AI Generated Ghibli Style Image Trends (2025)_EDA_PROJECT_VIVEK_CHAUHAN

AI Generated Ghibli Style Image Trends (2025)

The analysis is divided into four main parts:

1. Data understanding

2. Data cleaning (cleaning missing values, removing redundant columns etc.)

3. Data Analysis

4. Recommendations

In [1]:
```
# first of all load the necessary library to work with datasets and visualization

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# load the dataset

data = pd.read_csv("C:/Users/VIVEK CHAUHAN/Desktop/AI Generated Ghibli Style Image Trends (2025)/ai_ghibli_trend_dataset_v2.cs
data
```

Out[2]:

	image_id	user_id	prompt	likes	shares	comments	platform	generation_time	gpu_usage	file_size_kb	resolution	style_
0	77ce5c72-eb45-4651-bcb1-c0677c0fceaf	6a7adf3d	Studio Ghibli-inspired ocean with giant fish	916	410	555	Reddit	4.80	49	1684	1024x1024	
1	7d66c67f-0d11-4ef9-895c-d865ef11fe40	523b8706	Ghibli-style village at sunset	2965	1361	417	Reddit	11.11	81	2808	1024x1024	
2	d7978afd-3932-4cce-9a21-5f9bf2bc1f64	0e02592a	A lone traveler exploring an enchanted ruin	4727	655	785	Instagram	5.56	41	1800	2048x2048	
3	cb34636a-a15c-4b15-999c-759dbb8896fe	9ed78a42	Spirited Away-style bustling market street	1629	1954	212	TikTok	12.45	88	479	2048x2048	
4	7511fbb8-db05-4584-a3a4-e8bb525ed58b	69ec8f02	Magical Ghibli forest with floating lanterns	2573	1281	913	TikTok	4.80	64	1789	512x512	
...	...	...	...	...	...	...	...	...	...	...	...	
495	135267ba-3941-42ae-a421-3be20e3856d1	c80e6866	Anime-style train passing through a fantasy world	1634	1328	624	TikTok	10.56	73	3255	1024x1024	

	image_id	user_id	prompt	likes	shares	comments	platform	generation_time	gpu_usage	file_size_kb	resolution	style_
496	b2ced831-5b08-403d-bfb3-b562e256f359	81669630	Serene meadow with a tiny spirit creature	4198	833	812	Reddit	8.41	89	4710	1024x1024	
497	8c87b8aa-b304-43cf-82b3-3a199367ec17	258613ea	Ghibli-style mountain with floating islands	1237	1703	530	Reddit	12.05	86	1545	2048x2048	
498	04dba0f4-fdb7-4341-8bb6-dca5a595cd81	23b56439	Cozy tea shop in a mystical town, Ghibli style	1852	1158	259	Twitter	5.86	56	2796	512x512	
499	1960cd2d-0e99-4e09-b426-62212dd8b37f	7dea460a	Mysterious temple hidden in a magical forest	3944	1059	149	Twitter	5.81	35	1314	1024x1024	

500 rows × 16 columns

Data Understanding

In [3]:
```
# print all the column names

data.columns
```

Out[3]:
```
Index(['image_id', 'user_id', 'prompt', 'likes', 'shares', 'comments',
       'platform', 'generation_time', 'gpu_usage', 'file_size_kb',
       'resolution', 'style_accuracy_score', 'is_hand_edited',
       'ethical_concerns_flag', 'creation_date', 'top_comment'],
      dtype='object')
```

In [4]:
```
# print the shape of the dataset

data.shape
```

Out[4]:
```
(500, 16)
```

In [5]:
```
# print all the statistics about the dataset

data.describe().T
```

Out[5]:

	count	mean	std	min	25%	50%	75%	max
likes	500.0	2601.26200	1429.433498	105.00	1343.5000	2566.50	3913.25	4944.00
shares	500.0	1040.18200	562.668738	13.00	587.7500	1092.00	1502.00	1999.00
comments	500.0	506.87200	283.384066	5.00	276.7500	518.00	744.25	998.00
generation_time	500.0	8.31778	3.903103	1.54	5.0275	8.38	11.54	14.99
gpu_usage	500.0	61.12400	18.151131	30.00	45.0000	63.00	77.00	90.00
file_size_kb	500.0	2511.82200	1390.178578	101.00	1374.7500	2498.00	3729.00	4973.00
style_accuracy_score	500.0	74.62600	14.679001	50.00	62.0000	74.00	87.25	100.00

In [6]:
```
# print the information about the dataset

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   image_id              500 non-null    object 
 1   user_id               500 non-null    object 
 2   prompt                500 non-null    object 
 3   likes                 500 non-null    int64  
 4   shares                500 non-null    int64  
 5   comments              500 non-null    int64  
 6   platform              500 non-null    object 
 7   generation_time       500 non-null    float64
 8   gpu_usage             500 non-null    int64  
 9   file_size_kb          500 non-null    int64  
 10  resolution            500 non-null    object 
 11  style_accuracy_score  500 non-null    int64  
 12  is_hand_edited        500 non-null    object 
 13  ethical_concerns_flag 500 non-null    object 
 14  creation_date         500 non-null    object 
 15  top_comment           500 non-null    object 
dtypes: float64(1), int64(6), object(9)
memory usage: 62.6+ KB
```

Remember all the datatypes and we changed it further for accurate Analysis.

Data-Cleaning If Any

In [7]:
```
# check the missing cells in dataset

data.isnull().sum()
```

Out[7]:
```
image_id                0
user_id                 0
prompt                  0
likes                   0
shares                  0
comments                0
platform                0
generation_time         0
gpu_usage               0
file_size_kb            0
resolution              0
style_accuracy_score    0
is_hand_edited          0
ethical_concerns_flag   0
creation_date           0
top_comment             0
dtype: int64
```

Data Analysis

Uni-Variate-Analysis

In [8]:
```
# we don't need the 'image_id', 'user_id' Column For Our Analysis Cause We Find the engagement and prompting of the user for
# creating ghibli style image.

data = data.drop(columns = ['image_id','user_id'])
```

In [9]:
```
# check if the column is deleted or not ?
data.head(1)
```

Out[9]:

	prompt	likes	shares	comments	platform	generation_time	gpu_usage	file_size_kb	resolution	style_accuracy_score	is_hand_edited
0	Studio Ghibli-inspired ocean with giant fish	916	410	555	Reddit	4.8	49	1684	1024x1024	89	Yes

In [10]:
```
# check the how many prompts are in our dataset
data.prompt.value_counts()
```

Out[10]:
```
prompt
Anime-style train passing through a fantasy world    57
Mysterious castle in the clouds, Ghibli-style        49
Ghibli-style mountain with floating islands          48
Cozy tea shop in a mystical town, Ghibli style       44
A lone traveler exploring an enchanted ruin          43
Ghibli-style night sky with glowing stars            43
Studio Ghibli-inspired ocean with giant fish         39
Spirited Away-style bustling market street           38
Mysterious temple hidden in a magical forest         38
Ghibli-style village at sunset                       34
Magical Ghibli forest with floating lanterns         34
Serene meadow with a tiny spirit creature            33
Name: count, dtype: int64
```

In [11]:
```
# let's find most frequent prompts

data.prompt.mode()
```

Out[11]:
```
0    Anime-style train passing through a fantasy world
Name: prompt, dtype: object
```

In [12]:
```
# check the how many types of comments in our dataset

data.top_comment.value_counts()
```

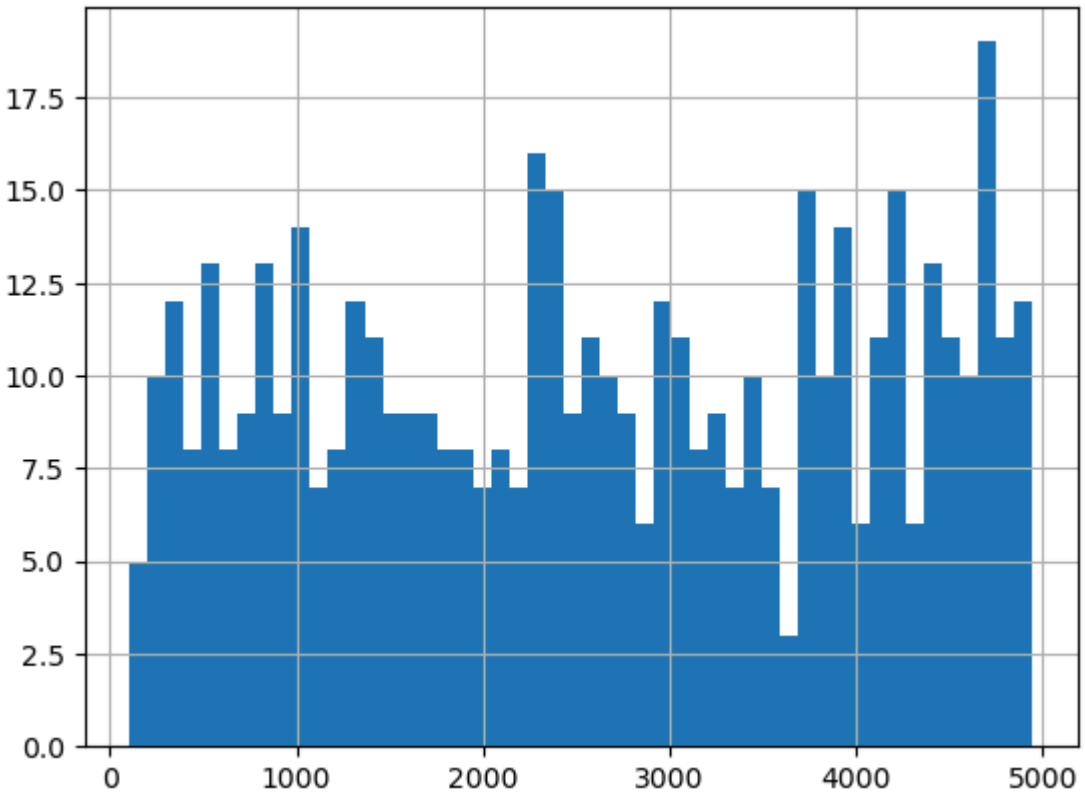
Out[12]:
```
top_comment
So nostalgic, feels like childhood memories. 🎥 #5019    1
AI art is getting too good! 🤖✨ #2799                     1
Absolutely stunning! Love the details. 🎨 #6674           1
This needs to be a real Ghibli film! #3895               1
The colors are so soft and dreamy! ❤️ #3365               1
                                                        ..
This is giving me serious Spirited Away vibes! #7888     1
I'd love to live in this world! #1442                    1
So nostalgic, feels like childhood memories. 🎥 #6534    1
Absolutely stunning! Love the details. 🎨 #7400           1
I'd love to live in this world! #5615                    1
Name: count, Length: 500, dtype: int64
```

In [13]:
```
# check the most common comment among the all the ghibli style images

data.top_comment.mode()
```

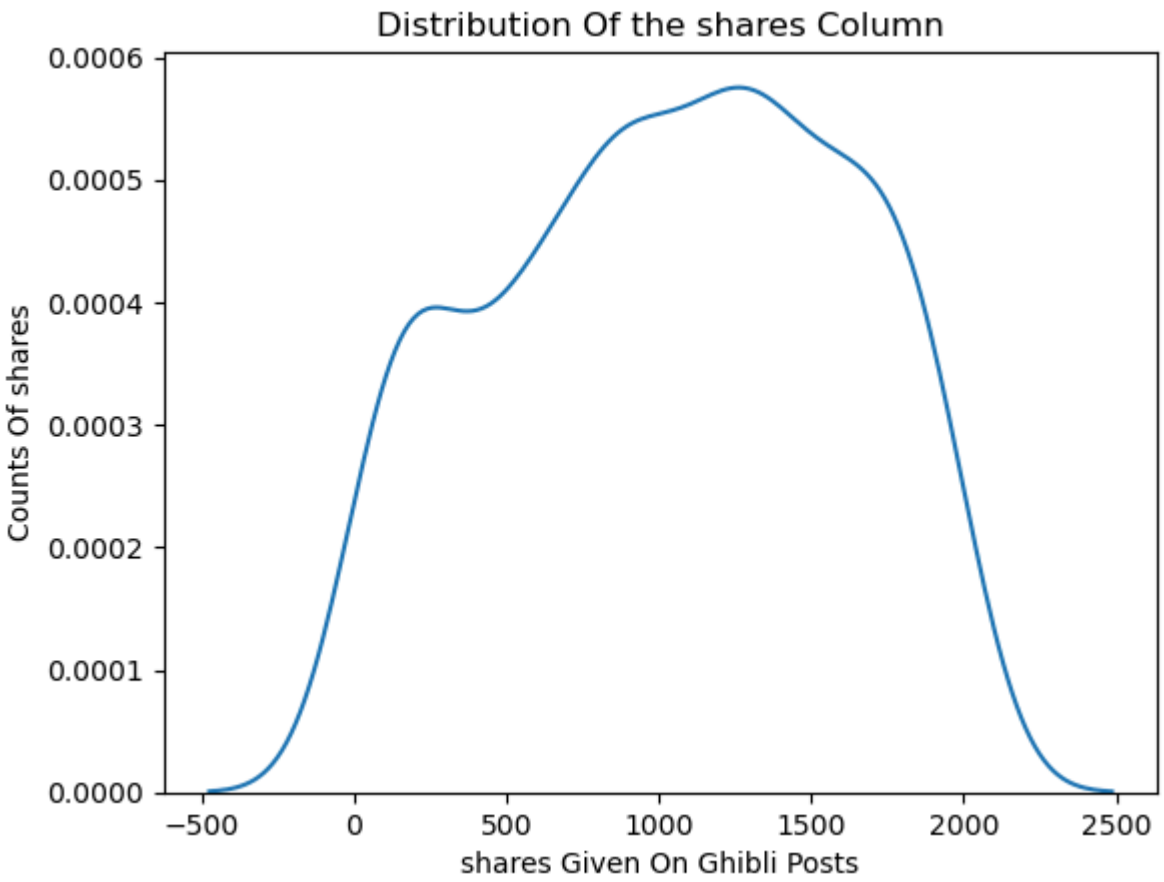
Out[13]:
```
0                  AI art is getting too good! 🤖✨ #1100
1                  AI art is getting too good! 🤖✨ #1119
2                  AI art is getting too good! 🤖✨ #1708
3                  AI art is getting too good! 🤖✨ #2177
4                  AI art is getting too good! 🤖✨ #2383
                           ...
495    This reminds me of Howl's Moving Castle! 🏰 #7843
496    This reminds me of Howl's Moving Castle! 🏰 #8094
497    This reminds me of Howl's Moving Castle! 🏰 #8740
498    This reminds me of Howl's Moving Castle! 🏰 #8819
499    This reminds me of Howl's Moving Castle! 🏰 #9911
Name: top_comment, Length: 500, dtype: object
```

In [14]:
```
# now the main important thing is check the outliers is present in our dataset if yes so remove it

data["likes"].hist(bins = 50)
```

Out[14]:
```
<Axes: >
```

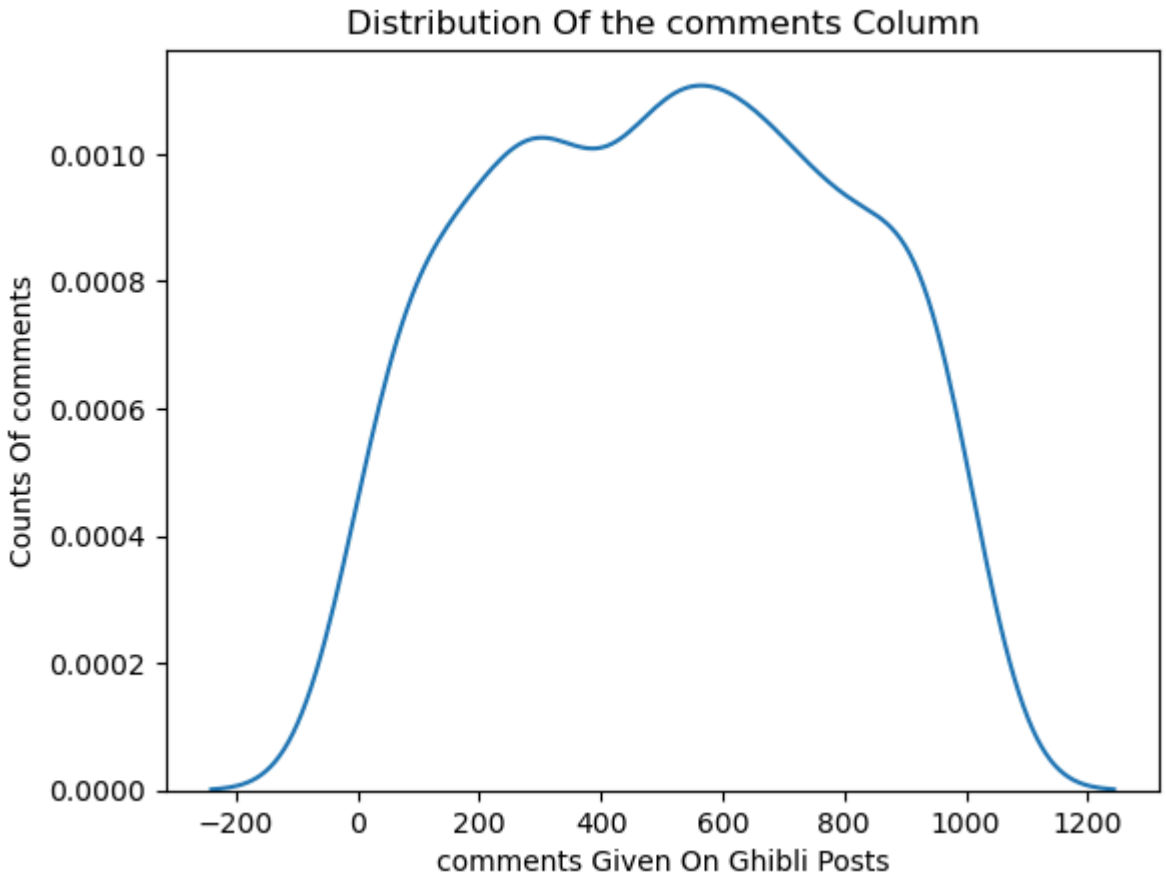


In [15]:
```
sns.kdeplot(x = 'shares',data = data)
plt.title("Distribution Of the shares Column")
plt.xlabel("shares Given On Ghibli Posts")
plt.ylabel("Counts Of shares")
plt.show()
```



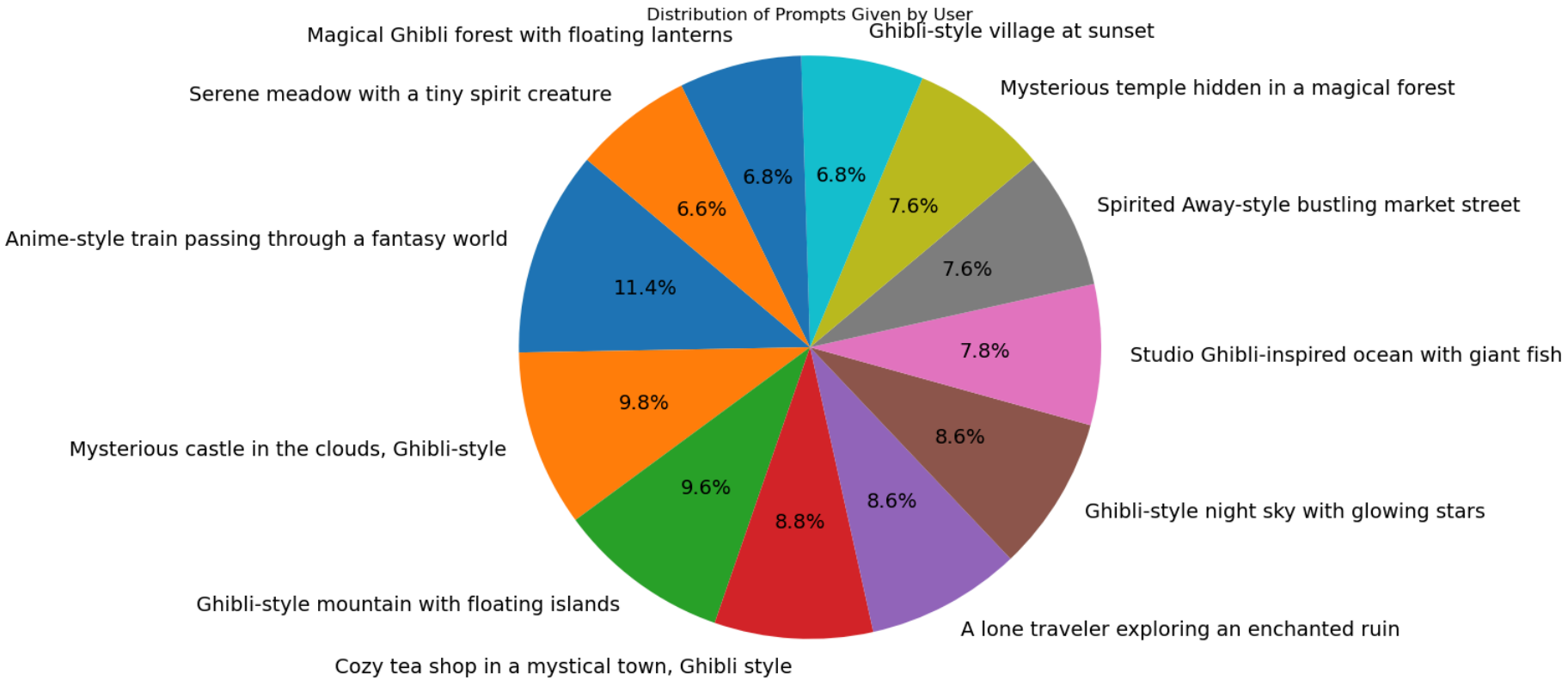
Here You Can See the Likes Column Not Generated The proper Bell Shape Curve Means The Data Is Averagely Distributed Not Much properly.

In [16]:
```
sns.kdeplot(x = 'comments',data = data)
plt.title("Distribution Of the comments Column")
plt.xlabel("comments Given On Ghibli Posts")
plt.ylabel("Counts Of comments")
plt.show()
```



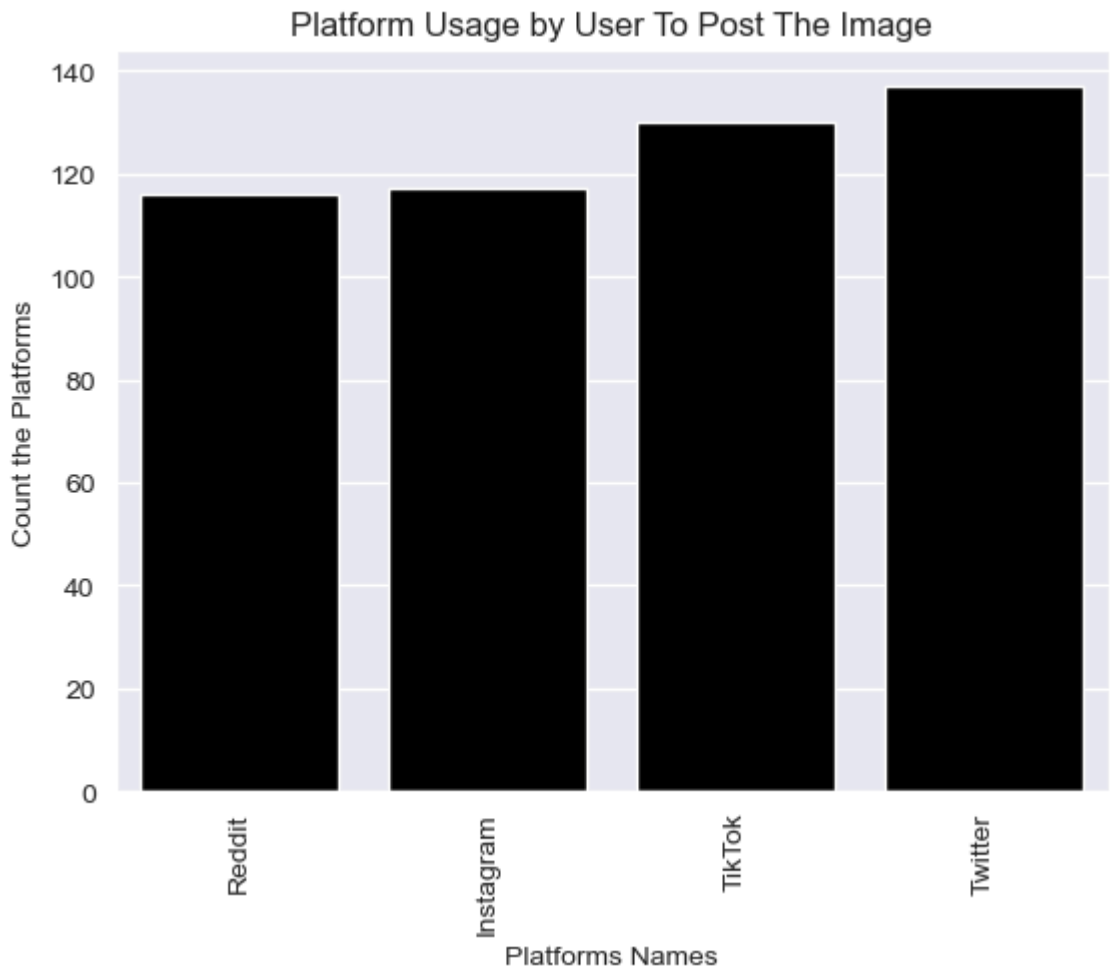
Here You Can See the Likes Column Not Generated The proper Bell Shape Curve Means The Data Is Averagely Distributed Not Much properly.

In [17]:
```
prompt_counts = data['prompt'].value_counts()

plt.figure(figsize=(8, 8))
plt.pie(prompt_counts, labels=prompt_counts.index, autopct='%1.1f%%', startangle=140,textprops={'color': 'black', 'fontsize':
plt.title("Distribution of Prompts Given by User")
plt.axis('equal')  # Equal aspect ratio ensures the pie chart is a circle
plt.show()
```



You can see the most using prompt by user is "Anime-Style train passing through a fantasy world"

In [18]:
```
# let's count the platform to create the Ghibli Image.

sns.set_style("darkgrid")
sns.countplot(x = "platform",data = data,color = "black")
plt.title("Platform Usage by User To Post The Image")
plt.xlabel("Platforms Names")
plt.ylabel("Count the Platforms")
plt.xticks(rotation = "vertical")
plt.show()
```

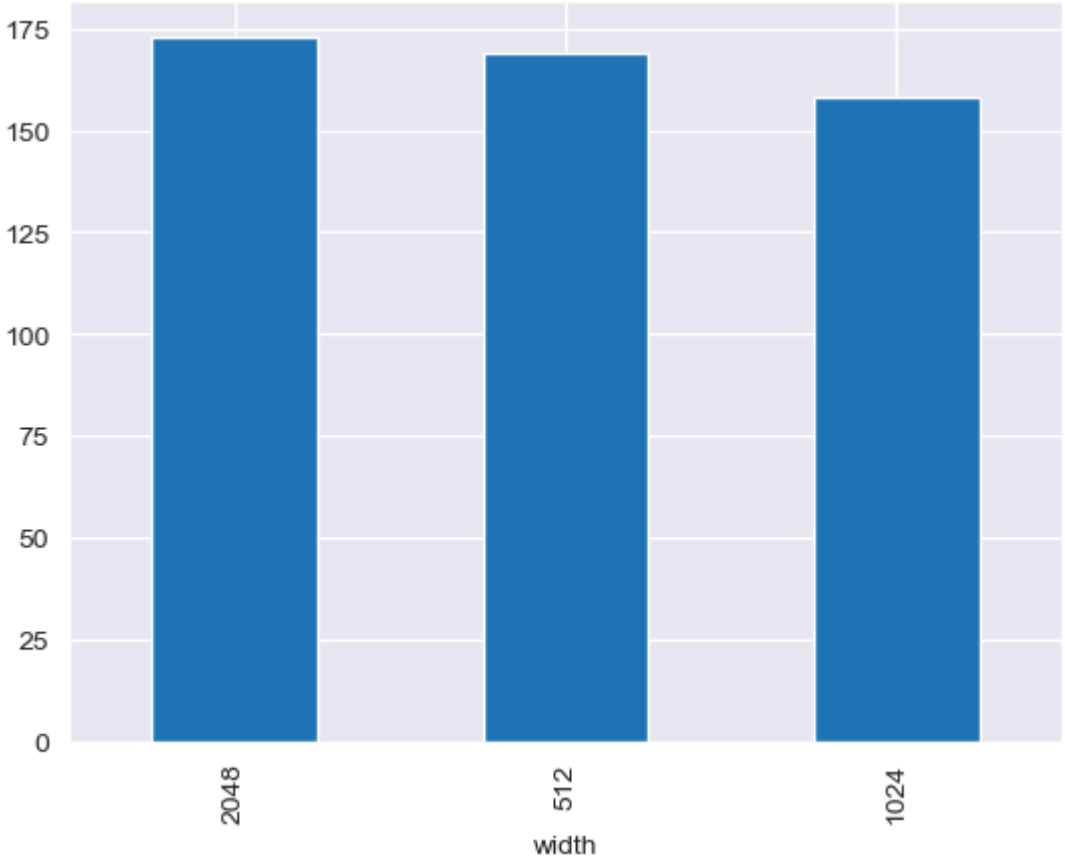


As per the Countplot the Twitter Platform Is most trending to post Ghibli Style Images.

In [19]:
```
# let's split the resolution column so we get better understand which width & heghit types images is mostly generates.

data[['width', 'height']] = data['resolution'].str.split('x', expand=True).astype(int)
```

In [20]:
```
# let's count which width types of images is mostly generated.

data.width.value_counts().plot(kind = "bar")
```

Out[20]: <Axes: xlabel='width'>

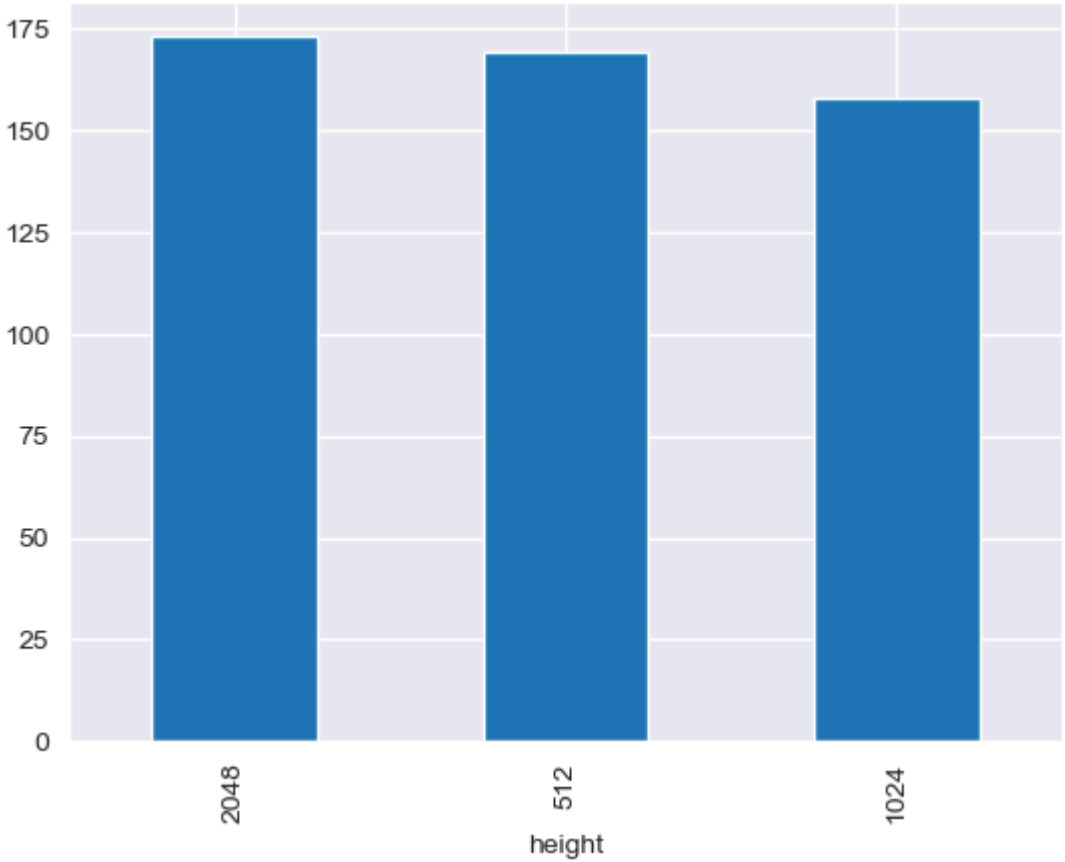


In above bar chart you can see clearly the 2048 width type images is mostly generated.

In [21]:
```
# let's count which height types of images is mostly generated.
```

```
data.height.value_counts().plot(kind = "bar")
```

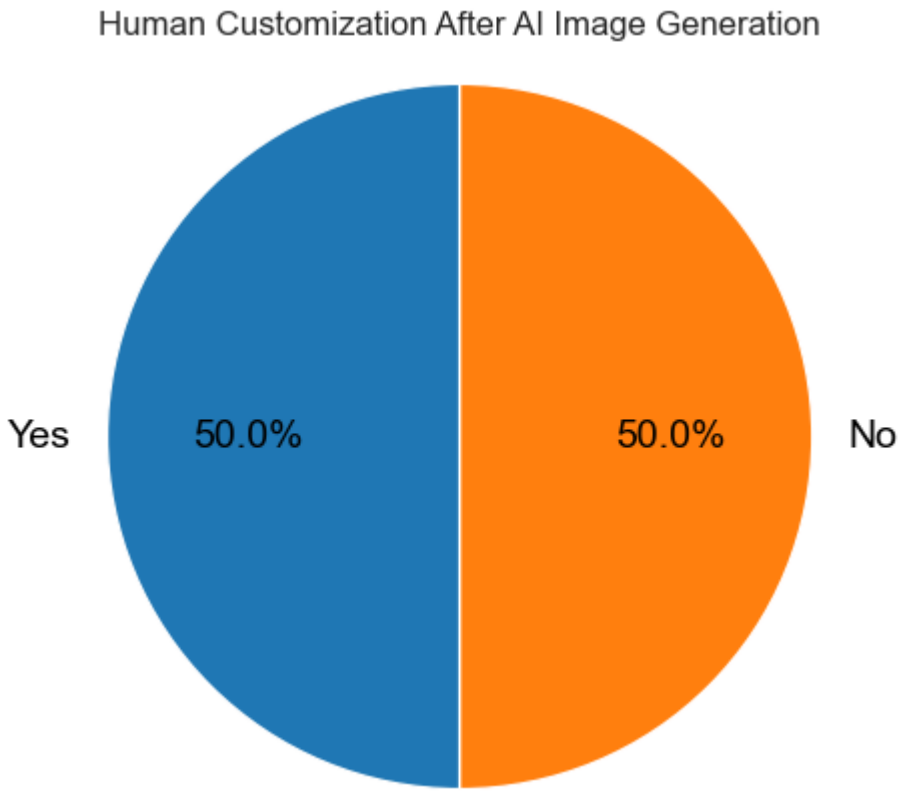
Out[21]: <Axes: xlabel='height'>



Now again the winner is you can see clearly the 2048 width type images is mostly generated.

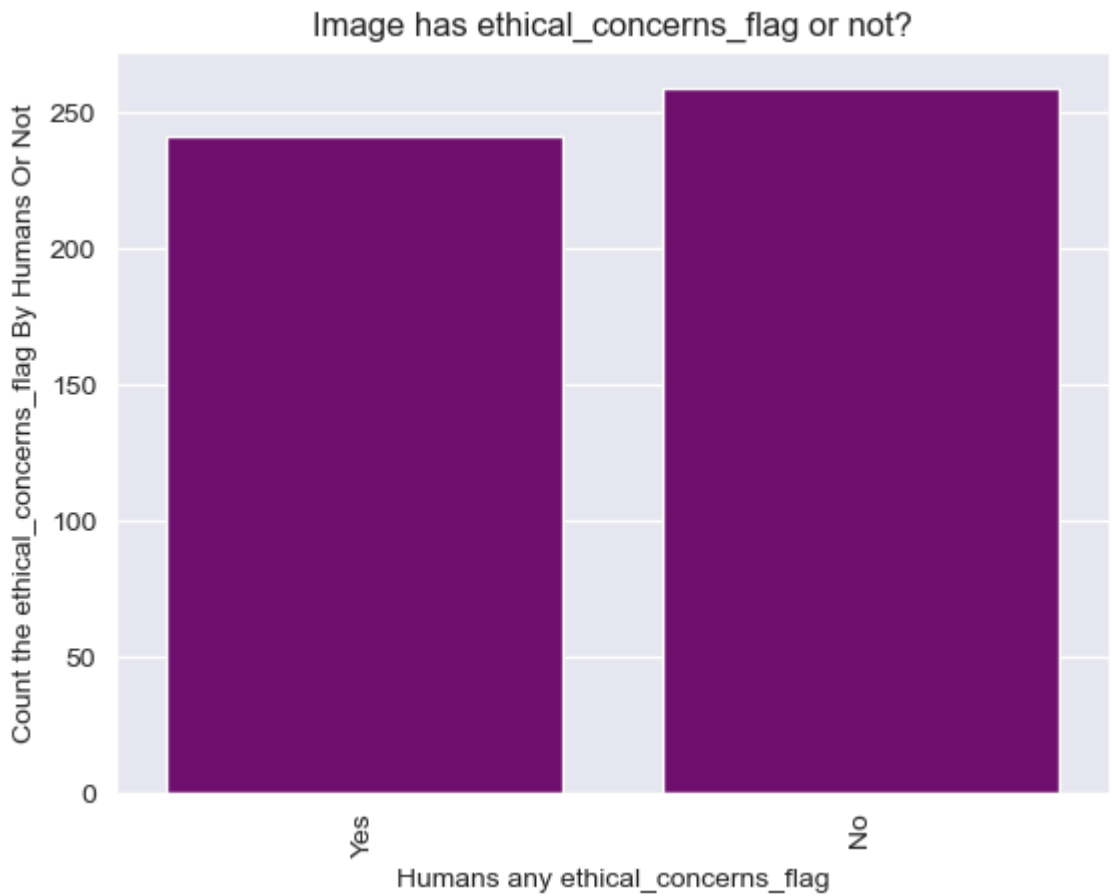
In [22]:
```
edit_counts = data['is_hand_edited'].value_counts()

plt.figure(figsize=(5,5))
plt.pie(edit_counts, labels=edit_counts.index, autopct='%1.1f%%', startangle=90,textprops={'color': 'black', 'fontsize': 14} )
plt.title("Human Customization After AI Image Generation")
```

```
plt.axis('equal')
plt.show()
```



Here the ration is 50-50% some people wants to editting in AI Generated Images and some peoples Wants to keep as it is.

In [23]:
```
sns.set_style("darkgrid")
sns.countplot(x = "ethical_concerns_flag",data = data,color = "purple")
plt.title("Image has ethical_concerns_flag or not?")
plt.xlabel("Humans any ethical_concerns_flag")
plt.ylabel("Count the ethical_concerns_flag By Humans Or Not")
plt.xticks(rotation = "vertical")
plt.show()
```



Most of the users has no ethical_concerns_flag to create the Ghibli style image.

In [24]:
```
# print the generation_time column

data.generation_time
```

Out[24]:
```
0       4.80
1      11.11
2       5.56
3      12.45
4       4.80
       ...
495    10.56
496     8.41
497    12.05
498     5.86
499     5.81
Name: generation_time, Length: 500, dtype: float64
```

In [25]:
```
# categorize the generation_time column

def generationtime_cat(x):
    if(x<=3):
        return "Very_Fast_Image_Generation"
    elif(x>3 and x<=6):
        return "Fast_Image_Generation"
    elif(x>6 and x<=10):
        return "Slow_Image_Generation"
    else:
        return "Very_slow_Image_Generation"

data["GenerationTime_Cat"] = data["generation_time"].apply(generationtime_cat)
```

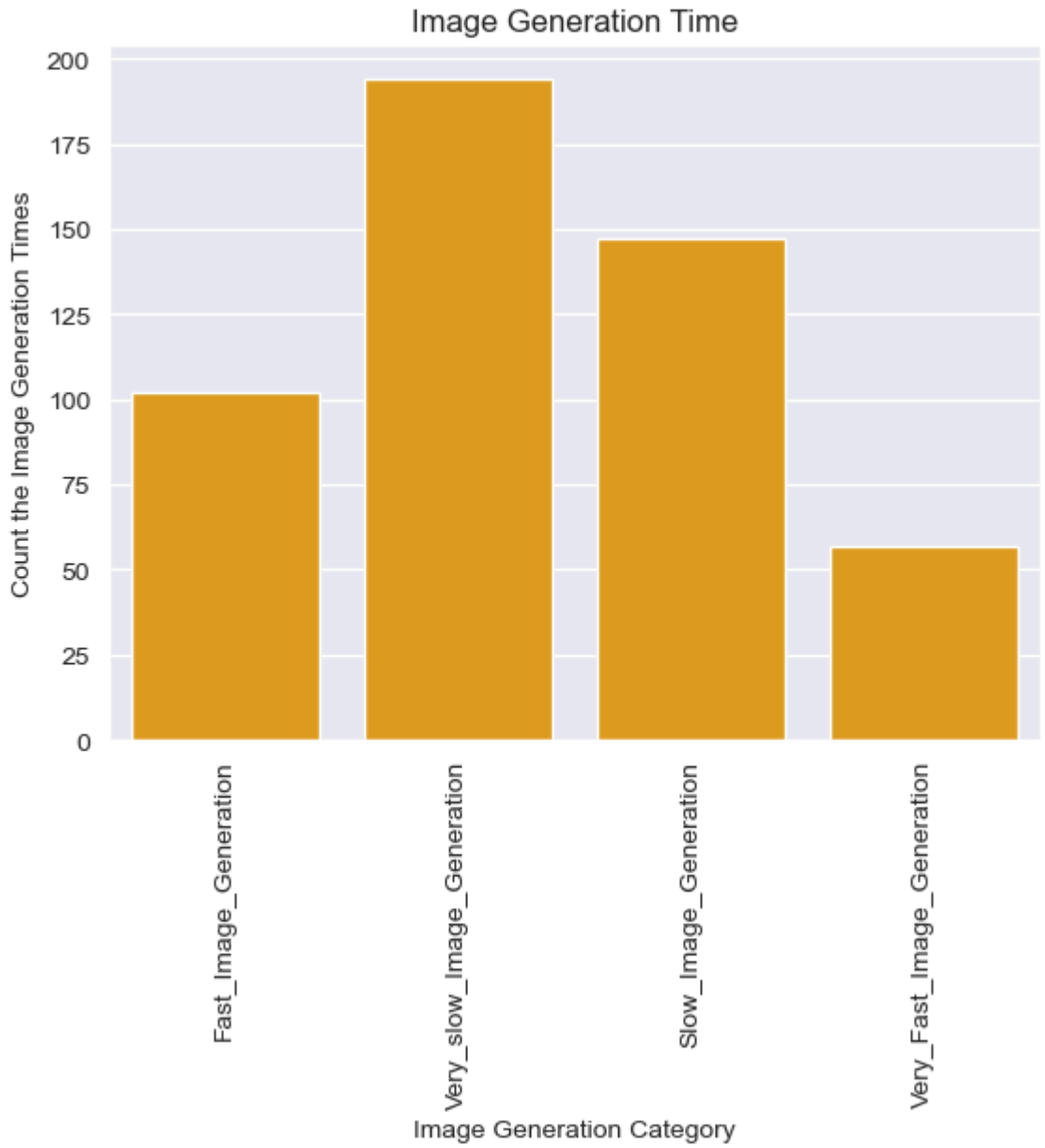
In [26]:
```
# print the data["GenerationTime_Cat"] column

data["GenerationTime_Cat"]
```

Out[26]:
```
0           Fast_Image_Generation
1      Very_slow_Image_Generation
2           Fast_Image_Generation
3      Very_slow_Image_Generation
4           Fast_Image_Generation
                   ...
495    Very_slow_Image_Generation
496         Slow_Image_Generation
497    Very_slow_Image_Generation
498         Fast_Image_Generation
499         Fast_Image_Generation
Name: GenerationTime_Cat, Length: 500, dtype: object
```

In [27]:
```
# count the values

data["GenerationTime_Cat"].value_counts()
```

Out[27]:
```
GenerationTime_Cat
Very_slow_Image_Generation    194
Slow_Image_Generation         147
Fast_Image_Generation         102
Very_Fast_Image_Generation     57
Name: count, dtype: int64
```

In [28]:
```
sns.set_style("darkgrid")
sns.countplot(x = "GenerationTime_Cat",data = data,color = "orange")
plt.title("Image Generation Time")
plt.xlabel("Image Generation Category")
plt.ylabel("Count the Image Generation Times")
plt.xticks(rotation = "vertical")
plt.show()
```



Most of the image is generated in a very slow image generation time.

In [29]:
```
data.gpu_usage.value_counts()
```

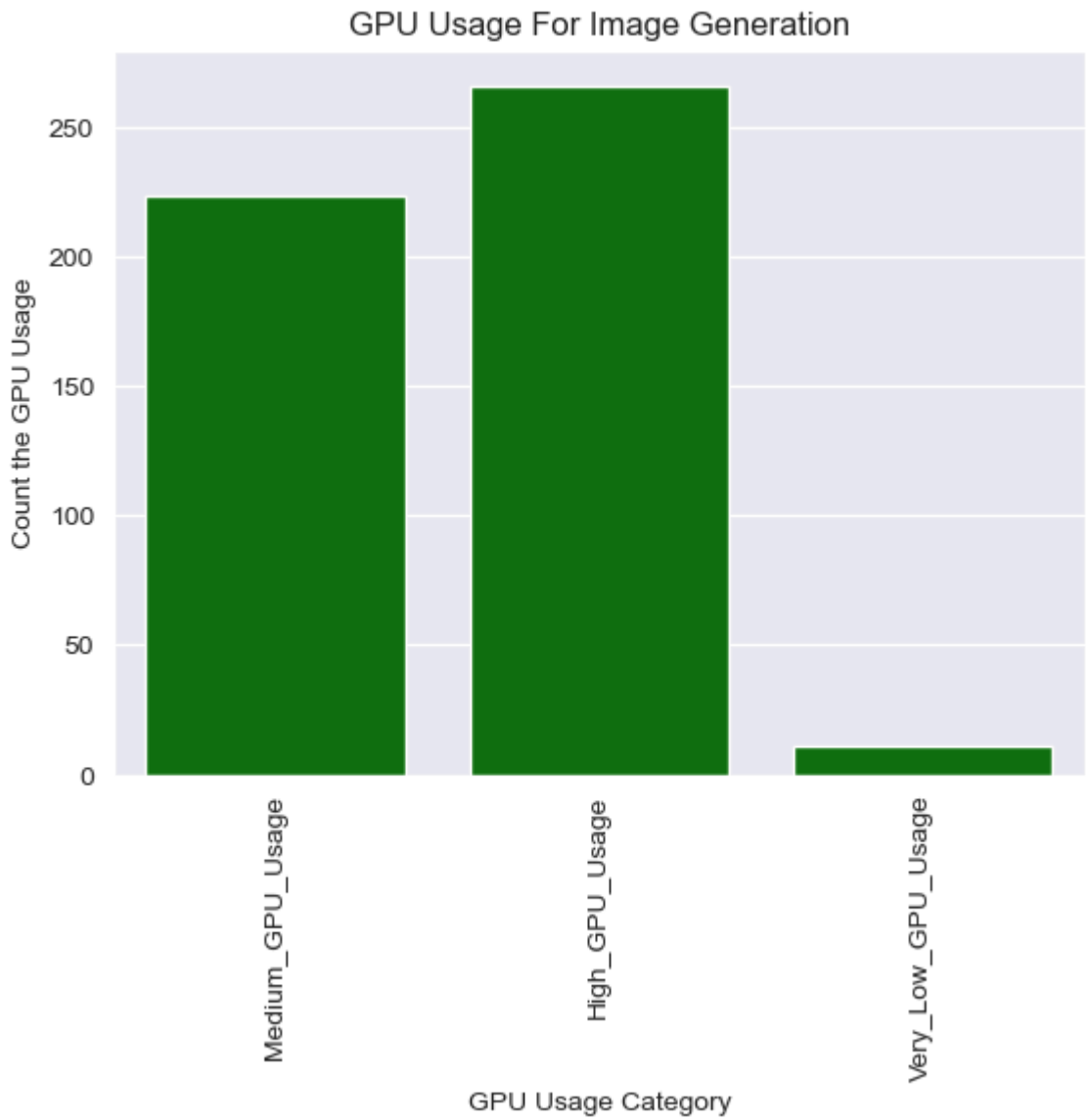
Out[29]:
```
gpu_usage
88    15
85    15
86    14
83    13
69    12
      ..
68     4
89     4
45     4
38     4
71     4
Name: count, Length: 61, dtype: int64
```

In [30]:
```
# categorize the GPU Usage time column

def gputime_cat(x):
    if(x<=30):
        return "Very_Low_GPU_Usage"
    elif(x>30 and x<=60):
        return "Medium_GPU_Usage"
    elif(x>60 and x<=100):
        return "High_GPU_Usage"
    else:
        return "Very_High_GPU_Usage"

data["Gputime_cat"] = data["gpu_usage"].apply(gputime_cat)
```

In [31]:
```
# print the GPU_Usage column

data.Gputime_cat
```

Out[31]:
```
0      Medium_GPU_Usage
1        High_GPU_Usage
2      Medium_GPU_Usage
3        High_GPU_Usage
4        High_GPU_Usage
            ...
495      High_GPU_Usage
496      High_GPU_Usage
497      High_GPU_Usage
498    Medium_GPU_Usage
499    Medium_GPU_Usage
Name: Gputime_cat, Length: 500, dtype: object
```

In [32]:
```
sns.set_style("darkgrid")
sns.countplot(x = "Gputime_cat",data = data,color = "green")
plt.title("GPU Usage For Image Generation")
plt.xlabel("GPU Usage Category")
plt.ylabel("Count the GPU Usage")
plt.xticks(rotation = "vertical")
plt.show()
```



Most of the image Gpu usage in a high gpu usage category.

In [33]:
```
data.file_size_kb
```

Out[33]:
```
0      1684
1      2808
2      1800
3       479
4      1789
       ...
495    3255
496    4710
497    1545
498    2796
499    1314
Name: file_size_kb, Length: 500, dtype: int64
```

In [34]:
```
def file_size_kb_cat(x):
    x = x/1000
    if(x<=1):
        return "Low_File_Size_In_MB"
    elif(x>1 and x<=3):
        return "Medium_File_Size_In_MB"
    else:
        return "High_File_Size_In_MB"

data["File_size_mb_cat"] = data["file_size_kb"].apply(file_size_kb_cat)
```

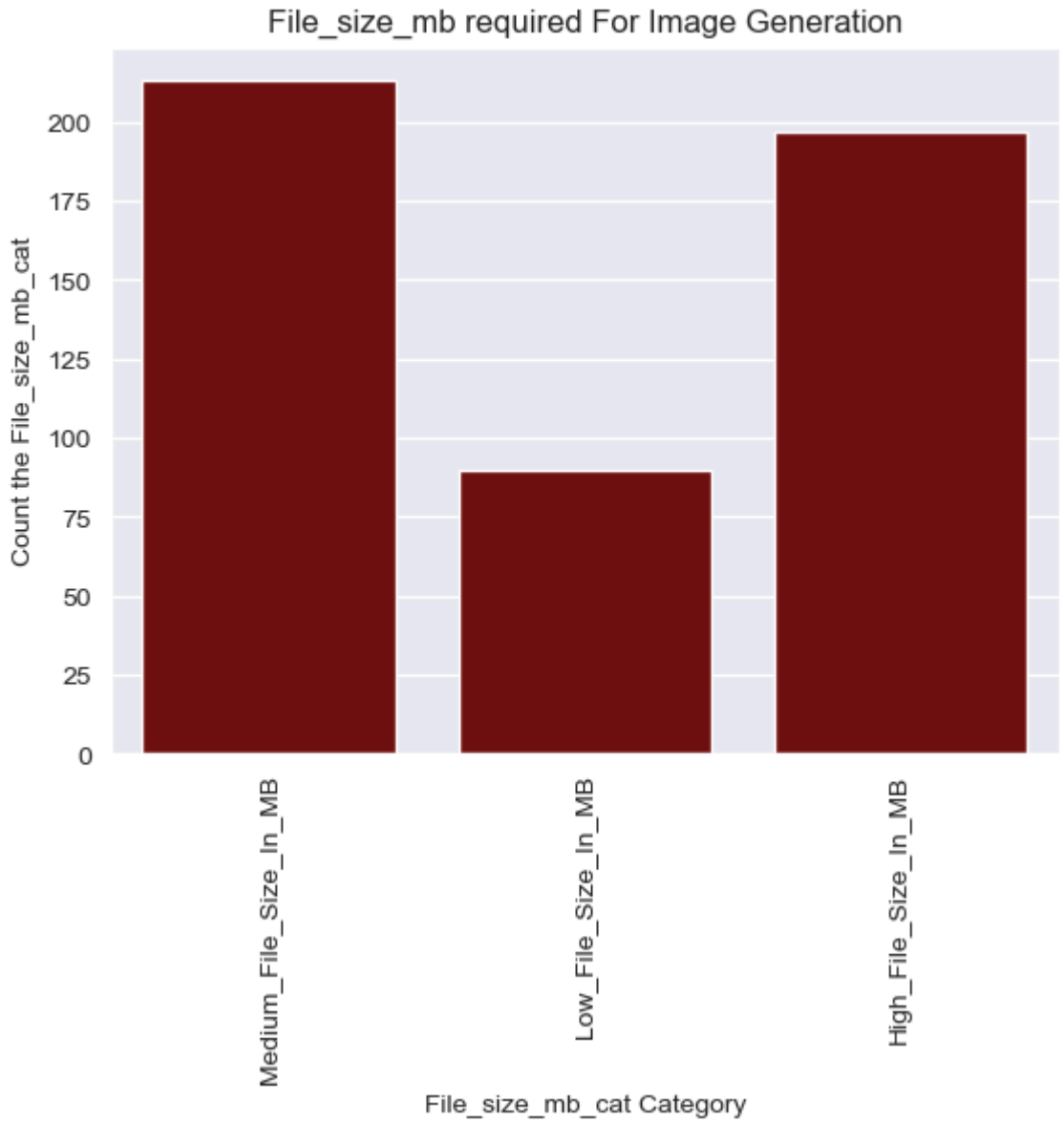
In [35]:
```
data["File_size_mb_cat"]
```

Out[35]:
```
0      Medium_File_Size_In_MB
1      Medium_File_Size_In_MB
2      Medium_File_Size_In_MB
3         Low_File_Size_In_MB
4      Medium_File_Size_In_MB
                ...
495      High_File_Size_In_MB
496      High_File_Size_In_MB
497    Medium_File_Size_In_MB
498    Medium_File_Size_In_MB
499    Medium_File_Size_In_MB
Name: File_size_mb_cat, Length: 500, dtype: object
```

In [36]:
```
data["File_size_mb_cat"].value_counts()
```

Out[36]:
```
File_size_mb_cat
Medium_File_Size_In_MB    213
High_File_Size_In_MB      197
Low_File_Size_In_MB        90
Name: count, dtype: int64
```

In [37]:
```
sns.set_style("darkgrid")
sns.countplot(x = "File_size_mb_cat",data = data,color = "maroon")
plt.title("File_size_mb required For Image Generation")
plt.xlabel("File_size_mb_cat Category")
plt.ylabel("Count the File_size_mb_cat")
plt.xticks(rotation = "vertical")
plt.show()
```



Most of the images is generated in the medium file size in mb means majority of the images are generated in the category of 1 to 3 mb.

In [38]:
```
data.style_accuracy_score
```

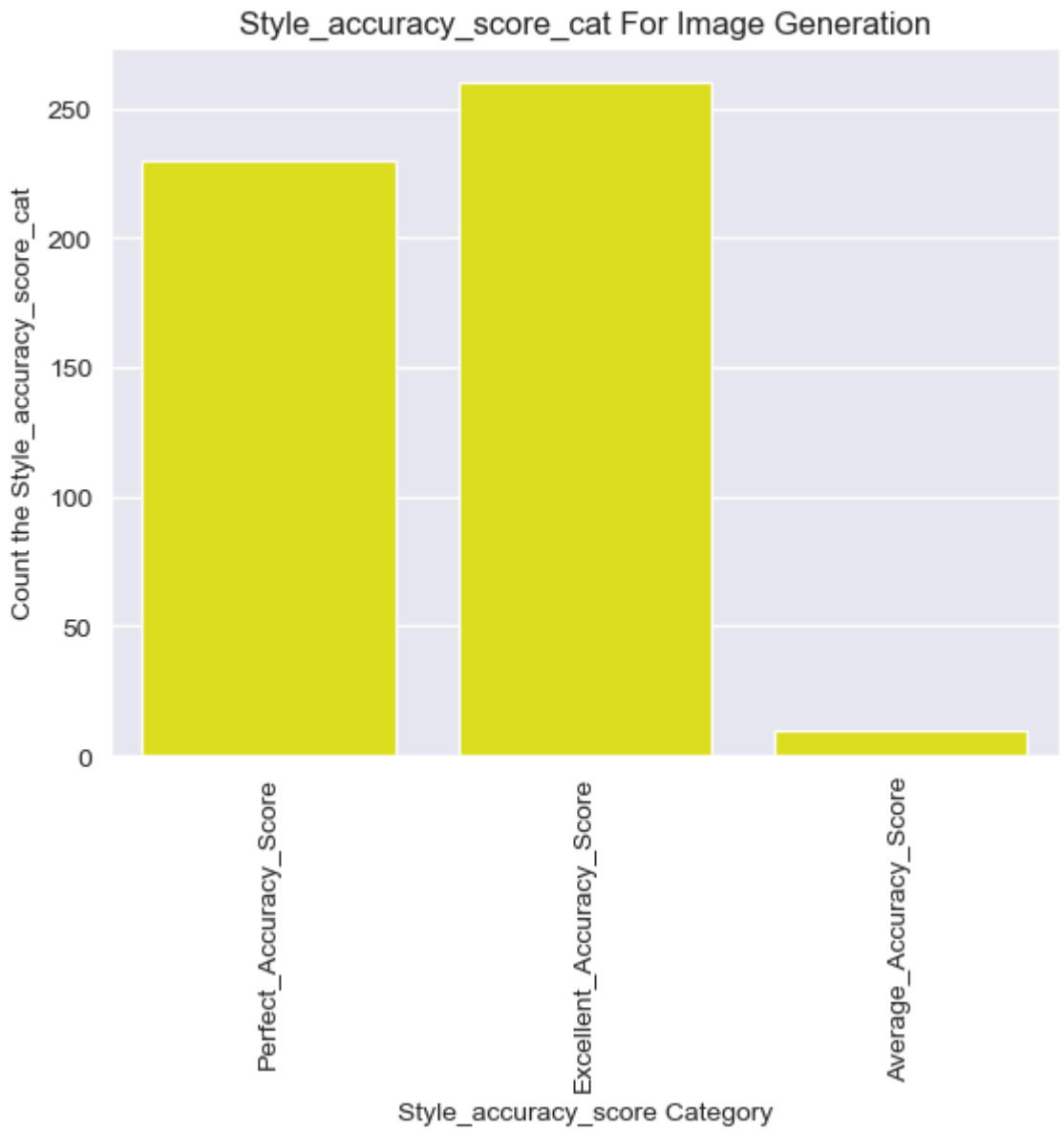
Out[38]:
```
0      89
1      92
2      61
3      76
4      58
       ..
495    96
496    83
497    81
498    78
499    82
Name: style_accuracy_score, Length: 500, dtype: int64
```

In [39]:
```
# let's convert the style accuracy score into category.

def style_accuracy_score_cat(x):
    if(x<=25):
        return "Poor_Accuracy_Score"
    elif(x>25 and x<=50):
        return "Average_Accuracy_Score"
    elif(x>50 and x<=75):
        return "Excellent_Accuracy_Score"
    else:
        return "Perfect_Accuracy_Score"


data["Style_accuracy_score_cat"] = data["style_accuracy_score"].apply(style_accuracy_score_cat)
```

In [40]:
```
data["Style_accuracy_score_cat"]
```

Out[40]:
```
0        Perfect_Accuracy_Score
1        Perfect_Accuracy_Score
2      Excellent_Accuracy_Score
3        Perfect_Accuracy_Score
4      Excellent_Accuracy_Score
                 ...
495      Perfect_Accuracy_Score
496      Perfect_Accuracy_Score
497      Perfect_Accuracy_Score
498      Perfect_Accuracy_Score
499      Perfect_Accuracy_Score
Name: Style_accuracy_score_cat, Length: 500, dtype: object
```

In [41]:
```
sns.set_style("darkgrid")
sns.countplot(x = "Style_accuracy_score_cat",data = data,color = "yellow")
plt.title("Style_accuracy_score_cat For Image Generation")
plt.xlabel("Style_accuracy_score Category")
plt.ylabel("Count the Style_accuracy_score_cat")
plt.xticks(rotation = "vertical")
plt.show()
```



Most of the image is generated and user gives Excellent Accuracy Score.

In [42]:
```
# let's check the most frequent comments

data.top_comment.value_counts().sort_values(ascending = False).head(1)
```

Out[42]:
```
top_comment
So nostalgic, feels like childhood memories. 🎥 #5019    1
Name: count, dtype: int64
```

the top comment is So nostalgic, feels like childhood memories. 🎥 #5019 on Ghibli Image.

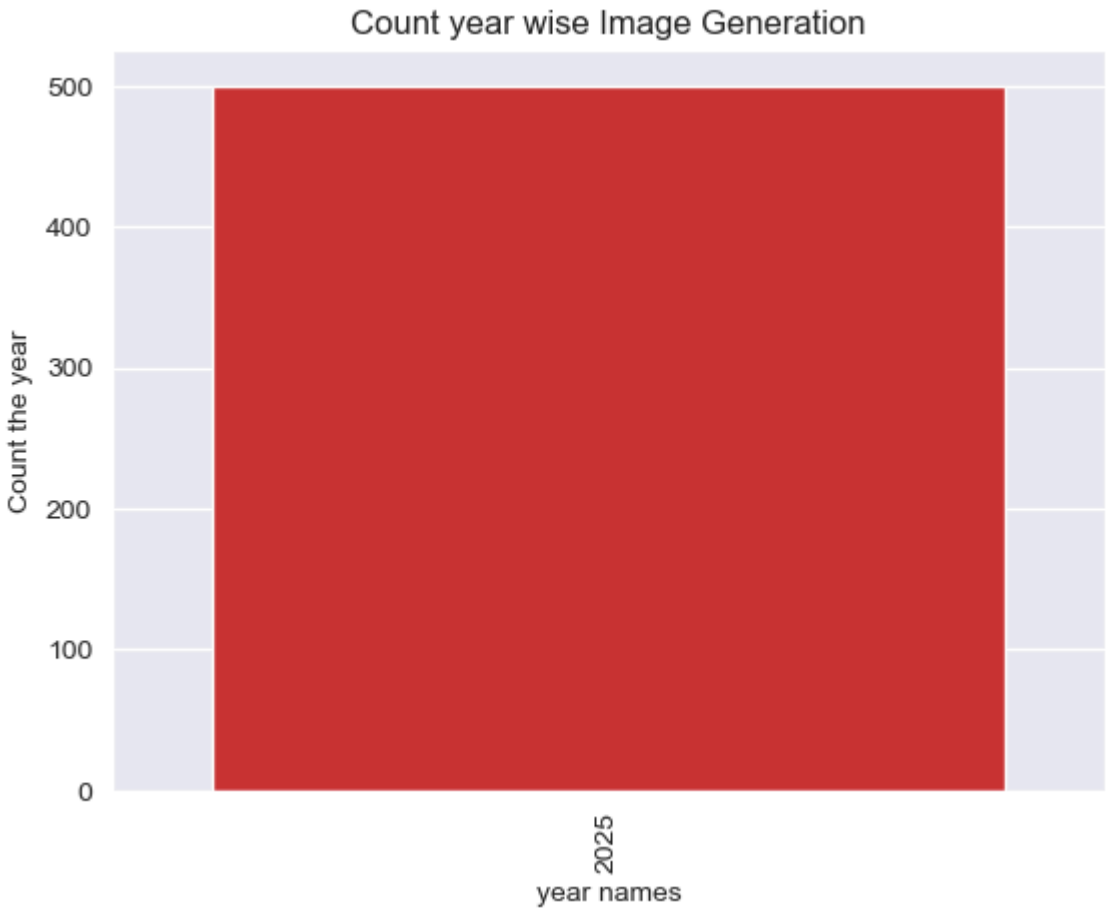
In [43]:
```
# let's convert the dtype of the creation_date

data["creation_date"] = pd.to_datetime(data["creation_date"])
```

In [44]:
```
# cross check the dtype of the creation_date column

data.creation_date.dtypes
```

Out[44]:
```
dtype('<M8[ns]')
```

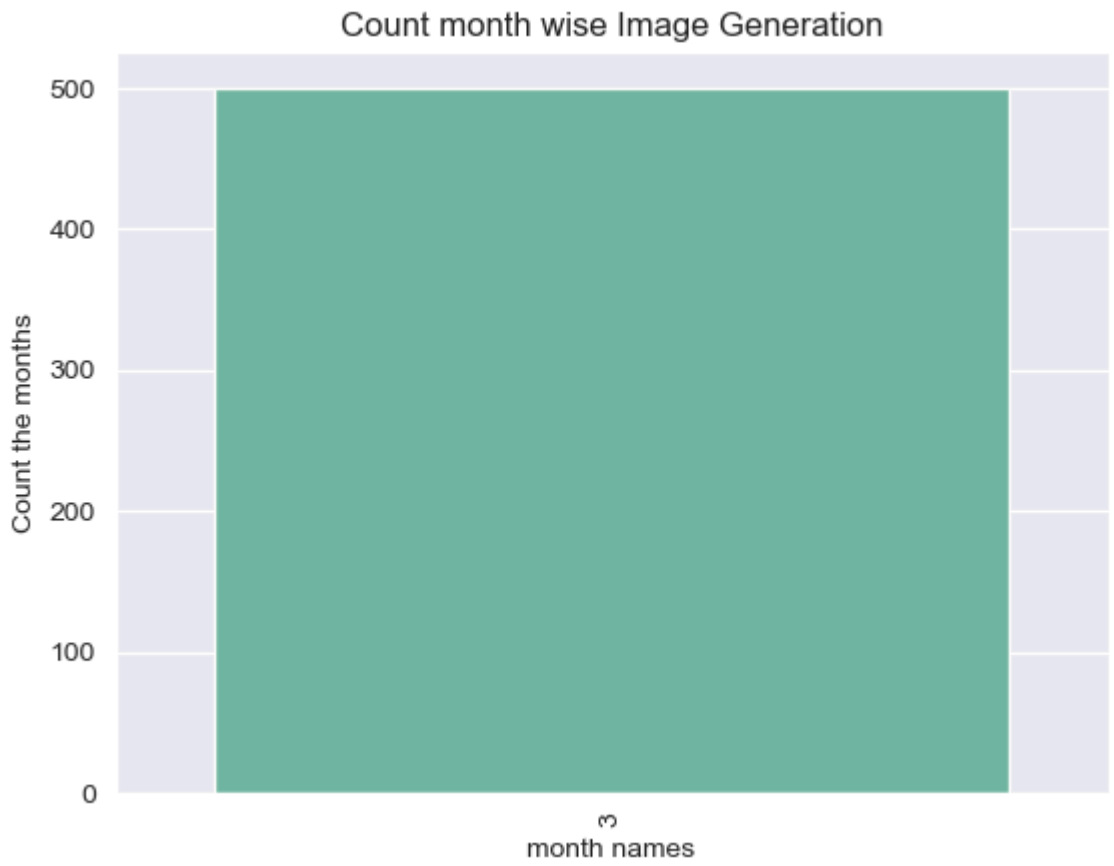
In [45]:
```
# so now convert the year,month & day & day-name column for deeper analysis

data["year"] = data["creation_date"].dt.year
data["month"] = data["creation_date"].dt.month
data["day"] = data["creation_date"].dt.day
data["day-name"] = data["creation_date"].dt.day_name()
```

In [46]:
```
sns.set_style("darkgrid")
sns.countplot(x = "year",data = data,palette = "Set1")
plt.title("Count year wise Image Generation")
plt.xlabel("year names")
plt.ylabel("Count the year")
plt.xticks(rotation = "vertical")
plt.show()
```



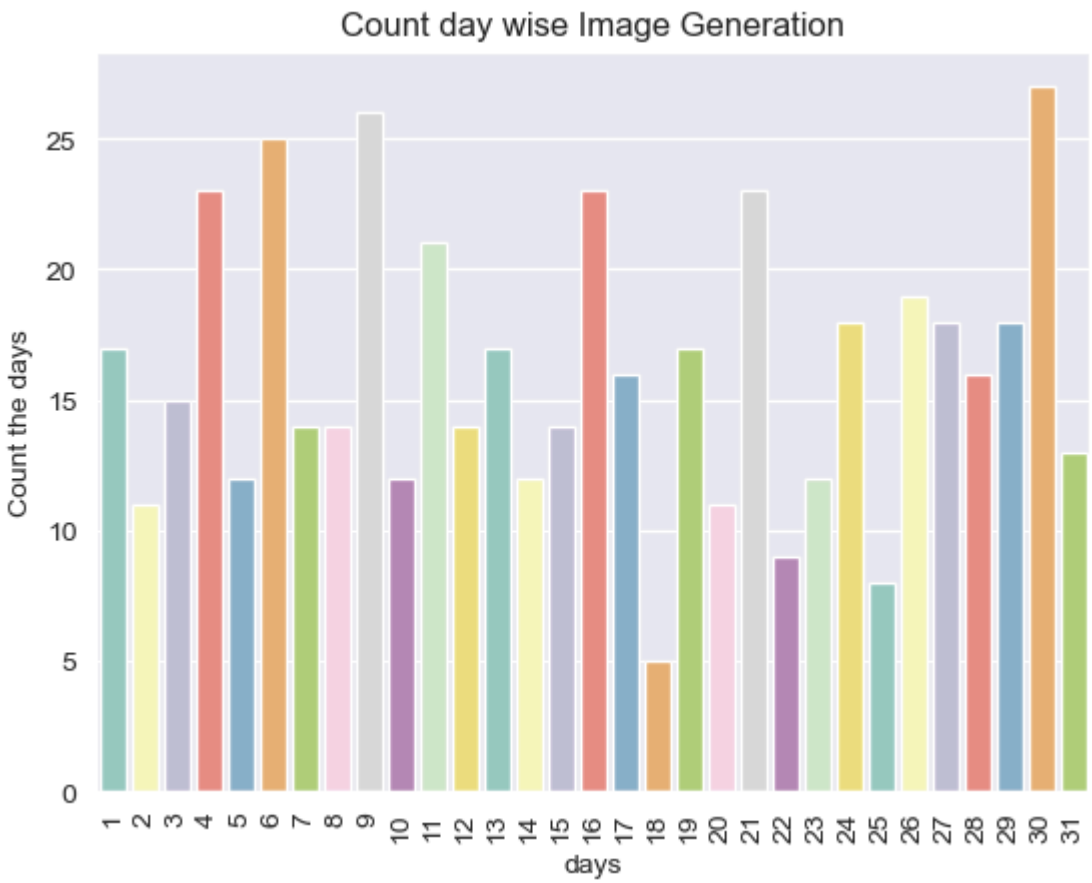
This dataset is contains 2025 records so only year is 2025.

In [47]:
```
sns.set_style("darkgrid")
sns.countplot(x = "month",data = data,palette = "Set2")
plt.title("Count month wise Image Generation")
plt.xlabel("month names")
plt.ylabel("Count the months")
plt.xticks(rotation = "vertical")
plt.show()
```



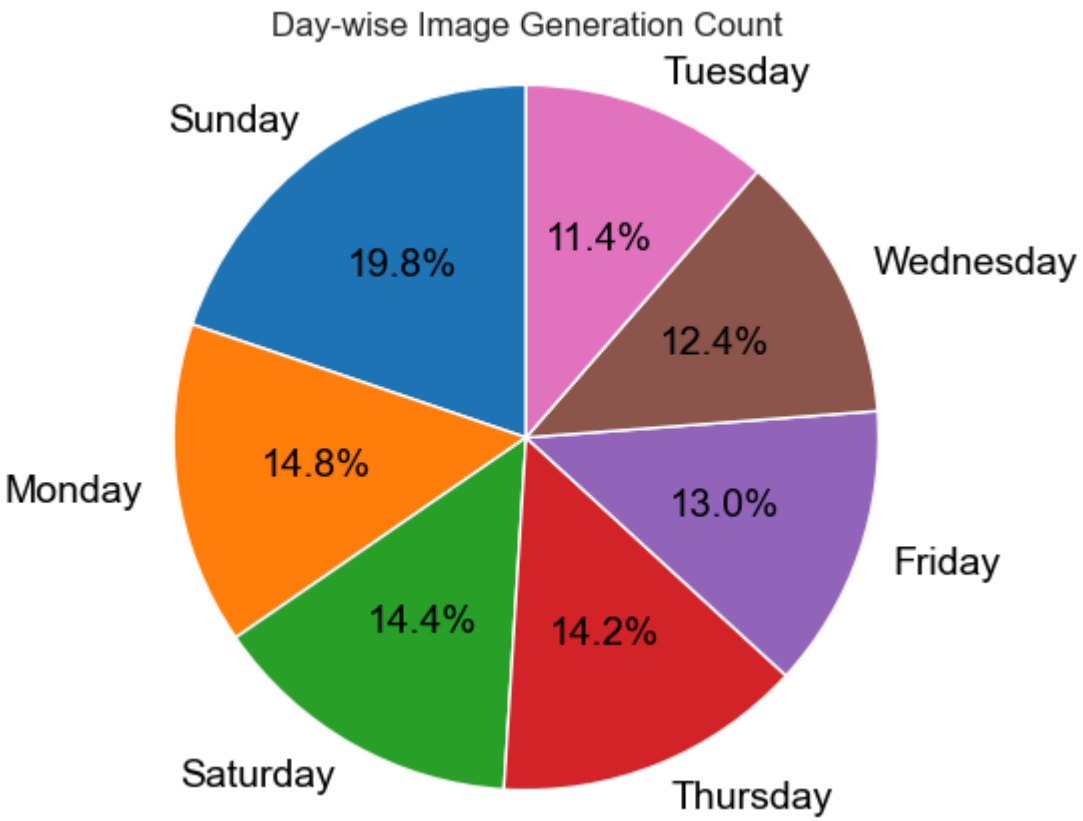
All the images are generated in the month of March.

In [48]:
```
sns.set_style("darkgrid")
sns.countplot(x = "day",data = data,palette = "Set3")
plt.title("Count day wise Image Generation")
plt.xlabel("days")
plt.ylabel("Count the days")
plt.xticks(rotation = "vertical")
plt.show()
```



Most of the images are generated on 3rd & 30th March.

In [49]:
```
day_counts = data['day-name'].value_counts()

plt.figure(figsize=(5,5))
plt.pie(day_counts, labels=day_counts.index, autopct='%1.1f%%', startangle=90,textprops={'color': 'black', 'fontsize': 14} )
plt.title("Day-wise Image Generation Count")
plt.axis('equal')  # Keep the pie chart circular
plt.show()
```



Most of the images are generated on Sunday.

Word Frequency Count

In [50]:
```
from collections import Counter
import nltk
from nltk.corpus import stopwords

# Download stopwords (only needed once)
nltk.download('stopwords')
```

```
# Get English stopwords
stop_words = set(stopwords.words('english'))

# Create an empty list to store words
all_words = []

# Extend all words from the prompt column into one list
for prompt_text in data['prompt']:  # Loop through each entry in the prompt column
    all_words.extend(prompt_text.split())  # Split the text into words and extend the list

# Filter out stopwords and single-character words
filtered_words = [word for word in all_words if word.lower() not in stop_words and len(word) > 1]

# Count the frequency of each word
word_freq = Counter(filtered_words)

# Print the 5 most common words
print(word_freq.most_common(5))
```

```
[('Ghibli-style', 174), ('Mysterious', 87), ('floating', 82), ('Ghibli', 78), ('forest', 72)]
```

```
[nltk_data] Downloading package stopwords to C:\Users\VIVEK
[nltk_data]     CHAUHAN\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

You can see the 'Ghibli-style','Mysterious','with' is the most common words while prompting.

In [100…

```
from collections import Counter
import nltk
from nltk.corpus import stopwords

# Ensure stopwords are downloaded (only needed once)
nltk.download('stopwords')

# Get English stopwords
stop_words = set(stopwords.words('english'))

# Create an empty list to store words
all_comment_words = []
```

```
# Loop through each entry in the top_comment column
for comment_text in data['top_comment']:
    if isinstance(comment_text, str):  # Ensure it's a string
        all_comment_words.extend(comment_text.split())  # Split into words

# Filter out stopwords, single-character words, and numeric words
filtered_comment_words = [word for word in all_comment_words
                          if word.lower() not in stop_words
                          and len(word) > 1
                          and not word.isdigit()  # Exclude numeric words
                          and word.isalpha()]  # Only alphabetic words

# Count the frequency of each word
comment_word_freq = Counter(filtered_comment_words)

# Print the 5 most common words
print(comment_word_freq.most_common(5))
```

```
[('Ghibli', 95), ('AI', 76), ('believe', 54), ('lighting', 53), ('atmosphere', 53)]
```

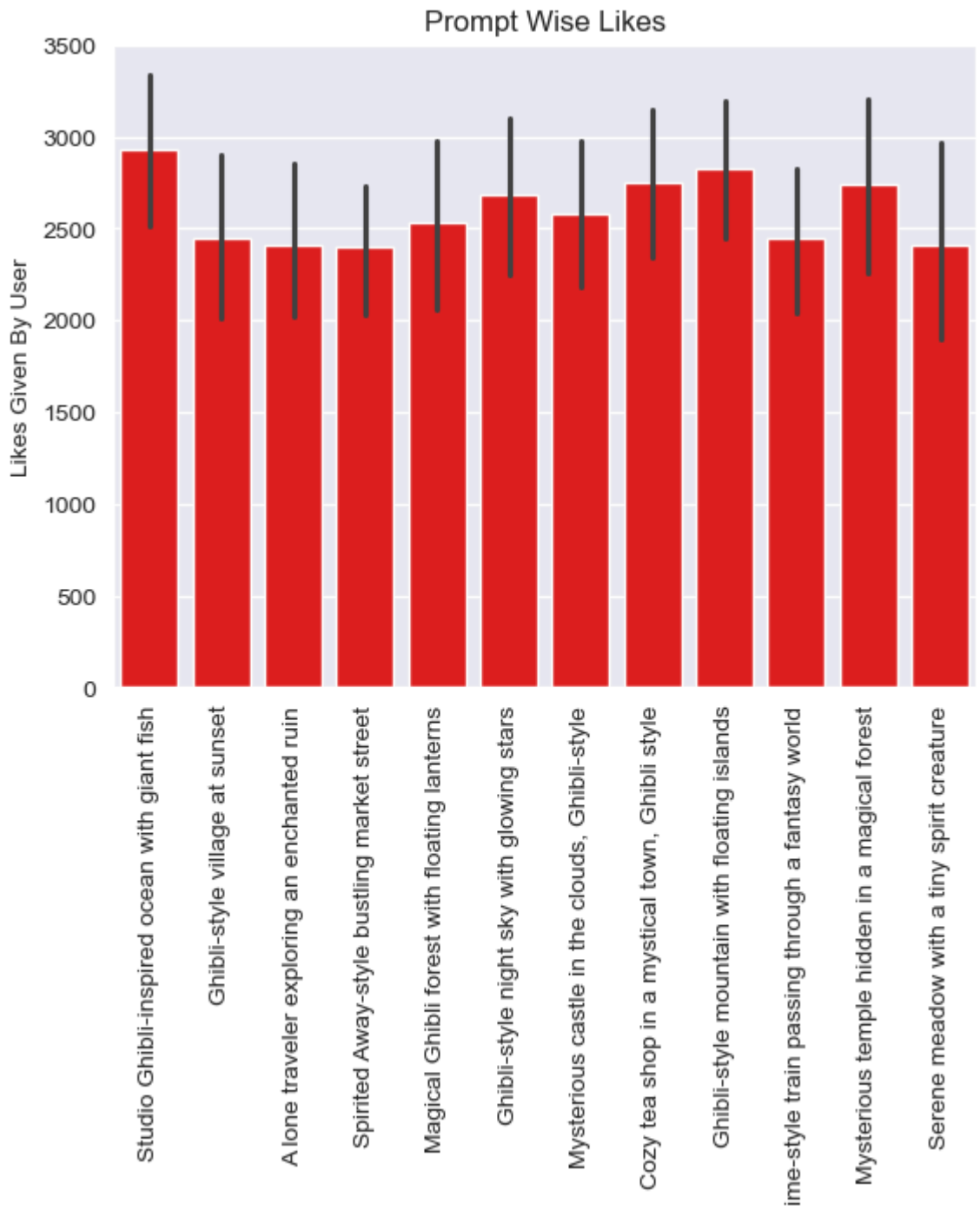
```
[nltk_data] Downloading package stopwords to C:\Users\VIVEK
[nltk_data]     CHAUHAN\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Most common word in the top comments is 'Ghibli','AI','believe','lighting','atmosphere'.

Bi-Variate-Analysis

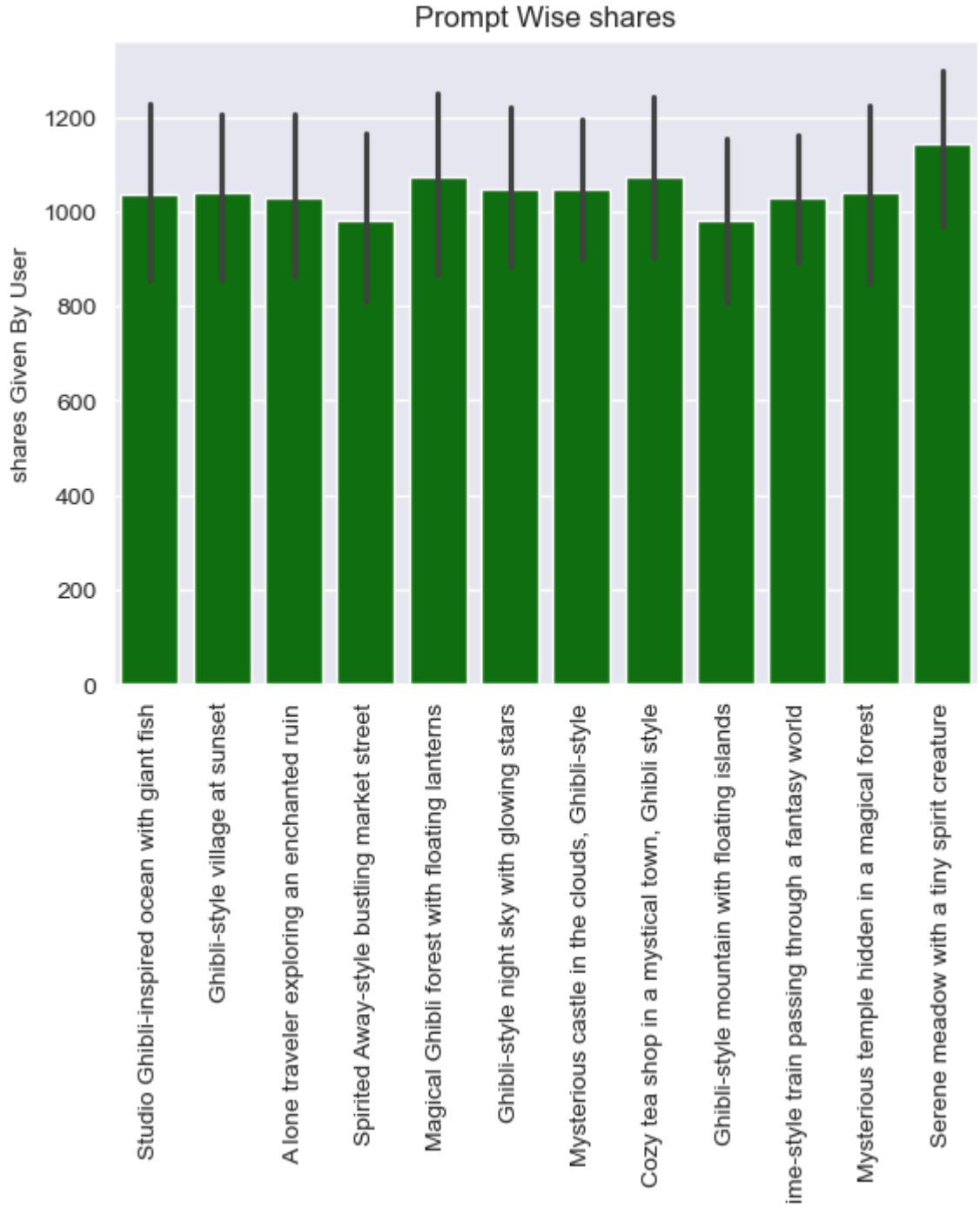
In [52]:
```
# Let's Check the which prompt get the highest likes.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "likes",data = data,color = "red")
plt.title("Prompt Wise Likes")
plt.xlabel("Prompts Given By User")
plt.ylabel("Likes Given By User")
```

```
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise Likes
Likes Given By User
3500
3000
2500
2000
1500
1000
500
0
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

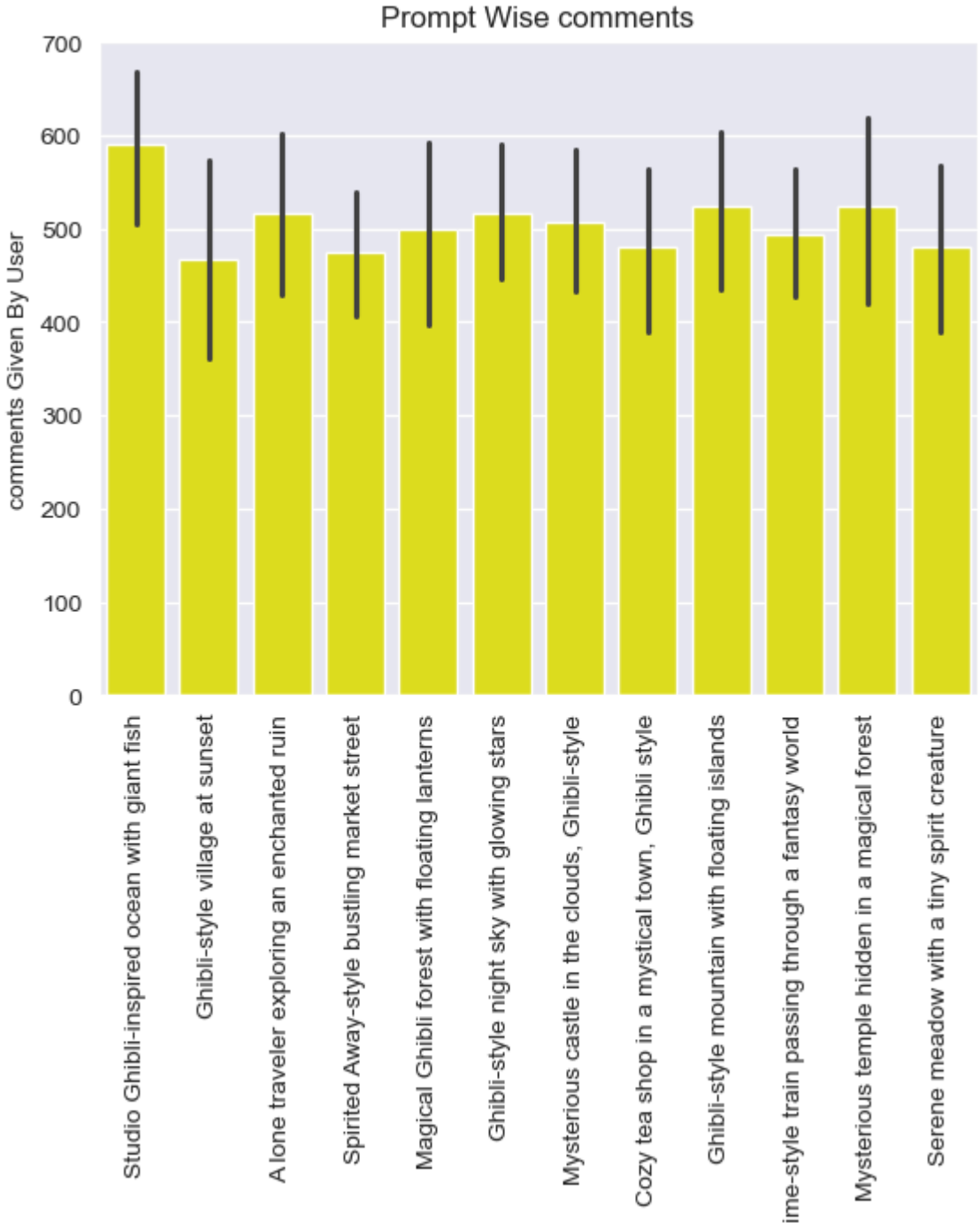
In Above Bar-Chart you can see clearly the "Studio Ghibli-Inspired Ocean With giant fish" Prompt get the highest likes.

In [53]:
```
# Let's Check the which prompt get the highest shares.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "shares",data = data,color = "green")
plt.title("Prompt Wise shares")
plt.xlabel("Prompts Given By User")
plt.ylabel("shares Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise shares
shares Given By User
0
200
400
600
800
1000
1200
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

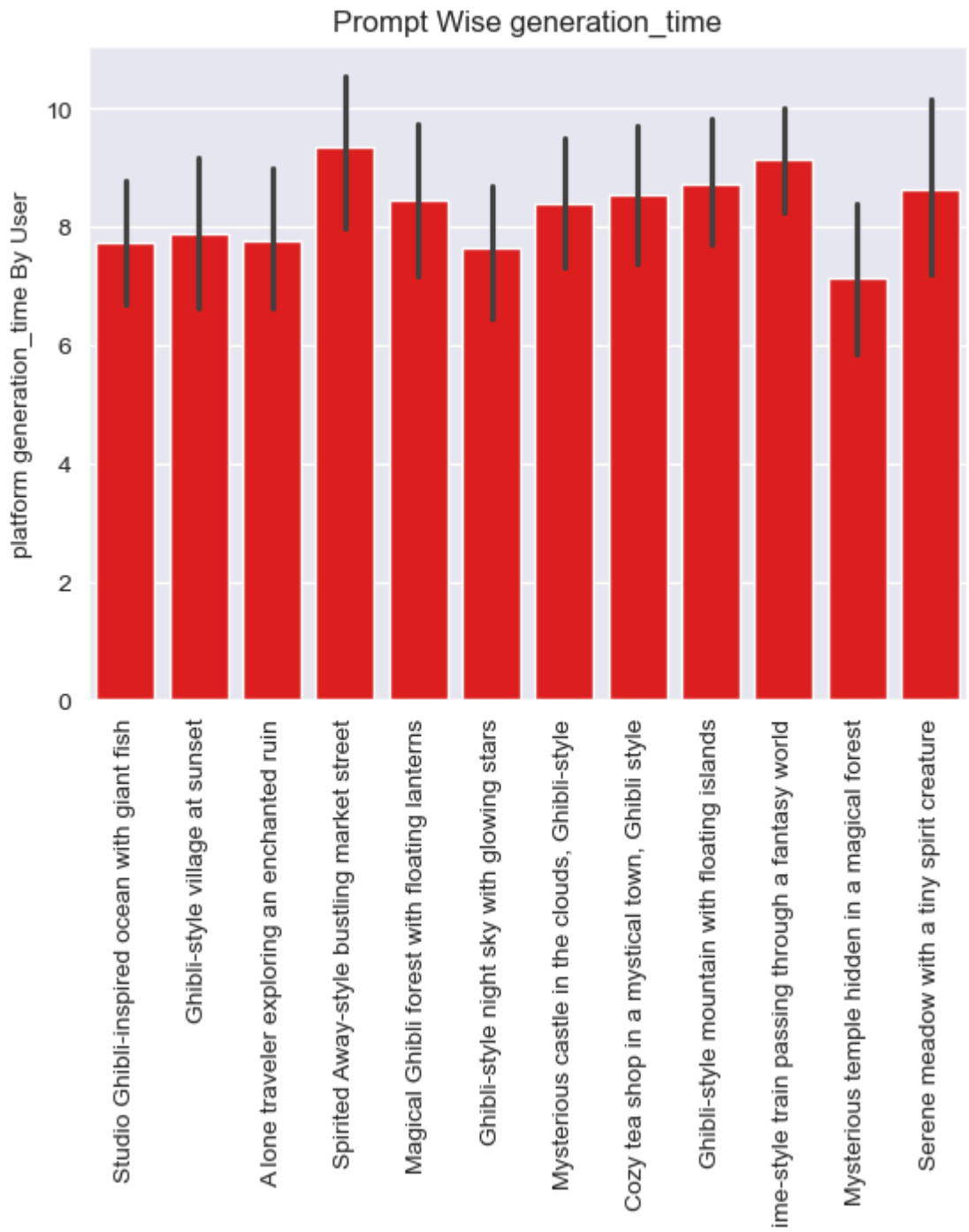
Totally Opposite Thing is "Serene medow with a tiny spirit creature" Prompt Generated Image is highes shares On social-Media.

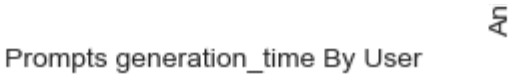
In [54]:
```
# Let's Check the which prompt get the highest comments.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "comments",data = data,color = "yellow")
plt.title("Prompt Wise comments")
plt.xlabel("Prompts Given By User")
plt.ylabel("comments Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise comments
comments Given By User
700
600
500
400
300
200
100
0
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

In Above Bar-Chart you can see clearly the "Studio Ghibli-Inspired Ocean With giant fish" Prompt get the highest Comments.

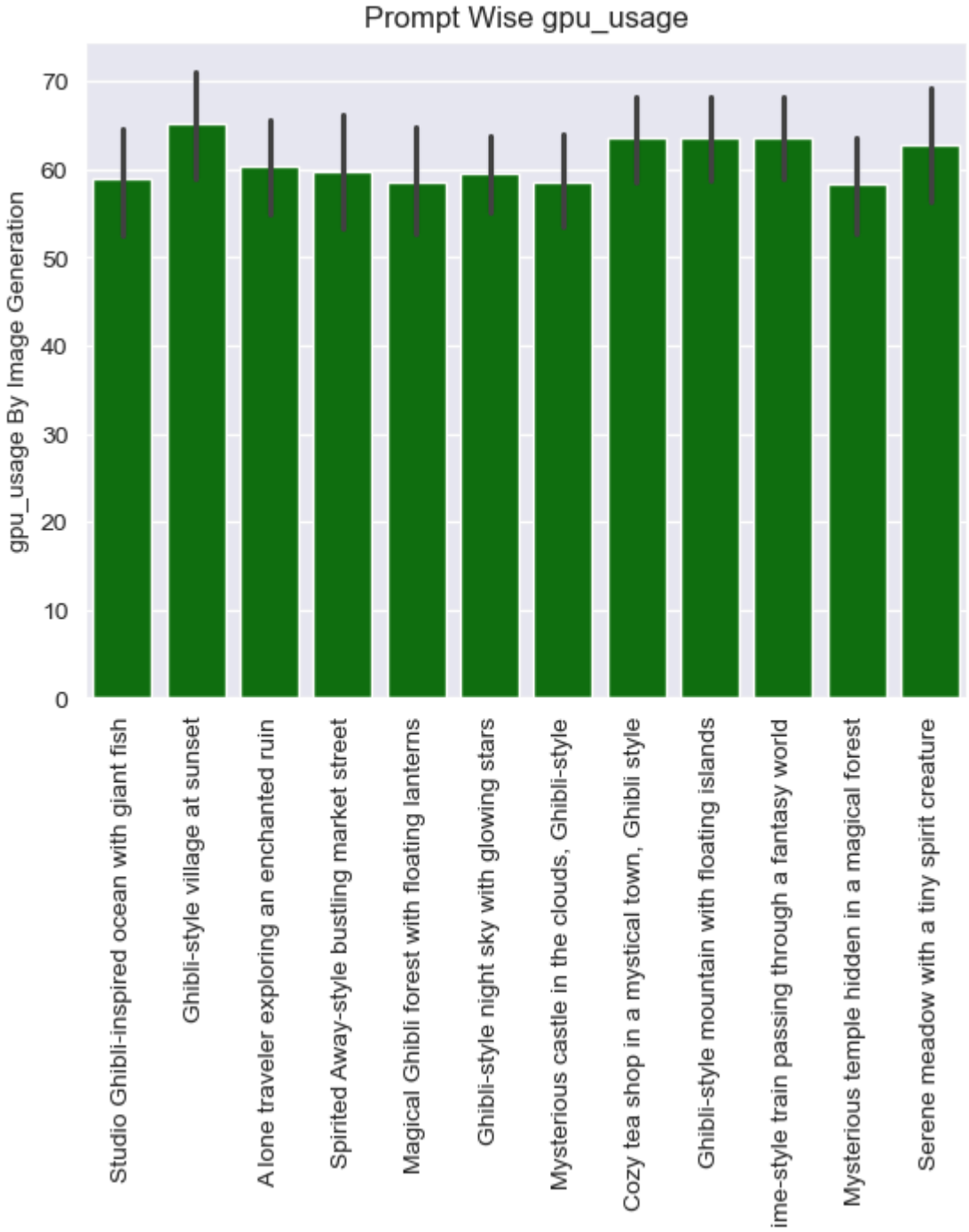
In [55]:
```
# Let's Check the which prompt get the highest generation_time.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "generation_time",data = data,color = "red")
plt.title("Prompt Wise generation_time")
plt.xlabel("Prompts generation_time By User")
plt.ylabel("platform generation_time By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise generation_time
platform generation_time By User
0
2
4
6
8
10
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature



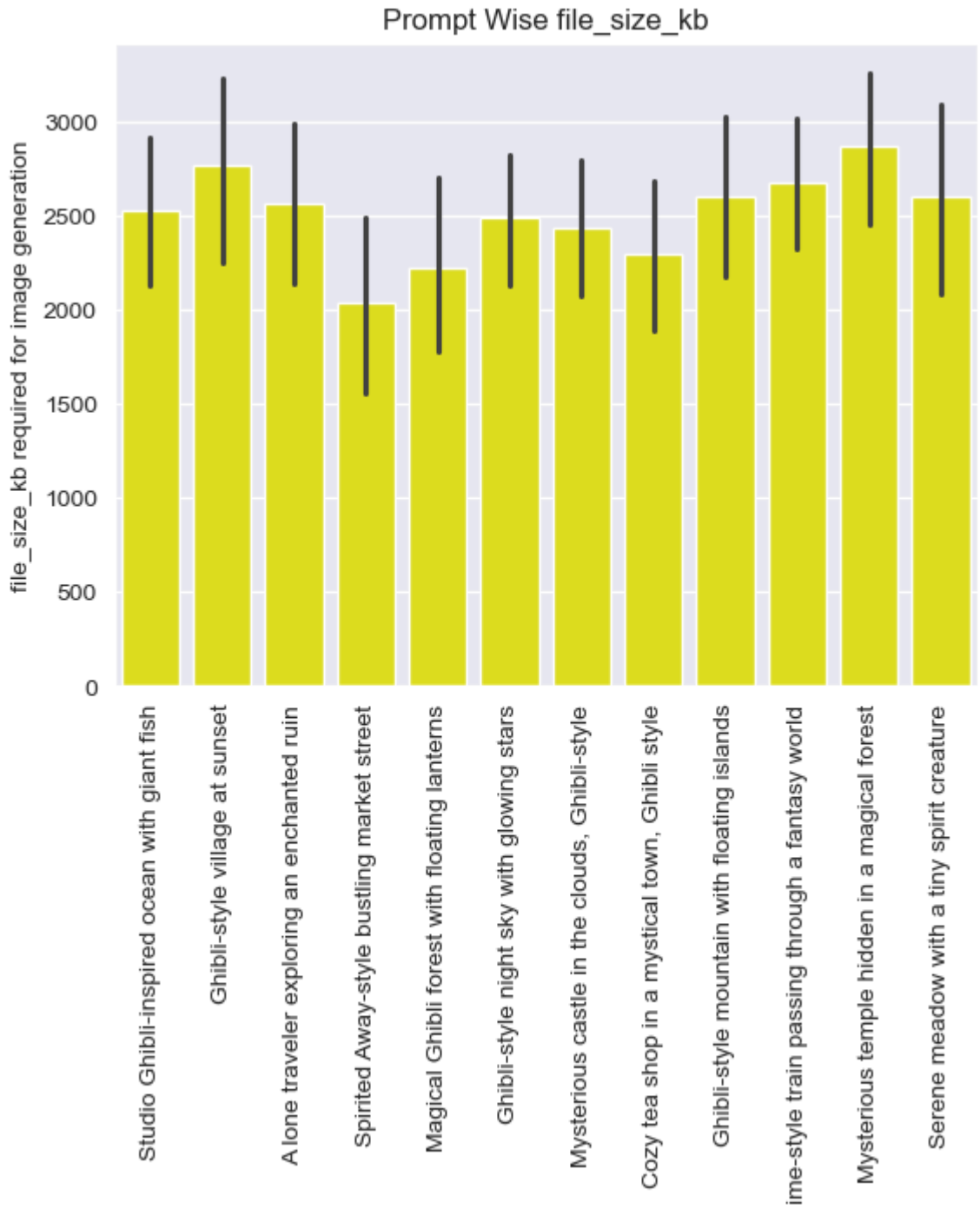
You can see clearly "Sprinted Away Style bustiing market street" prompt takes higher image generation time among the all the prompts.

In [56]:
```
# Let's Check the which prompt get the highest gpu_usage.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "gpu_usage",data = data,color = "green")
plt.title("Prompt Wise gpu_usage")
plt.xlabel("Prompts Given By User")
plt.ylabel("gpu_usage By Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise gpu_usage
gpu_usage By Image Generation
0
10
20
30
40
50
60
70
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

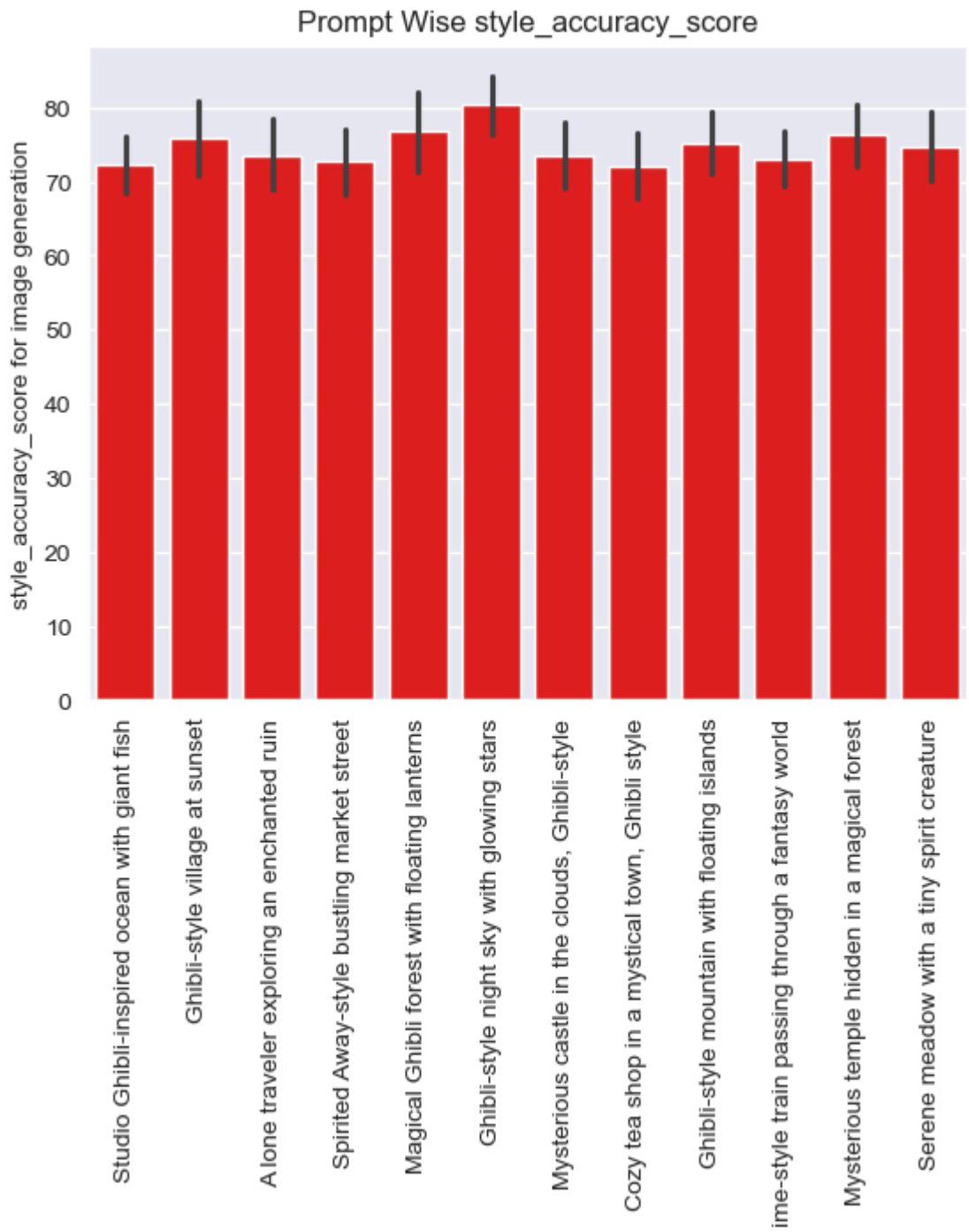
Here the interesting thing is prompt "Ghibli-style village at sunset" take the higher gpu time create image.

In [57]:
```
# Let's Check the which prompt get the highest file_size_kb.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "file_size_kb",data = data,color = "yellow")
plt.title("Prompt Wise file_size_kb")
plt.xlabel("Prompts Given By User")
plt.ylabel("file_size_kb required for image generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise file_size_kb
file_size_kb required for image generation
0
500
1000
1500
2000
2500
3000
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

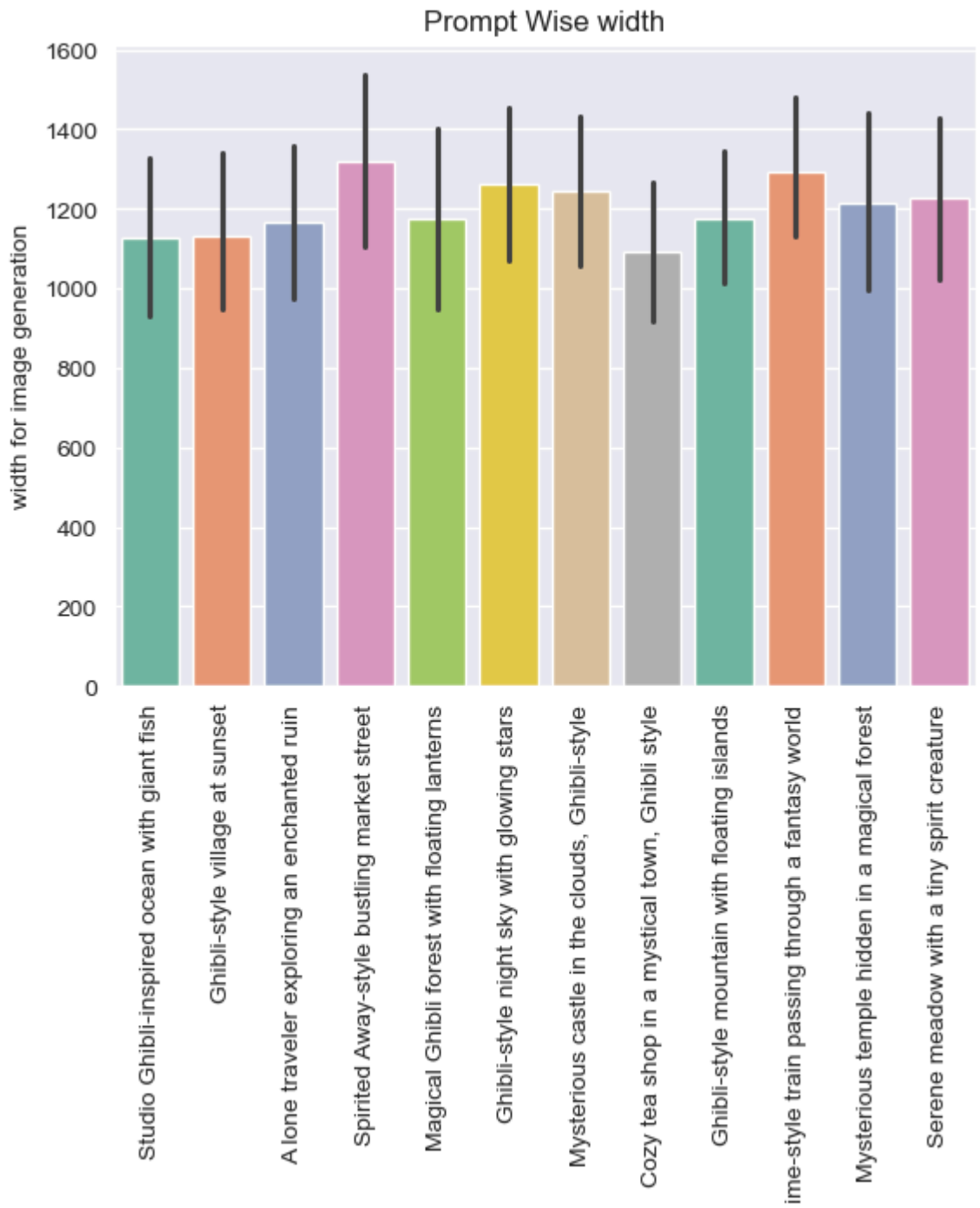
One Another interesting thing is prompt "Mysterious temple hidden in a magical forest" required high spaces to create that type of images.

In [58]:
```
# Let's Check the which prompt get the highest style_accuracy_score.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "style_accuracy_score",data = data,color = "red")
plt.title("Prompt Wise style_accuracy_score")
plt.xlabel("Prompts Given By User")
plt.ylabel("style_accuracy_score for image generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise style_accuracy_score
style_accuracy_score for image generation
0
10
20
30
40
50
60
70
80
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

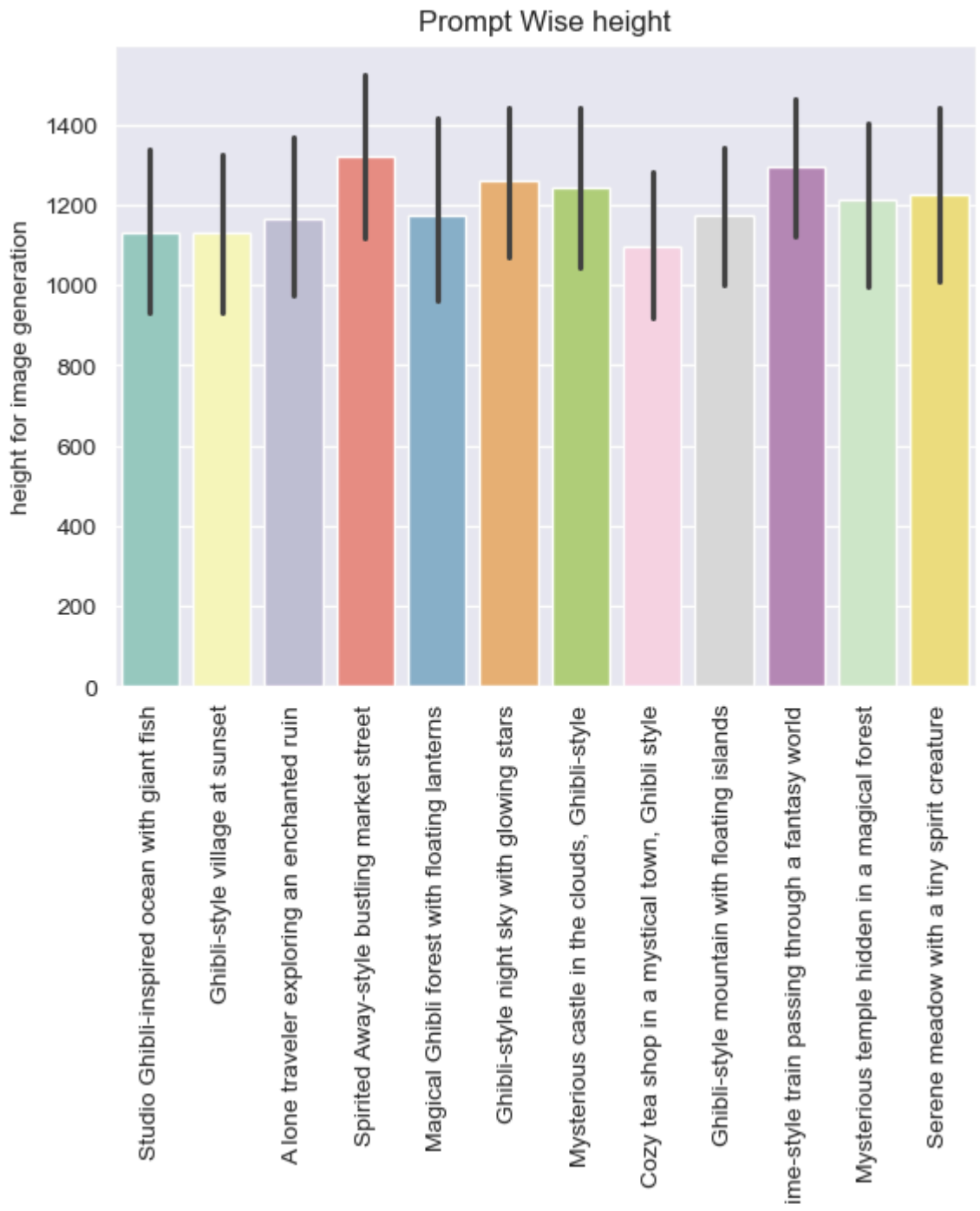
You can see clearly the prompt "Ghibli-style night sky with glowing stars" generates the images with high accuracy score.

In [59]:
```
# Let's Check the which prompt get the highest width.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "width",data = data,palette = "Set2")
plt.title("Prompt Wise width")
plt.xlabel("Prompts Given By User")
plt.ylabel("width for image generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise width
width for image generation
1600
1400
1200
1000
800
600
400
200
0
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

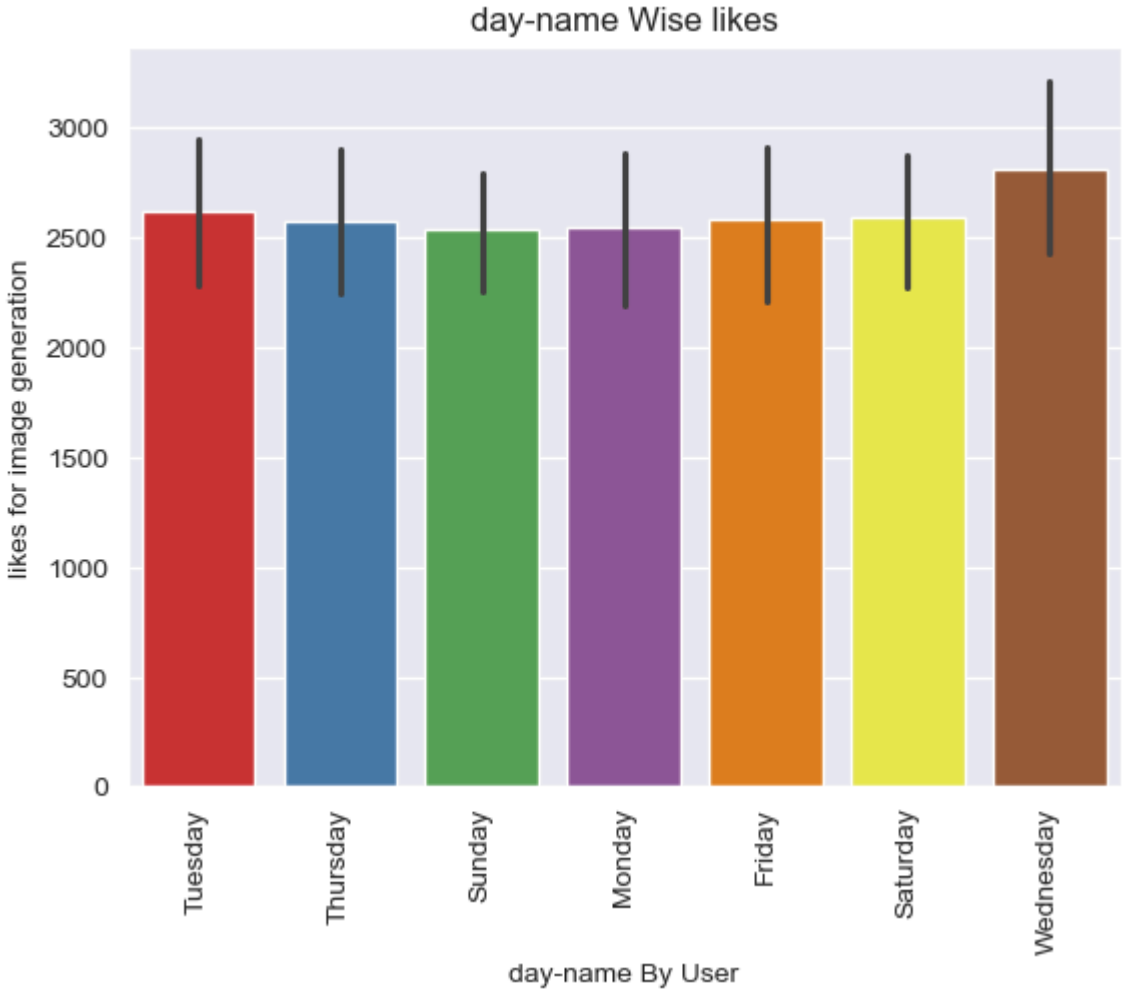
You can see broder images generated in the prompt of "Sprinted Away Style bustling market street".

In [60]:
```
# Let's Check the which prompt get the highest height.

sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "height",data = data,palette = "Set3")
plt.title("Prompt Wise height")
plt.xlabel("Prompts Given By User")
plt.ylabel("height for image generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise height
height for image generation
0
200
400
600
800
1000
1200
1400
Studio Ghibli-inspired ocean with giant fish
Ghibli-style village at sunset
A lone traveler exploring an enchanted ruin
Spirited Away-style bustling market street
Magical Ghibli forest with floating lanterns
Ghibli-style night sky with glowing stars
Mysterious castle in the clouds, Ghibli-style
Cozy tea shop in a mystical town, Ghibli style
Ghibli-style mountain with floating islands
ime-style train passing through a fantasy world
Mysterious temple hidden in a magical forest
Serene meadow with a tiny spirit creature

Now again the winner is you can see biggest images generated in the prompt of "Sprinted Away Style bustling market street".

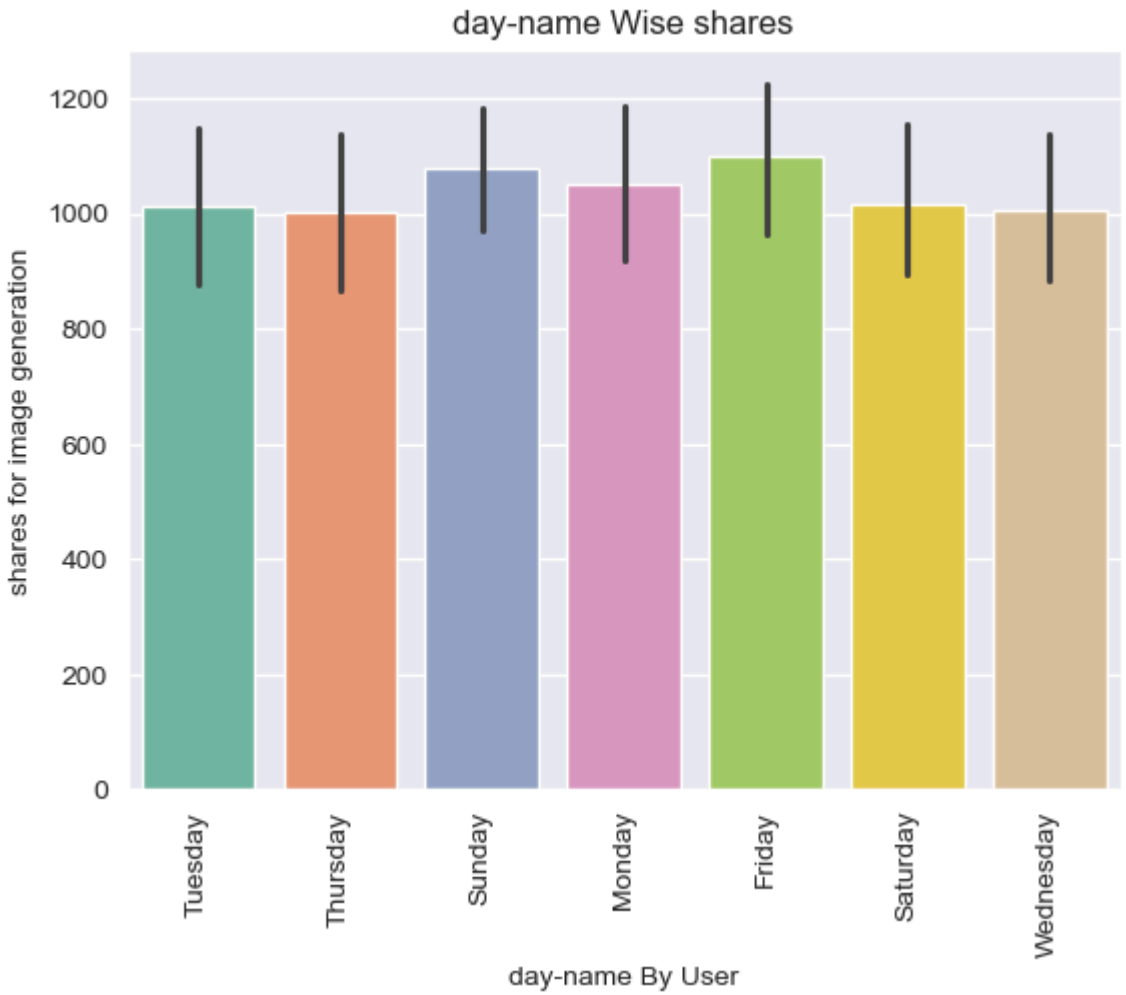
In [61]:
```
# Let's Check the which day-name get the highest likes.

sns.set_style("darkgrid")
sns.barplot(x = "day-name",y = "likes",data = data,palette = "Set1")
plt.title("day-name Wise likes")
plt.xlabel("day-name By User")
plt.ylabel("likes for image generation")
plt.xticks(rotation = "vertical")
plt.show()
```



Wednesday posted Ghibli Images getting the highes likes on platforms.

In [62]:
```
# Let's Check the which day-name get the highest shares.

sns.set_style("darkgrid")
sns.barplot(x = "day-name",y = "shares",data = data,palette = "Set2")
plt.title("day-name Wise shares")
plt.xlabel("day-name By User")
```

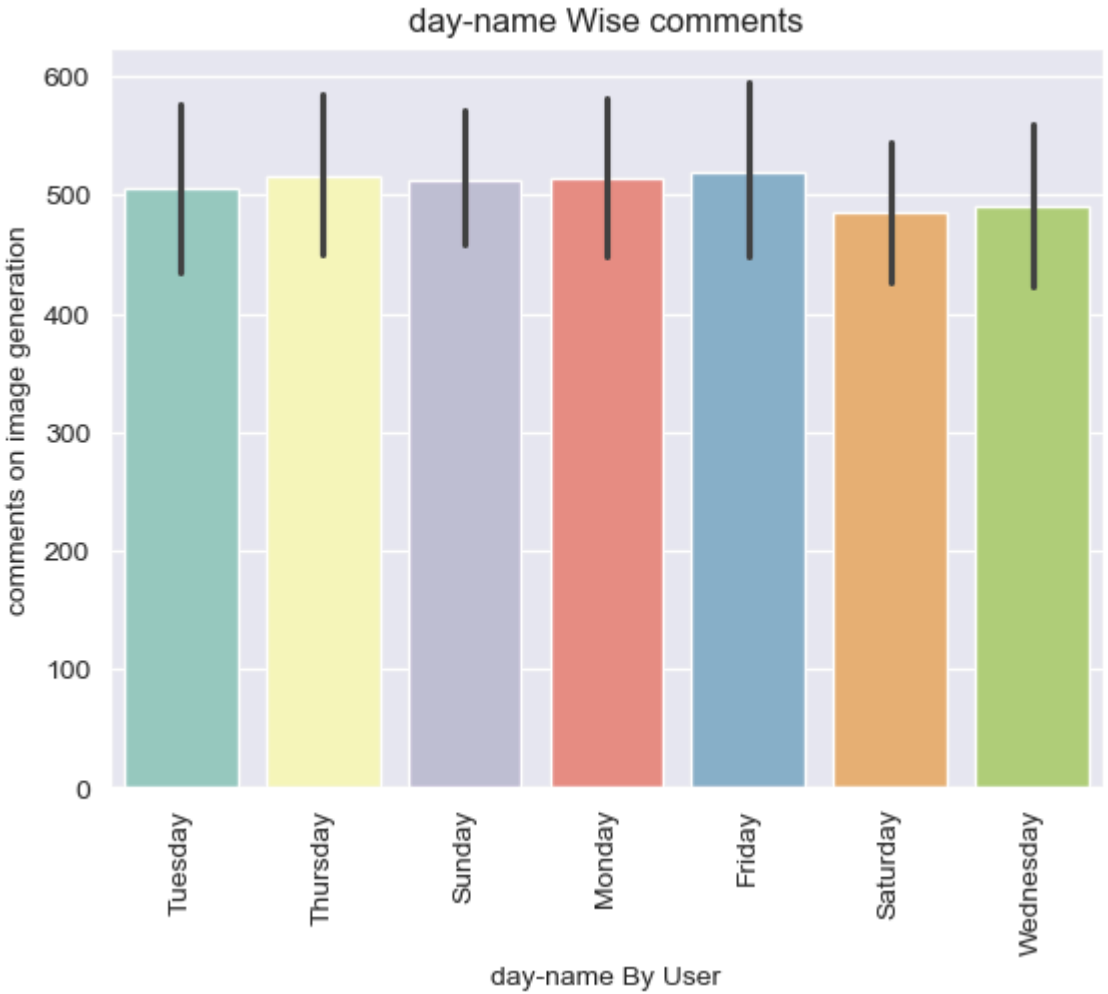
```
plt.ylabel("shares for image generation")
plt.xticks(rotation = "vertical")
plt.show()
```



Friday posted images getting the highest shares on platforms.

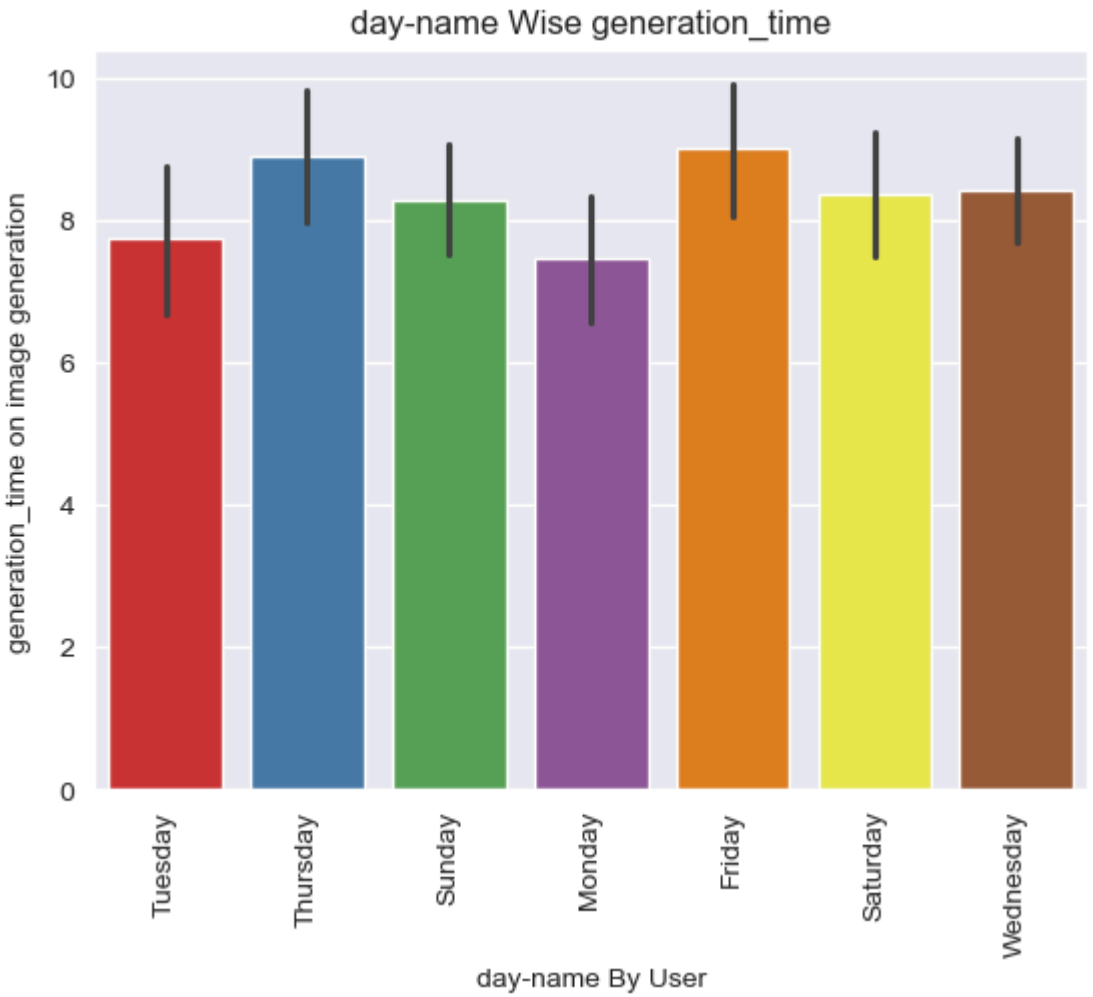
In [63]:
```
# Let's Check the which day-name get the highest comments.
```

```
sns.set_style("darkgrid")
sns.barplot(x = "day-name",y = "comments",data = data,palette = "Set3")
plt.title("day-name Wise comments")
plt.xlabel("day-name By User")
plt.ylabel("comments on image generation")
plt.xticks(rotation = "vertical")
plt.show()
```



Again the winner is Friday getting the highest comments on posts.

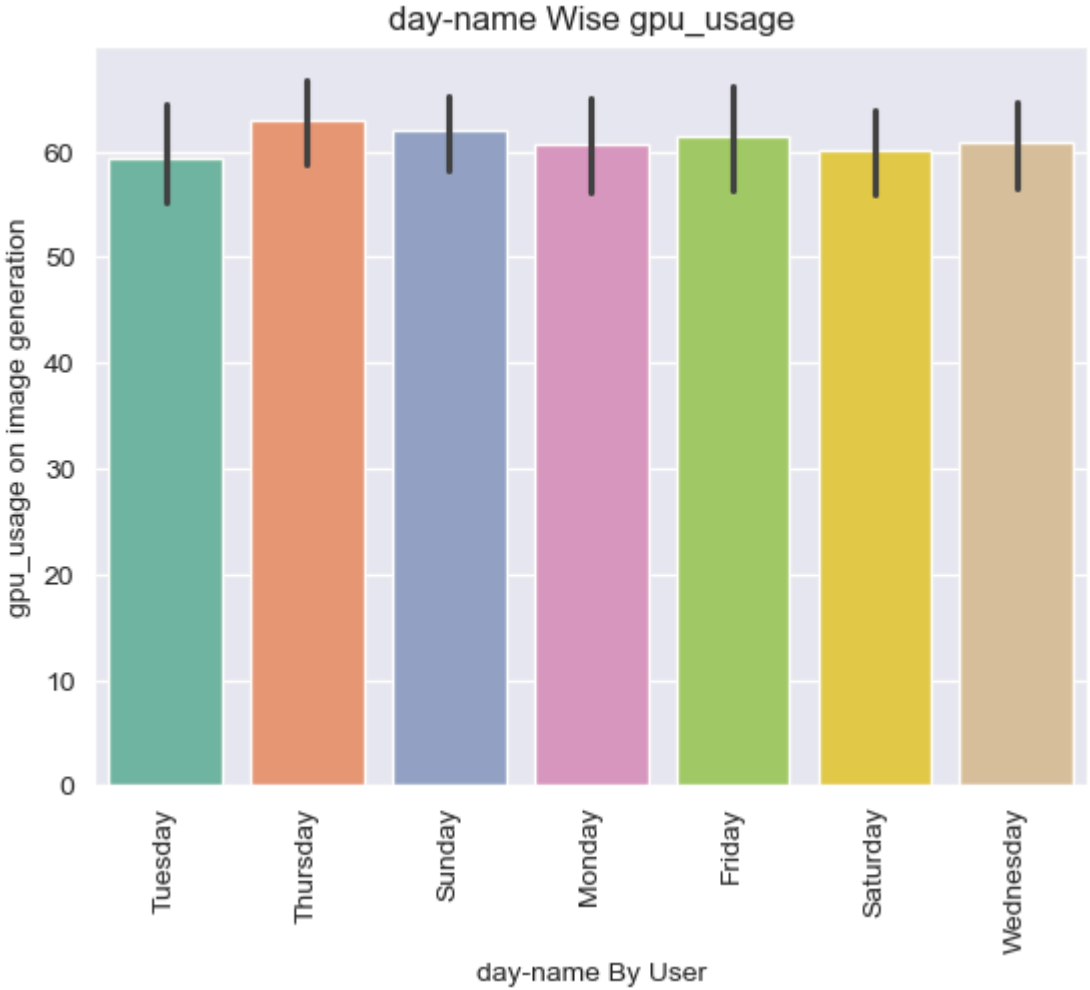
In [64]:
```
# Let's Check the which day-name get the highest generation_time.

sns.set_style("darkgrid")
sns.barplot(x = "day-name",y = "generation_time",data = data,palette = "Set1")
plt.title("day-name Wise generation_time")
plt.xlabel("day-name By User")
plt.ylabel("generation_time on image generation")
plt.xticks(rotation = "vertical")
plt.show()
```



Obeviously Friday taking the highest image generation time cause all the images are posted on Friday.

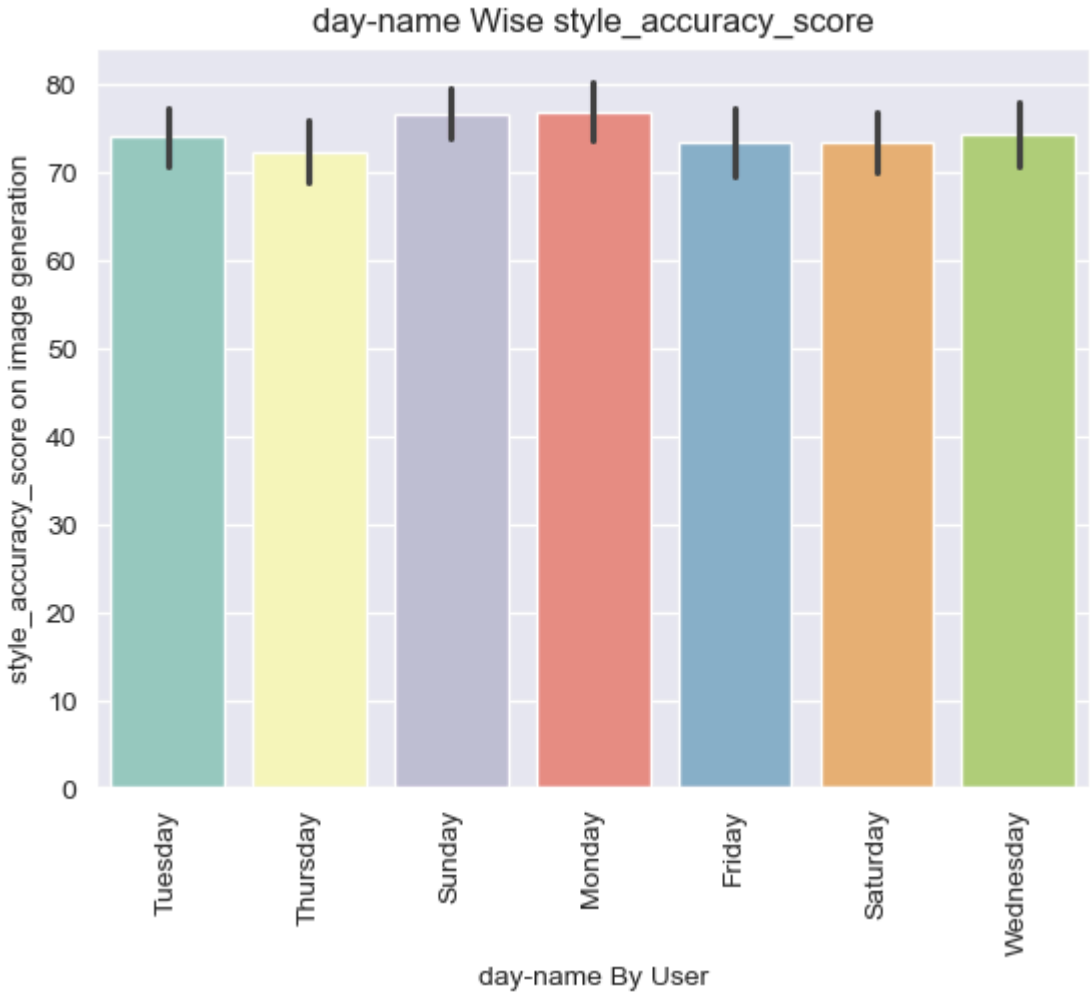
In [65]:
```
# Let's Check the which day-name get the highest gpu_usage.

sns.set_style("darkgrid")
sns.barplot(x = "day-name",y = "gpu_usage",data = data,palette = "Set2")
plt.title("day-name Wise gpu_usage")
```

```
plt.xlabel("day-name By User")
plt.ylabel("gpu_usage on image generation")
plt.xticks(rotation = "vertical")
plt.show()
```



Interesting thing is Gpu usage is high in the day of Thursay.

In [66]:
```
# Let's Check the which day-name get the highest style_accuracy_score.

sns.set_style("darkgrid")
sns.barplot(x = "day-name",y = "style_accuracy_score",data = data,palette = "Set3")
plt.title("day-name Wise style_accuracy_score")
plt.xlabel("day-name By User")
plt.ylabel("style_accuracy_score on image generation")
plt.xticks(rotation = "vertical")
plt.show()
```

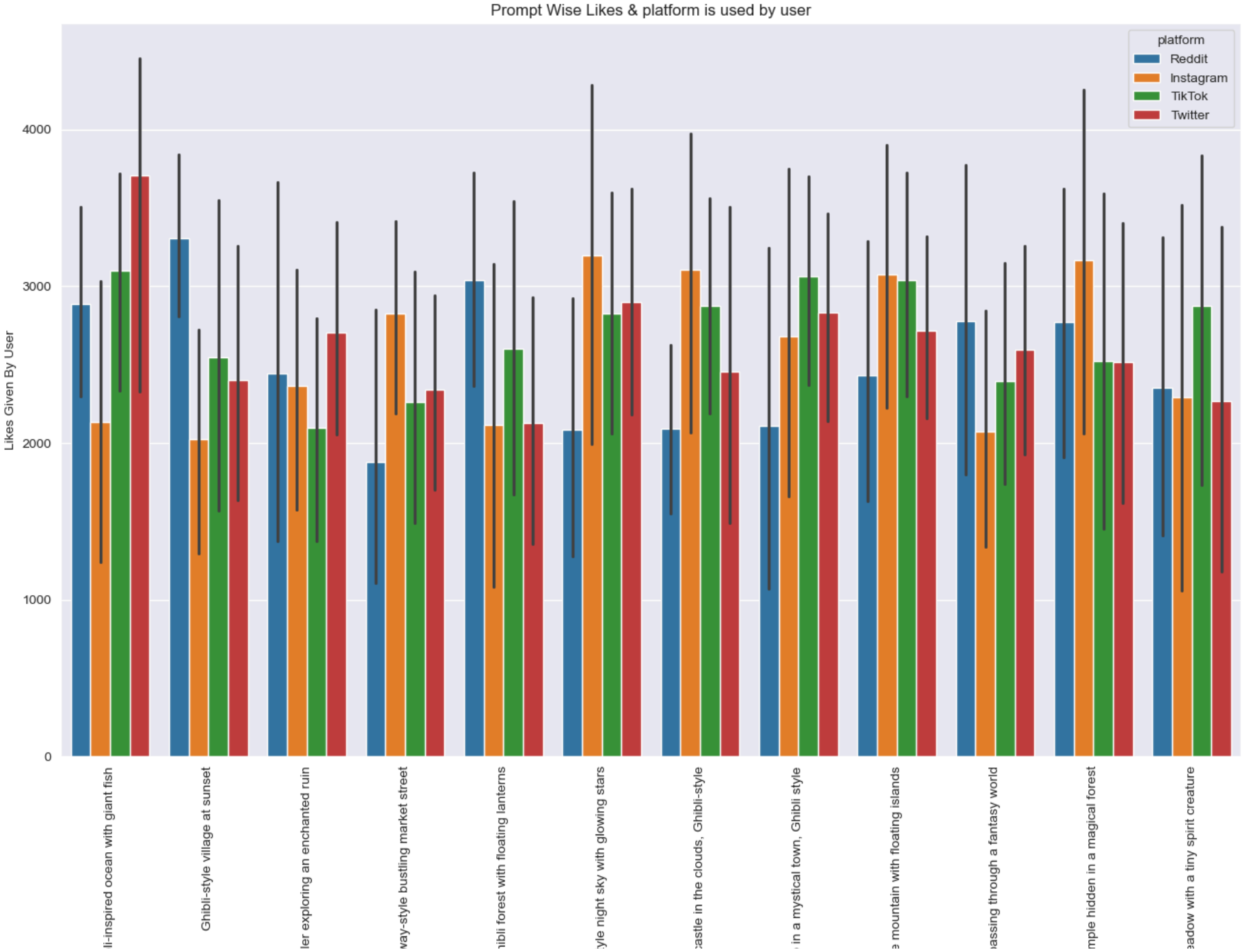


Another interesting thing is Sunday & Monday Images Accuracy score is high.

Multi-Variate-Analysis

Hue is platform.

In [67]:
```
# Let's Check the promt wise likes and platform is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "likes",hue = "platform",data = data)
plt.title("Prompt Wise Likes & platform is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("Likes Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise Likes & platform is used by user
platform
Reddit
Instagram
TikTok
Twitter
Likes Given By User
0
1000
2000
3000
4000
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

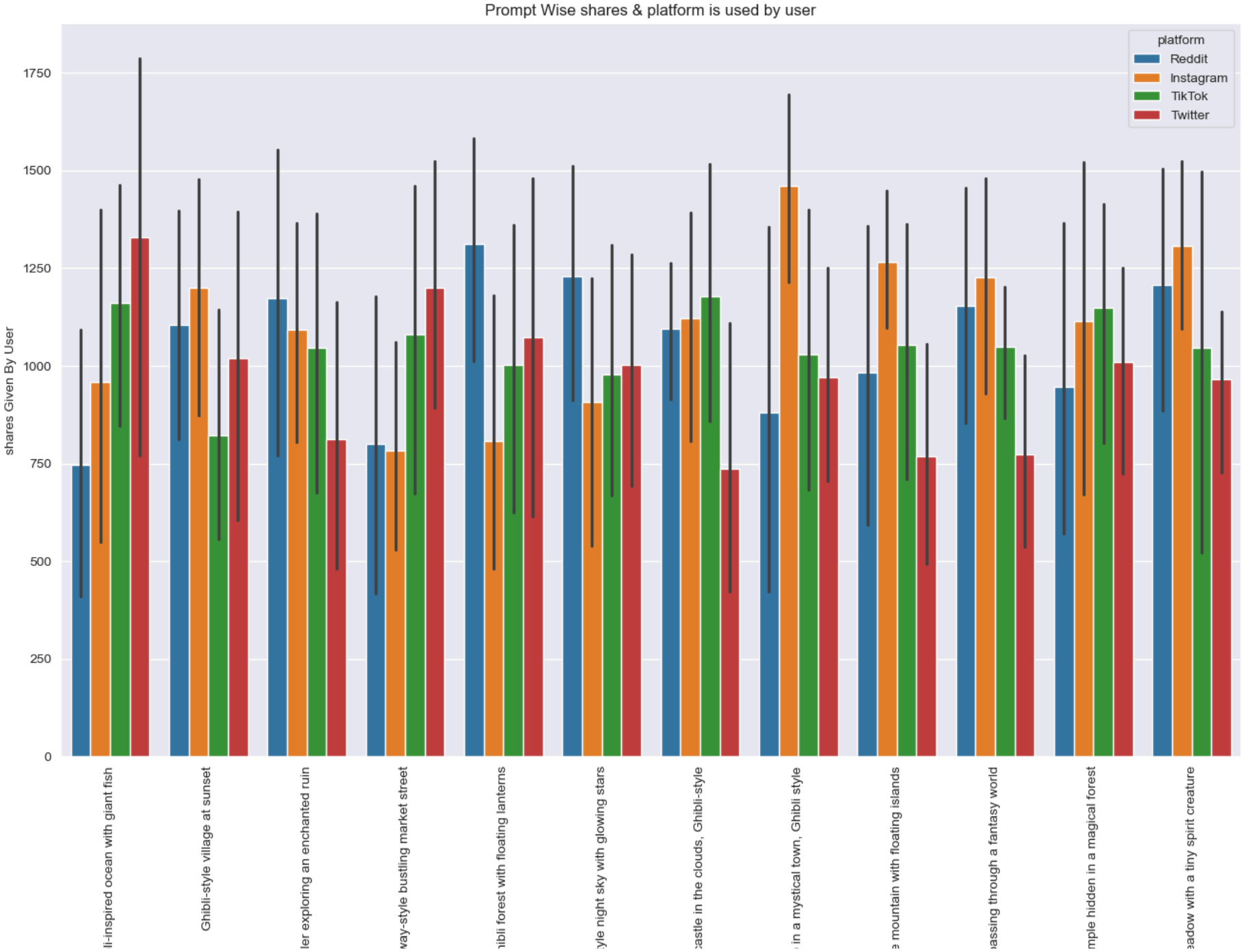
You can see clearly the prompt "Studio Ghibli-Inspired ocaean with gian fish" generated images are posted on the platform of Twitter.

The second prompt "Ghibli Style nigh sky with glowing starts" generated images are posted on the platform of Instagram.

The third prompt "Mysterious castle in the clouds.Ghibli-Style" generated images are posted on the platform of Instagram.

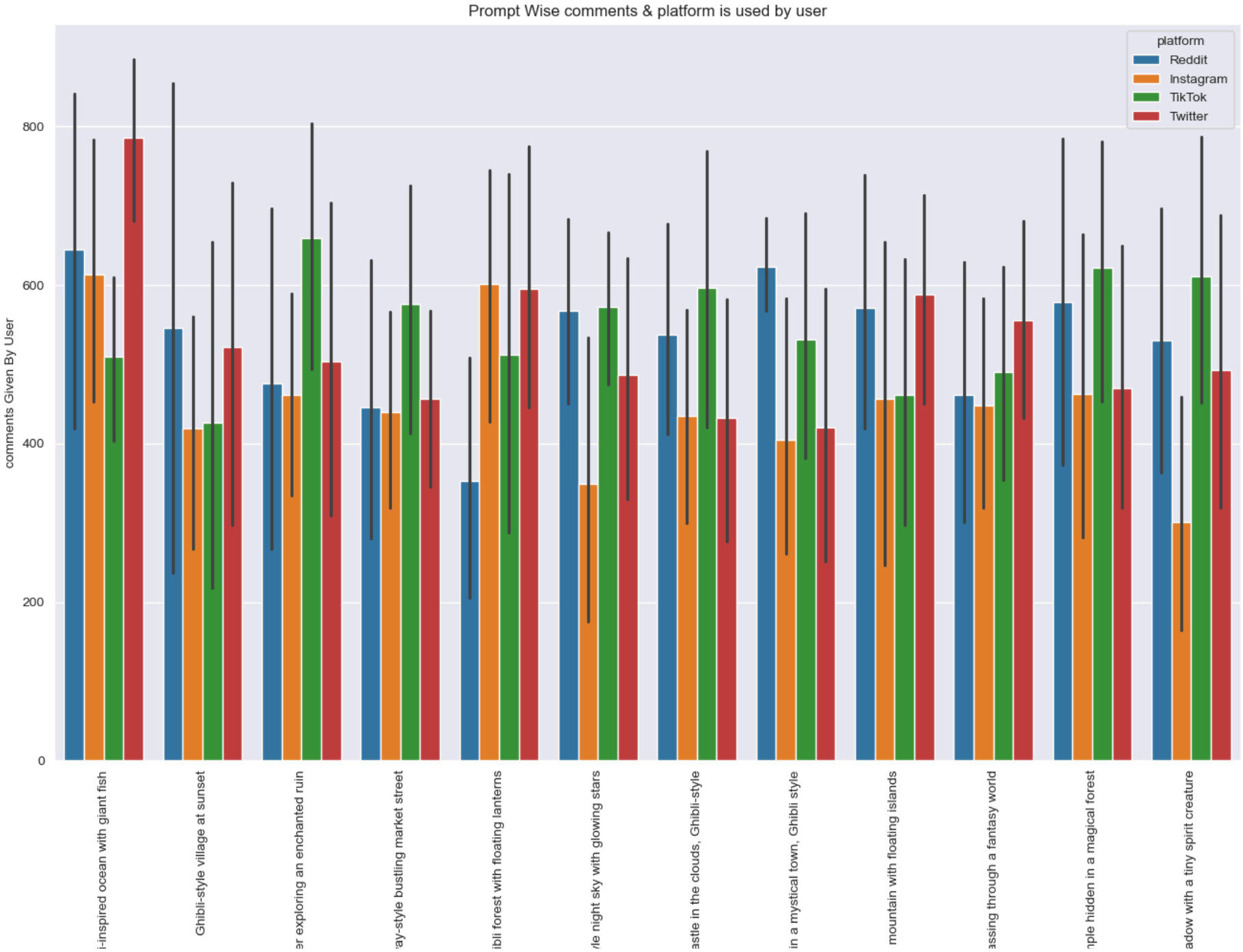
The forth prompt "Mysterious temple hidden in a magical forest" generated images are posted on the platform of Instagram.

In [68]:
```
# Let's Check the promt wise shares and platform is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "shares",hue = "platform",data = data)
plt.title("Prompt Wise shares & platform is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("shares Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise shares & platform is used by user
platform
Reddit
Instagram
TikTok
Twitter
shares Given By User
0
250
500
750
1000
1250
1500
1750
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

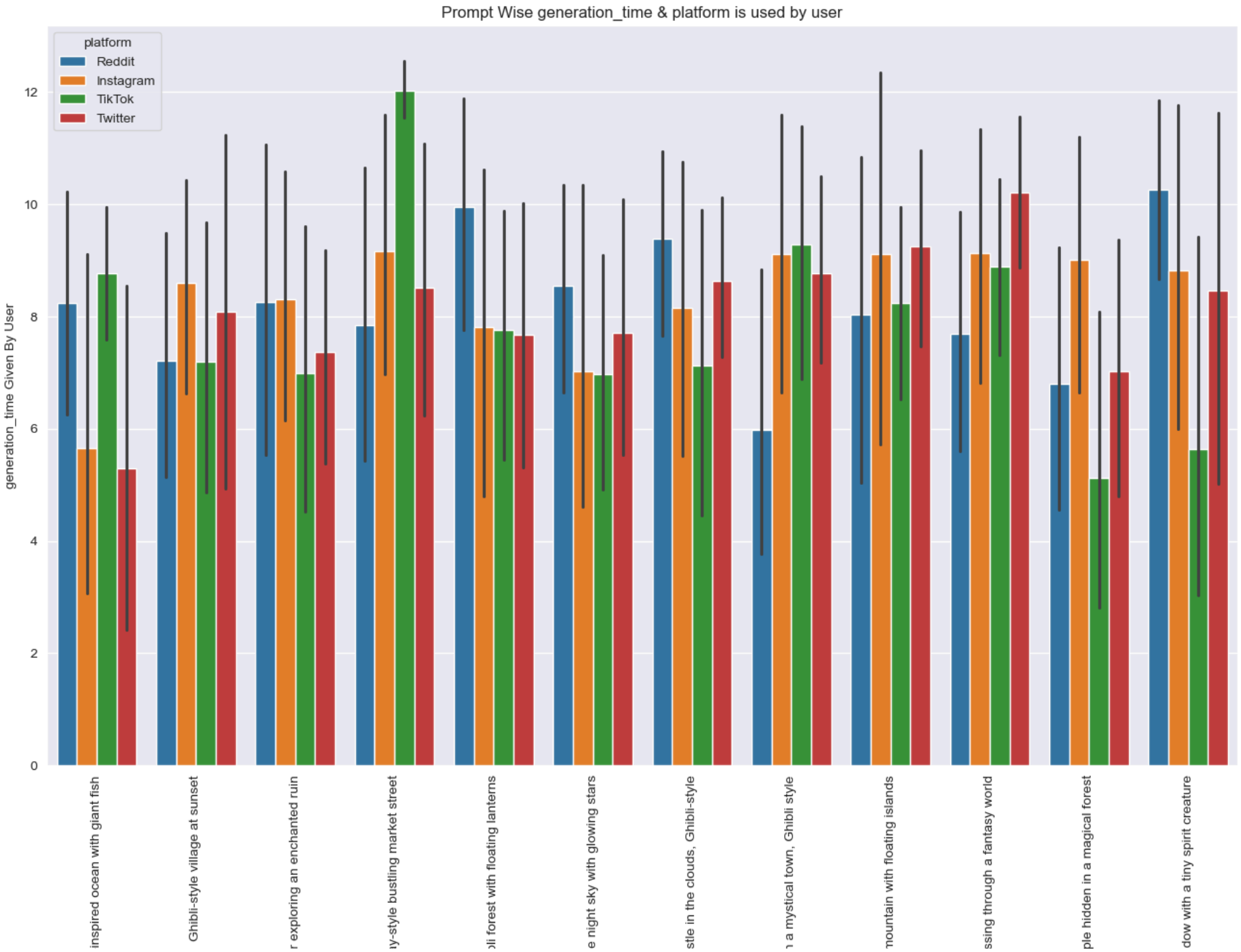
Most of the Ghibli Images are share on the platform of Reddit & Instagram but the interesting thing is prompt "Studio Ghibli-Inspired ocaean with gian fish" images are highest shares on the platform of Twitter.

In [69]:
```
# Let's Check the promt wise comments and platform is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "comments",hue = "platform",data = data)
plt.title("Prompt Wise comments & platform is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("comments Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise comments & platform is used by user
platform
Reddit
Instagram
TikTok
Twitter
comments Given By User
0
200
400
600
800
i-inspired ocean with giant fish
Ghibli-style village at sunset
ar exploring an enchanted ruin
ay-style bustling market street
ibli forest with floating lanterns
rle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
mountain with floating islands
assing through a fantasy world
nple hidden in a magical forest
adow with a tiny spirit creature

You can see clearly the prompt "Studio Ghibli-Inspired ocaean with gian fish" images are getting highest comments on the platform of Twitter.

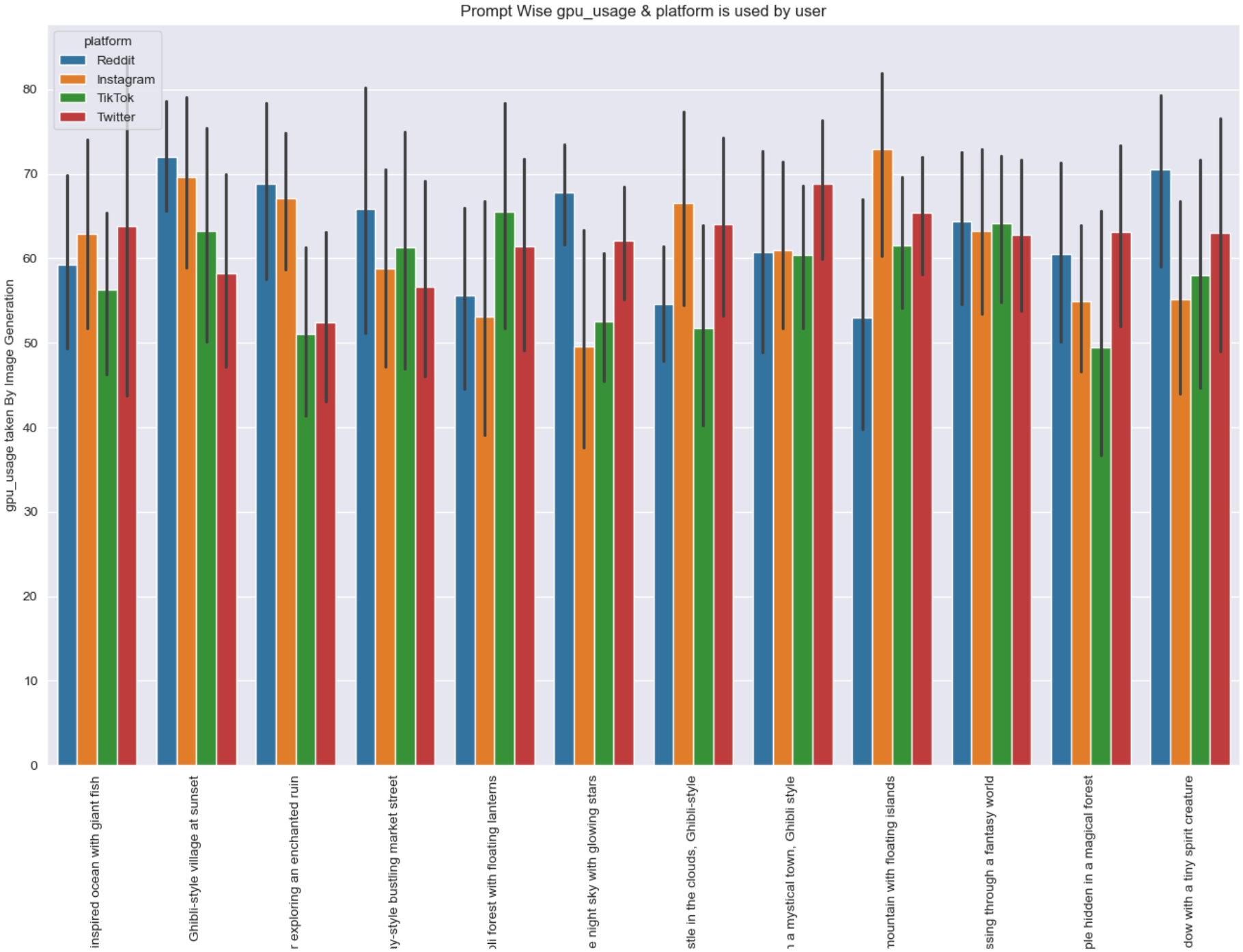
In [70]:
```
# Let's Check the promt wise generation_time and platform is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "generation_time",hue = "platform",data = data)
plt.title("Prompt Wise generation_time & platform is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("generation_time Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise generation_time & platform is used by user
platform
Reddit
Instagram
TikTok
Twitter
generation_time Given By User
0
2
4
6
8
10
12
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
y-style bustling market street
li forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature

The prompt "Spirited Away style bustling market street" highest generating time taken and posten on the platform of Tik-Tok.

Other Interesting thing is The prompt "Ghibli style mountain with floaring islands" highest generating time taken and posten on the platform of Instagram.

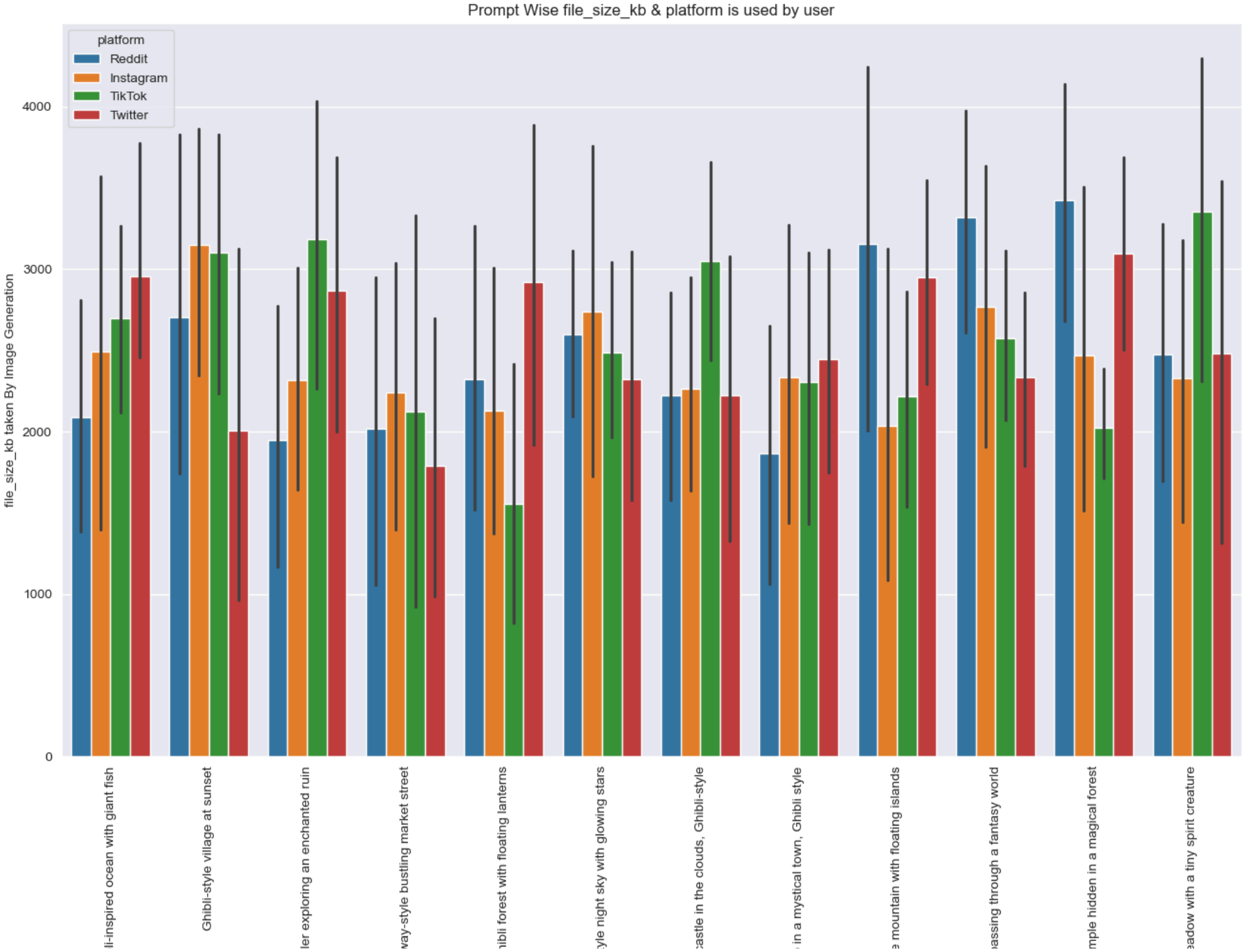
In [71]:
```
# Let's Check the promt wise gpu_usage and platform is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "gpu_usage",hue = "platform",data = data)
plt.title("Prompt Wise gpu_usage & platform is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("gpu_usage taken By Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise gpu_usage & platform is used by user
platform
Reddit
Instagram
TikTok
Twitter
gpu_usage taken By Image Generation
0
10
20
30
40
50
60
70
80
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
y-style bustling market street
li forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature

Again the winner is prompt "Studio Ghibli inspired ocean with giant fish" takes the highest gpu time to generate the images and posted on the platform of Twitter.

The second winner is prompt "Ghibli style mountain wiht floating islands" takes the highest gpu time to generate the images and posted on the platform of Instagram.

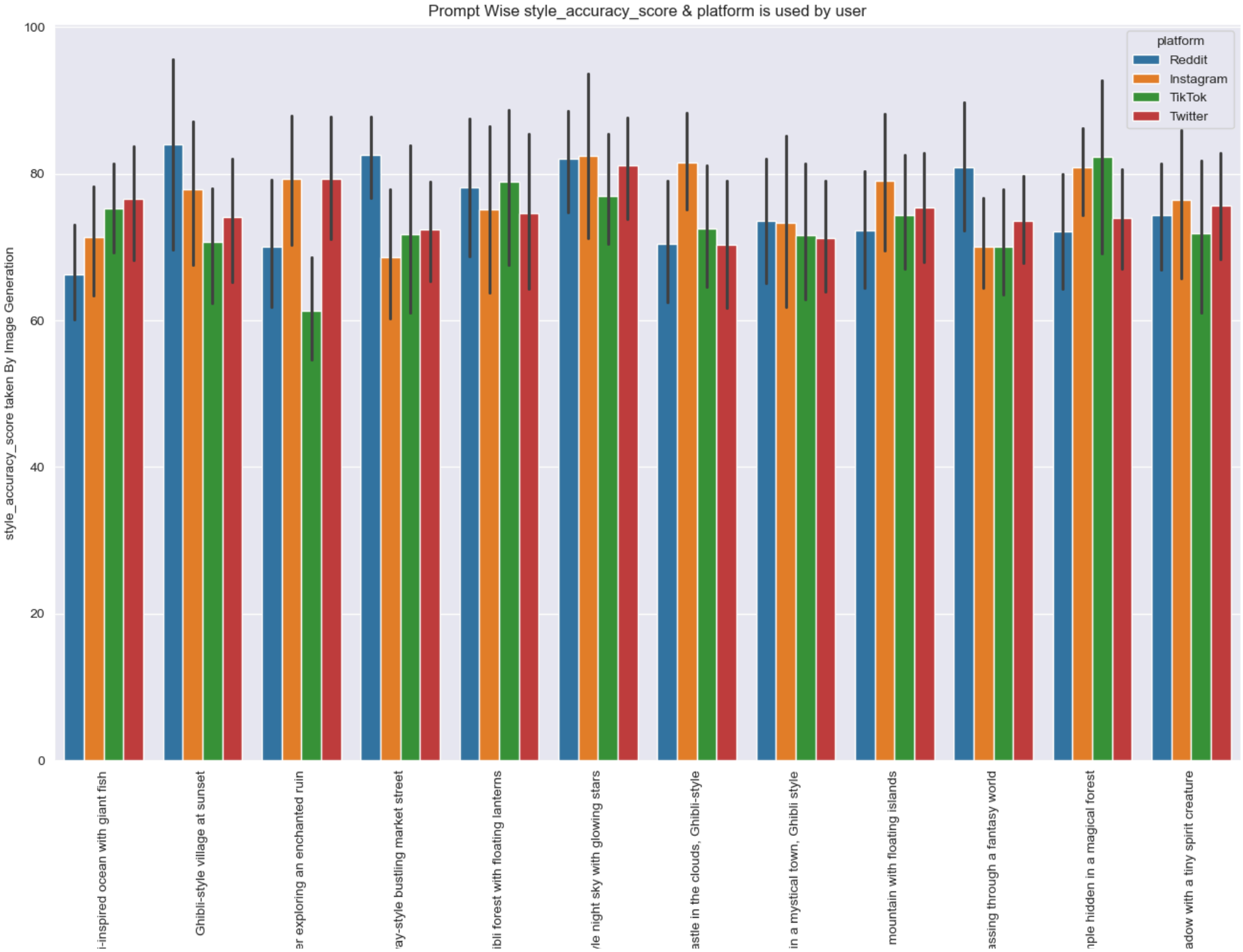
In [72]:
```
# Let's Check the promt wise file_size_kb and platform is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "file_size_kb",hue = "platform",data = data)
plt.title("Prompt Wise file_size_kb & platform is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("file_size_kb taken By Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise file_size_kb & platform is used by user
platform
Reddit
Instagram
TikTok
Twitter
file_size_kb taken By Image Generation
0
1000
2000
3000
4000
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

Prompt "Ghibli style mountain with floating islands" & "Mysterious temple hidden in a magical forest" takes the highest sizes of the images and posted on the platform of Reddit.

But the interesting thing is prompt "Serene meadow with a tiny spirit creature" images take the highest sizes and posted on the platform of Tik-Tok.

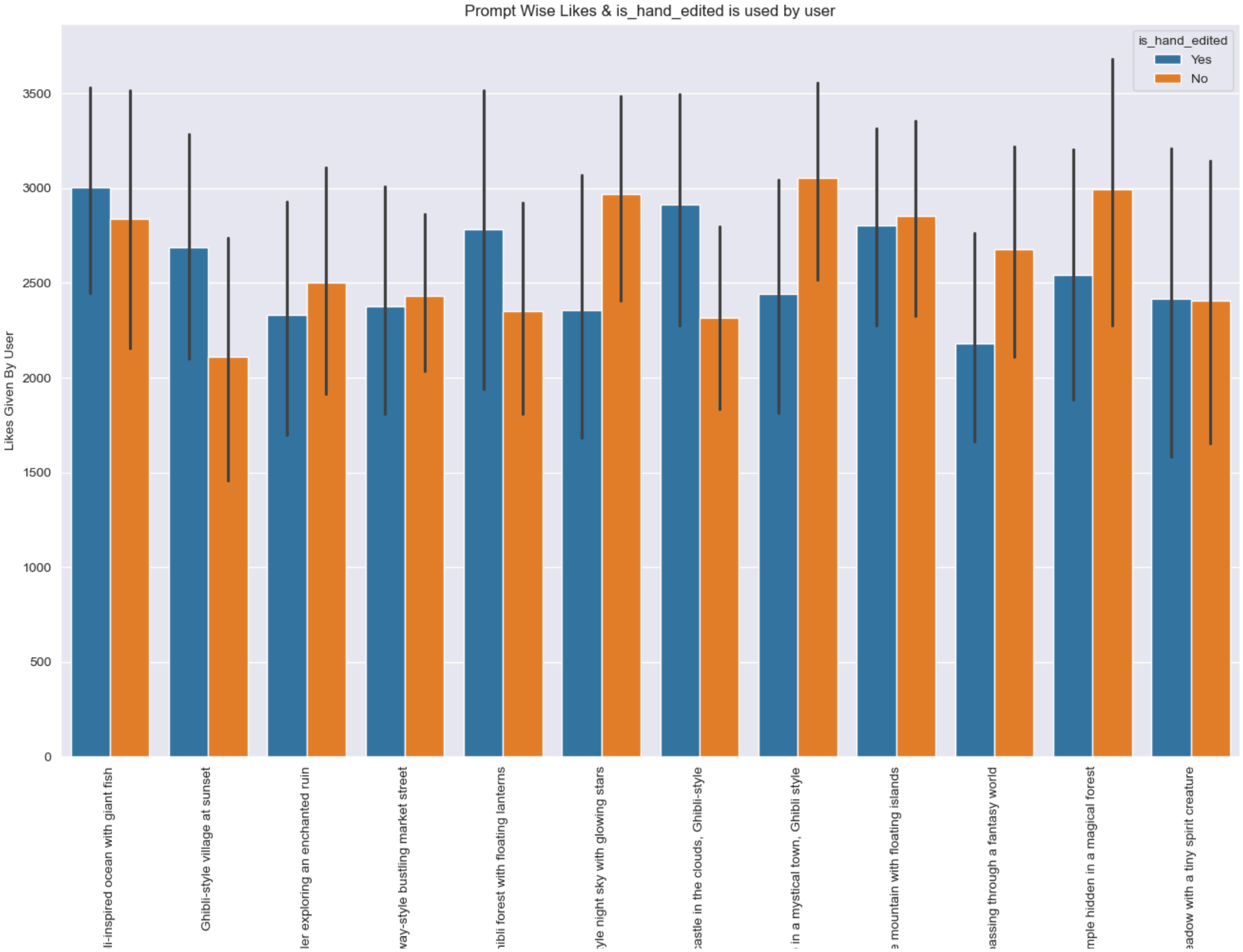
In [73]:
```
# Let's Check the promt wise style_accuracy_score and platform is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "style_accuracy_score",hue = "platform",data = data)
plt.title("Prompt Wise style_accuracy_score & platform is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("style_accuracy_score taken By Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise style_accuracy_score & platform is used by user
style_accuracy_score taken By Image Generation
platform
Reddit
Instagram
TikTok
Twitter
100
80
60
40
20
0
i-inspired ocean with giant fish
Ghibli-style village at sunset
ar exploring an enchanted ruin
ay-style bustling market street
ibli forest with floating lanterns
rle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
mountain with floating islands
assing through a fantasy world
nple hidden in a magical forest
adow with a tiny spirit creature

Prompt "Ghibli style village at sunset" getting the highest image accuracy score and images are posted on the platform of Reddit.

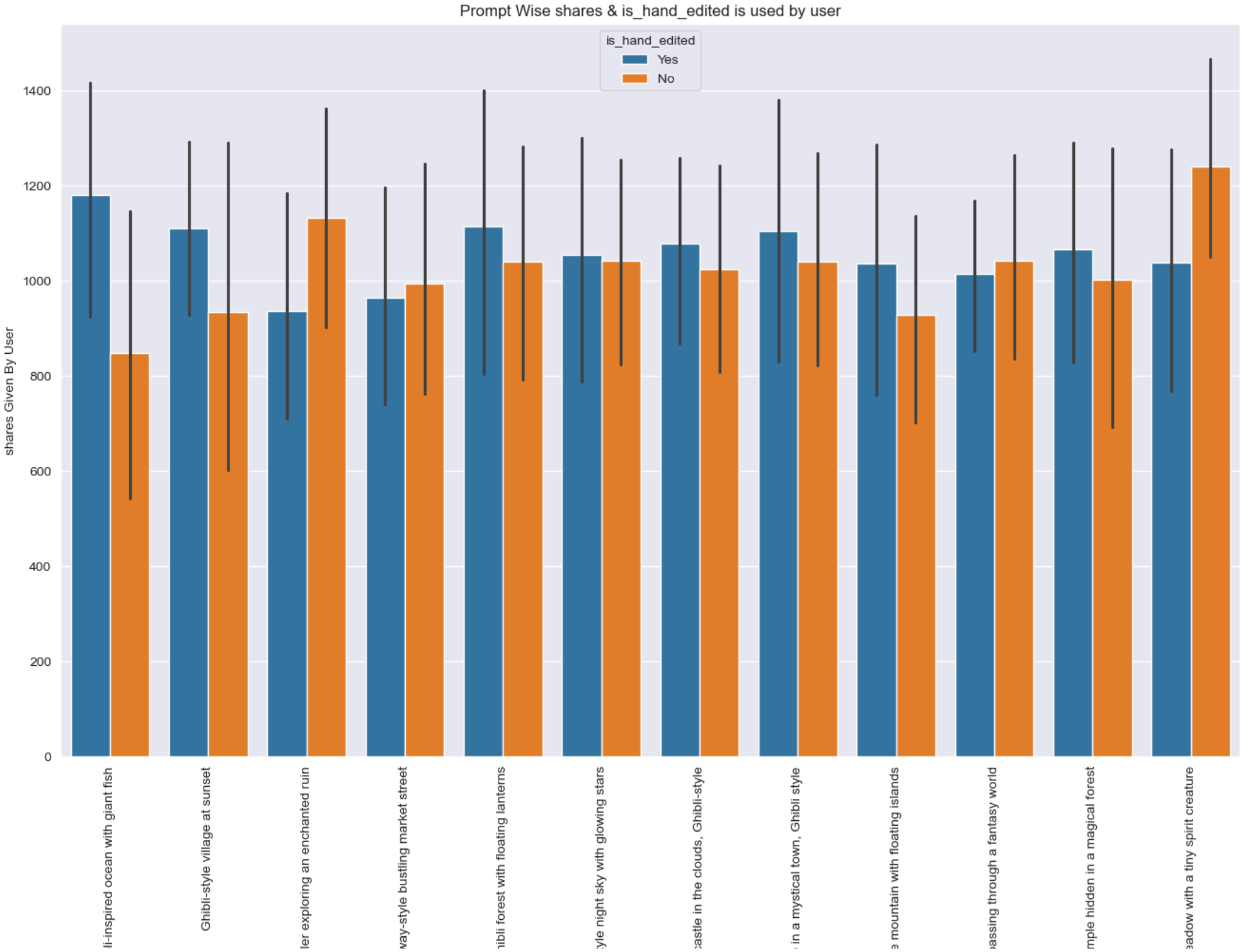
Hue is is_hand_edited

In [74]:
```
# Let's Check the promt wise likes and is_hand_edited is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "likes",hue = "is_hand_edited",data = data)
plt.title("Prompt Wise Likes & is_hand_edited is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("Likes Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise Likes & is_hand_edited is used by user
is_hand_edited
Yes
No
Likes Given By User
0
500
1000
1500
2000
2500
3000
3500
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

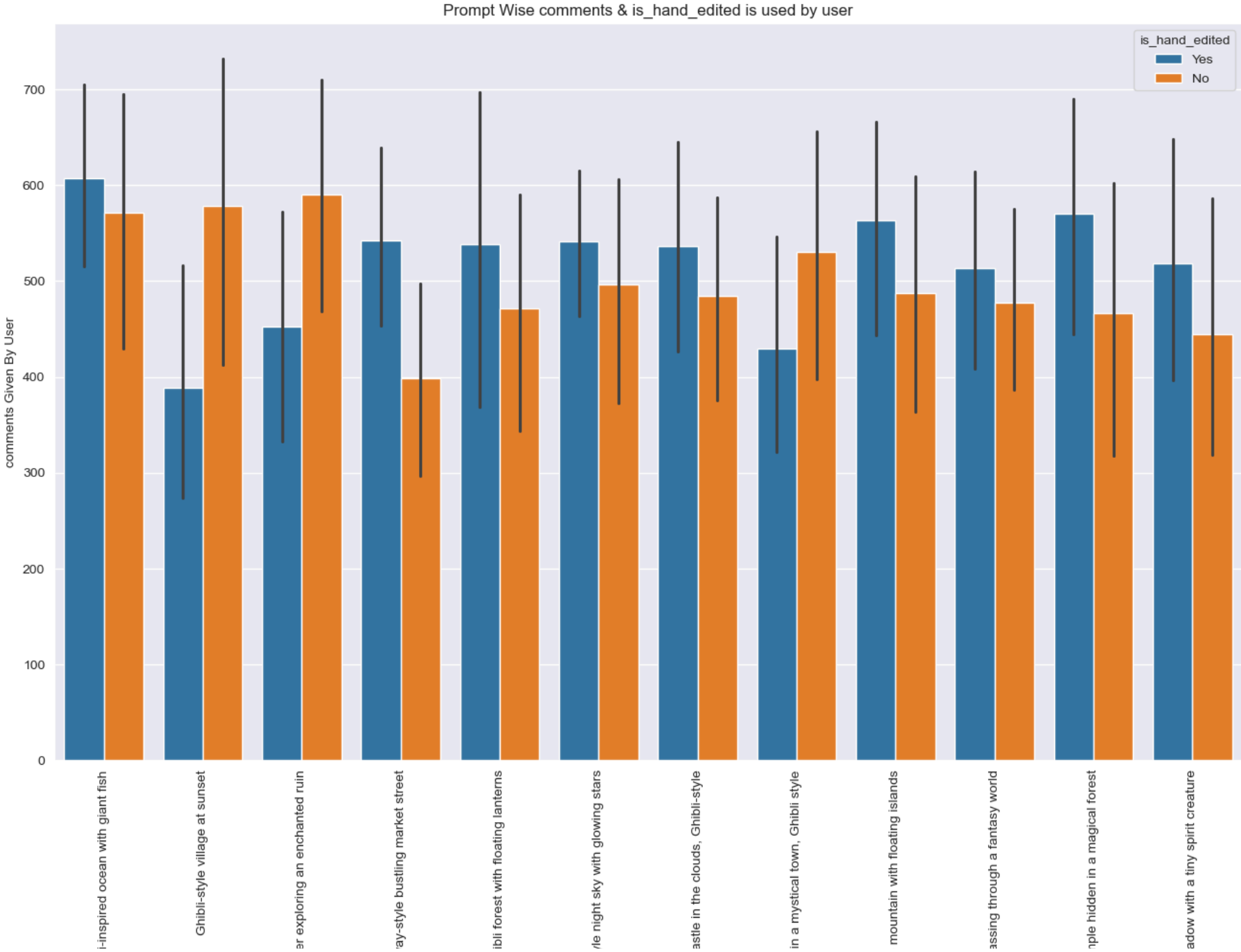
Most of the images is generated then after the user is ownly edited and those types of images gets the highest likes on their posts.

In [75]:
```
# Let's Check the promt wise shares and is_hand_edited is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "shares",hue = "is_hand_edited",data = data)
plt.title("Prompt Wise shares & is_hand_edited is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("shares Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise shares & is_hand_edited is used by user
is_hand_edited
Yes
No
shares Given By User
0
200
400
600
800
1000
1200
1400
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

Most of the images is generated then after the user is ownly edited and those types of images gets the highest shares on their posts.

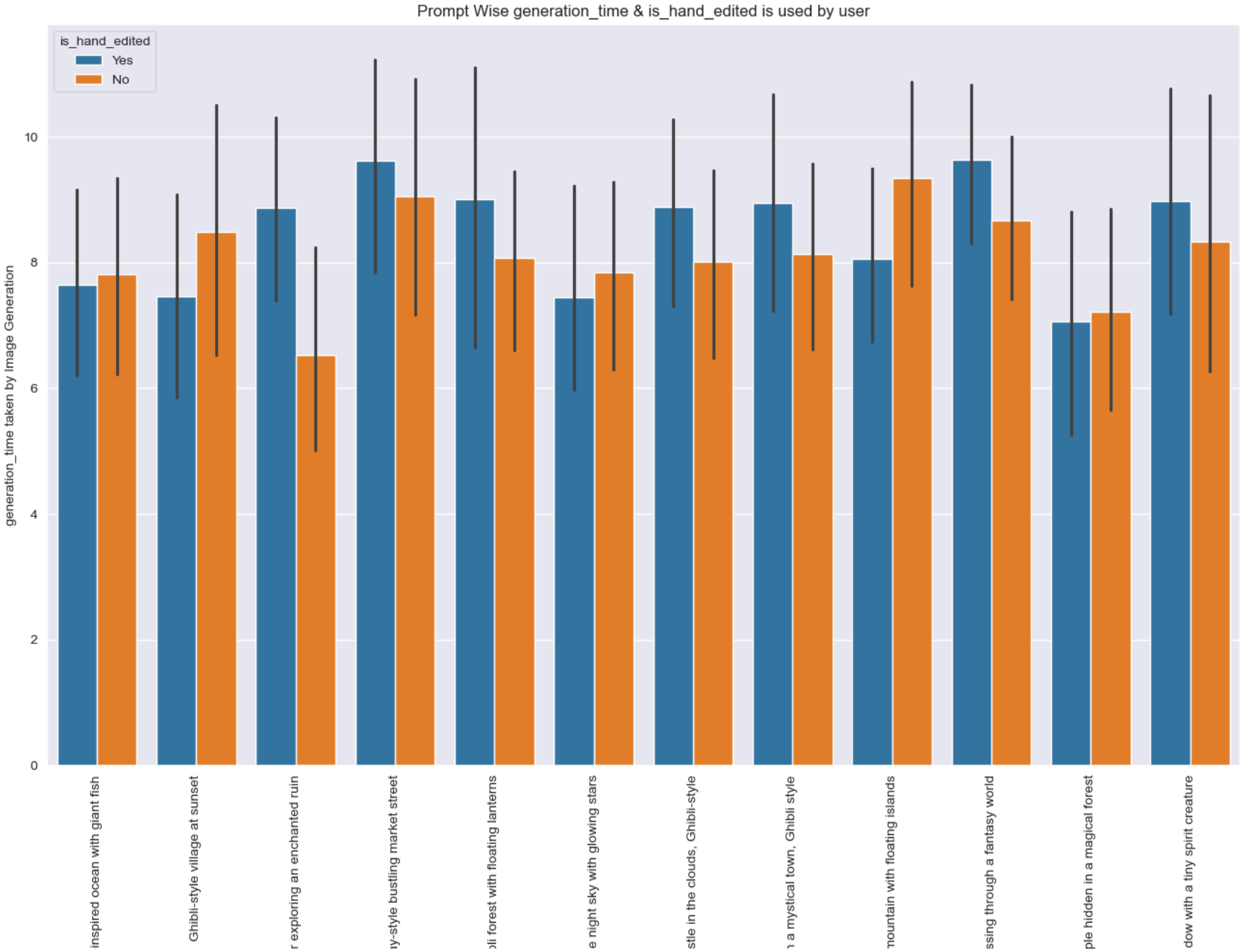
In [76]:
```
# Let's Check the promt wise comments and is_hand_edited is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "comments",hue = "is_hand_edited",data = data)
plt.title("Prompt Wise comments & is_hand_edited is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("comments Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise comments & is_hand_edited is used by user
is_hand_edited
Yes
No
comments Given By User
0
100
200
300
400
500
600
700
i-inspired ocean with giant fish
Ghibli-style village at sunset
ar exploring an enchanted ruin
ay-style bustling market street
ibli forest with floating lanterns
rle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
mountain with floating islands
assing through a fantasy world
nple hidden in a magical forest
adow with a tiny spirit creature

Most of the images is generated then after the user is ownly edited and those types of images gets the highest comments on their posts.

Here again the winner is those user edited their images after the generation they getting the highest people comments on their posts.

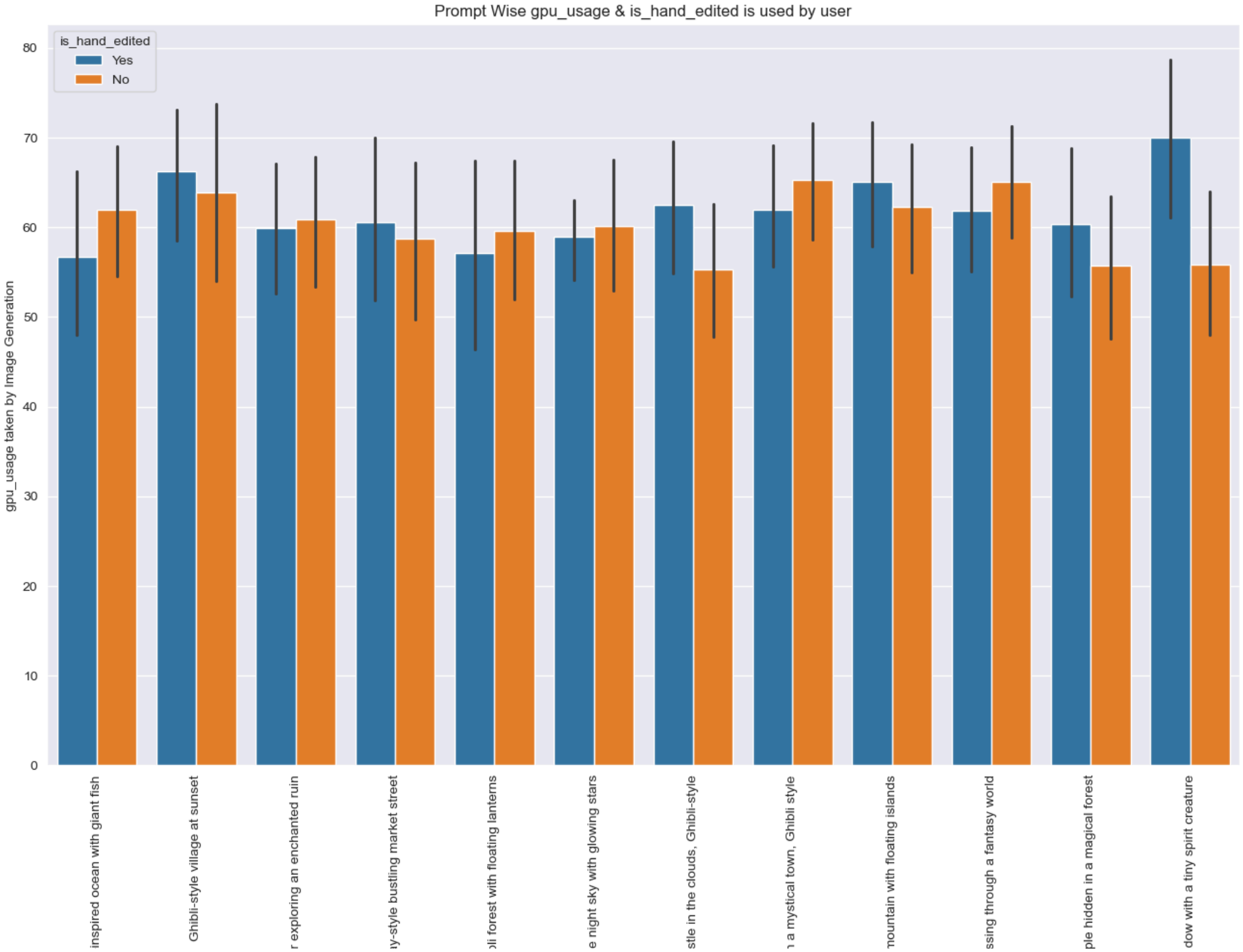
In [77]:
```
# Let's Check the promt wise generation_time and is_hand_edited is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "generation_time",hue = "is_hand_edited",data = data)
plt.title("Prompt Wise generation_time & is_hand_edited is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("generation_time taken by Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise generation_time & is_hand_edited is used by user
is_hand_edited
Yes
No
generation_time taken by Image Generation
0
2
4
6
8
10
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
ıy-style bustling market street
ıli forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
ı a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature

You can see clearly those images takes the higher generation time people also edited after the image generation.

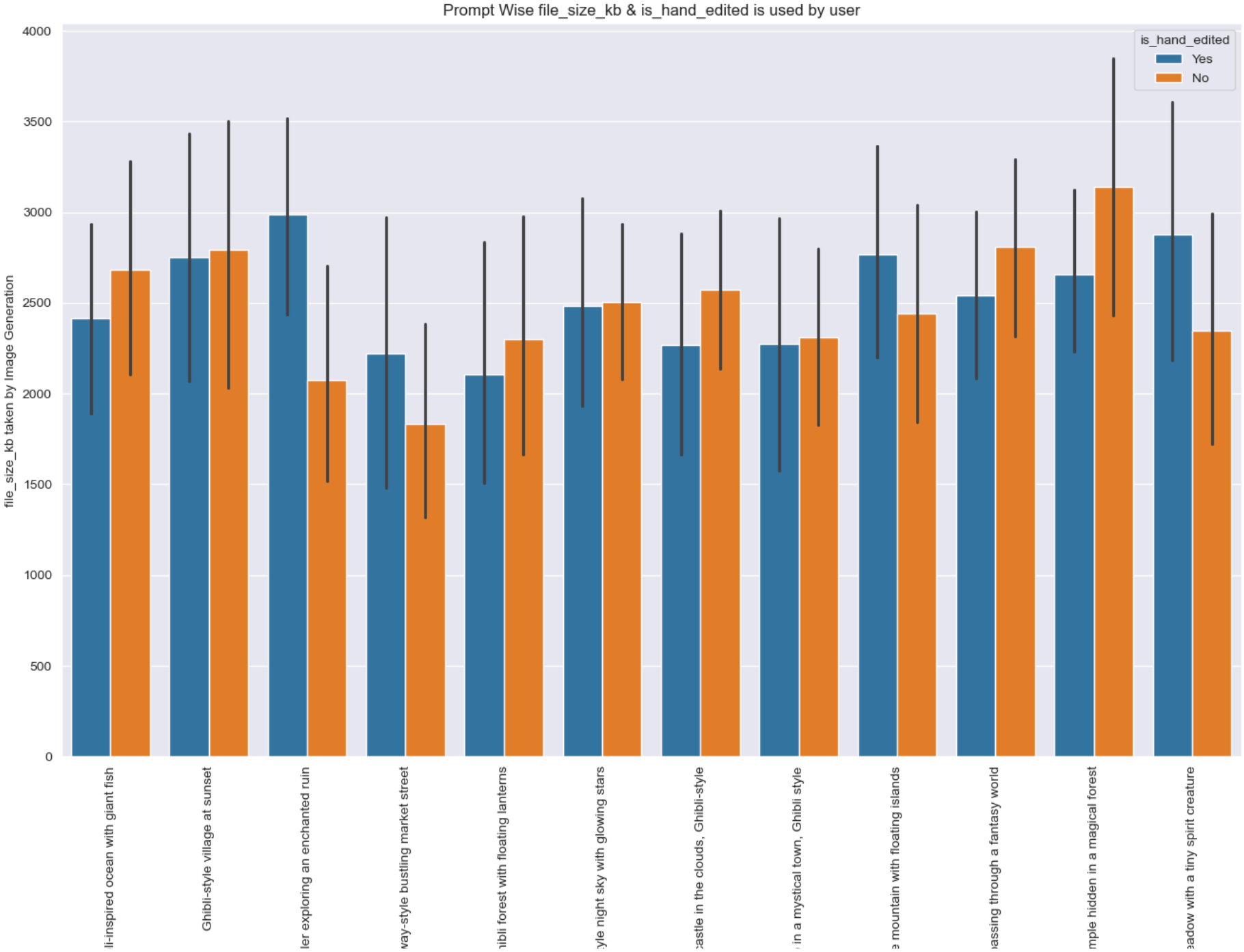
We can say that the image generation time is high but the user favourite image is not generated may be.

In [78]:
```
# Let's Check the promt wise gpu_usage and is_hand_edited is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "gpu_usage",hue = "is_hand_edited",data = data)
plt.title("Prompt Wise gpu_usage & is_hand_edited is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("gpu_usage taken by Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise gpu_usage & is_hand_edited is used by user
is_hand_edited
Yes
No
gpu_usage taken by Image Generation
0
10
20
30
40
50
60
70
80
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
y-style bustling market street
li forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature

You can see clearly the prompt "Serene meadow with a tiny spirit creature" getting the highest gpu time for image generation and people also want to edit in their images.

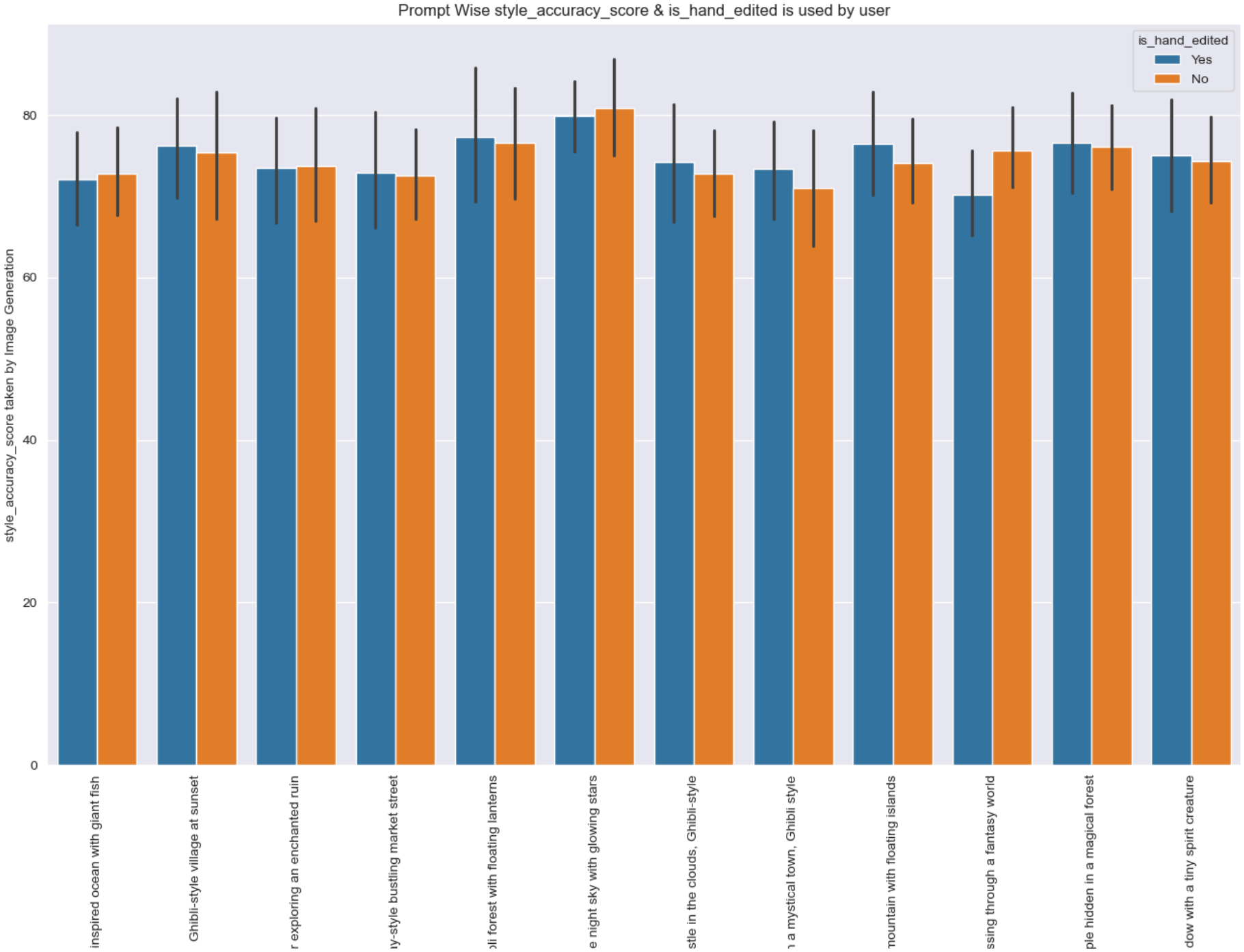
In [79]:
```
# Let's Check the promt wise file_size_kb and is_hand_edited is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "file_size_kb",hue = "is_hand_edited",data = data)
plt.title("Prompt Wise file_size_kb & is_hand_edited is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("file_size_kb taken by Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise file_size_kb & is_hand_edited is used by user
file_size_kb taken by Image Generation
is_hand_edited
Yes
No
0
500
1000
1500
2000
2500
3000
3500
4000
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

Everything is okay not the major diffrent in all the prompts whether the hand is used and their images sizes are high.

But Interesting thing with the prompt "Mysterious temple hidden in a magical forest" images required higher spaces and there is no user edited images.

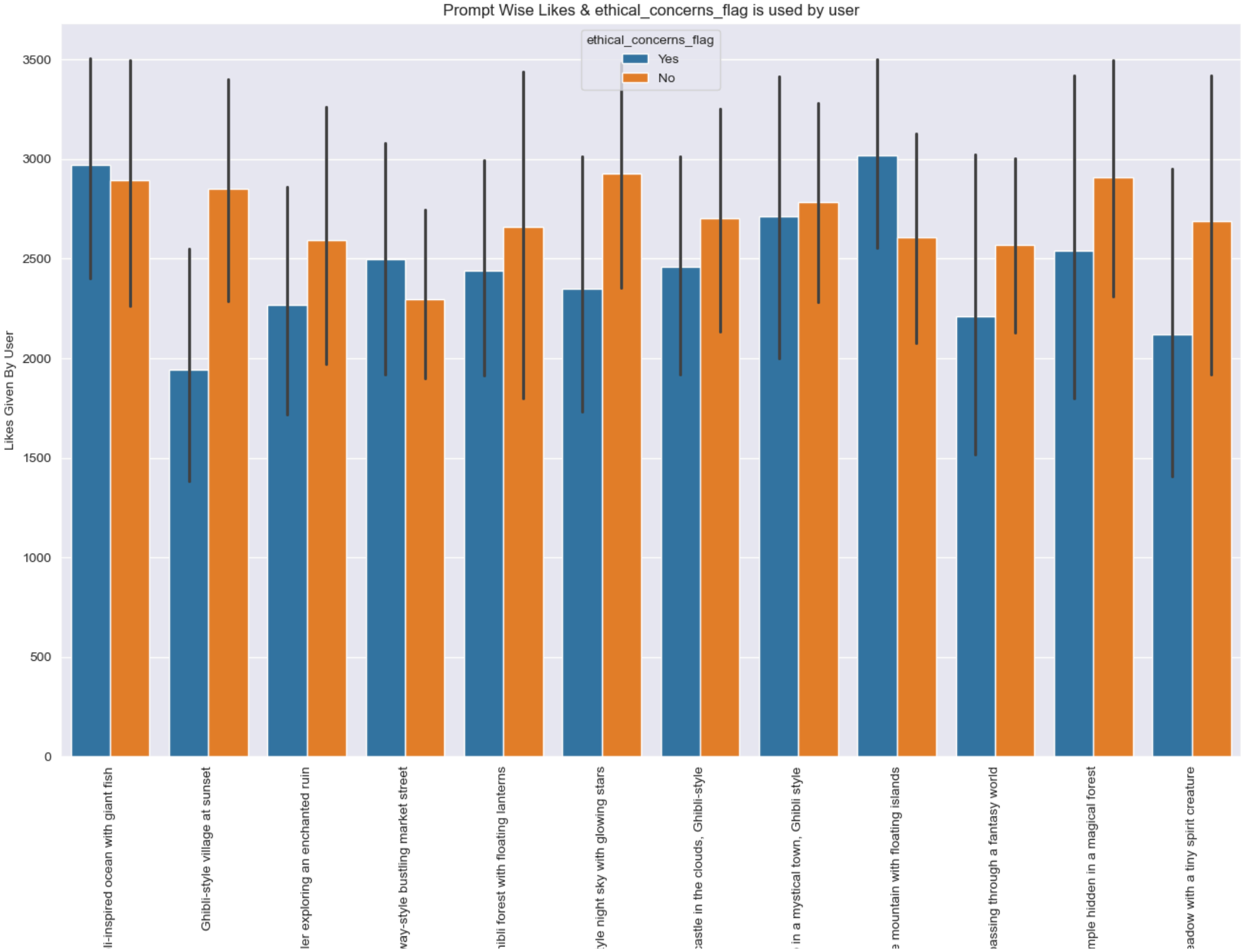
In [80]:
```
# Let's Check the promt wise style_accuracy_score and is_hand_edited is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "style_accuracy_score",hue = "is_hand_edited",data = data)
plt.title("Prompt Wise style_accuracy_score & is_hand_edited is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("style_accuracy_score taken by Image Generation")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise style_accuracy_score & is_hand_edited is used by user
is_hand_edited
Yes
No
style_accuracy_score taken by Image Generation
0
20
40
60
80
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
y-style bustling market street
li forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature

There no major difference in all the prompts but the interesting thing is prompt "Ghibli style night sky with glowing stars" getting the highest image accuracy score and people don't want to edit.

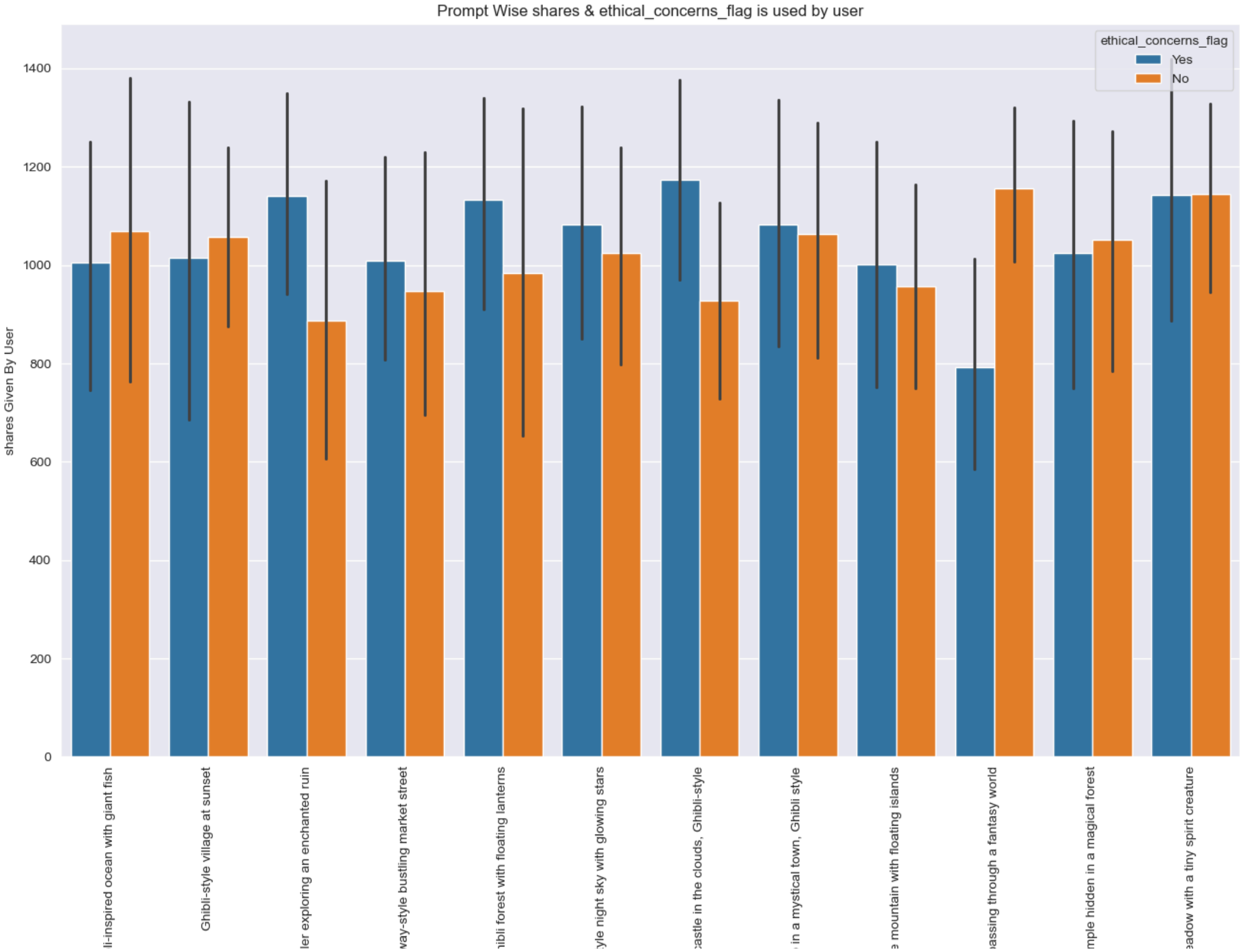
Hue is ethical_concerns_flag

In [81]:
```
# Let's Check the promt wise likes and ethical_concerns_flag is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "likes",hue = "ethical_concerns_flag",data = data)
plt.title("Prompt Wise Likes & ethical_concerns_flag is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("Likes Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise Likes & ethical_concerns_flag is used by user
ethical_concerns_flag
Yes
No
Likes Given By User
0
500
1000
1500
2000
2500
3000
3500
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

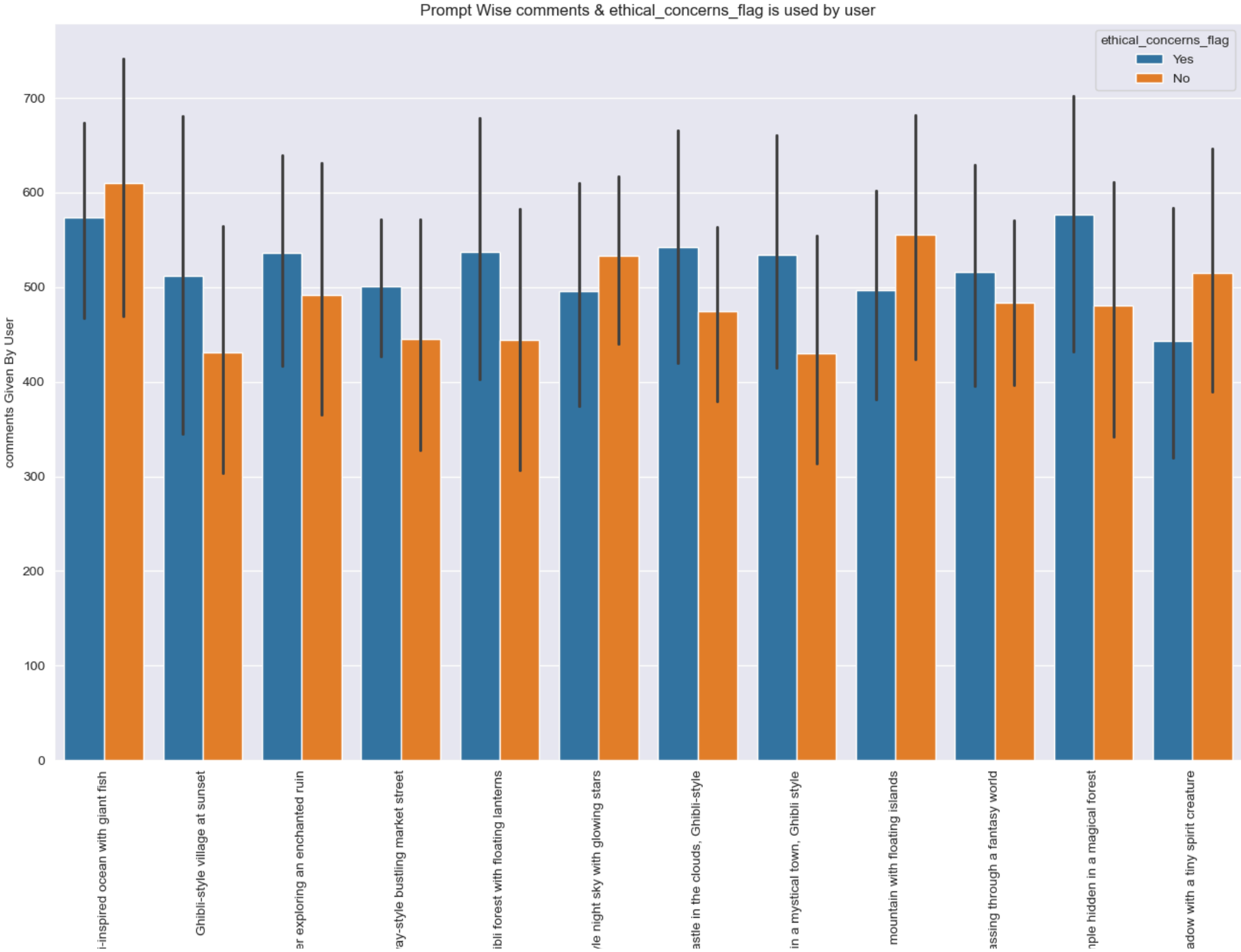
You can see clearly those images users has no ethical concern flags getting the highest likes.

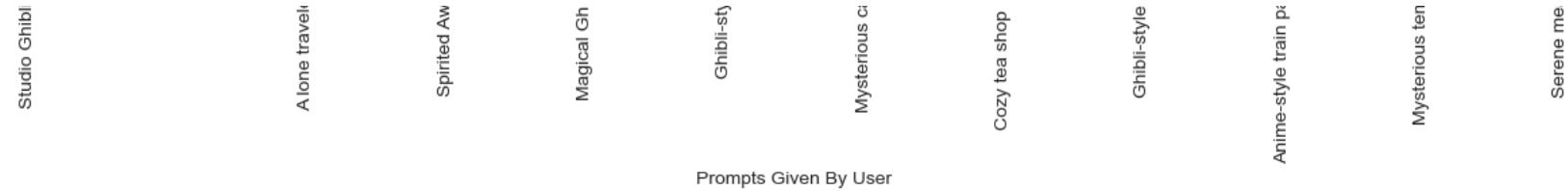
In [82]:
```
# Let's Check the promt wise shares and ethical_concerns_flag is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "shares",hue = "ethical_concerns_flag",data = data)
plt.title("Prompt Wise shares & ethical_concerns_flag is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("shares Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise shares & ethical_concerns_flag is used by user
ethical_concerns_flag
Yes
No
shares Given By User
0
200
400
600
800
1000
1200
1400
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

You can see clearly those images are fall in ethical concern flag category cause highest sharing.

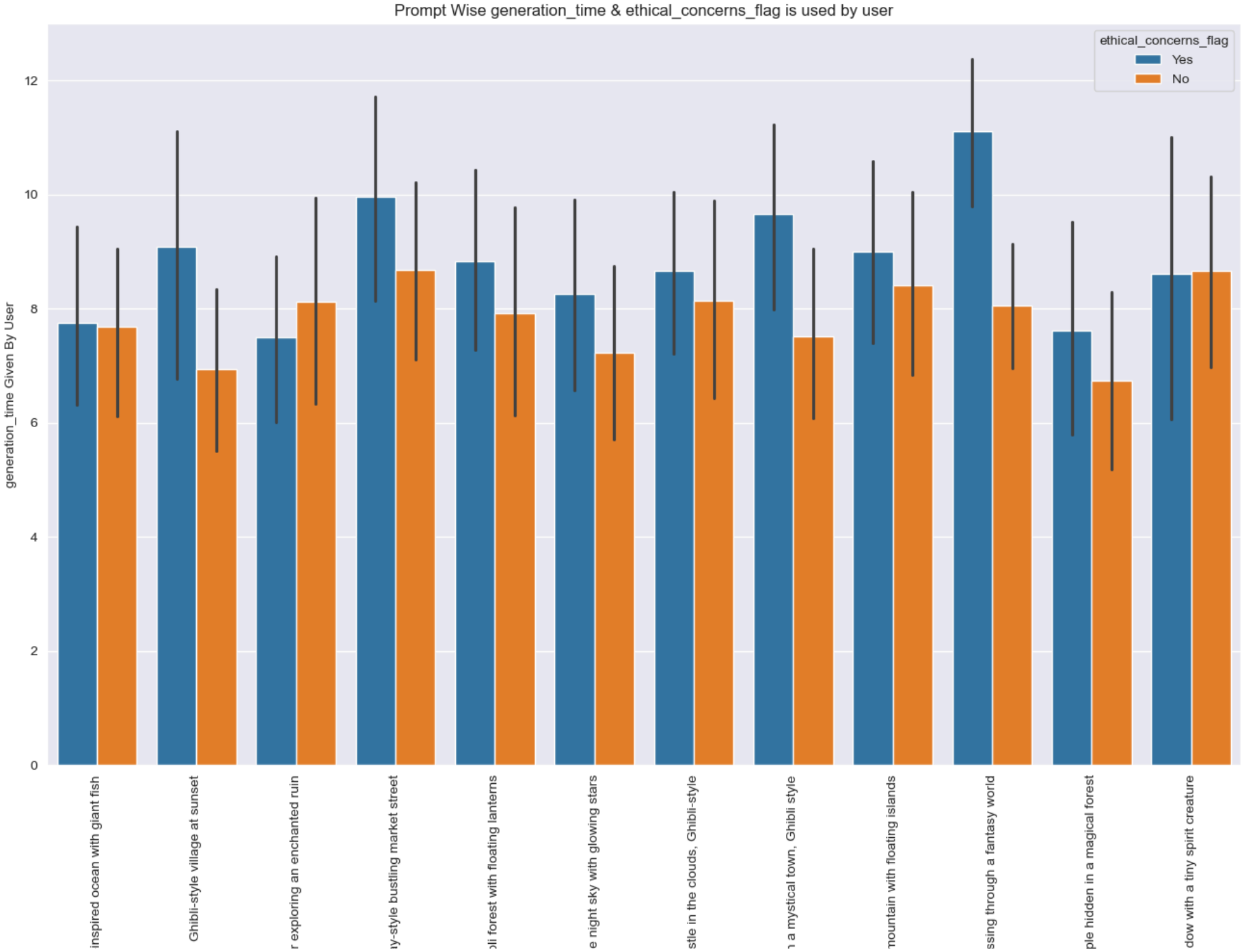
In [83]:
```
# Let's Check the promt wise comments and ethical_concerns_flag is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "comments",hue = "ethical_concerns_flag",data = data)
plt.title("Prompt Wise comments & ethical_concerns_flag is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("comments Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise comments & ethical_concerns_flag is used by user
ethical_concerns_flag
Yes
No
comments Given By User
0
100
200
300
400
500
600
700
i-inspired ocean with giant fish
Ghibli-style village at sunset
ar exploring an enchanted ruin
ay-style bustling market street
ibli forest with floating lanterns
le night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
mountain with floating islands
assing through a fantasy world
nple hidden in a magical forest
adow with a tiny spirit creature



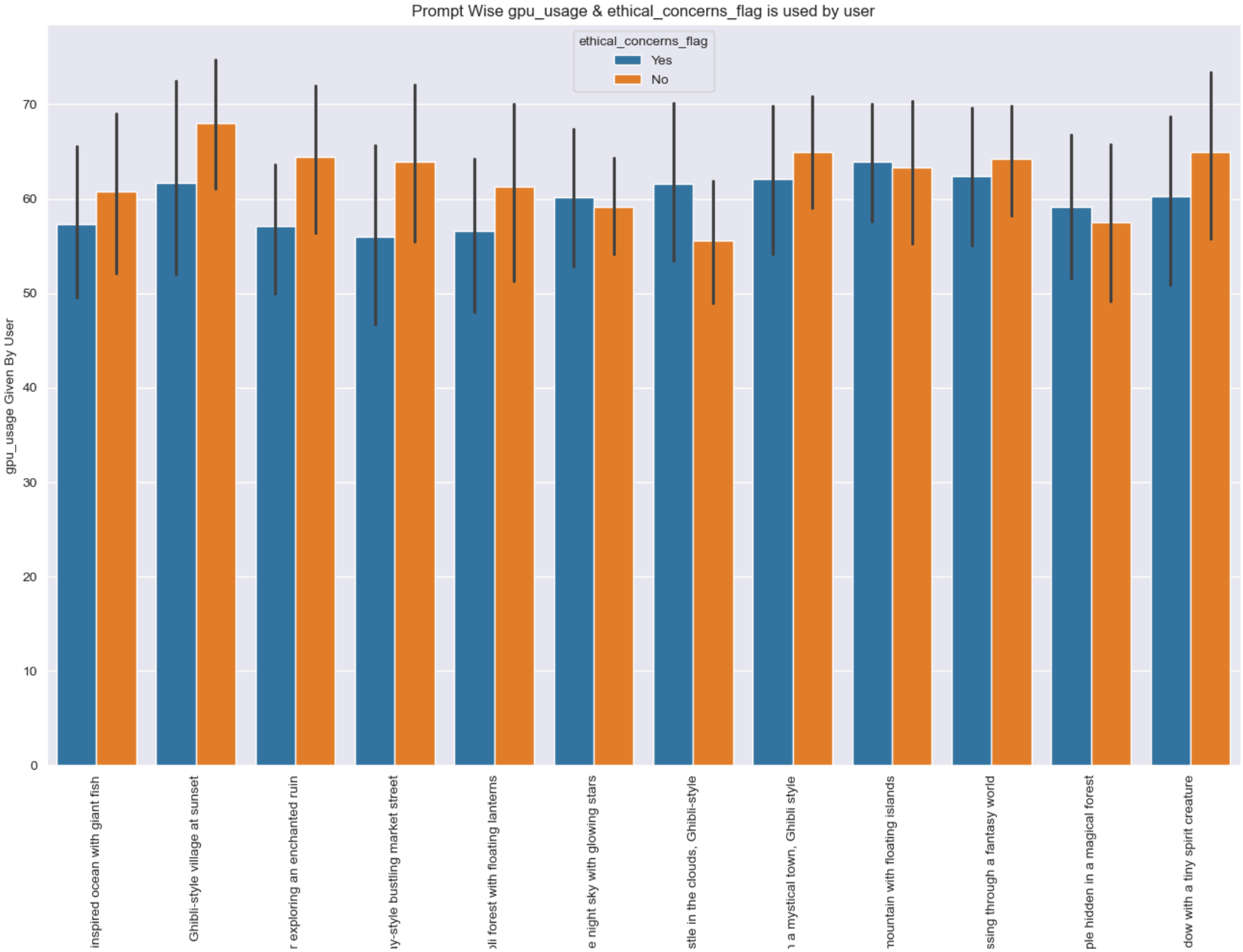
You can see Prompt "Studio Ghibli Inspired ocean with giant fish" images getting highest comments and interesting thing is there is no ethical concern flag.

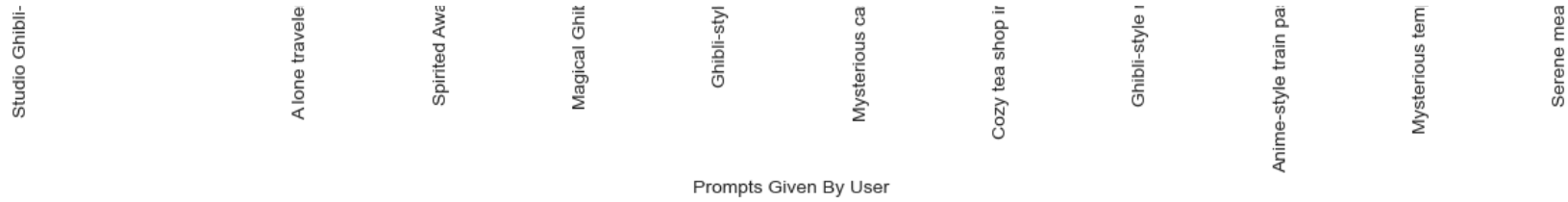
In [84]:
```
# Let's Check the promt wise generation_time and ethical_concerns_flag is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "generation_time",hue = "ethical_concerns_flag",data = data)
plt.title("Prompt Wise generation_time & ethical_concerns_flag is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("generation_time Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise generation_time & ethical_concerns_flag is used by user
ethical_concerns_flag
Yes
No
generation_time Given By User
0
2
4
6
8
10
12
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
y-style bustling market street
li forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature

You can see clearly the prompt "Anime style passing through a fantasy world" images are taking highest image generation time and that majority of all images are fall in ethical concern yes category.

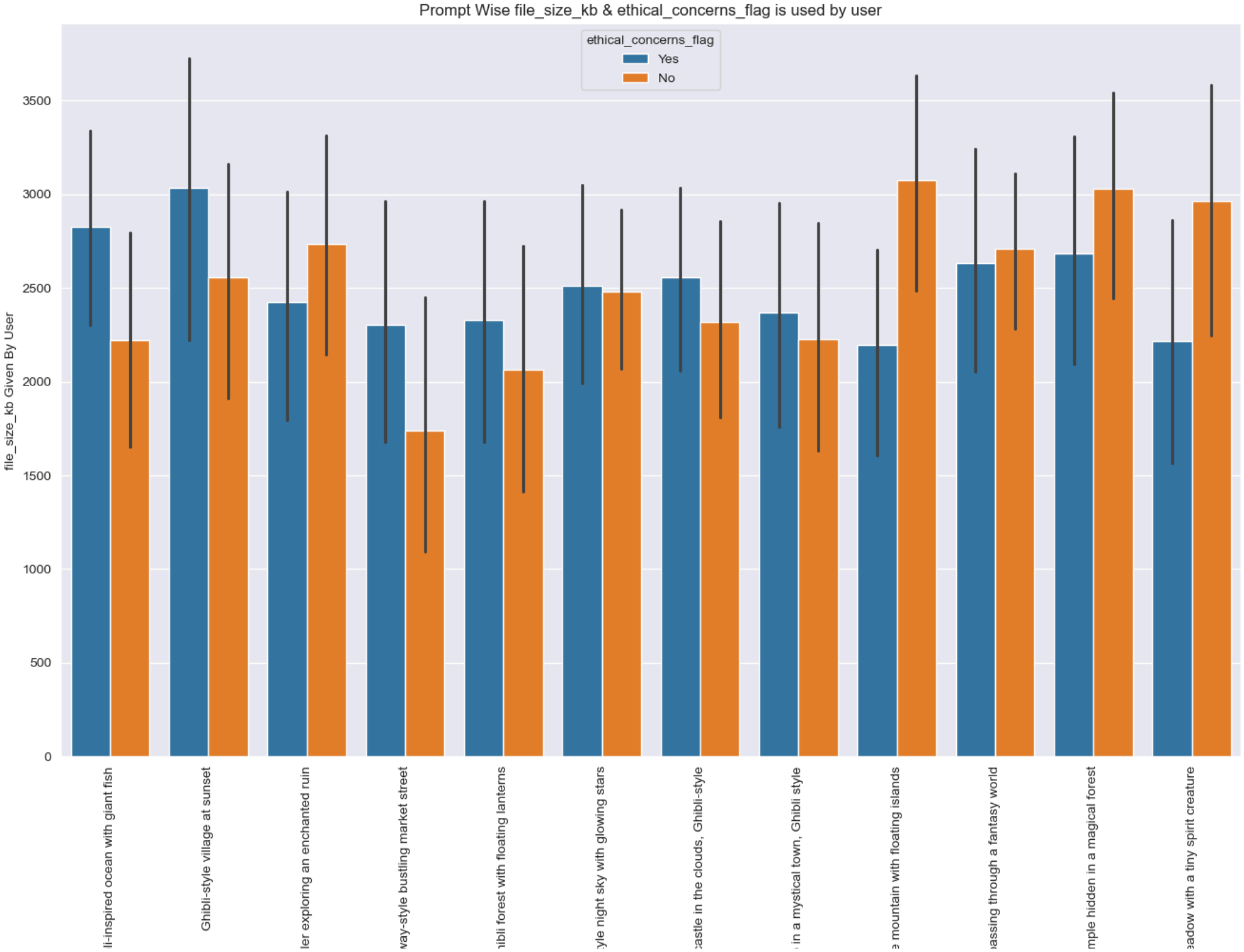
In [85]:
```
# Let's Check the promt wise gpu_usage and ethical_concerns_flag is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "gpu_usage",hue = "ethical_concerns_flag",data = data)
plt.title("Prompt Wise gpu_usage & ethical_concerns_flag is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("gpu_usage Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise gpu_usage & ethical_concerns_flag is used by user
ethical_concerns_flag
Yes
No
gpu_usage Given By User
0
10
20
30
40
50
60
70
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
y-style bustling market street
li forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature



Prompt "Ghibli style village at sunset" taking the highest gpu time & fall in no ethical concern category among the all prompts.

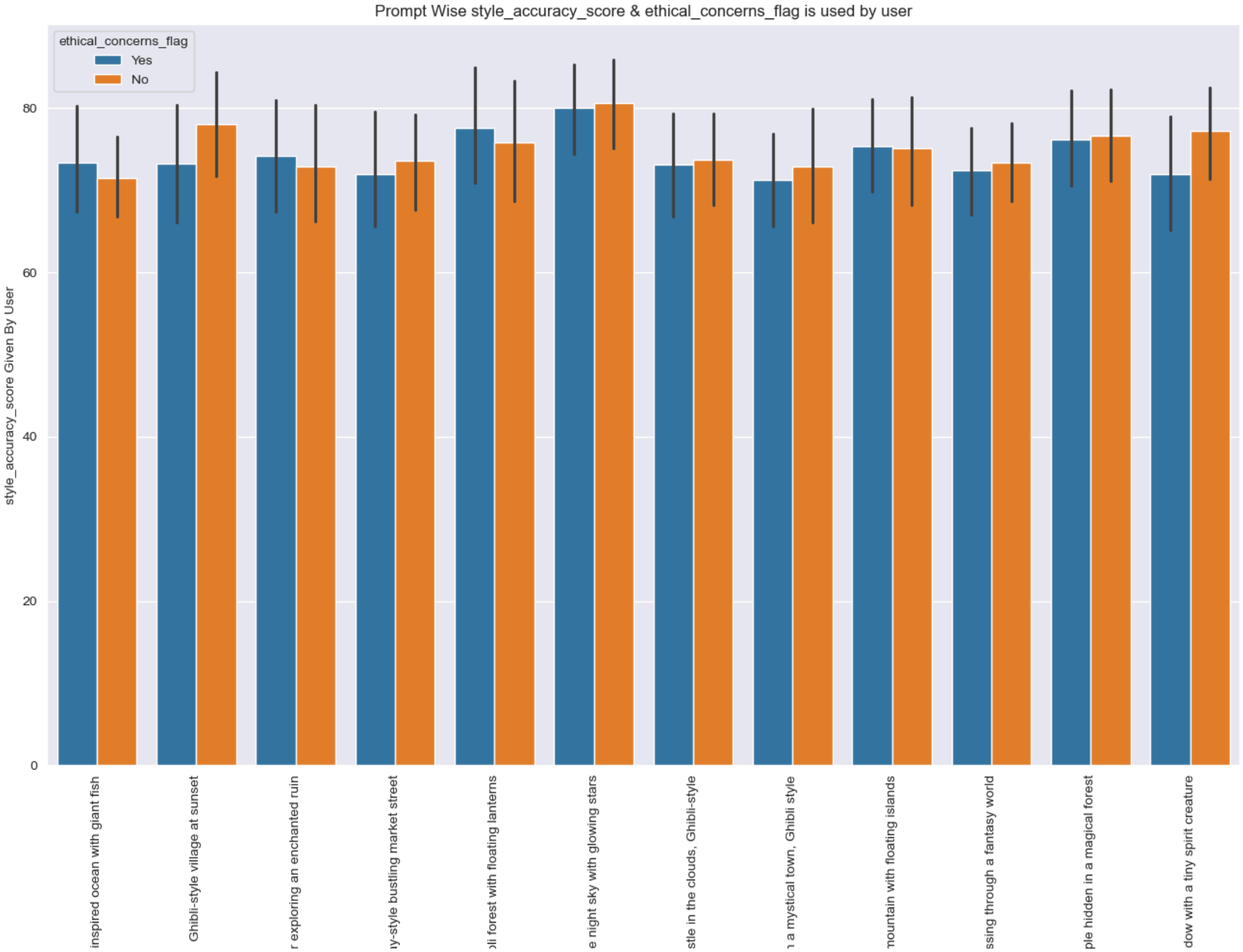
In [86]:
```
# Let's Check the promt wise file_size_kb and ethical_concerns_flag is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "file_size_kb",hue = "ethical_concerns_flag",data = data)
plt.title("Prompt Wise file_size_kb & ethical_concerns_flag is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("file_size_kb Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise file_size_kb & ethical_concerns_flag is used by user
ethical_concerns_flag
Yes
No
file_size_kb Given By User
0
500
1000
1500
2000
2500
3000
3500
li-inspired ocean with giant fish
Ghibli-style village at sunset
ler exploring an enchanted ruin
way-style bustling market street
ibli forest with floating lanterns
yle night sky with glowing stars
astle in the clouds, Ghibli-style
in a mystical town, Ghibli style
a mountain with floating islands
assing through a fantasy world
mple hidden in a magical forest
eadow with a tiny spirit creature

Majority of the all the images above 2500 kbs but the few are 3500 kbs variations and interesting thing is that prompt,"Ghibli style village at sunset" is yes ehical concern.

In the opposite side prompt "Ghibli style mountain with floating islands" take the highest size & no generate the images with no ethical concern.

In [87]:
```
# Let's Check the promt wise style_accuracy_score and ethical_concerns_flag is used by user.

plt.figure(figsize=(16, 10))
sns.set_style("darkgrid")
sns.barplot(x = "prompt",y = "style_accuracy_score",hue = "ethical_concerns_flag",data = data)
plt.title("Prompt Wise style_accuracy_score & ethical_concerns_flag is used by user")
plt.xlabel("Prompts Given By User")
plt.ylabel("style_accuracy_score Given By User")
plt.xticks(rotation = "vertical")
plt.show()
```


Prompt Wise style_accuracy_score & ethical_concerns_flag is used by user
ethical_concerns_flag
Yes
No
style_accuracy_score Given By User
0
20
40
60
80
inspired ocean with giant fish
Ghibli-style village at sunset
r exploring an enchanted ruin
y-style bustling market street
li forest with floating lanterns
e night sky with glowing stars
stle in the clouds, Ghibli-style
a mystical town, Ghibli style
ountain with floating islands
ssing through a fantasy world
ple hidden in a magical forest
dow with a tiny spirit creature

The winner is prompt "Ghibli style nighe sky with glowing stars" getting the highest accuracy score and there is no diffrence between ethical concerns but slightly no ethical concerns images are higher.

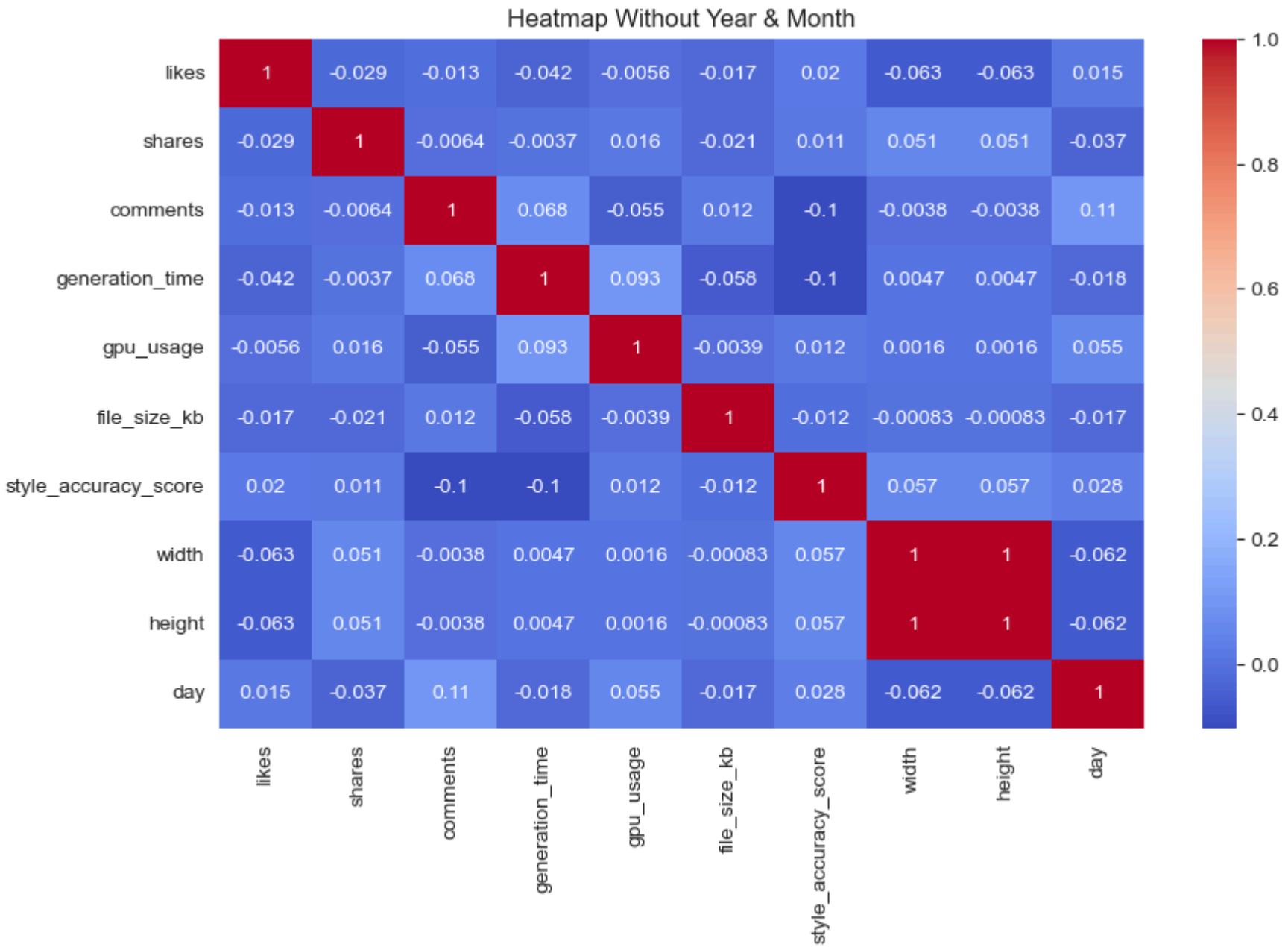
In [88]:
```
# let's check the relationship of all the variables

relation_variables = data.corr(numeric_only = True)
relation_variables
```

Out[88]:

	likes	shares	comments	generation_time	gpu_usage	file_size_kb	style_accuracy_score	width	height
likes	1.000000	-0.029318	-0.013297	-0.041881	-0.005609	-0.016744	0.019640	-0.063076	-0.063076
shares	-0.029318	1.000000	-0.006400	-0.003688	0.015755	-0.020748	0.011279	0.050954	0.050954
comments	-0.013297	-0.006400	1.000000	0.068484	-0.054742	0.012197	-0.102865	-0.003820	-0.003820
generation_time	-0.041881	-0.003688	0.068484	1.000000	0.092766	-0.057717	-0.101120	0.004742	0.004742
gpu_usage	-0.005609	0.015755	-0.054742	0.092766	1.000000	-0.003873	0.011885	0.001576	0.001576
file_size_kb	-0.016744	-0.020748	0.012197	-0.057717	-0.003873	1.000000	-0.011579	-0.000831	-0.000831
style_accuracy_score	0.019640	0.011279	-0.102865	-0.101120	0.011885	-0.011579	1.000000	0.056687	0.056687
width	-0.063076	0.050954	-0.003820	0.004742	0.001576	-0.000831	0.056687	1.000000	1.000000
height	-0.063076	0.050954	-0.003820	0.004742	0.001576	-0.000831	0.056687	1.000000	1.000000
year	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
month	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
day	0.015398	-0.036627	0.107089	-0.017634	0.054681	-0.016966	0.028469	-0.062244	-0.062244

In [89]:

```
# let's generate the heatmap so we can identify the relationship of all the variables

filtered_data = data.drop(columns=['year', 'month'])
plt.figure(figsize=(10, 6))
sns.heatmap(filtered_data.corr(numeric_only= True), annot=True, cmap='coolwarm')
plt.title("Heatmap Without Year & Month")
plt.show()
```



There is no strong relationships between all the variables.

Insights from Ghibli-Style Image Trends Dataset EDA Project

1.Most frequent prompt is Anime-Style train passing through a fantasy world with 2048×2048 resolution, high GPU usage, very slow generation time, medium file size, excellent accuracy, no ethical concern, and mostly generated on March 3rd, 31st, and Sundays.

2.Twitter is the top platform with highest likes, comments, and shares especially for Studio Ghibli-Inspired Ocean with giant fish which also shows high GPU time and image accuracy.

3.Instagram hosts popular prompts like Ghibli-style night sky with glowing stars, Mysterious castle in the clouds, and Mysterious temple hidden in a magical forest with large file sizes and high accuracy.

4.Reddit features Ghibli-style content with high accuracy and larger image sizes, especially Ghibli-style village at sunset which also shows high GPU time.

5.TikTok hosts heavy prompts like Spirited Away style bustling market street and Serene meadow with a tiny spirit creature with highest image dimensions, large sizes, and longer generation times.

6.Most images were created in March 2025 with peak activity on Wednesdays, Fridays, Sundays, and Mondays based on likes, shares, comments, generation time, and accuracy.

7.Hand-edited images tend to get more likes and comments, especially when GPU time and generation time are high, while non-edited images receive higher shares.

8.Prompts with no ethical concern flags consistently receive higher likes, comments, and accuracy, but interestingly, flagged prompts tend to get more shares and longer generation times.

9.Ghibli-style mountain with floating islands is notable for high GPU usage, large file size, and high generation time with no ethical concerns or editing.

10.Across platforms, Studio Ghibli themed prompts dominate trends with high engagement, performance scores, and aesthetic preference.

11.'Ghibli-style','Mysterious','with' is the most common words while prompting.

12.Most common word in the top comments is 'Ghibli'.