Fintech Group Project 01

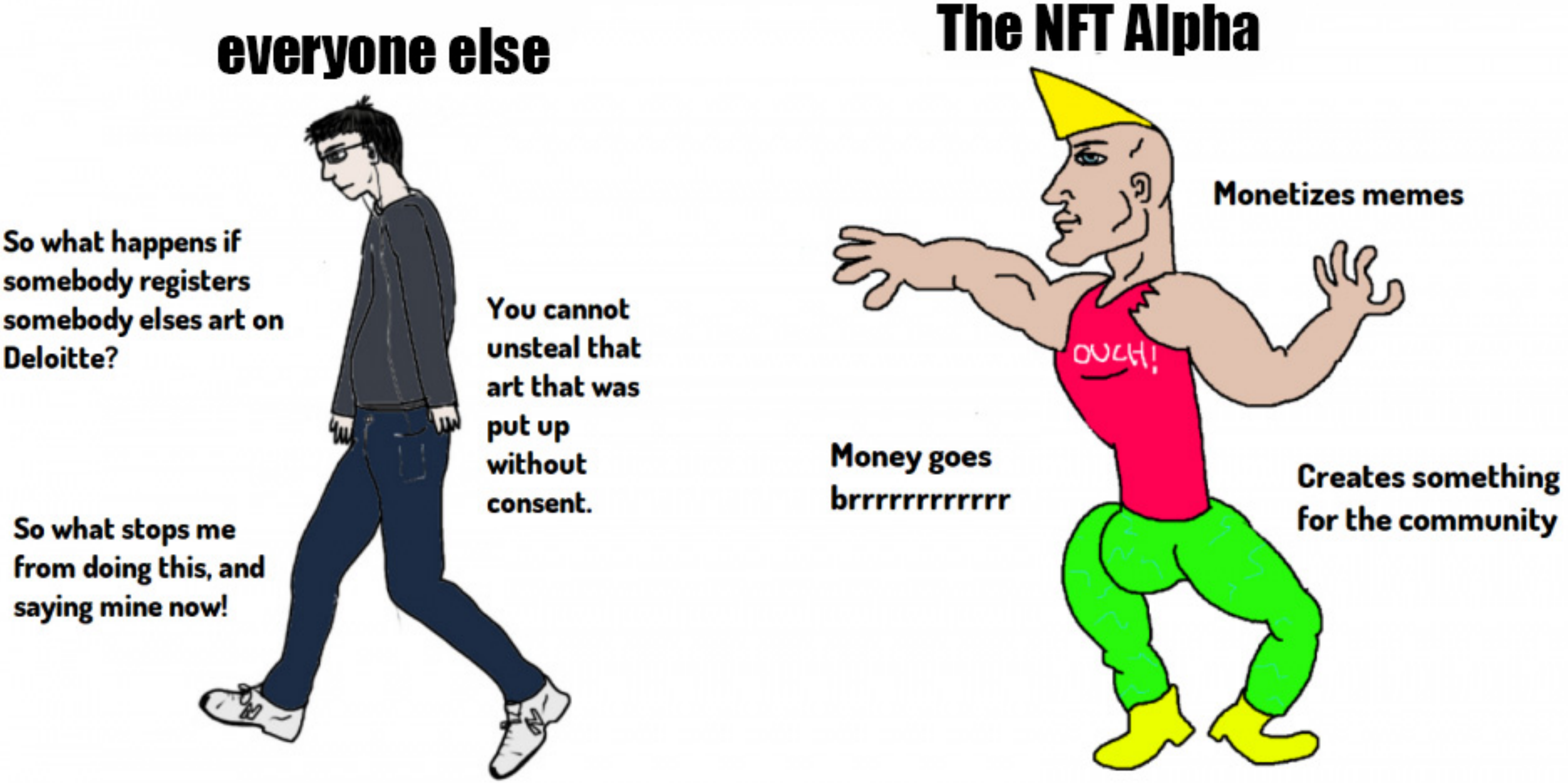
Azuki, BAYC and Crypto Punks NFTs.

by Github users /@dockingbay24 /@angel-estrada7 and /@mmsaki, **2022.**

Table of Contents

- Hypothesis of Project
- Data Collection
- Data Cleanup & Exploration
- Data Analysis
- Discussion
- Postmortem
- Your Questions?

Azuki Token ID: 7558 **I** **Current Owner:** ron1n.eth **I** **Sale:** 27ETH


everyone else
So what happens if somebody registers somebody elses art on Deloitte?
You cannot unsteal that art that was put up without consent.
So what stops me from doing this, and saying mine now!
The NFT Alpha
Monetizes memes
OUCH!
Money goes brrrrrrrrrrrr
Creates something for the community

1. Hypothesis

Hypothesis of Project

Our motivation and summary…

- Should you invest in Azuki, BAYC or Crypto Punks?
- How can you tell which collection is performing well?

Data Collection

Describing what kinds of data we needed and where to find it.

2. Data Collection

Collecting data for NFT Collections

- Covalent API
- Etherscan Python Dependacy

May 5, 2022

0.1 Import dependancies

[10]:

```
# Import dependancies
import os
import requests
import pandas as pd
import json
from dotenv import load_dotenv
from etherscan_py import etherscan_py
import plotly.express as px
```

[2]:

```
# Loading .env containing our keys
load_dotenv()
```

[2]:

```
True
```

[3]:

```
# create variable for api key
api_key = os.getenv('COVALENT_API_KEY')
type(api_key)
```

[3]:

```
str
```

0.2 Current value of ETH

[4]:

```
# import dependancy
from etherscan_py import etherscan_py
client = etherscan_py.Client(os.getenv('ETHERSCAN_API'))

# Print current eth price and latest block height
eth_value = client.get_eth_price()
eth_value
```

3. Data Cleanup & Exploration

Exploring our collection data through APIs preparing it for analysis

0.3 Set variables

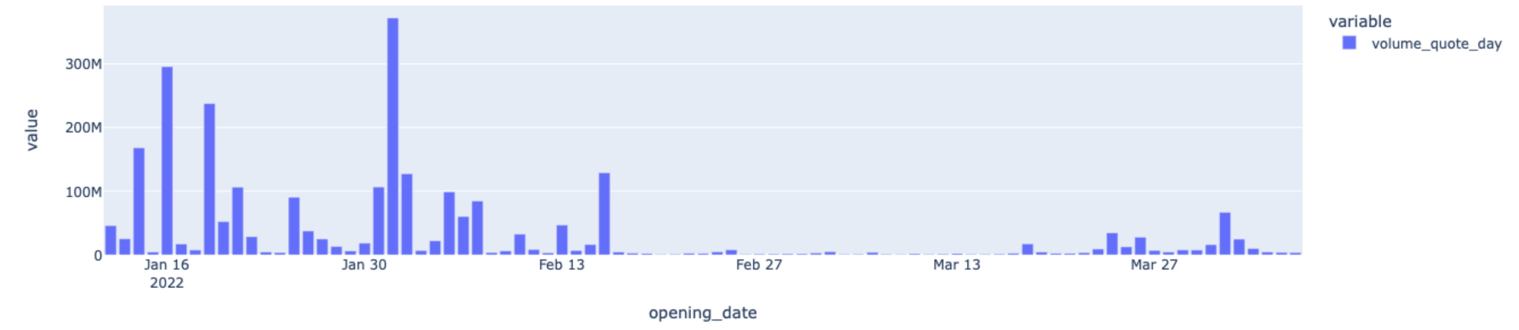
[5]:
```
# Append url for our api
url = "https://api.covalenthq.com/v1"
chain_id = "/1"
azuki_address = "/0xED5AF388653567Af2F388E6224dC7C4b3241C544"
cryptopunks_address = "/0xb47e3cd837dDF8e4c57F05d70Ab865de6e193BBB"
BAYC_address = "/0xBC4CA0EdA7647A8aB7C2061c2E118A18a936f13D"
date_option = '/?quote-currency=USD&format=JSON&from=2017-01-01&to=2022-05-01'
page_option = '/transactions_v2/?
 ↪quote-currency=USD&format=JSON&block-signed-at-asc=false&no-logs=false&page-number=0&page-s
api_option = "&key=" + api_key
api_no_option = '/?key=' + api_key
```

0.4 1. Azuki Daily Volume

[6]:
```
# Create variables needed for owner data and add to url
historical_url = url + chain_id + "/nft_market/collection" + azuki_address +␣
 ↪api_no_option

# Get request
azuki_historical_json = requests.get(historical_url).json()

# Convert historical json data to a dataframe and view data
azuki_df = pd.DataFrame(azuki_historical_json['data']['items'])

# Set index to date
azuki_df = azuki_df.set_index('opening_date')

# Create Volume dataframe
azuki_vol_df = pd.DataFrame(azuki_df, columns = ['volume_quote_day',␣
 ↪'unique_token_ids_sold_count_day']).sort_index()
azuki_vol_df.head()
```

[6]:
```
              volume_quote_day  unique_token_ids_sold_count_day
opening_date
2022-01-12          45941404.0                             2402
2022-01-13          25129178.0                             1318
2022-01-14         168151840.0                              470
2022-01-15           4408686.0                              499
2022-01-16         295638336.0                              368
```

[40]:
```
# Plot Volume quote per day
azuki_volume = azuki_vol_df['volume_quote_day'].astype(int)

# Plot Historical daily volume
px.bar(azuki_volume)
```



0.5 1. Azuki Historical transactions

[8]:
```
# Quering the API for transaction data
azuki_tx_url = url + chain_id + "/address" + azuki_address + page_option +␣
 ↪api_option
azuki_tx = requests.get(azuki_tx_url).json()

# Convert transactions data to dataframe
azuki_tx_df = pd.DataFrame(azuki_tx['data']['items'], columns =␣
 ↪['to_address_label','fees_paid', 'value_quote','block_signed_at']).
 ↪set_index('block_signed_at').sort_index()

azuki_tx_df.head()
```

[8]:
```
                       to_address_label          fees_paid   value_quote
block_signed_at
2022-05-02T17:19:43Z  LooksRare: Exchange  19172817560393340  84346.297729
2022-05-02T17:20:39Z                 None   7005576883041388      0.000000
2022-05-02T17:23:13Z                 None  16639408965196144      0.000000
2022-05-02T17:24:03Z  LooksRare: Exchange  26602204157466837  85775.895996
2022-05-02T17:27:41Z                 None   2184930000000000      0.000000
```

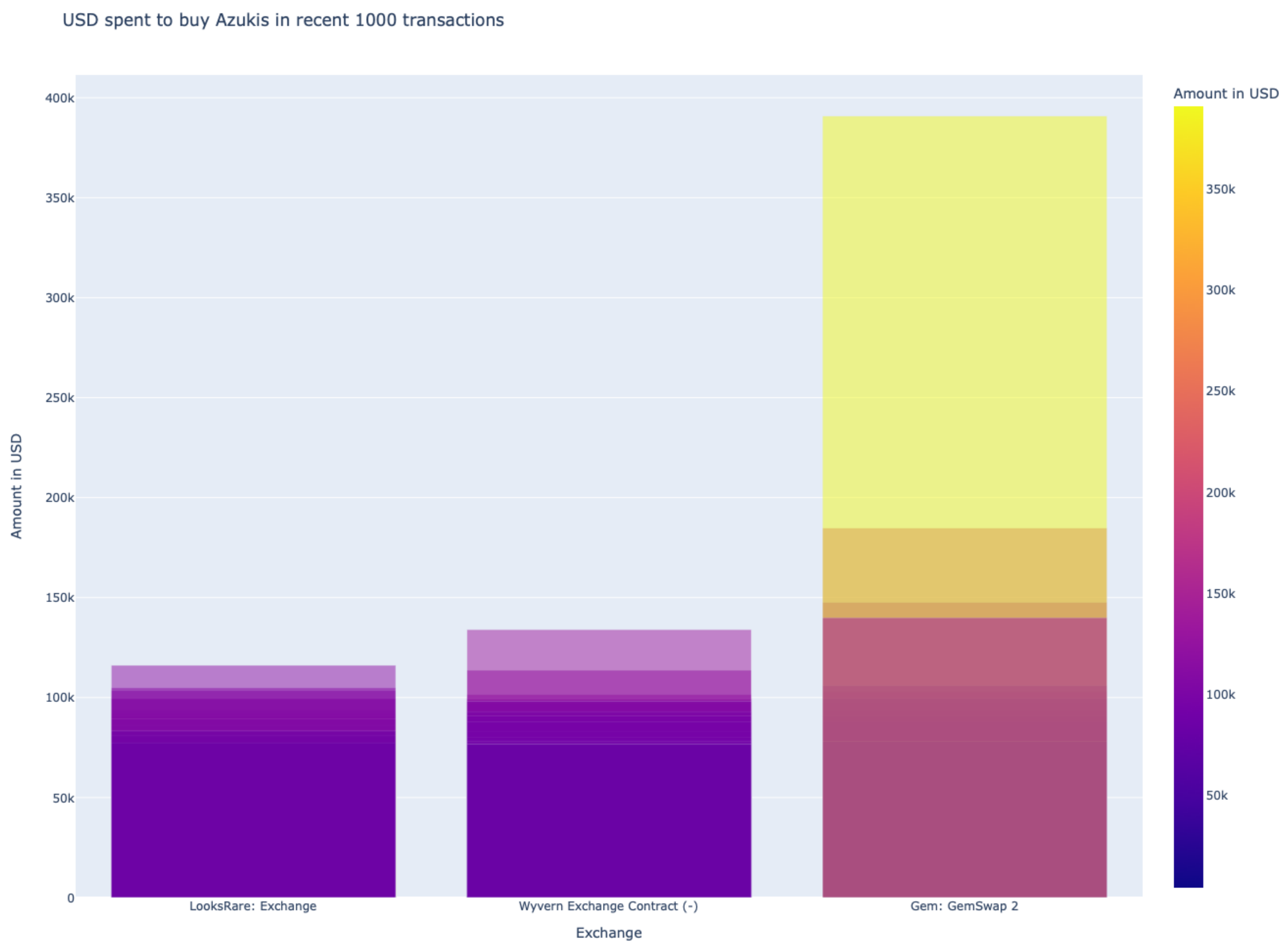
0.6 1.a Azuki Historical Sales

[41]:
```
# Filter Through data for non null transactions
azuki_sales_df = azuki_tx_df[azuki_tx_df['value_quote'] != 0]
azuki_sales = azuki_sales_df[azuki_sales_df['to_address_label'].notnull()]

# Creating the plot using plotly express
azuki_fig = px.bar(azuki_sales,
                   x='to_address_label',
                   y= 'value_quote',
                   color='value_quote',
                   height=1020,
                   width = 1000,
```

4. Data Analysis

Analyzing our data and developing figures to answer our questions

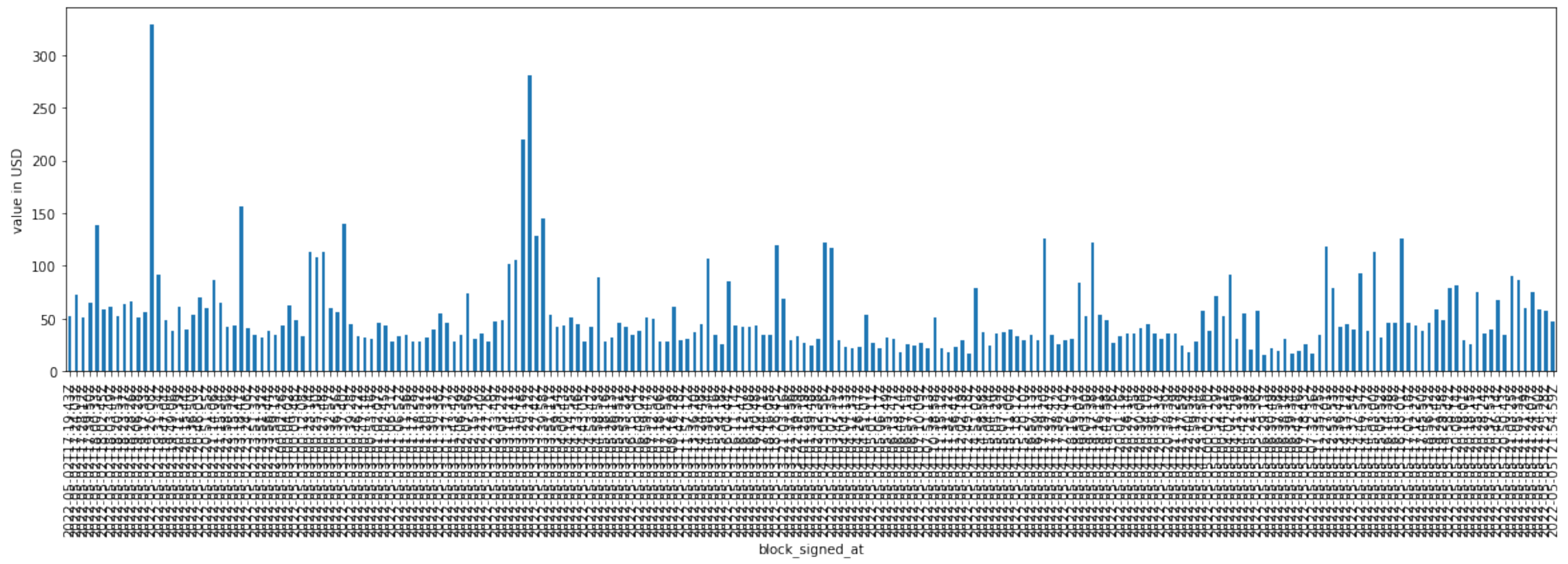
```
                    barmode='overlay',
                    labels={'value_quote':'Amount in USD', 'to_address_label':␣
↪'Exchange'},
                    title='USD spent to buy Azukis in recent 1000 transactions'
                   )
azuki_fig.show()
```



0.7 1.b Azuki transaction fees paid

```
[30]: # Filter Through data for non null transactions
      azuki_fees = azuki_sales_df['fees_paid'].astype(int)/10**18*eth_value

      azuki_fees.plot.bar(rot = 90, figsize = (20,5), ylabel = 'value in USD')
```

```
[30]: <AxesSubplot:xlabel='block_signed_at', ylabel='value in USD'>
```



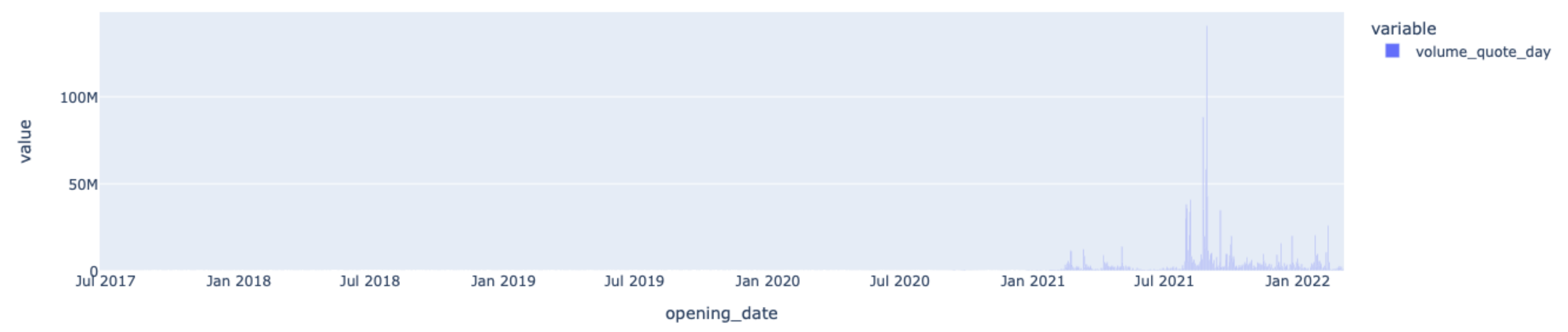
0.8 2. Cryptopunks Daily Volume

```
[13]: # Create variables needed for owner data and append to url
      cryptopunks_historical_url = url + chain_id + "/nft_market/collection" +␣
       ↪cryptopunks_address + api_no_option

      # Get request
      cryptopunks_historical_json = requests.get(cryptopunks_historical_url).json()

      # Convert historical json data to a dataframe and view data
      cryptopunks_df = pd.DataFrame(cryptopunks_historical_json['data']['items'])

      # Set index to date
      cryptopunks_df = cryptopunks_df.set_index('opening_date')

      # Create Volume dataframe
      cryptopunks_vol_df = pd.DataFrame(cryptopunks_df, columns =␣
       ↪['volume_quote_day', 'unique_token_ids_sold_count_day']).sort_index()
      cryptopunks_vol_df.head()
```

```
[13]:               volume_quote_day  unique_token_ids_sold_count_day
      opening_date
      2017-06-23                 0.0                               19
      2017-06-24                 0.0                               22
      2017-06-25                 0.0                               11
      2017-06-26                 0.0                               18
      2017-06-27                 0.0                               35
```

```
[42]: # Plot Volume quote per day
      cryptopunks_volume = cryptopunks_vol_df['volume_quote_day'].astype(int)

      # cryptopunks_volume.plot.line(figsize = (20,4))
```

```
px.bar(cryptopunks_volume)
```



0.9 2a Cryptopunks Historical transactions

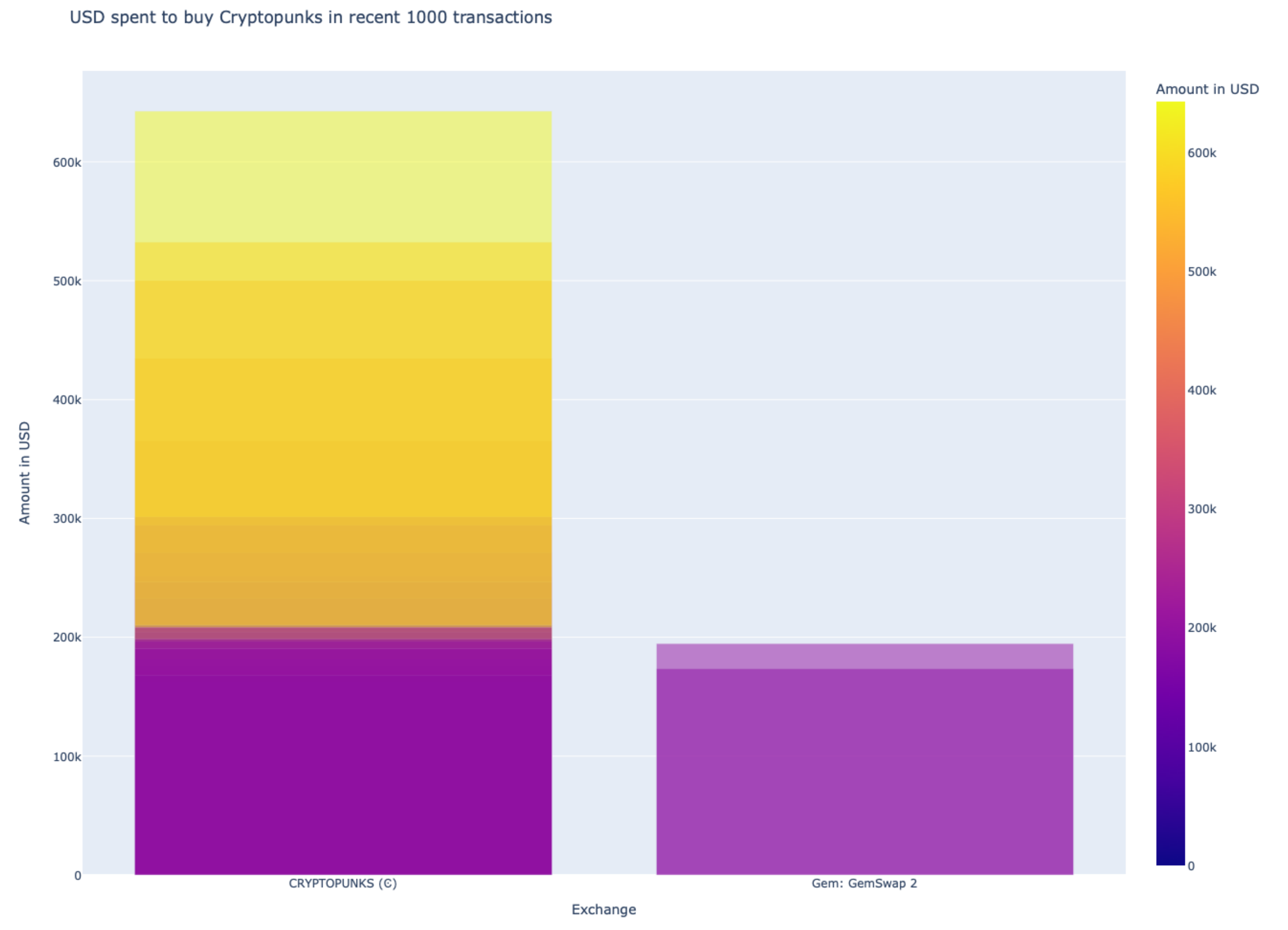
```
[15]: # Quering the API for transaction data
cryptopunks_tx_url = url + chain_id + "/address" + cryptopunks_address +␣
 ↪page_option + api_option
cryptopunks_tx = requests.get(cryptopunks_tx_url).json()

# Convert transactions data to dataframe
cryptopunks_tx_df = pd.DataFrame(cryptopunks_tx['data']['items'], columns =␣
 ↪['to_address_label','fees_paid', 'value_quote','block_signed_at']).
 ↪set_index('block_signed_at').sort_index()

cryptopunks_tx_df.head()
```

```
[15]:                      to_address_label          fees_paid  value_quote
block_signed_at
2022-04-27T17:02:10Z  CRYPTOPUNKS ( )   9248158760553474          0.0
2022-04-27T17:02:42Z  CRYPTOPUNKS ( )   9375445908544005          0.0
2022-04-27T17:03:33Z              None  46953125223110119          0.0
2022-04-27T17:05:29Z              None  48539446229017550          0.0
2022-04-27T17:20:09Z  CRYPTOPUNKS ( )   6381749930928020          0.0
```

0.10 2.a Cryptopunks Historical Sales

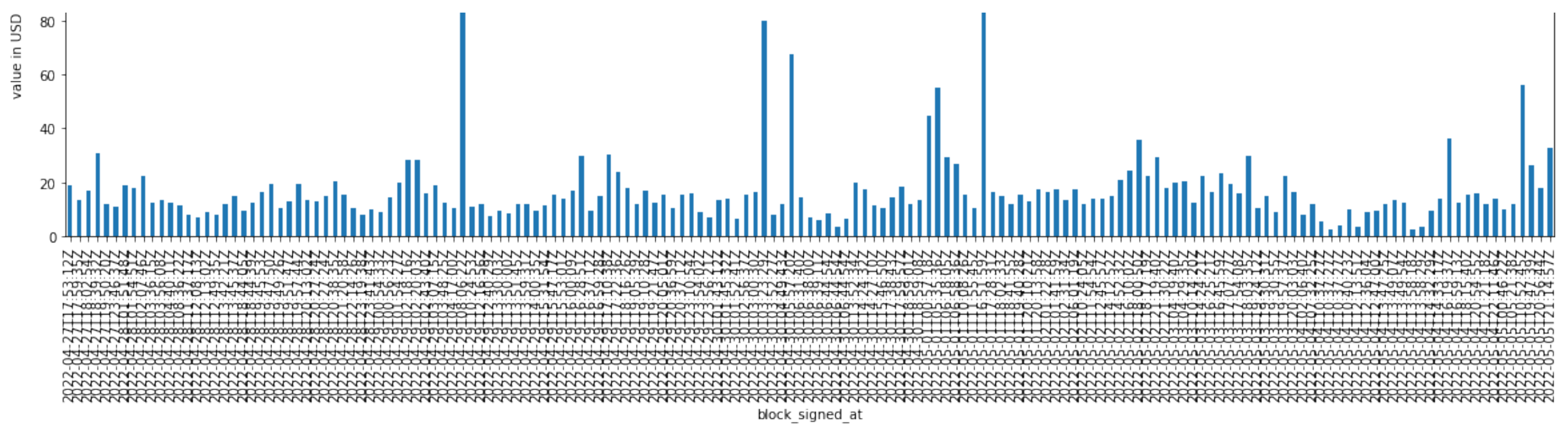
```
[36]: # Filter Through data for non null transactions
cryptopunks_sales_df = cryptopunks_tx_df[cryptopunks_tx_df['value_quote'] != 0]
cryptopunks_sales =␣
 ↪cryptopunks_sales_df[cryptopunks_sales_df['to_address_label'].notnull()].
 ↪dropna()


# Creating the plot using plotly express
```

```
cryptopunks_fig = px.bar(cryptopunks_sales,
                           x='to_address_label',
                           y= 'value_quote',
                           color='value_quote',
                           height=1020,
                           width = 1000,
                           barmode = 'overlay',
                           labels={'value_quote':'Amount in USD',␣
 ↪'to_address_label': 'Exchange'},
                           title='USD spent to buy Cryptopunks in recent 1000␣
 ↪transactions'
                          )
cryptopunks_fig.show()
```



0.11 2.b Cryptopunks Fees paid

```
[17]: # Filter Through data for non null transactions
cryptopunks_fees = cryptopunks_sales_df['fees_paid'].astype(int)/
 ↪10**18*eth_value
```

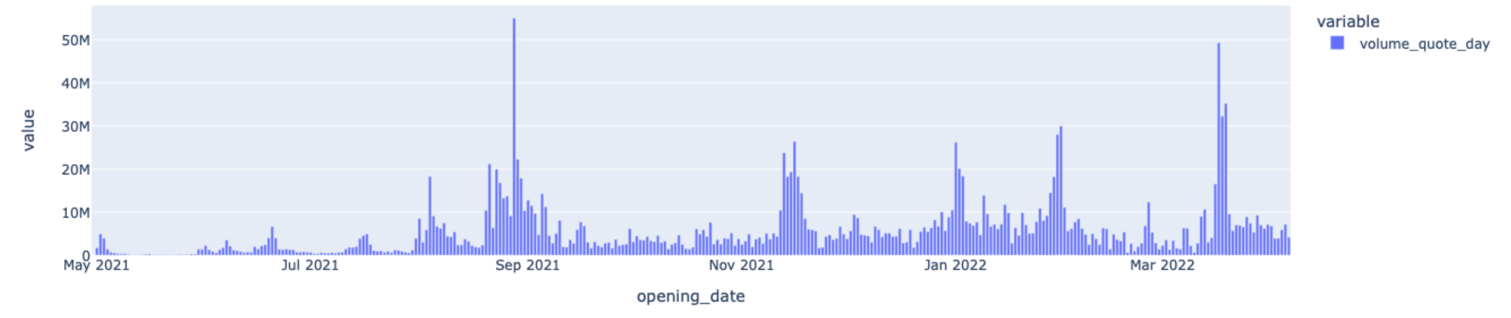


0.12 3. BAYC Daily Volume

```
[37]: # Create variables needed for owner data and add to url
      BAYC_historical_url = url + chain_id + "/nft_market/collection" + BAYC_address␣
       ↪+ api_no_option

      # Get request
      BAYC_historical_json = requests.get(BAYC_historical_url).json()

      # Convert historical json data to a dataframe and view data
      BAYC_df = pd.DataFrame(BAYC_historical_json['data']['items'])

      # Set index to date
      BAYC_df = BAYC_df.set_index('opening_date')

      # Create Volume dataframe
      BAYC_vol_df = pd.DataFrame(BAYC_df, columns = ['volume_quote_day',␣
       ↪'unique_token_ids_sold_count_day']).sort_index()
      BAYC_vol_df.head()
```

```
[37]:              volume_quote_day  unique_token_ids_sold_count_day
      opening_date
```

```
[43]: # Plot Volume quote per day
      BAYC_volume = BAYC_vol_df['volume_quote_day'].astype(int)

      # BAYC_volume.plot.bar(figsize = (20,4))
      px.bar(BAYC_volume)
```

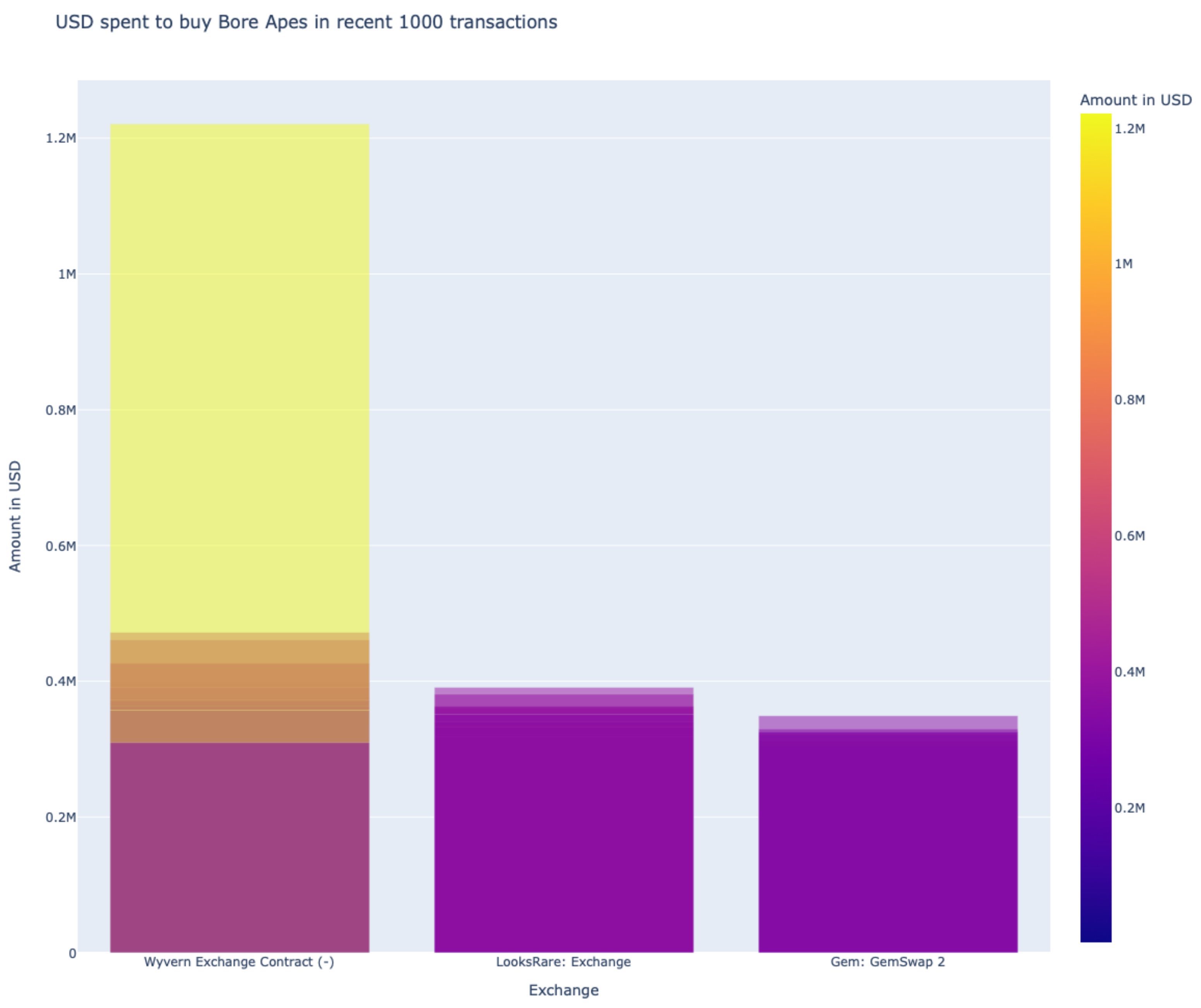


0.13 3a BAYC Historical Sales

```
[20]: # Quering the API for transaction data
      BAYC_tx_url = url + chain_id + "/address" + BAYC_address + page_option +␣
       ↪api_option
      BAYC_tx = requests.get(BAYC_tx_url).json()

      # Convert transactions data to dataframe
      BAYC_tx_df = pd.DataFrame(BAYC_tx['data']['items'], columns =␣
       ↪['to_address_label','fees_paid', 'value_quote','block_signed_at']).
       ↪set_index('block_signed_at').sort_index()

      BAYC_tx_df.head()
```

```
[20]:                               to_address_label          fees_paid  \
      block_signed_at
      2022-05-02T19:17:57Z                            None   5514685994543960
      2022-05-02T19:22:45Z                            None  10393902970606424
      2022-05-02T19:22:45Z                            None   9087780139176796
      2022-05-02T19:23:32Z  Wyvern Exchange Contract (-)  25104762520975735
      2022-05-02T19:30:41Z                            None   3462010508841040

                              value_quote
      block_signed_at
      2022-05-02T19:17:57Z       0.000000
      2022-05-02T19:22:45Z       0.000000
      2022-05-02T19:22:45Z       0.000000
      2022-05-02T19:23:32Z  328807.601318
      2022-05-02T19:30:41Z       0.000000
```

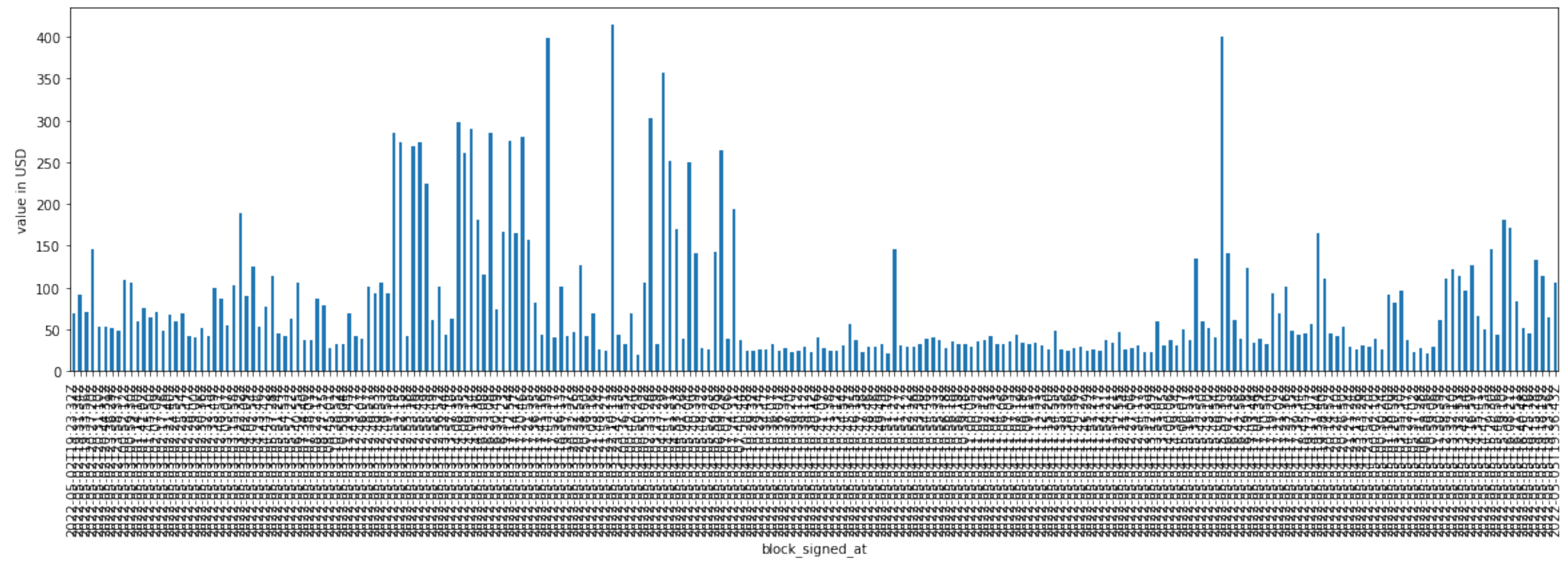
[21]:
```
# Filter Through data for non null transactions
BAYC_sales_df = BAYC_tx_df[BAYC_tx_df['value_quote'] != 0]
BAYC_sales = BAYC_sales_df[BAYC_sales_df['to_address_label'].notnull()].dropna()

# Creating the plot using plotly express
BAYC_fig = px.bar(BAYC_sales,
             x='to_address_label',
             y= 'value_quote',
             color='value_quote',
             height=1020,
             width = 1000,
                barmode = 'overlay',
                labels={'value_quote':'Amount in USD', 'to_address_label':␣
 ↪'Exchange'},
             title='USD spent to buy Bore Apes in recent 1000 transactions'
            )
BAYC_fig.show()
```



0.14 3.b BAYC Fees paid

[22]:
```
# Filter Through data for non null transactions
BAYC_fees = BAYC_sales_df['fees_paid'].astype(int)/10**18*eth_value

BAYC_fees.plot.bar(rot = 90, figsize = (20,5), ylabel = 'value in USD')
```

```
[22]: <AxesSubplot:xlabel='block_signed_at', ylabel='value in USD'>
```

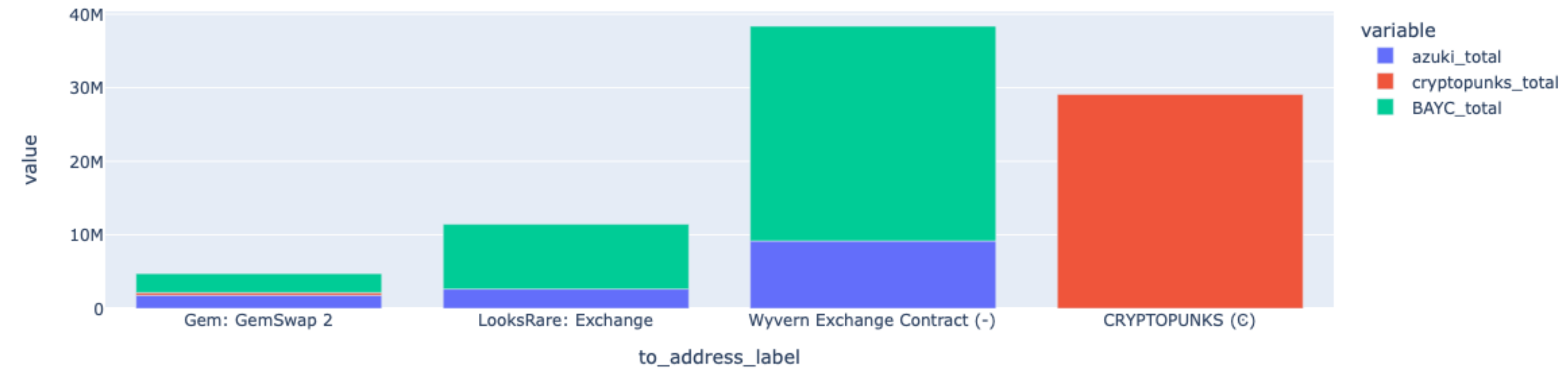


0.15 Combine Total Sales

[23]:
```
# Group by address label and sum the value
azuki_total = azuki_sales.groupby('to_address_label').sum()
cryptopunks_total = cryptopunks_sales.groupby('to_address_label').sum()
BAYC_total = BAYC_sales.groupby('to_address_label').sum()
```

[24]:
```
# Combine and rename columns for our total sales data
combined_totals = pd.concat([azuki_total,cryptopunks_total,BAYC_total], axis=1)
combined_totals.columns = ['azuki_total', 'cryptopunks_total','BAYC_total']
```

[25]:
```
# Plot for combined figure
combined_total_fig = px.bar(combined_totals)


# Show Figure
combined_total_fig.show()
```



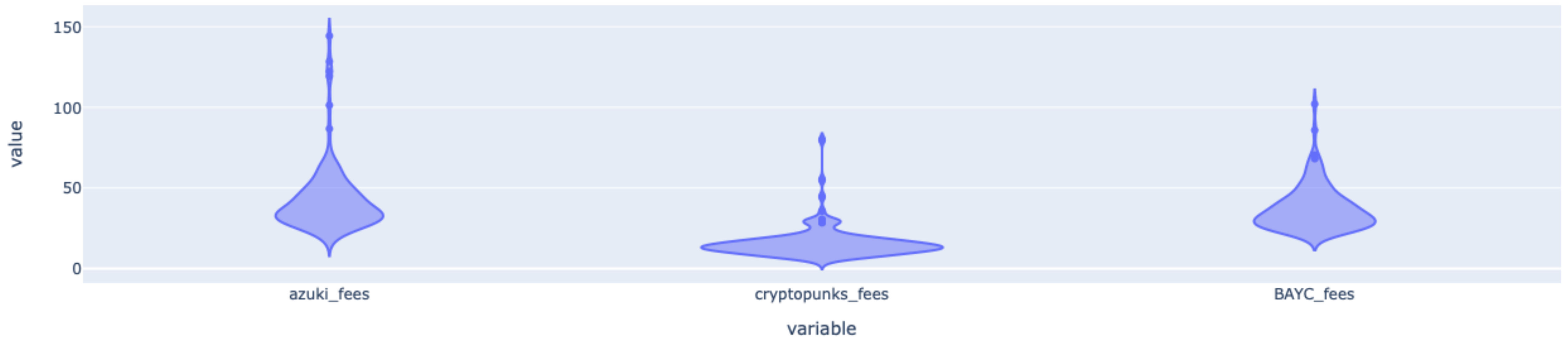
0.16 Combine Total Fees

[26]:
```
# Group by address label and sum the value
combined_totals
```

[26]:
```
                             azuki_total  cryptopunks_total    BAYC_total
to_address_label
Gem: GemSwap 2              1.791827e+06       3.683260e+05  2.568193e+06
LooksRare: Exchange         2.623563e+06                NaN  8.814426e+06
Wyvern Exchange Contract (-) 9.173662e+06               NaN  2.918673e+07
CRYPTOPUNKS ( )                      NaN       2.909927e+07           NaN
```

[38]:
```
# Combine and rename columns for our total sales data
azuki_usd_fees = azuki_sales['fees_paid'].astype(int)/10**18*eth_value
cryptopunks_usd_fees = cryptopunks_sales['fees_paid'].astype(int)/
 ↪10**18*eth_value
BAYC_usd_fees = BAYC_sales['fees_paid'].astype(int)/10**18*eth_value

# Combine dataframe and drop nulls
combined_usd_fees = pd.concat([azuki_usd_fees.reset_index(drop=True),
                               cryptopunks_usd_fees.reset_index(drop=True),
                               BAYC_usd_fees.reset_index(drop=True)],
                              axis=1
                             ).dropna()
combined_usd_fees.columns = ['azuki_fees', 'cryptopunks_fees','BAYC_fees']
```

[39]:
```
# Plot for combined figure
combined_fees_fig = px.violin(combined_usd_fees)

# Show Figure
combined_fees_fig.show()
```



5. Discussion

Combining our data and discussing our findings

Postmoterm

Did we find everything we expected to find?

- Our difficulties and how we dealt with them
- Additional questions that came up that we didn’t answer
- What would we research next if we had more time?

8. Questions

Open floor Q&A with the audience

thank you,