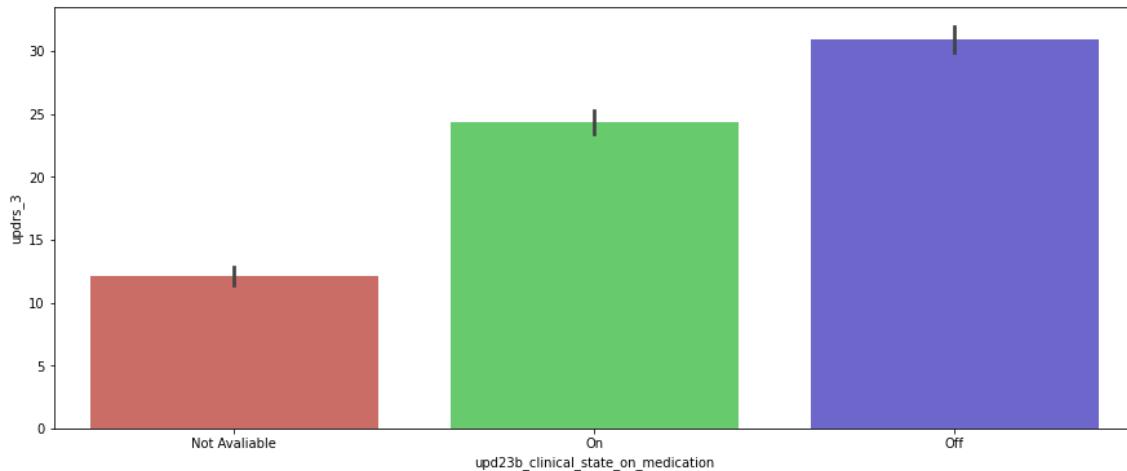
In [150]:

```
plt.figure(figsize=(15,6))
sns.barplot(y = df2['updrs_2'], x = df2['upd23b_clinical_state_on_medication'], palette
plt.show()
```



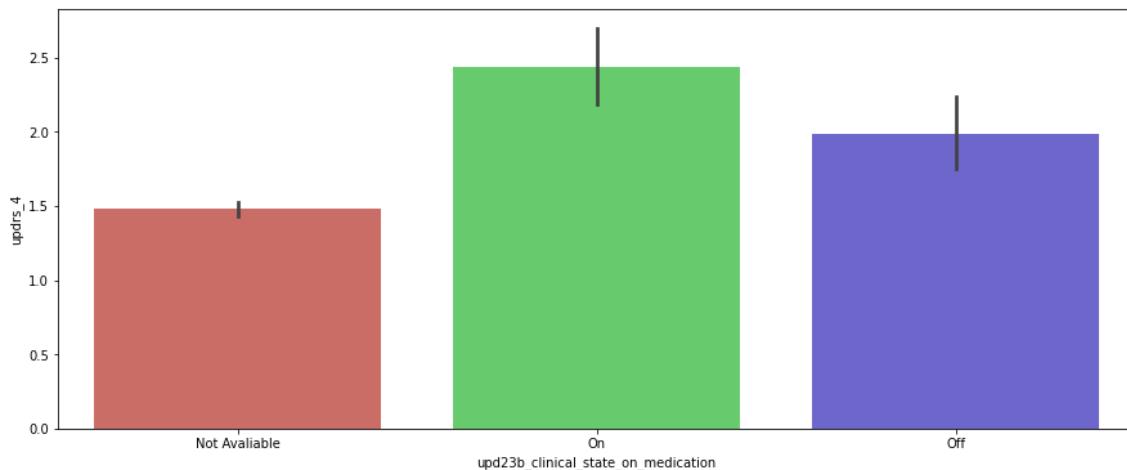
In [151]:

```
plt.figure(figsize=(15,6))
sns.barplot(y = df2['updrs_3'], x = df2['upd23b_clinical_state_on_medication'], palette
plt.show()
```



In [152]:

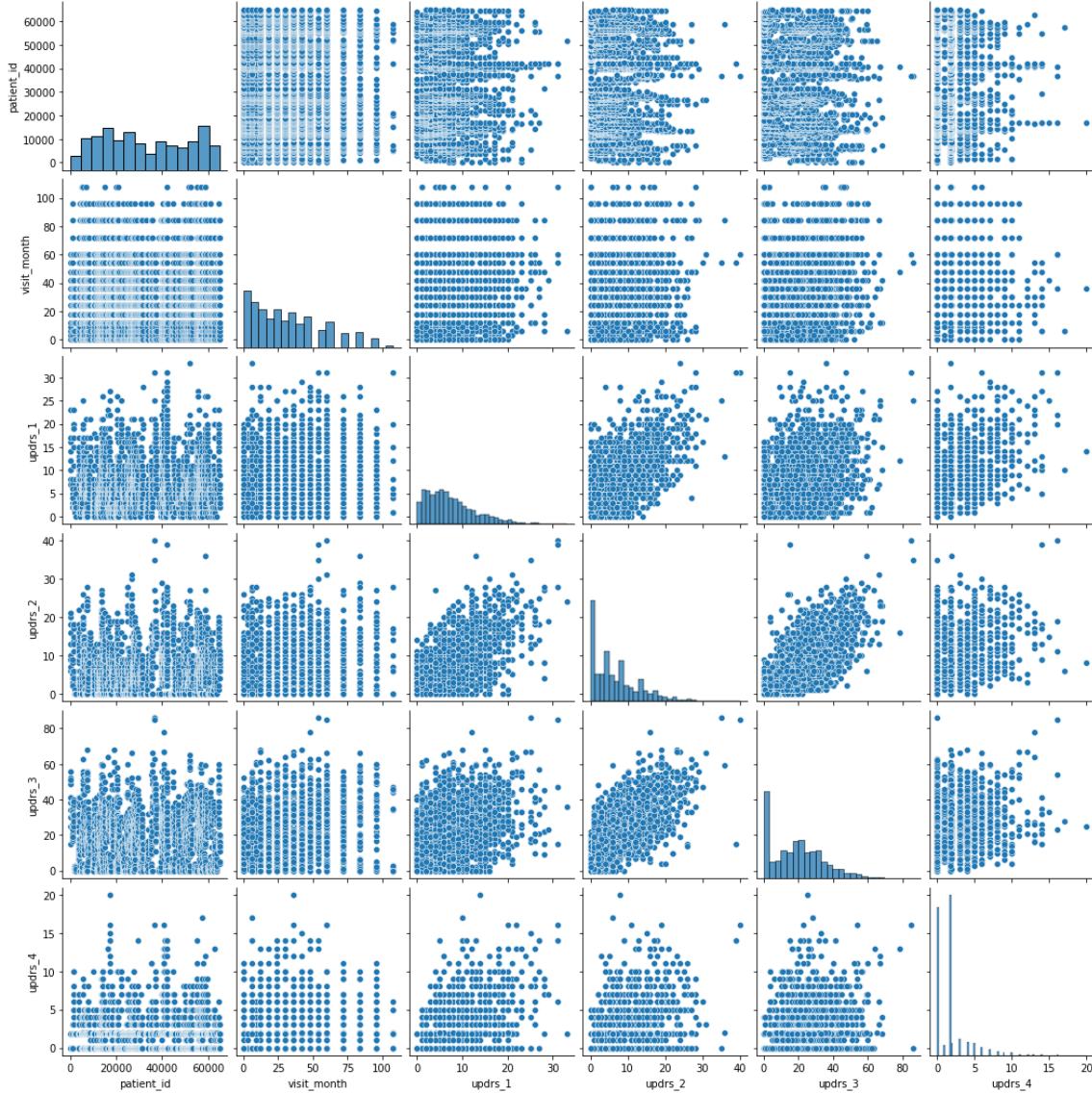
```
plt.figure(figsize=(15,6))
sns.barplot(y = df2['updrs_4'], x = df2['upd23b_clinical_state_on_medication'], palette
plt.show()
```



In [155]:

```
plt.figure(figsize=(15,6))
sns.pairplot(df2, palette = 'hls')
plt.show()
```

<Figure size 1080x432 with 0 Axes>

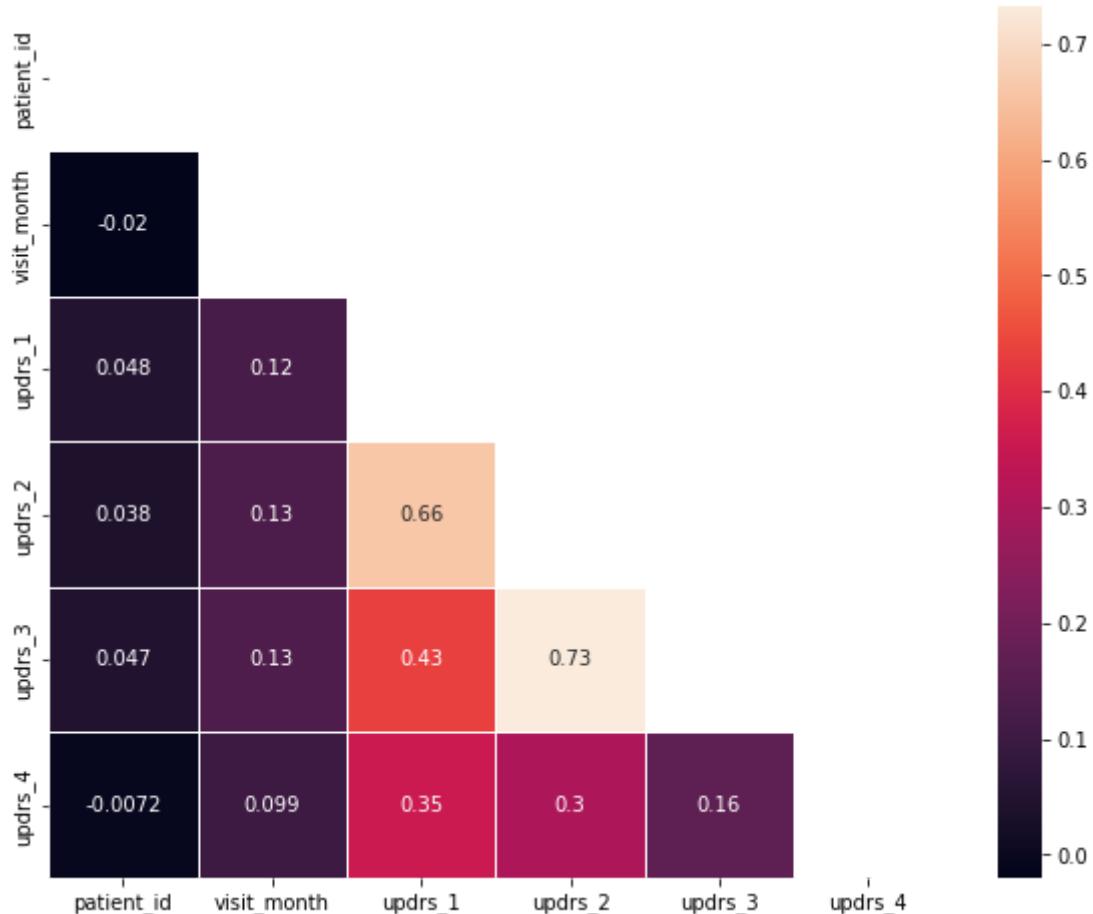


In [153]:

```
df2_corr = df2.corr()
```

In [154]:

```
plt.figure(figsize=(10, 8))
matrix = np.triu(df2_corr)
sns.heatmap(df2_corr, annot=True, linewidth=.8, mask=matrix, cmap="rocket");
plt.show()
```



In [168]:

```
df.shape
```

Out[168]:

(847363, 6)

In [169]:

```
df1.shape
```

Out[169]:

(200981, 5)

In [170]:

```
df2.shape
```

Out[170]:

(2615, 8)

In [171]:

```
df.columns
```

Out[171]:

```
Index(['visit_id', 'visit_month', 'patient_id', 'UniProt', 'Peptide',
       'PeptideAbundance'],
      dtype='object')
```

In [172]:

```
df1.columns
```

Out[172]:

```
Index(['visit_id', 'visit_month', 'patient_id', 'UniProt', 'NPX'],
      dtype='object')
```

In [173]:

```
df2.columns
```

Out[173]:

```
Index(['visit_id', 'patient_id', 'visit_month', 'updrs_1', 'updrs_2',
       'updrs_3', 'updrs_4', 'upd23b_clinical_state_on_medication'],
      dtype='object')
```

In [174]:

```
df_new = df.merge(df1, on=['visit_id', 'visit_month', 'patient_id', 'UniProt'])
df_new = df_new.merge(df2, on=['visit_id', 'visit_month', 'patient_id'])
```

In [175]:

```
df_new.head()
```

Out[175]:

	visit_id	visit_month	patient_id	UniProt	Peptide	I
0	55_0	0	55	O00391		NEEQPLGQWHL
1	55_0	0	55	O00533		GNPEPTFSWTK
2	55_0	0	55	O00533		IEIPSSVQQVPTI
3	55_0	0	55	O00533	KPQSAVYSTGSNGILLC(UniMod_4)	EAEGEPQPTIK
4	55_0	0	55	O00533		SMEQNPGLEYR

In [177]:

```
df_new.tail()
```

Out[177]:

	visit_id	visit_month	patient_id	UniProt	Peptide	F
539644	58648_108	108	58648	Q9UHG2	ILAGSADSEGVAAAPR	
539645	58648_108	108	58648	Q9UKV8	SGNIPAGTTVDTK	
539646	58648_108	108	58648	Q9Y646	LALLVDTVGPR	
539647	58648_108	108	58648	Q9Y6R7	AGC(UniMod_4)VAESTAVC(UniMod_4)R	
539648	58648_108	108	58648	Q9Y6R7	GATTSPGVYELSSR	

◀ ▶

In [176]:

```
df_new.shape
```

Out[176]:

```
(539649, 12)
```

In [178]:

```
df_new.columns
```

Out[178]:

```
Index(['visit_id', 'visit_month', 'patient_id', 'UniProt', 'Peptide',
       'PeptideAbundance', 'NPX', 'updrs_1', 'updrs_2', 'updrs_3', 'updrs_4',
       'upd23b_clinical_state_on_medication'],
      dtype='object')
```

In [179]:

```
df_new.duplicated().sum()
```

Out[179]:

```
0
```

In [180]:

```
df_new.isnull().sum()
```

Out[180]:

```
visit_id          0  
visit_month       0  
patient_id        0  
UniProt           0  
Peptide           0  
PeptideAbundance  0  
NPX               0  
updrs_1           0  
updrs_2           0  
updrs_3           0  
updrs_4           0  
upd23b_clinical_state_on_medication 0  
dtype: int64
```

In [181]:

```
df_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
Int64Index: 539649 entries, 0 to 539648  
Data columns (total 12 columns):  
 #   Column            Non-Null Count  Dtype     
---  --     
 0   visit_id          539649 non-null   object    
 1   visit_month       539649 non-null   int64     
 2   patient_id        539649 non-null   int64     
 3   UniProt           539649 non-null   object    
 4   Peptide           539649 non-null   object    
 5   PeptideAbundance  539649 non-null   float64   
 6   NPX               539649 non-null   float64   
 7   updrs_1           539649 non-null   float64   
 8   updrs_2           539649 non-null   float64   
 9   updrs_3           539649 non-null   float64   
 10  updrs_4           539649 non-null   float64   
 11  upd23b_clinical_state_on_medication 539649 non-null   object    
dtypes: float64(6), int64(2), object(4)  
memory usage: 53.5+ MB
```

In [182]:

df_new.describe()

Out[182]:

	visit_month	patient_id	PeptideAbundance	NPX	updrs_1	updrs_2
count	539649.000000	539649.000000	539649.000000	5.396490e+05	539649.000000	539649.000000
mean	26.758738	32652.382856	71166.236653	4.322718e+05	6.538131	2.000000
std	22.912143	18626.760180	81205.642003	3.938369e+05	5.329968	2.000000
min	0.000000	55.000000	46.765000	8.460820e+01	0.000000	0.000000
25%	6.000000	16566.000000	19537.900000	8.643140e+04	2.000000	2.000000
50%	24.000000	29417.000000	42528.700000	3.096000e+05	5.000000	5.000000
75%	48.000000	50611.000000	90343.100000	7.138080e+05	9.000000	9.000000
max	108.000000	65043.000000	511044.000000	1.365430e+06	33.000000	2.000000

◀ ▶

In [183]:

df_new.nunique()

Out[183]:

```

visit_id                      1068
visit_month                   15
patient_id                    248
UniProt                        226
Peptide                        844
PeptideAbundance                430488
NPX                           178139
updrs_1                       30
updrs_2                       29
updrs_3                       65
updrs_4                       18
upd23b_clinical_state_on_medication    3
dtype: int64

```

In [184]:

df_new['visit_month'].unique()

Out[184]:

```
array([  0,   3,   6,  12,  18,  24,  30,  36,  48,  54,  60,  72,  84,
       96, 108], dtype=int64)
```

In [185]:

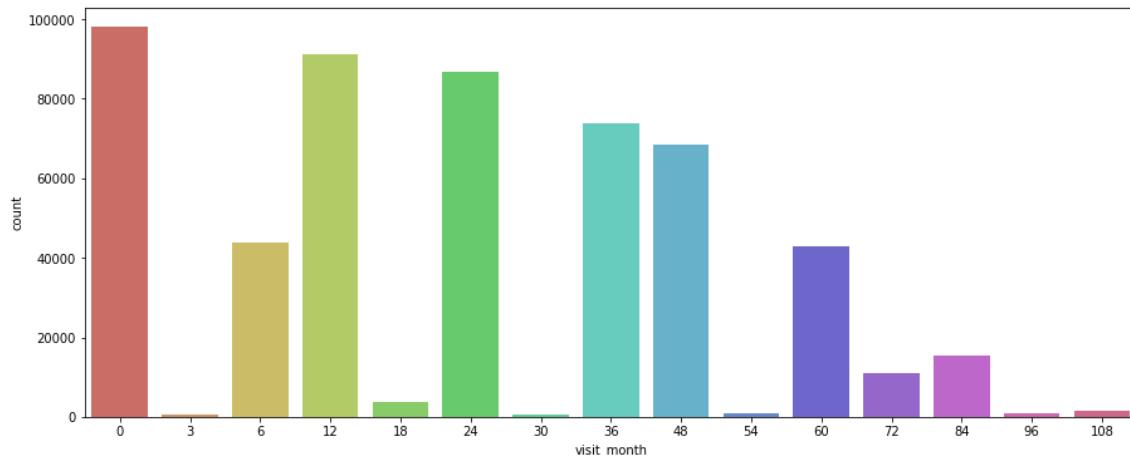
```
df_new['visit_month'].value_counts()
```

Out[185]:

```
0      98054  
12     91046  
24     86690  
36     73811  
48     68616  
6      43886  
60     42897  
84     15400  
72     10943  
18     3815  
108    1516  
96     1012  
54     969  
3      514  
30     480  
Name: visit_month, dtype: int64
```

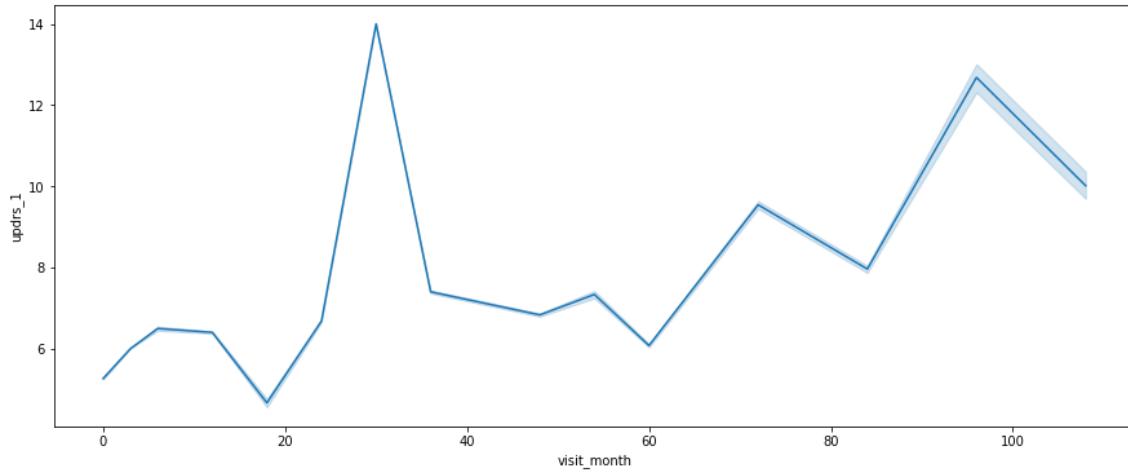
In [186]:

```
plt.figure(figsize=(15,6))  
sns.countplot(df_new['visit_month'], data = df_new, palette = 'hls')  
plt.show()
```



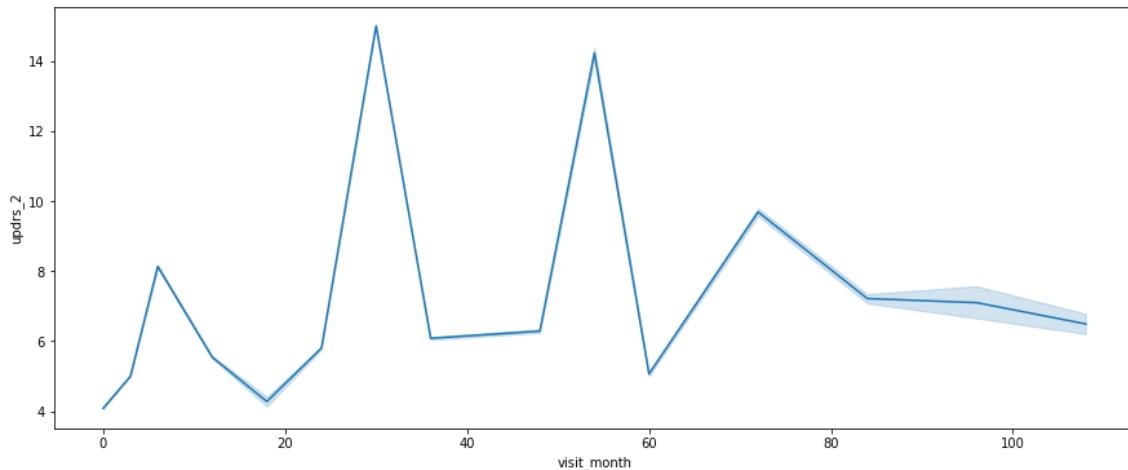
In [190]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df_new['visit_month'], y = df_new['updrs_1'], data = df_new, palette =
plt.show()
```



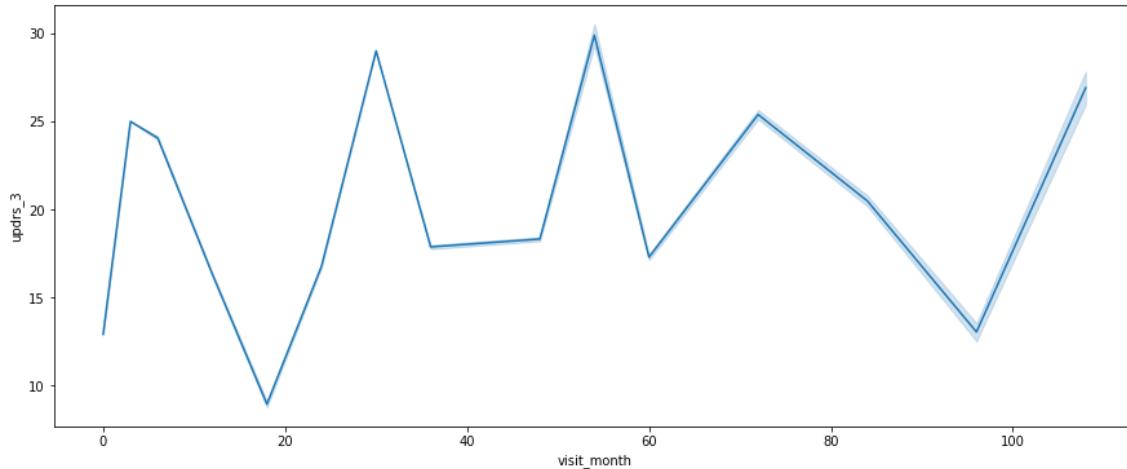
In [191]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df_new['visit_month'], y = df_new['updrs_2'], data = df_new, palette =
plt.show()
```



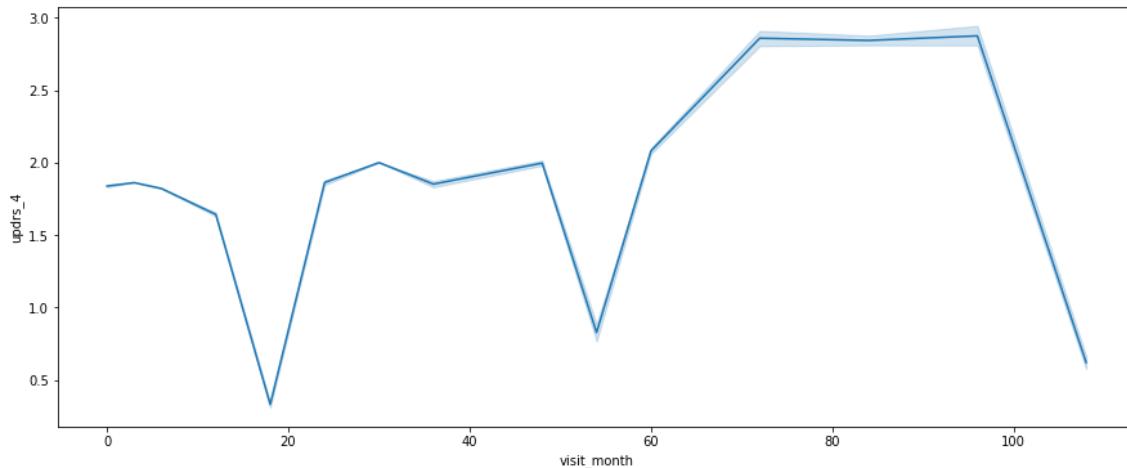
In [192]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df_new['visit_month'], y = df_new['updrs_3'], data = df_new, palette =
plt.show()
```



In [193]:

```
plt.figure(figsize=(15,6))
sns.lineplot(x = df_new['visit_month'], y = df_new['updrs_4'], data = df_new, palette =
plt.show()
```



In [202]:

```
df_new.columns
```

Out[202]:

```
Index(['visit_id', 'visit_month', 'patient_id', 'UniProt', 'Peptide',
       'PeptideAbundance', 'NPX', 'updrs_1', 'updrs_2', 'updrs_3', 'updrs_4',
       'upd23b_clinical_state_on_medication'],
      dtype='object')
```

In [206]:

```
df_new_1 = df_new[['PeptideAbundance', 'NPX', 'updrs_1']]
```

In [207]:

```
df_new_1
```

Out[207]:

	PeptideAbundance	NPX	updrs_1
0	11254.30	11254.3	10.0
1	102060.00	732430.0	10.0
2	174185.00	732430.0	10.0
3	27278.90	732430.0	10.0
4	30838.70	732430.0	10.0
...
539644	202820.00	369437.0	6.0
539645	105830.00	105830.0	6.0
539646	21257.60	21257.6	6.0
539647	5127.26	17953.1	6.0
539648	12825.90	17953.1	6.0

539649 rows × 3 columns

In [208]:

```
X = df_new_1.drop('updrs_1', axis=1)
y = df_new_1['updrs_1']
```

In [209]:

```
from sklearn import preprocessing
scaler = preprocessing.MinMaxScaler()
X = scaler.fit_transform(X)
```

In [210]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.30)
```

In [211]:

```
from sklearn.linear_model import LinearRegression
```

In [212]:

```
lr = LinearRegression()
```

In [213]:

```
lr.fit(X_train,y_train)
```

Out[213]:

```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [214]:

```
y_pred = lr.predict(X_test)
```

In [215]:

```
from sklearn.metrics import mean_squared_error
```

In [216]:

```
RMSE_lr = mean_squared_error(y_test, y_pred, squared = False)  
RMSE_lr
```

Out[216]:

```
5.30827489979071
```

In [219]:

```
from sklearn.metrics import r2_score
```

In [220]:

```
r2 = r2_score(y_test, y_pred)  
r2
```

Out[220]:

```
-2.6721576411148362e-05
```

In [217]:

```
from sklearn.model_selection import cross_val_score
```

In [218]:

```
cross_val_score(lr, X_train, y_train, cv=10)
```

Out[218]:

```
array([-5.27375550e-05,  3.37670713e-05, -1.58026404e-06, -1.99379959e-05,
       -7.52700358e-05,  1.86911454e-05, -1.63380315e-05, -6.99537494e-06,
       1.35883401e-05,  6.12006435e-05])
```

In [265]:

```
from sklearn.model_selection import GridSearchCV

# Define the grid of hyperparameters to search
param_grid = {'fit_intercept': [True, False],
              'normalize': [True, False]}

# Create a Linear regression model
lin_reg = LinearRegression()

# Create a grid search object
grid_search = GridSearchCV(lin_reg, param_grid, )
cv=5
# Fit the grid search to the data
grid_search.fit(X_train, y_train)

# Get the best hyperparameters
best_fit_intercept = grid_search.best_params_['fit_intercept']
best_normalize = grid_search.best_params_['normalize']

# Train a Linear regression model with the best hyperparameters
lin_reg_best = LinearRegression(fit_intercept=best_fit_intercept,
                                 normalize=best_normalize)
lin_reg_best.fit(X_train, y_train)
```

Out[265]:

```
LinearRegression(normalize=True)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [266]:

```
y_pred = lin_reg_best.predict(X_test)
```

In [267]:

```
r2 = r2_score(y_test, y_pred)
r2
```

Out[267]:

```
-2.672157641148362e-05
```

In [221]:

```
from sklearn.tree import DecisionTreeRegressor
```

In [222]:

```
dt = DecisionTreeRegressor(random_state=0)
```

In [223]:

```
dt.fit(X_train, y_train)
```

Out[223]:

```
DecisionTreeRegressor(random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [224]:

```
y_pred = dt.predict(X_test)
```

In [225]:

```
RMSE_dt = mean_squared_error(y_test, y_pred, squared = False)  
RMSE_dt
```

Out[225]:

```
5.770982107341489
```

In [226]:

```
cross_val_score(dt, X_train, y_train, cv=10)
```

Out[226]:

```
array([-0.26053411, -0.28087271, -0.23617434, -0.21194405, -0.20917041,  
      -0.22644514, -0.43629063, -0.2066975 , -0.15947303, -0.24562276])
```

In [227]:

```
r2 = r2_score(y_test, y_pred)  
r2
```

Out[227]:

```
-0.1819640005104295
```

In [268]:

```
# Define the grid of hyperparameters to search
param_grid = {'max_depth': [1, 2, 3, 4, 5],
              'min_samples_split': [2, 3, 4]}

# Create a decision tree regression model
dtr = DecisionTreeRegressor()

# Create a grid search object
grid_search = GridSearchCV(dtr, param_grid, cv=5)

# Fit the grid search to the data
grid_search.fit(X_train, y_train)

# Get the best hyperparameters
best_max_depth = grid_search.best_params_['max_depth']
best_min_samples_split = grid_search.best_params_['min_samples_split']

# Train a decision tree regression model with the best hyperparameters
dtr_best = DecisionTreeRegressor(max_depth=best_max_depth,
                                  min_samples_split=best_min_samples_split)
dtr_best.fit(X_train, y_train)
```

Out[268]:

```
DecisionTreeRegressor(max_depth=5, min_samples_split=4)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with [nbviewer.org](#).

In [228]:

```
from sklearn.ensemble import RandomForestRegressor
```

In [229]:

```
rf = RandomForestRegressor(max_depth=5, random_state=0)
```

In [230]:

```
rf.fit(X_train, y_train)
```

Out[230]:

```
RandomForestRegressor(max_depth=5, random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with [nbviewer.org](#).

In [231]:

```
y_pred = rf.predict(X_test)
```

In [232]:

```
RMSE_rf = mean_squared_error(y_test, y_pred, squared = False)
RMSE_rf
```

Out[232]:

5.30474267933126

In [233]:

```
r2 = r2_score(y_test, y_pred)
r2
```

Out[233]:

0.0013037067378985912

In [269]:

```
# Define the grid of hyperparameters to search
param_grid = {'n_estimators': [10, 50, 100],
              'max_depth': [1, 2, 3, 4, 5]}

# Create a random forest regression model
rfr = RandomForestRegressor()

# Create a grid search object
grid_search = GridSearchCV(rfr, param_grid, cv=5)

# Fit the grid search to the data
grid_search.fit(X_train, y_train)

# Get the best hyperparameters
best_n_estimators = grid_search.best_params_['n_estimators']
best_max_depth = grid_search.best_params_['max_depth']

# Train a random forest regression model with the best hyperparameters
rfr_best = RandomForestRegressor(n_estimators=best_n_estimators,
                                  max_depth=best_max_depth)
rfr_best.fit(X_train, y_train)
```

Out[269]:

RandomForestRegressor(max_depth=5, n_estimators=10)

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [234]:

```
from xgboost import XGBRegressor
```

In [235]:

```
model = XGBRegressor()
model.fit(X_train, y_train)
```

Out[235]:

```
XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
             early_stopping_rounds=None, enable_categorical=False,
             eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
             importance_type=None, interaction_constraints='',
             learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
             max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
             missing=nan, monotone_constraints='()', n_estimators=100, n_jobs=0,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, ...)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with [nbviewer.org](#).

In [236]:

```
y_pred = model.predict(X_test)
```

In [237]:

```
RMSE_xg = mean_squared_error(y_test, y_pred, squared = False)
RMSE_xg
```

Out[237]:

5.21380418046862

In [238]:

```
r2 = r2_score(y_test, y_pred)
r2
```

Out[238]:

0.03525124767452825

In []:

```
from sklearn.model_selection import GridSearchCV
param_grid = {
    'max_depth': [3, 5, 7],
    'learning_rate': [0.01, 0.1, 0.5],
    'n_estimators': [50, 100, 200]
}

model = XGBRegressor()
grid_search = GridSearchCV(model, param_grid, cv=5, scoring='neg_mean_squared_error')
grid_search.fit(X_train, y_train)

y_pred = grid_search.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
print("MSE:", mse)

# Print the best hyperparameters from GridSearchCV
print("Best hyperparameters:", grid_search.best_params_)
```