Odd Semester (2022)

BINUS UNIVERSITY INTERNATIONAL

Algorithm and Programming Final Project

Cryptocurrency Dashboard

Student Information:

Surname: Hartono	**Given Name:** Lenno Aubert	**Student ID:** 2602116983

Course Code : COMP6047001	**Course Name :** Algorithm and Programming
Class : **L1AC**	**Lecturer:** Jude Joseph Lamug Martinez, MCS

Type of Assignment: Final Project Report

Submission Pattern

Due Date: January 15th, 2023 **Submission Date:** January 15th, 2023

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer's instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

Plagiarism/Cheating

Binus International seriously regards all forms of plagiarism, cheating, and collusion as academic offences which may result in severe penalties, including loss/drop of marks, course/class discontinuity, and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

Declaration of Originality

By signing this assignment, I understand, accept, and consent to Binus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student:

Lenno Aubert Hartono

Table of Contents

Project Specification

1. **Background**

   Every semester 1 student was instructed to make a Python program using the concept of Object-Oriented Programming or OOP. We need to make a program that is beyond what has been taught in class. So, I made a Cryptocurrency Dashboard that I deployed on a website called Streamlit.

2. **Problem Analysis**

   The rise of cryptocurrency has been one of the most significant technological and economic developments of the 21st century. With the creation of Bitcoin in 2009, a new form of decentralized digital currency was introduced that operates independently of a central bank or government. Since then, the number of cryptocurrencies in circulation has grown exponentially, with many different types and uses. In addition to being used as a form of currency, cryptocurrencies are also being used for a wide range of other applications such as smart contracts, decentralized applications, and more. The goal of this project is to create a comprehensive cryptocurrency dashboard that allows users to track, monitor, compare, and analyze the performance of different cryptocurrencies in real time. Also, I make a 1-month prediction for the selected cryptocurrency. To achieve this, I utilized various data sources and libraries to gather and display relevant information in an easy-to-use and visually appealing format. This dashboard provides valuable insights into the performance of different cryptocurrencies and has the potential to be a valuable tool for anyone interested in the cryptocurrency market.

3. **Libraries**

   To help me achieve a fully working cryptocurrency dashboard, I use some libraries for Python that are available on the internet. The libraries are:

   - **Streamlit**

     Streamlit is an open-source Python library that can create interactive web-based visualizations and applications in a simple and easy-to-use way. Streamlit is also highly customizable, providing a wide range of built-in components and widgets that can be used to create a unique and customized user interface. Additionally, I use Streamlit because it allows me to easily integrate with other popular Python libraries, for example, Plotly which I used for making the graph. Lastly, I am using Streamlit to easily deploy and share my dashboard with others.

   - **yfinance**

     The yfinance library is a popular Python library used for obtaining financial market data from the Yahoo Finance website. For this project, I use this library as part of my cryptocurrency dashboard to retrieve financial data such as historical pricing data, market capitalization, and other financial indicators.

This data can then be used to create visualizations and analyze the performance of the different cryptocurrencies.

- **Pandas**

  Pandas is a popular open-source Python library for data manipulation and analysis. In this project, I use Pandas for handling the financial data obtained from the yfinance library. The data is first retrieved using the yfinance library and then converted into a Pandas DataFrame. This allows the data to be easily manipulated and analyzed.

- **Plotly**

  Plotly is an open-source library for creating interactive data visualizations. In this project, I use plotly to make line charts, bar charts, and candlestick charts as it is highly customizable in terms of appearance, such as different chart colors and layouts. Also, it provides built-in support for web-based visualization and it's easy to integrate with other libraries like Streamlit in my case.

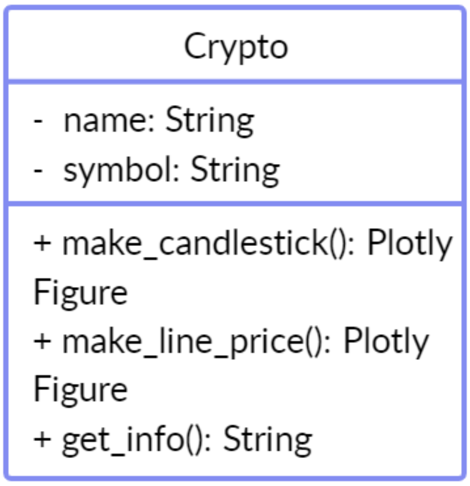
- **Prophet**

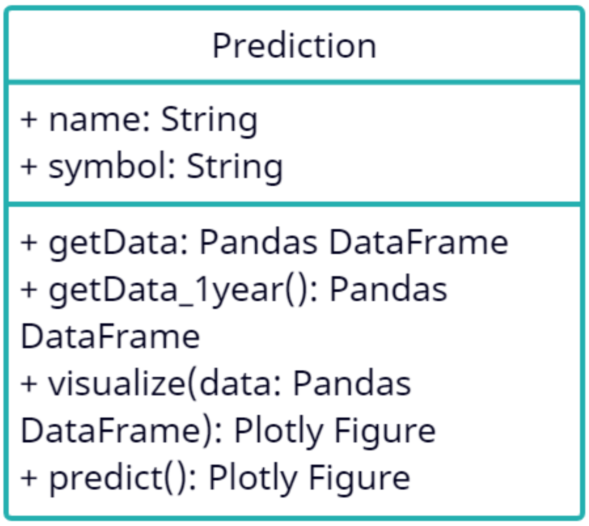
  Prophet is a library developed by Facebook that is used for time series forecasting. I use Prophet to forecast the cryptocurrency price because this library is specifically designed for time series forecasting. I choose Prophet over some other similar time series forecasting library because Prophet is easy to use even for me who does not have a deep understanding of time series forecasting.

  Also, it can make pretty accurate predictions because of its ability to handle missing data and outliers, as well as changing trends. Moreover, it uses a decomposable time series model that can handle trends, seasonality, and holidays, to produce more accurate predictions. Lastly, it can integrate with Plotly to create a line chart that shows the actual historical price data of the cryptocurrency along with the predicted future prices.

4. Classes

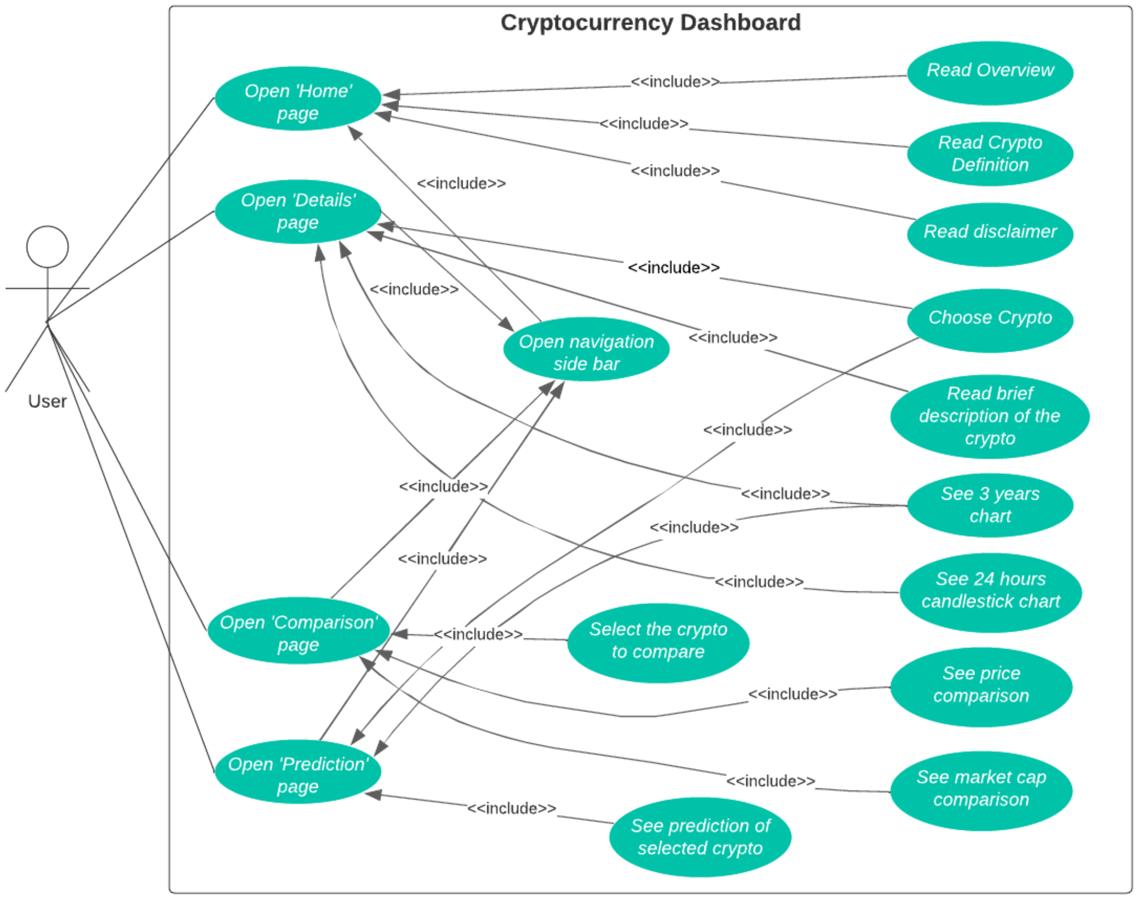
To make this program I use 2 classes located in 2 different files, which are 2_Details.py and 4_Prediction.py. To make it clearer, I make a class diagram.


Crypto
- name: String
- symbol: String
+ make_candlestick(): Plotly Figure
+ make_line_price(): Plotly Figure
+ get_info(): String


Prediction
+ name: String
+ symbol: String
+ getData: Pandas DataFrame
+ getData_1year(): Pandas DataFrame
+ visualize(data: Pandas DataFrame): Plotly Figure
+ predict(): Plotly Figure

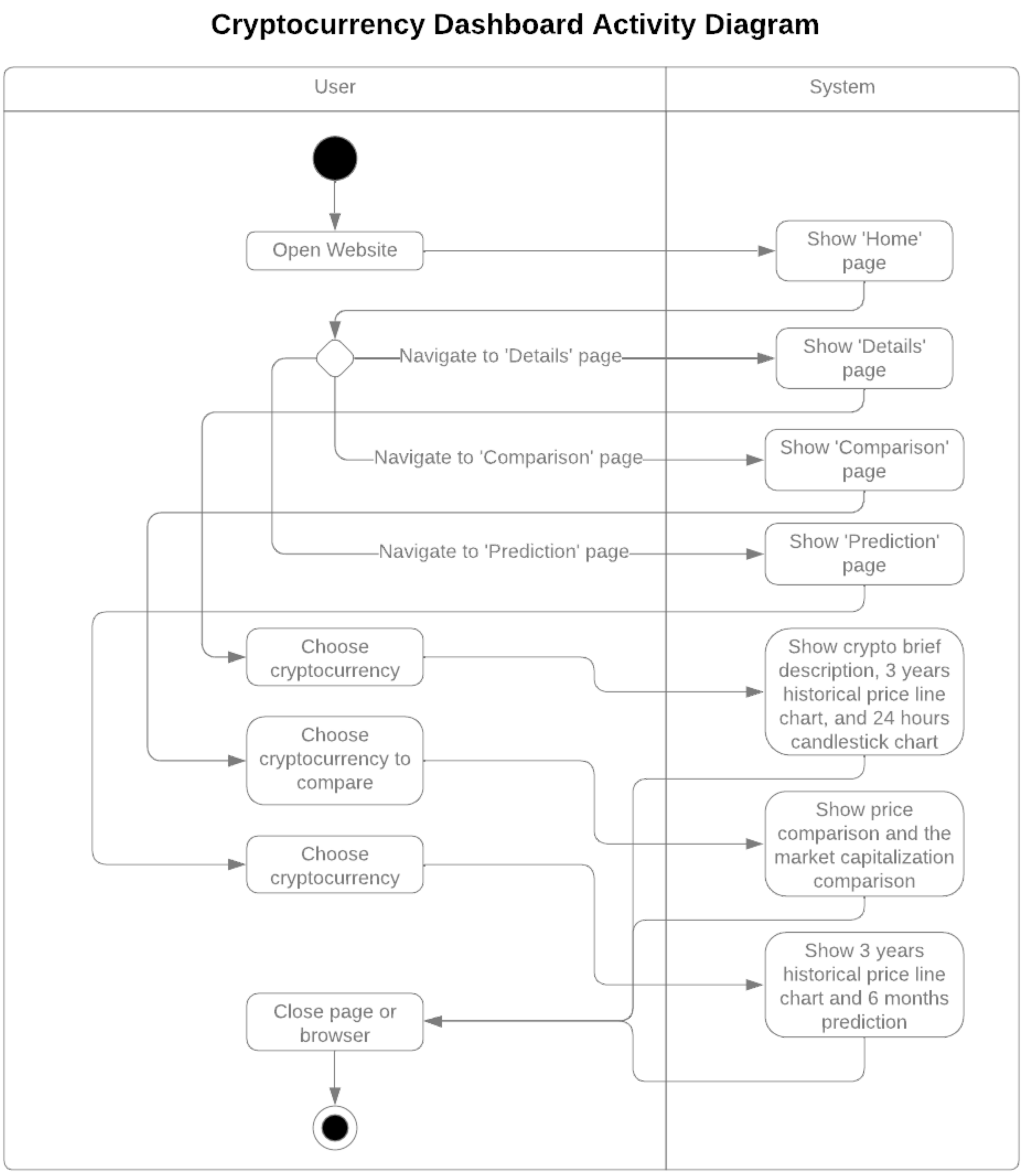
5. Use Case Diagram

To know what the user can do, I make a use case diagram



6. Activity Diagram

To understand the step-by-step process of this web app, I make an activity diagram


Cryptocurrency Dashboard Activity Diagram
User
System
Open Website
Show 'Home' page
Navigate to 'Details' page
Show 'Details' page
Navigate to 'Comparison' page
Show 'Comparison' page
Navigate to 'Prediction' page
Show 'Prediction' page
Choose cryptocurrency
Show crypto brief description, 3 years historical price line chart, and 24 hours candlestick chart
Choose cryptocurrency to compare
Show price comparison and the market capitalization comparison
Choose cryptocurrency
Show 3 years historical price line chart and 6 months prediction
Close page or browser

Essential Algorithm

In this part of the report, I will explain each section of the code that build my cryptocurrency dashboard. Starting from the file structure. There are 5 files and 1 folder named ‘pages’ that also contains 3 files. From a total of 8 files, 4 of them are the codes that consist of:

a. 1_Home.py

This is the file that will be executed when you open the site. I divide this code into 4 sections. Below is the explanation of each section:

- *Importing module*

```
import streamlit as st
```

In the first section, I am importing the *Streamlit* module. In this Python file, I only upload Streamlit library because this page is only containing the information about the web application.

- *Setting the page configuration*

```
st.set_page_config(
    page_title="Home",
    page_icon="🏠",
    initial_sidebar_state="expanded",
    layout='centered'
)
```

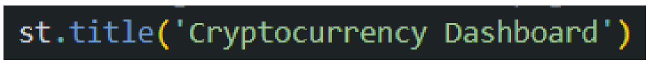
The second section uses the ‘*st.set_page_config()*’ method to set the page configuration. This includes the title of the page, which is set to "Home", an icon for the page, which is set to a house symbol "🏠", the initial state of the sidebar, which is set to "*expanded*", and the layout of the page, which is set to "*centered*".

- *Adding the information for data source in the sidebar*

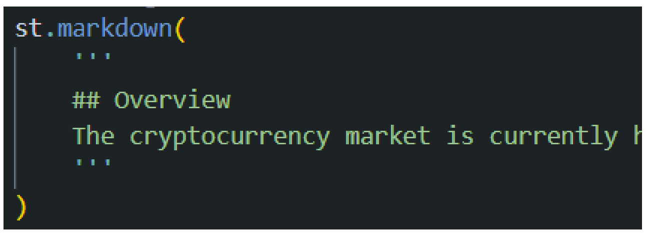
```
sources = st.sidebar.expander('Data source')
sources.write('''[Yahoo Finance](https://finance.yahoo.com)''')
```

The third section creates a sidebar using the ‘*st.sidebar.expander()*’ method. The ‘*expander()*’ method creates a section that can be expanded or collapsed by the user. When expanded, there will be a link directed to ‘https://finance.yahoo.com’.

- *Printing the information*



```
st.title('Cryptocurrency Dashboard')
```



```
st.markdown(
    '''
    ## Overview
    The cryptocurrency market is currently h
    '''
)
```

In this section, I am printing the title of the page using ‘*st.title()*’ function, also I am using ‘*st.markdown()*’ function to print the ‘Overview’, ‘Definition’, and ‘Disclaimer’.

b. 2_Details.py

This is the file that I put inside the ‘pages’ folder. I put it inside the ‘pages’ folder because *Streamlit* needed it to make a multipage web application. In this file, I divide the code into 5 sections. Below is the explanation for each section:

- Importing the libraries

  The code imports the ‘*Streamlit*’ library for creating the app, the ‘*yfinance*’ library for getting financial data, and ‘*plotly.graph_objs*’ and ‘*plotly.express*’ for creating plots.

```
import streamlit as st
import plotly.graph_objs as go
import yfinance as yf
import plotly.express as px
```

- *Setting page configuration*

  The ‘*st.set_page_config*’ function is used to set the title, icon, and initial state of the sidebar for the app.

```
st.set_page_config(
    page_title="Details",
    page_icon="🔍",
    initial_sidebar_state="expanded"
)
```

- *Sidebar configuration*

  Same as in the ‘*1_Home.py*’ file, the sidebar contains the information about the data source for the visualization.

```
sources = st.sidebar.expander('Data source')
sources.write('''[Yahoo Finance](https://finance.yahoo.com)''')
```

- *Making the class*

```
class Crypto:
```

  The code defines a class called Crypto which has the following methods:

  - **__init__**:
    - o The constructor of the class that takes two parameters (name and symbol) and assigns them to the instance variables ‘*self.__name*’ and ‘*self.__symbol*’. It also constructs the ticker

variable by concatenating the symbol and "-USD". Then it uses the '*yfinance*' library to get the historical data for 24 hours with 5-minute intervals and 3 years with 1-day intervals.

```
def __init__(self, name, symbol):
    self.__name = name
    self.__symbol = symbol
    self.ticker = symbol + "-USD"
    self.initialize = yf.Ticker(self.ticker)
    self.data_1d = self.initialize.history(period="1d",
interval="5m")
    self.data_3y = self.initialize.history(period="3y",
interval="1d")
```

- **make_candlestick**:
    - This function creates a candlestick chart of the crypto currency's historical data for the last 24 hours using the plotly.graph_objs library. It also updates the chart's title, y-axis label, and adds a slider below the graph to change the range of the data shown.

```
def make_candlestick(self):
    fig = go.Figure()

    fig.add_trace(go.Candlestick(x=self.data_1d.index,
        open=self.data_1d['Open'],
        high=self.data_1d['High'],
        low=self.data_1d['Low'],
        close=self.data_1d['Close'], name = 'market data'))

    fig.update_layout(
        title=self.__name + ' 24 Hours Candlestick Chart',
        yaxis_title=self.__name + ' Price (in US Dollars)'
        )

    fig.update_xaxes(
        rangeslider_visible=True,
        rangeselector=dict(
            buttons=list([
                dict(count=15, label="15m", step="minute",
stepmode="backward"),
                dict(count=45, label="45m", step="minute",
stepmode="backward"),
                dict(count=1, label="HTD", step="hour",
stepmode="todate"),
                dict(count=6, label="6h", step="hour",
stepmode="backward"),
                dict(step="all")
            ])
        )
    )
    cs = fig.data[0]

    cs.increasing.fillcolor = '#3D9970'
    cs.increasing.line.color = '#3D9970'
    cs.decreasing.fillcolor = '#FF4136'
    cs.decreasing.line.color = '#FF4136'

    return fig
```

- **make_line_price**:
  - This function creates a line chart of the crypto currency's historical data for the last 3 years using the '*plotly.graph_objs*' library. It also updates the chart's title, y-axis label, and adds a slider below the graph to change the range of the data shown.

```
def make_line_price(self):
    df = self.data_3y.reset_index()
    for i in ['Open', 'High', 'Close', 'Low']:
        df[i] = df[i].astype('float64')

    # Making the line chart
    fig = go.Figure([go.Scatter(x=df['Date'],
y=df['High'])])

    fig.update_layout(
        title=self.__name + ' 3 Years Chart',
        yaxis_title=self.__name + ' Price (in US Dollars)'
        )

    fig.update_xaxes(
        rangeslider_visible=True,
        rangeselector=dict(
            buttons=list([
                dict(count=1, label="1mo", step="month",
stepmode="backward"),
                dict(count=6, label="6mo", step="month",
stepmode="backward"),
                dict(count=1, label="YTD", step="year",
stepmode="todate"),
                dict(count=1, label="1y", step="year",
stepmode="backward"),
                dict(step="all")
            ])
        )
    )

    return fig
```

- *Printing on Streamlit page*

  After making the class, I print everything on the *Streamlit* such as listing the names of the cryptocurrency.

```
crypto_name = ["Bitcoin (BTC)", "Ethereum (ETH)", "XRP (XRP)",
"Tether (USDT)", "Dogecoin (DOGE)", "Cardano (ADA)", "Polygon
(MATIC)", "Binance Coin (BNB)", "USD Coin (USDC)", "Binance USD
(BUSD)"]
```

  After listing the names, I make a selectbox to select the cryptocurrency.

```
selected = st.selectbox("Cryptocurrency", crypto_name)
```

  To use the class, I split the name and the symbol of the selected cryptocurrency first with

```
words = selected.split(" ")
crypto_name = " ".join(words[:-1])
crypto_symbol = words[-1].strip("()")
```

  Then, I execute the functions from the class to make the charts

```
crypto_details = Crypto(crypto_name, crypto_symbol)
crypto_candlestick = crypto_details.make_candlestick()
crypto_info = crypto_details.get_info()
crypto_line = crypto_details.make_line_price()
```

  And finally, print everything to *Streamlit*

```
# making the crypto info section
st.subheader(f"About {crypto_name} ({crypto_symbol})")
st.info(f"{crypto_info}")

# printing the line and candlestick chart to the streamlit
st.plotly_chart(crypto_line)
st.plotly_chart(crypto_candlestick)
```

c. 3_Comparison.py

This is another file from the ‘pages’ folder for the ‘Comparison’ page. In this file I make a price and market capitalization comparison, then visualize it in charts. I divide this file of Python code into 6 sections. Below are the explanations for each section:

- *Importing* libraries

  This code imports ‘*Streamlit*’, ‘*yfinance*’, ‘*plotly.express*’, and ‘*pandas*’ libraries.

```
import streamlit as st
import yfinance as yf
import pandas as pd
import plotly.express as px
```

- *Setting the page configuration and sidebar*

  Same as the previous two files, this file contains the code to configure the page and sidebar.

```
# Setting the page configuration
st.set_page_config(
    page_title="Comparison",
    page_icon="💡",
    initial_sidebar_state="expanded"
)

# Making the sidebar
sources = st.sidebar.expander('Data source')
sources.write('''[Yahoo Finance](https://finance.yahoo.com)''')
```

- *Printing on Streamlit page*

  For this file, I do not use the class to write the functions. First, I print the title for the page.

```
st.title("Cryptocurrencies Comparison")
```

The code uses the '*st.title*' function to set the title of the app as "Cryptocurrencies Comparison".

Then, I make a list that contain the list of cryptocurrency names and symbols

```
crypto_name = ["Bitcoin (BTC)", "Ethereum (ETH)", "XRP (XRP)",
"Tether (USDT)", "Dogecoin (DOGE)", "Cardano (ADA)", "Polygon
(MATIC)", "Binance Coin (BNB)", "USD Coin (USDC)", "Binance USD
(BUSD)"]
```

After that, I split the name and the symbol of each cryptocurrency and store it in 2 different lists. This makes me easier when it comes to processing the data

```
for crypto in crypto_name:
    name, symbol = crypto.split("(")
    names.append(name.strip())
    symbols.append(symbol.strip().replace(")", ""))
```

Then, I make the '*st.multiselect*' function to create a multi-select widget where the user can select the cryptocurrencies they want to compare. The default selection is the first two cryptocurrencies in the symbols list, which is Bitcoin and Ethereum.

```
selected = st.multiselect("Choose the cryptocurrencies you want
to compare", symbols, [symbols[0], symbols[1]])
```

- *Processing the data*

  In this section I will divide it into three parts. Below is the explanation of each part:

  - Initializing empty list to store the DataFrame and Market Capitalization of the selected cryptocurrencies.

    ```
    dataframe_list = []
    marketcap_list = []
    ```

    The code creates two empty lists called ‘*dataframe_list*’ and ‘*marketcap_list*’ that will be used to store the DataFrame and Market Capitalization of the selected cryptocurrencies.

  - Using for loop to get the DataFrame and MarketCap.

    ```
    for crypto in selected:
        ticker = yf.Ticker(crypto + '-USD')
        hist = ticker.history(period='5y', interval='1d')
        info = ticker.info
        marketcap_list.append(info['marketCap'])
        hist['Name'] = crypto
        dataframe_list.append(hist)
    ```

    The code iterates over the selected list (the list of selected cryptocurrencies), gets the historical data for the last 5 years for each crypto, retrieves the market capitalization, creates a new column for the crypto name, and appends the DataFrame and MarketCap to the corresponding lists.

  - Combining the DataFrame of selected crypto to be compared.

    ```
    combined = pd.concat(dataframe_list)
    combined = combined.reset_index()
    ```

    The code uses the ‘*pd.concat()*’ function to combine the DataFrames of the selected cryptocurrencies into one DataFrame called ‘combined’, then I use the ‘*reset_index()*’ to reset the index. I reset the index because the default DataFrame index when retrieved from yfinance library is the ‘Date’. Using ‘*reset_index()*’ function will make it easier for me to visualize the data because the DataFrame will now have the default index which is 0, 1, 2, etc. and will make a new column for the ‘Date’.

- *Making the chart*

  First, I make the line chart for the price comparison

```
fig = px.line(
    data_frame=combined,
    x="Date",
    y="Close",
    color="Name",
    labels={
        "Close": "Closing Price (in USD)"
    }
)
```

  Then, I make the bar chart for the Market Capitalization comparison

```
marketcap_df =
pd.DataFrame({'Name':selected,'Marketcap':marketcap_list})
marketcap_fig = px.bar(marketcap_df, 'Name', 'Marketcap')
```

- *Continue printing on Streamlit page*

```
st.subheader("Price Comparison")
st.plotly_chart(fig)
st.subheader("Marketcap Comparison")
st.plotly_chart(marketcap_fig)
```

  The code uses the '*st.subheader*' and '*st.plotly_chart*' functions to display the "Price Comparison"

d. 4_Prediction.py

This is another file from the 'pages' folder. In this file, I make a price prediction for the future. I divide this code into 5 sections. Below is the explanation for each section:

- *Importing modules*

  The code imports the necessary modules for the app, including the '*streamlit*', '*yfinance*', '*plotly.express*', '*pandas*', '*plotly.graph_objs*' and '*prophet*' libraries for creating the app, getting financial data, manipulating data and creating plots and time series forecasting.

```
import streamlit as st
import yfinance as yf
import plotly.express as px
import pandas as pd
import plotly.graph_objs as go
from prophet import Prophet
from prophet.plot import plot_plotly
```

- *Setting the page configuration and side bar*

  Same as the previous two files, this file contains the code to configure the page and sidebar.

```
# Setting the page configuration
st.set_page_config(
    page_title="Prediction",
    page_icon="🔮",
    initial_sidebar_state="expanded",
)

# Making the sidebar
sources = st.sidebar.expander('Data source')
sources.write('''[Yahoo Finance](https://finance.yahoo.com)''')
```

- *Defining class name*

  I define a class called 'Prediction' that takes in name and symbol of the crypto when it is initialized.

```
class Prediction:
    def __init__(self, name, symbol):
        self.name = name
        self.symbol = symbol
```

- *Make a function to get 3 years data of the cryptocurrency*

  This function is used to retrieve historical data of the selected cryptocurrency. It does this by first creating a ticker symbol by concatenating the symbol of the cryptocurrency with "-USD", which is the ticker format used by Yahoo Finance.

```
def getData(self):
    ticker = self.symbol + "-USD"
```

  It then uses this ticker symbol to create an instance of the yfinance Ticker class, passing in the ticker symbol as an argument, and the '*history()*' function is then called on this instance, with the period set to '3y' (3 years) and the interval set to '1d' (daily) to retrieve historical data for the cryptocurrency for the last 3 years with daily intervals.

```
initialize = yf.Ticker(ticker)
data = initialize.history(period='3y', interval='1d')
```

Then, the '*reset_index()*' function is called on the data to reset the index of the DataFrame and make it easier when analyzing the data. Then the date column is converted to datetime format and only the date is retrieved. Finally, the cleaned dataframe is returned.

```
data.reset_index(inplace=True)
data['Date'] = pd.to_datetime(data['Date'])
data['Date'] = data['Date'].dt.date
return data
```

- *Make function for getting the 3 years for the prediction*

  This function is basically the same as the previous function, the only difference is that the code renames the 'Date' column to 'ds' and the 'Close' column to 'y' as these are the names that Prophet library uses for date column and target column respectively.

```
def getDataForPrediction(self):
    ticker = self.symbol + "-USD"
    initialize = yf.Ticker(ticker)
    data = initialize.history(period='3y', interval='1d')
    data.reset_index(inplace=True)
    data['Date'] = pd.to_datetime(data['Date'])
    data['Date'] = data['Date'].dt.date
    data = data.rename(columns={"Date": "ds", "Close": "y"})
    return data
```

- *Make function for making the comparison chart*

  This function takes in a DataFrame as an argument and creates a line chart visualization of the historical prices of the selected cryptocurrency.

```
def visualize(self,data):
```

It first converts the datatype of the 'Open', 'High', 'Close' and 'Low' columns to float64 using a for loop.

```
for i in ['Open', 'High', 'Close', 'Low']:
    data[i] = data[i].astype('float64')
```

It then creates an instance of the '*plotly*' Figure class, passing in a Scatter trace of the historical high prices of the cryptocurrency with the x-axis as the 'Date' column and the y-axis as the 'High' column.

```
fig = go.Figure([go.Scatter(x=data['Date'], y=data['High'])])
```

The layout of the chart is then updated by setting the title of the y-axis to 'Price (in US Dollars)'.

```
fig.update_layout(
    yaxis_title = 'Price (in US Dollars)'
)
```

It then updates the x-axis to show a range slider and range selector buttons with options to select time ranges of 1 month, 6 months, year to date, 1 year and all data.

```
fig.update_xaxes(
    rangeslider_visible=True,
    rangeselector=dict(
        buttons=list([
            dict(count=1, label="1mo", step="month", 
stepmode="backward"),
            dict(count=6, label="6mo", step="month", 
stepmode="backward"),
            dict(count=1, label="YTD", step="year", 
stepmode="todate"),
            dict(count=1, label="1y", step="year", 
stepmode="backward"),
            dict(step="all")
        ])
    )
)
```

Finally, it returns the figure instance, which can be rendered using the '*st.plotly_chart()*' function in the Streamlit app to display the chart.

- *Make the function for predicting the cryptocurrency prices*

This code defines a function named "predict" that takes one argument "data". Inside the function, it first creates a new instance of the Prophet model and fits it to the input "data".

```
def predict(self,data):
    model = Prophet()
    model.fit(data)
```

Then it creates a new dataframe named "future" which is used to predict the next 180 days (around 6 months) of data. The "predict" function is called on the model with the "future" DataFrame as an input, which generates a predicted DataFrame.

```
future = model.make_future_dataframe(periods=180)
prediction = model.predict(future)
```

Finally, it creates a '*plotly*' graph using '*plot_plotly*' function from the prophet library, the model and the predicted DataFrame, and the x-axis and y-axis labels are set. The graph is returned at the end of the function.

```
fig = plot_plotly(m=model, fcst=prediction, xlabel='Date',
ylabel='Closing Price (in USD)', figsize=(700,500))
return fig
```

- *Printing the title, listing the 10 cryptocurrencies, and making the selectbox for the cryptocurrencies in the Streamlit*

```
st.title('Prediction')

crypto_name = ["Bitcoin (BTC)", "Ethereum (ETH)", "XRP (XRP)",
"Tether (USDT)", "Dogecoin (DOGE)", "Cardano (ADA)", "Polygon
(MATIC)", "Binance Coin (BNB)", "USD Coin (USDC)", "Binance USD
(BUSD)"]

selected = st.selectbox(label='Select your cryptocurrency',
options=crypto_name)
```

- *Splitting the name and the symbol of the cryptocurrencies and initializing the class*

```
words = selected.split(" ")
crypto_name = " ".join(words[:-1])
crypto_symbol = words[-1].strip("()")

crypto_predict = Prediction(crypto_name, crypto_symbol)
```

- *Getting the historical data*

```
crypto_data_3y = crypto_predict.getData()
crypto_data_for_prediction =
crypto_predict.getDataForPrediction()
```

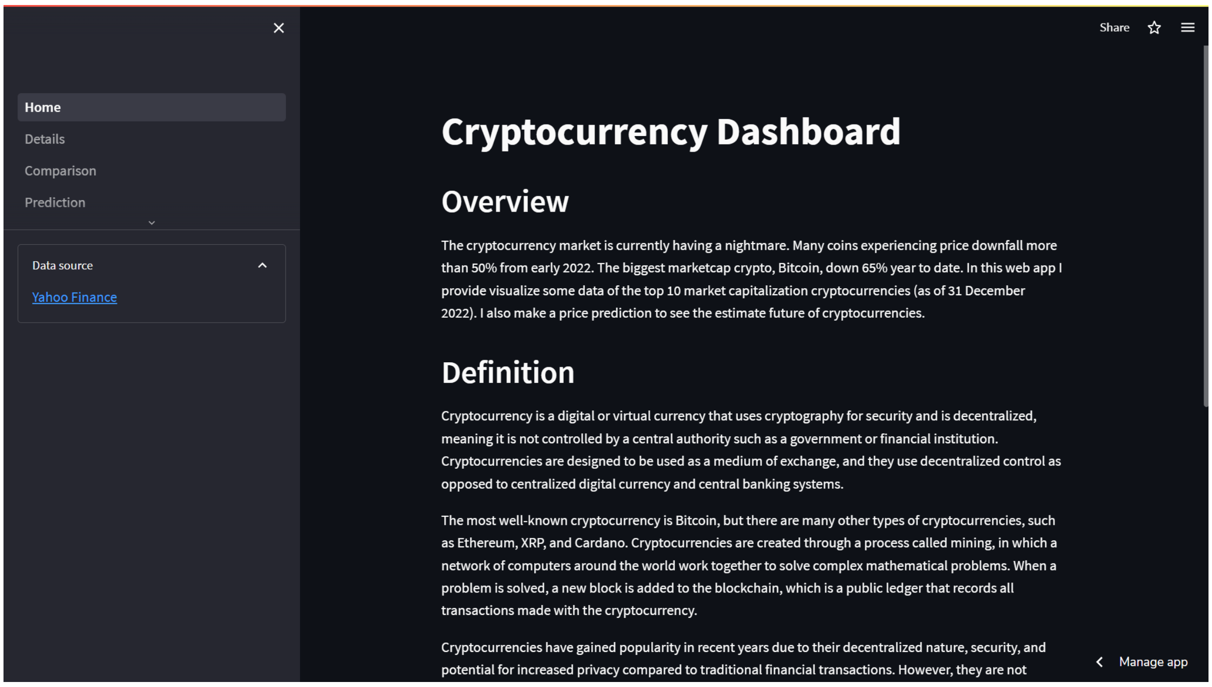
- *Printing the subheader on Streamlit and visualize the 3 years live price with Plotly*

```
st.subheader('Live 3 Years Chart')
st.plotly_chart(crypto_predict.visualize(crypto_data_3y))
```

- *Printing the subheader and making the Disclaimer expander widget in Streamlit, and printing the Plotly chart of the prediction*

```
st.subheader('6 Months Prediction')
disc = st.expander('Disclaimer. (Click to expand)')
disc.write("This is not a financial advise, buy at your own
risk.")
st.plotly_chart(crypto_predict.predict(crypto_data_for_prediction
))
```

Evidence of Working Program



Cryptocurrency Dashboard

Overview

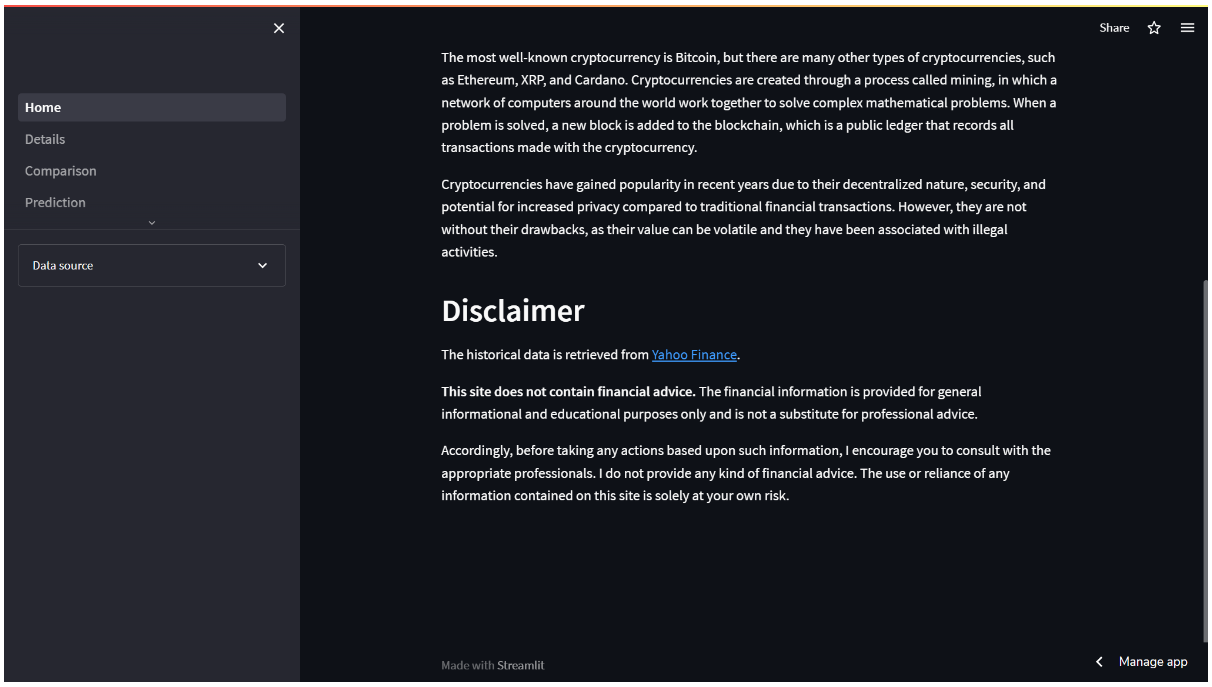
The cryptocurrency market is currently having a nightmare. Many coins experiencing price downfall more than 50% from early 2022. The biggest marketcap crypto, Bitcoin, down 65% year to date. In this web app I provide visualize some data of the top 10 market capitalization cryptocurrencies (as of 31 December 2022). I also make a price prediction to see the estimate future of cryptocurrencies.

Definition

Cryptocurrency is a digital or virtual currency that uses cryptography for security and is decentralized, meaning it is not controlled by a central authority such as a government or financial institution. Cryptocurrencies are designed to be used as a medium of exchange, and they use decentralized control as opposed to centralized digital currency and central banking systems.

The most well-known cryptocurrency is Bitcoin, but there are many other types of cryptocurrencies, such as Ethereum, XRP, and Cardano. Cryptocurrencies are created through a process called mining, in which a network of computers around the world work together to solve complex mathematical problems. When a problem is solved, a new block is added to the blockchain, which is a public ledger that records all transactions made with the cryptocurrency.

Cryptocurrencies have gained popularity in recent years due to their decentralized nature, security, and potential for increased privacy compared to traditional financial transactions. However, they are not



The most well-known cryptocurrency is Bitcoin, but there are many other types of cryptocurrencies, such as Ethereum, XRP, and Cardano. Cryptocurrencies are created through a process called mining, in which a network of computers around the world work together to solve complex mathematical problems. When a problem is solved, a new block is added to the blockchain, which is a public ledger that records all transactions made with the cryptocurrency.

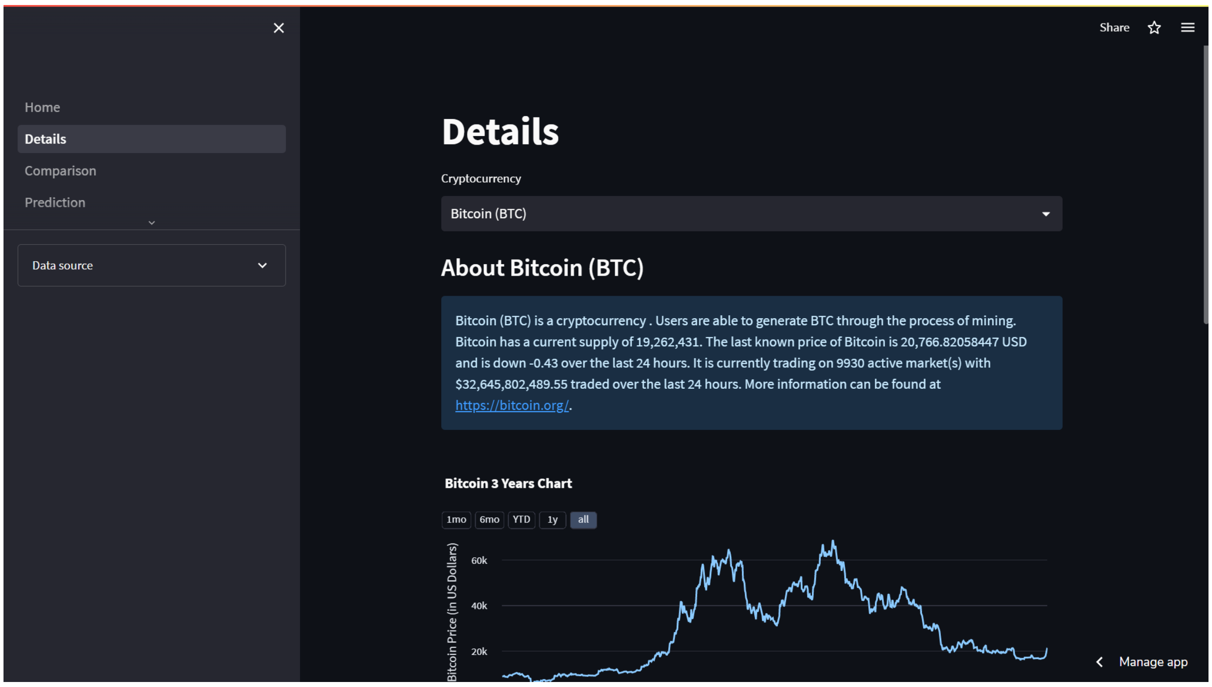
Cryptocurrencies have gained popularity in recent years due to their decentralized nature, security, and potential for increased privacy compared to traditional financial transactions. However, they are not without their drawbacks, as their value can be volatile and they have been associated with illegal activities.

Disclaimer

The historical data is retrieved from Yahoo Finance.

This site does not contain financial advice. The financial information is provided for general informational and educational purposes only and is not a substitute for professional advice.

Accordingly, before taking any actions based upon such information, I encourage you to consult with the appropriate professionals. I do not provide any kind of financial advice. The use or reliance of any information contained on this site is solely at your own risk.



Details

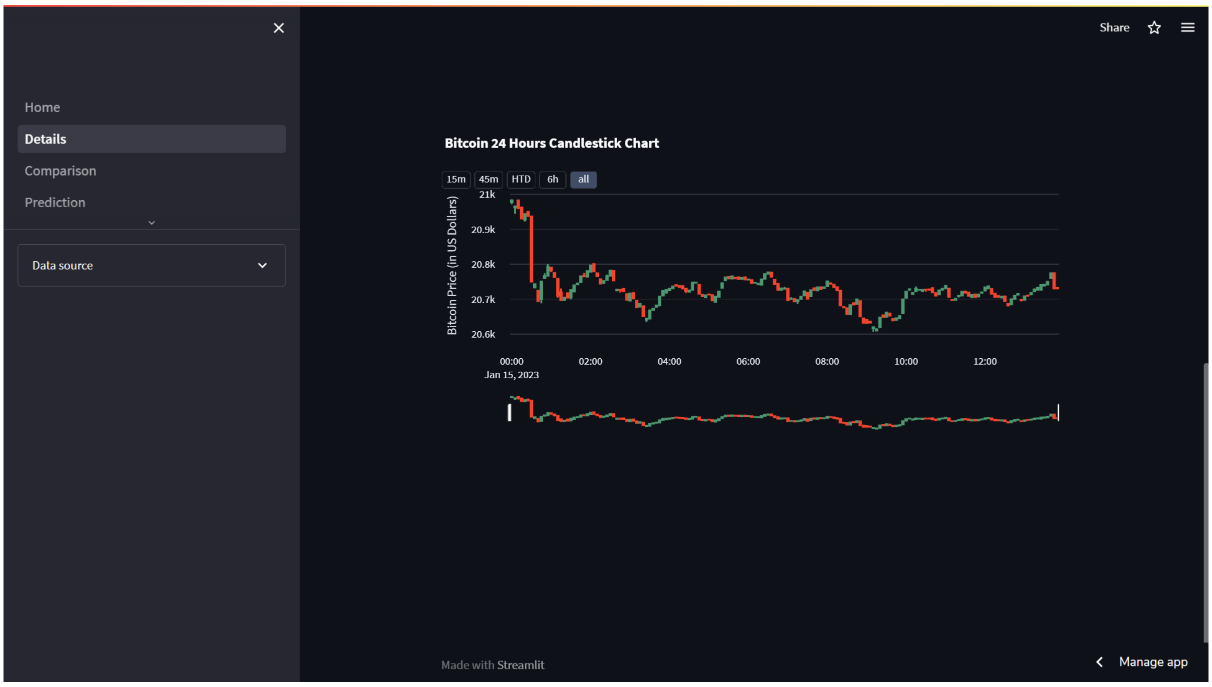
Cryptocurrency

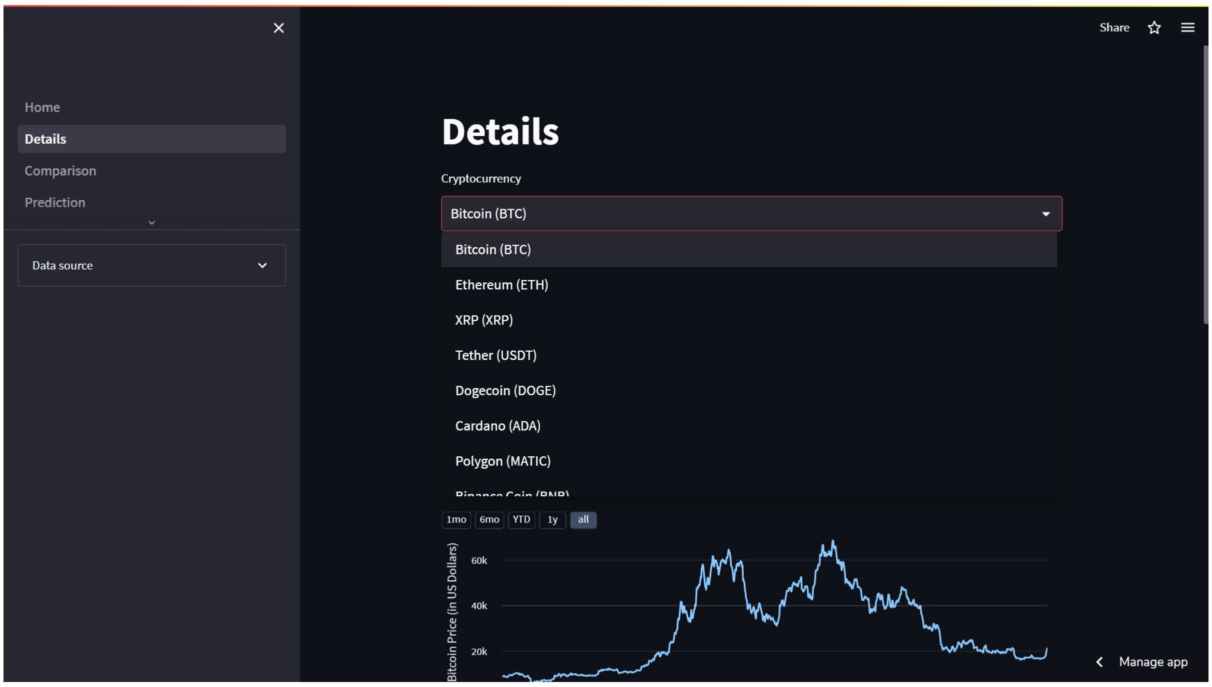
Bitcoin (BTC)

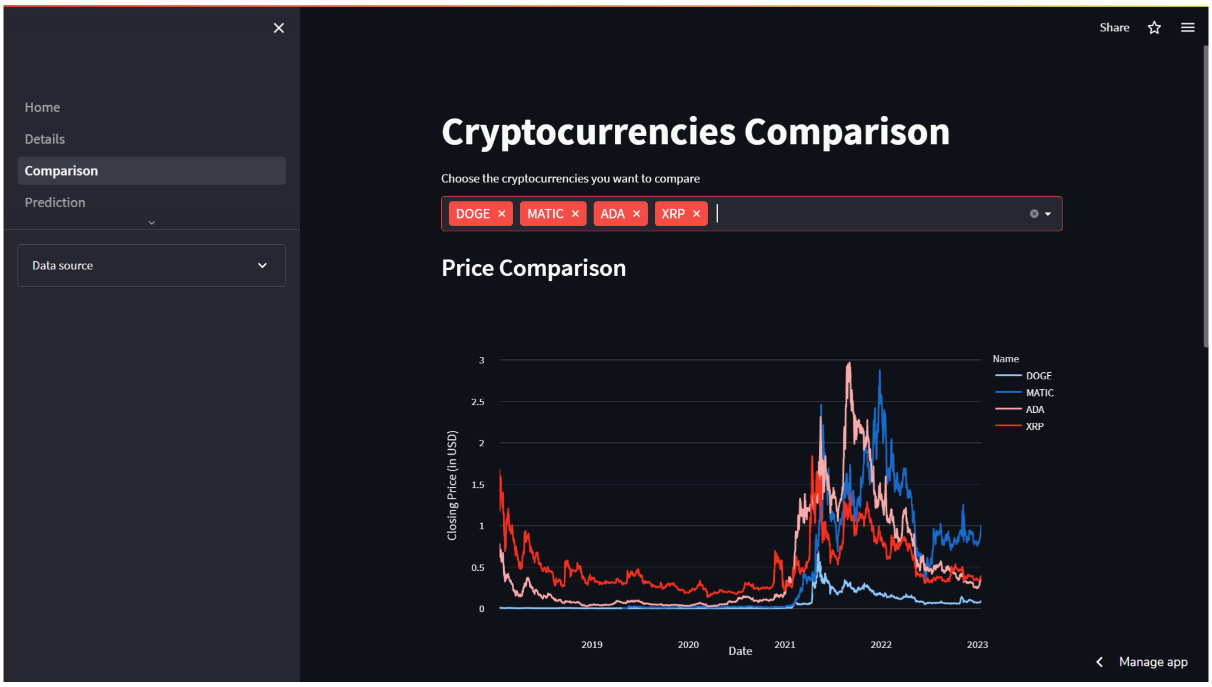
About Bitcoin (BTC)

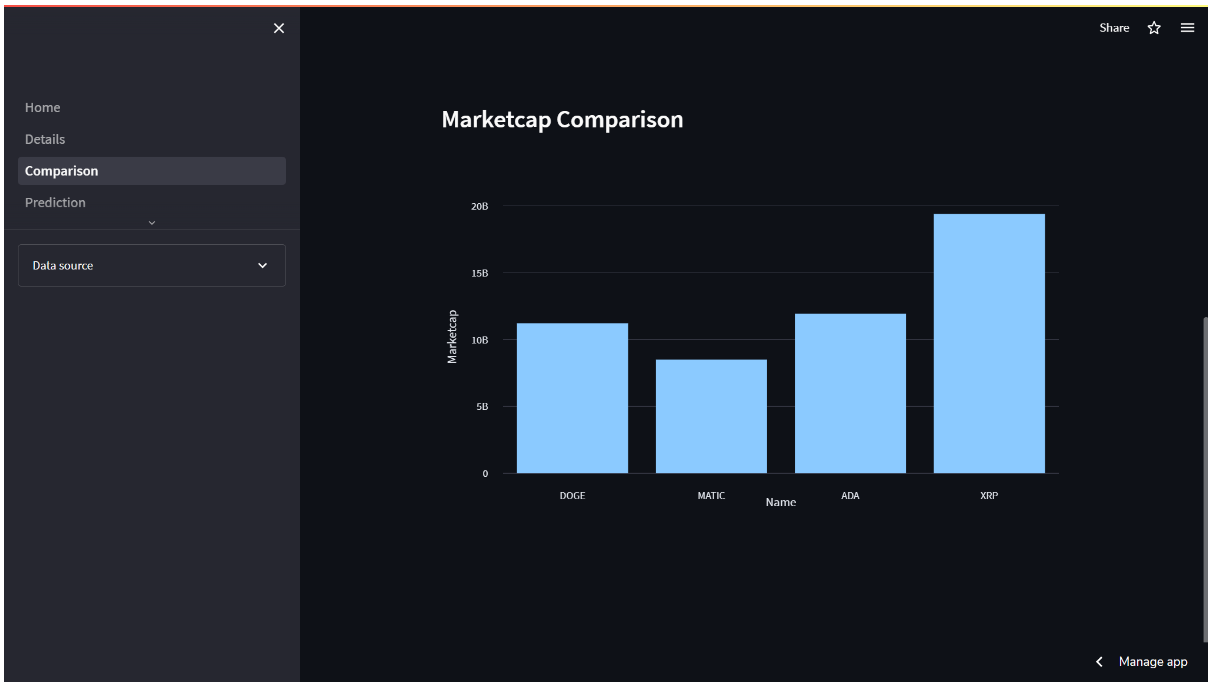
Bitcoin (BTC) is a cryptocurrency . Users are able to generate BTC through the process of mining. Bitcoin has a current supply of 19,262,431. The last known price of Bitcoin is 20,766.82058447 USD and is down -0.43 over the last 24 hours. It is currently trading on 9930 active market(s) with $32,645,802,489.55 traded over the last 24 hours. More information can be found at https://bitcoin.org/.

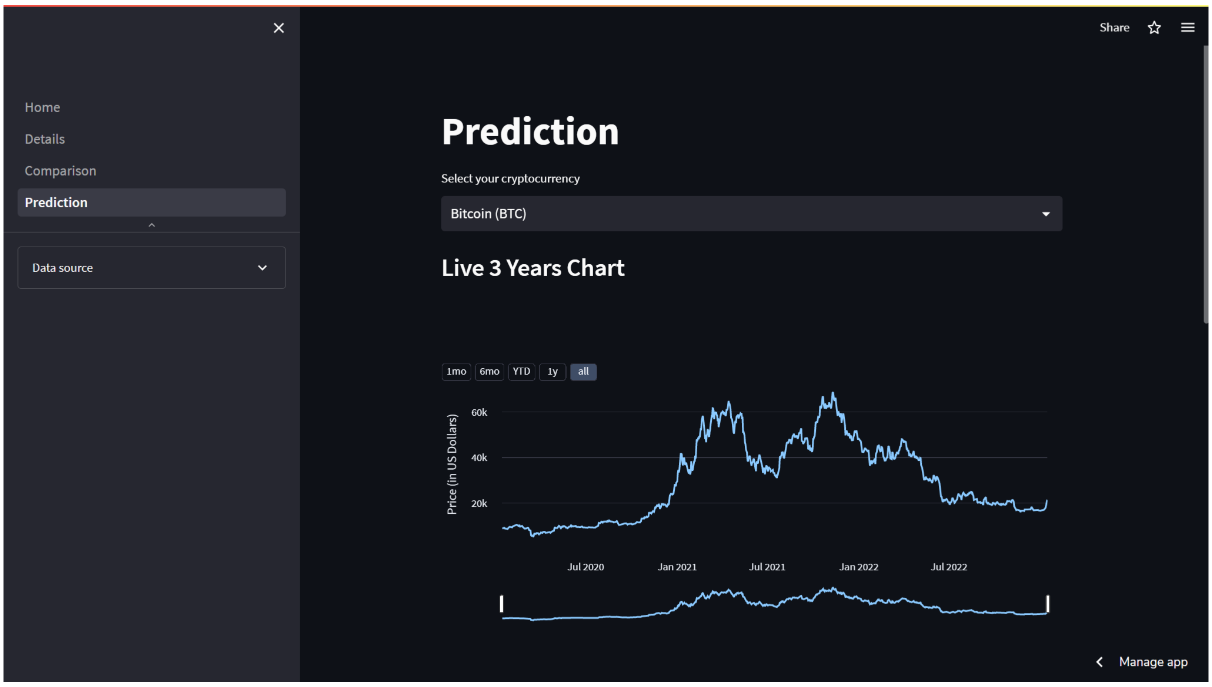
Bitcoin 3 Years Chart

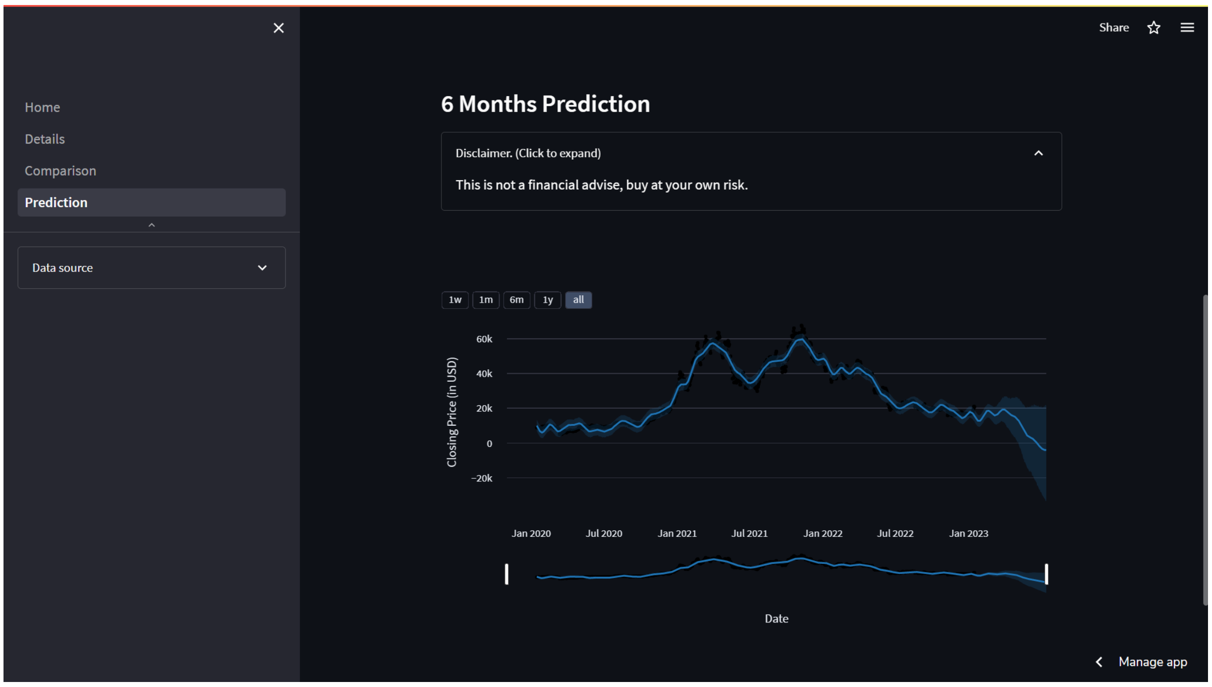



Share
Home
Details
Comparison
Prediction
Data source
Details
Cryptocurrency
Bitcoin (BTC)
Bitcoin (BTC)
Ethereum (ETH)
XRP (XRP)
Tether (USDT)
Dogecoin (DOGE)
Cardano (ADA)
Polygon (MATIC)
1mo
6mo
YTD
1y
all
Bitcoin Price (in US Dollars)
60k
40k
20k
Manage app


Share
Home
Details
Comparison
Prediction
Data source
Cryptocurrencies Comparison
Choose the cryptocurrencies you want to compare
DOGE ×
MATIC ×
ADA ×
XRP ×
Price Comparison
Closing Price (in USD)
3
2.5
2
1.5
1
0.5
0
2019
2020
2021
2022
2023
Date
Name
DOGE
MATIC
ADA
XRP
Manage app


Share
Home
Details
Comparison
Prediction
Data source
Marketcap Comparison
20B
15B
10B
5B
0
Marketcap
DOGE
MATIC
ADA
XRP
Name
Manage app


Share
Home
Details
Comparison
Prediction
Data source
Prediction
Select your cryptocurrency
Bitcoin (BTC)
Live 3 Years Chart
1mo
6mo
YTD
1y
all
60k
40k
20k
Price (in US Dollars)
Jul 2020
Jan 2021
Jul 2021
Jan 2022
Jul 2022
Manage app



6 Months Prediction

Disclaimer. (Click to expand)

This is not a financial advise, buy at your own risk.

Reflection

When creating this cryptocurrency dashboard project, I learned more than I had anticipated. I gained knowledge in data visualization using Pandas and Plotly, as well as Streamlit for deploying the app. I also learned about cryptocurrency market capitalization and various other topics. The process of making this project involved learning almost everything from scratch, starting with new libraries that I had not previously heard of and progressing to new concepts in data visualization. I would like to extend my gratitude to all those who provided tutorials, which I read and watched from various sources including online articles, official documentation, and YouTube videos. Through this project, I discovered that I am interested in learning more about data, such as data science and data analysis.

However, I did encounter some problems while working on this project. These included difficulties in selecting ideas for the project and issues with libraries just days before the submission deadline. To solve these problems, I had to consult additional sources such as StackOverflow, libraries' official documentation, and YouTube videos. Additionally, bugs randomly appeared, even when I thought my logic was correct. Furthermore, I was unable to use the concept of inheritance in my classes as the file name required by Streamlit to create a multipage web application has a special character _ (underscore), for example, 2_Details.py and 3_Comparison.py. This resulted in Python being unable to detect the file name when I tried to import the file containing the class.

Sources

Data Source:

Yahoo Finance - https://finance.yahoo.com/

Library official documentation:

Streamlit - https://docs.streamlit.io/

Plotly - https://plotly.com/python/

yfinance - https://pypi.org/project/yfinance/

Pandas - https://pandas.pydata.org/docs/

Prophet - https://facebook.github.io/prophet/docs/quick_start.html#python-api

External Resources:

https://www.forbes.com/advisor/investing/cryptocurrency/top-10-cryptocurrencies/

https://github.com/ranaroussi/yfinance/issues/1291

https://www.knowledgehut.com/blog/business-intelligence-and-visualization/python-data-visualization-libraries

https://towardsdatascience.com/top-6-python-libraries-for-visualization-which-one-to-use-fe43381cd658

https://www.kaggle.com/code/kanncaa1/plotly-tutorial-for-beginners

https://towardsdatascience.com/deploy-a-public-streamlit-web-app-for-free-heres-how-bf56d46b2abe

https://www.influxdata.com/what-is-time-series-data/

https://www.analyticsinsight.net/top-10-python-libraries-for-time-series-analysis-in-2022/

https://www.youtube.com/watch?v=zid-MVo7M-E&t=577s

https://www.youtube.com/watch?v=o5rQ69lB75k&t=591s