PROJECT REPORT

Team Id – 592419

1. **Introduction:**

1.1 Project Overview

The primary objective of this project is to develop a predictive model capable of estimating the future share prices of the top 5 GPU (Graphics Processing Unit) companies in the stock market. By leveraging historical stock data, financial indicators, market trends, and advanced machine learning techniques, the model aims to provide accurate and actionable predictions for investors and market participants.

1.2 Purpose

The purpose of the "Share Price Estimation of Top 5 GPU Companies" project is to empower investors with accurate share price predictions, enhance market predictability, facilitate effective risk management, and encourage informed and strategic investments in the dynamic GPU industry. By leveraging advanced machine learning, the project aims to contribute to financial education, support long-term investor confidence, and promote technological innovation within financial analysis.

2. **Literature Survey:**

2.1 Existing Problem

The existing problem is the inherent challenge of accurately predicting the future share prices of top GPU companies due to the complex and dynamic nature of stock markets, influenced by factors like technological shifts, market sentiment, and macroeconomic conditions. This uncertainty poses a barrier to informed decision-making for investors in the GPU industry.

2.2 References

https://www.youtube.com/watch?v=lj4I_CvBnt0

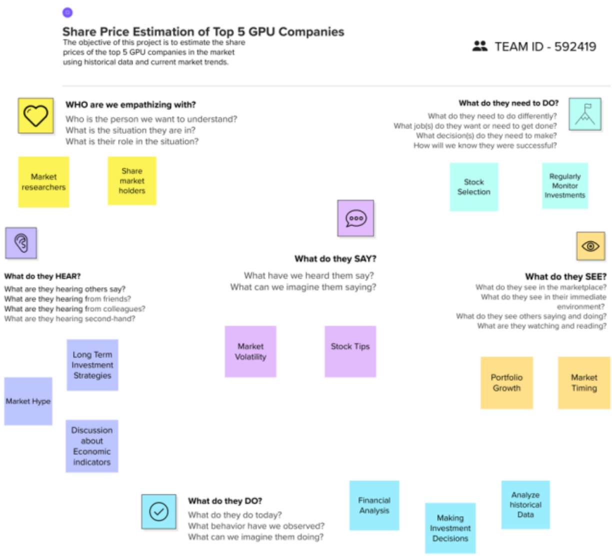
https://www.kaggle.com/code/subhajeetdas/nvidia-amd-intel-asus-msi-share-price-predict/notebook

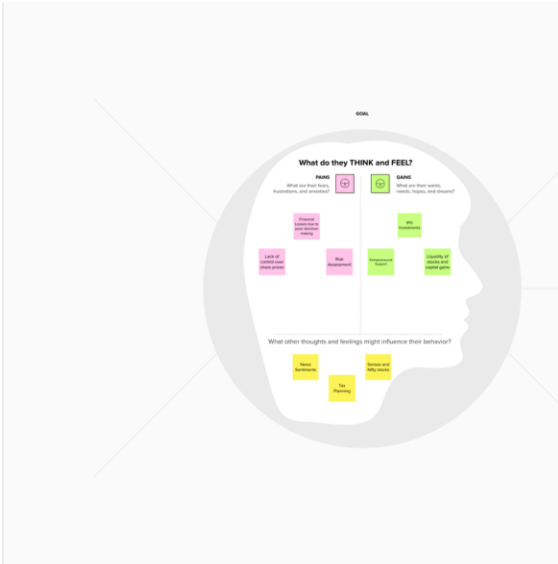
2.3 Project Statement Definition

The problem statement involves developing a solution to address the difficulty in predicting the future share prices of the top 5 GPU companies. The challenge lies in navigating the complex dynamics of the stock market, impacted by factors such as technological advancements, market sentiment, and macroeconomic conditions, hindering effective investment decision-making.

3. Ideation and Proposed Solution:

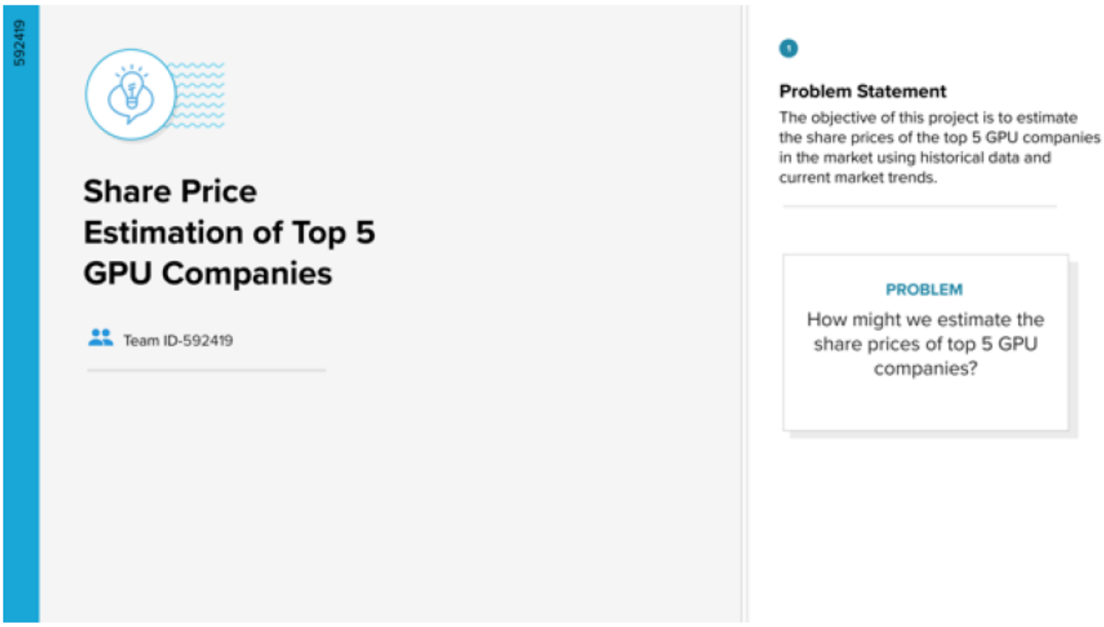
3.1 Empathy Map Canvas



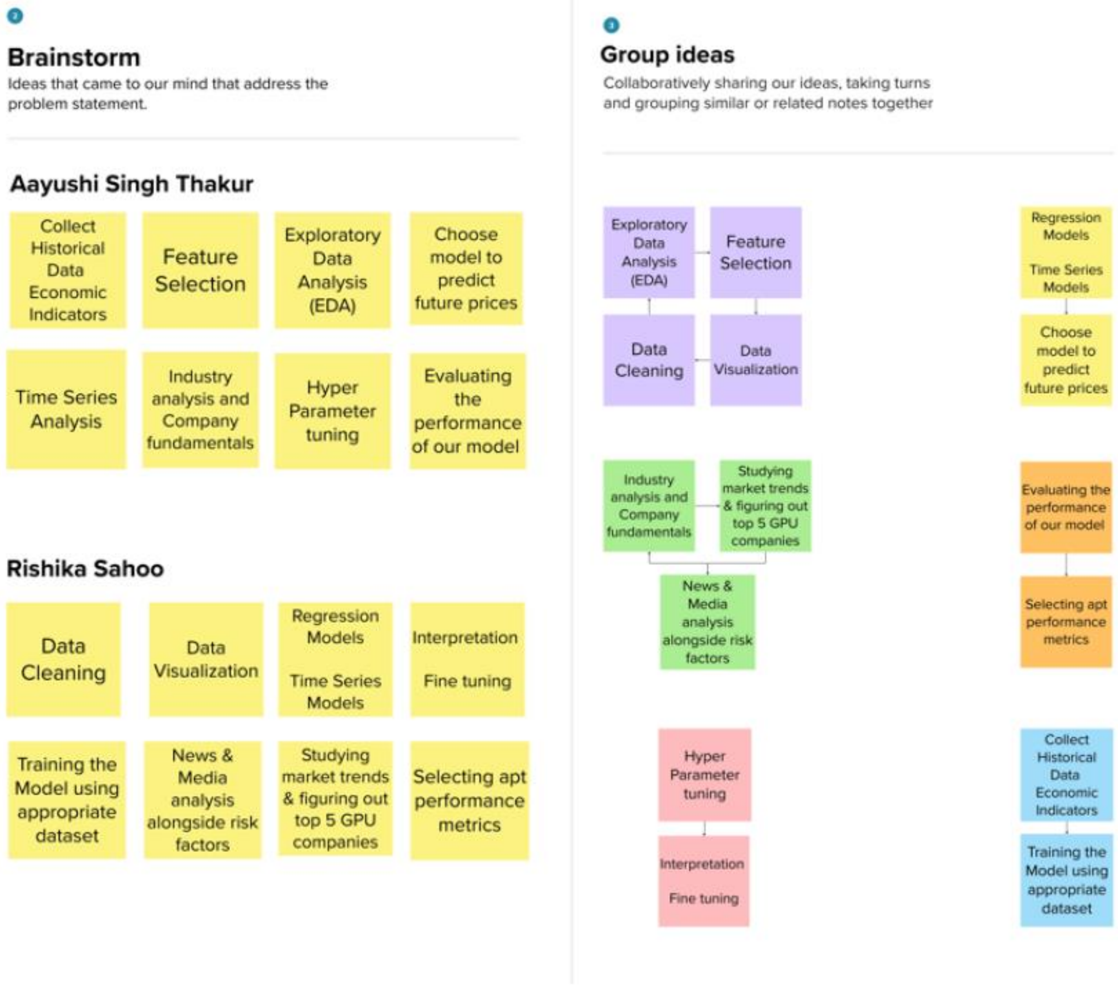


3.2 Ideation and Brainstorming

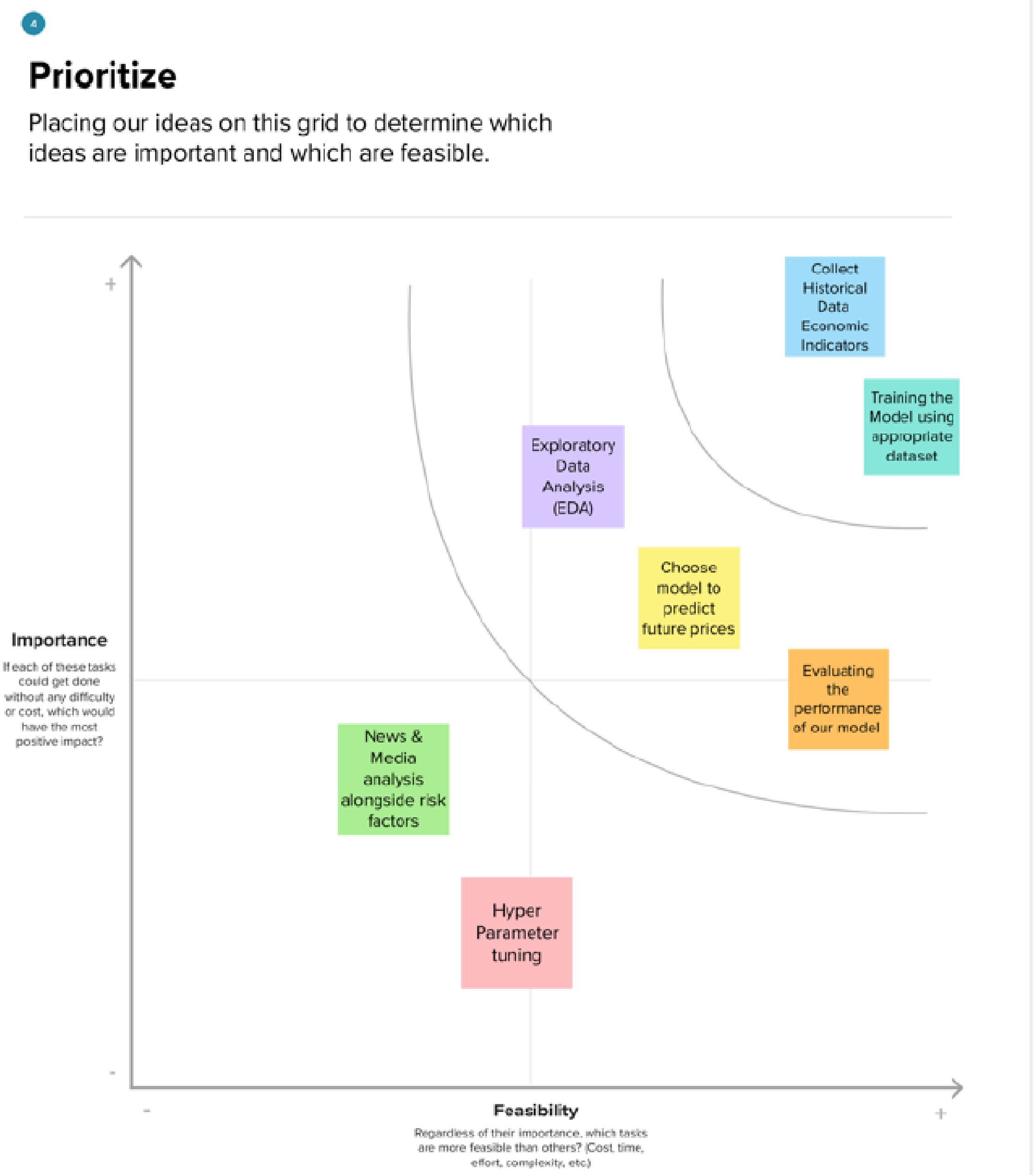
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



4. Requirement Analysis:

4.1 Functional Requirement

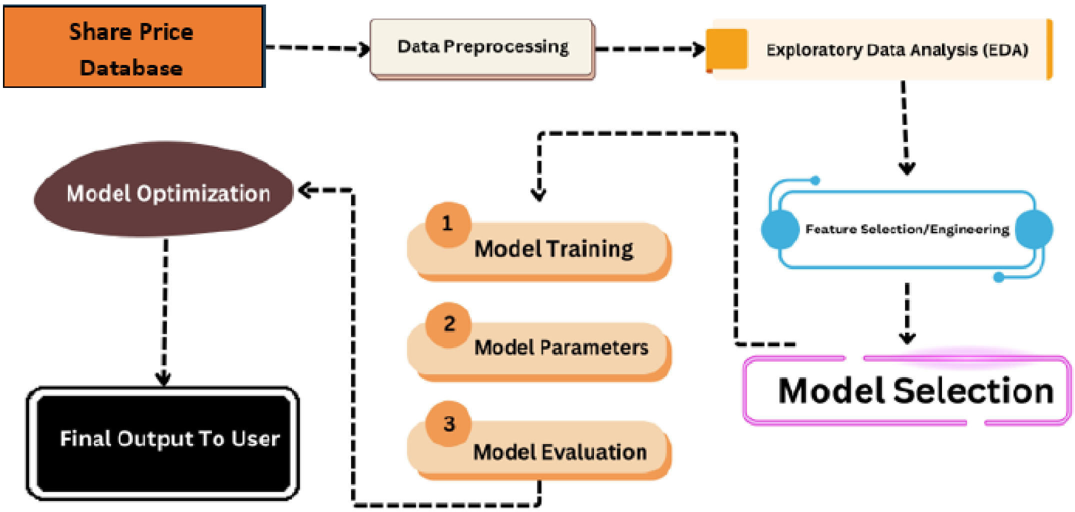
Functional requirements include collecting and preprocessing historical stock data, integrating financial indicators, market indices, and technical indicators, implementing advanced machine learning models for accurate predictions, and developing a user-friendly interface for investors to access real-time and predictive analytics on the top 5 GPU companies' share prices. These features aim to create a comprehensive and scalable solution to assist investors in making informed decisions.

4.2 Non-Functional Requirement

Non-functional requirements encompass ensuring the model's scalability to handle increased data volumes and user interactions efficiently. Additionally, the system should prioritize high reliability and low-latency response times for providing real-time share price predictions, contributing to a seamless and dependable user experience.

5. Project Design

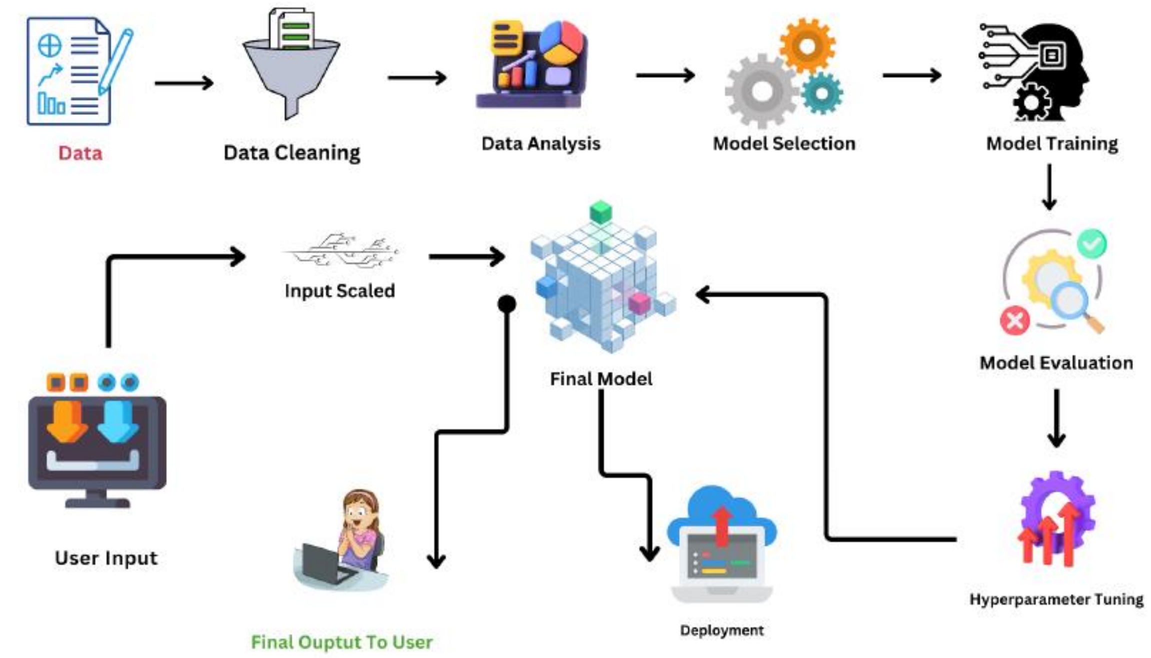
5.1 Dataflow Diagrams and User stories



User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
DSN-1	As a data scientist, I can preprocess the share price data to prepare it for analysis.	I can transform raw data into a suitable format for further analysis.	High	Sprint-1
DSN-2	As a data scientist, I can perform exploratory data analysis on the pre-processed data.	I can understand the relationships between different features and the target variable.	High	Sprint-1
DSN-3	As a data scientist, I can select and engineer features for the machine learning model.	I can identify the most relevant features for predicting share prices.	High	Sprint-2
DSN-4	As a data scientist, I can select an appropriate machine learning model for price prediction.	I can choose a model that is suitable for regression tasks.	High	Sprint-2

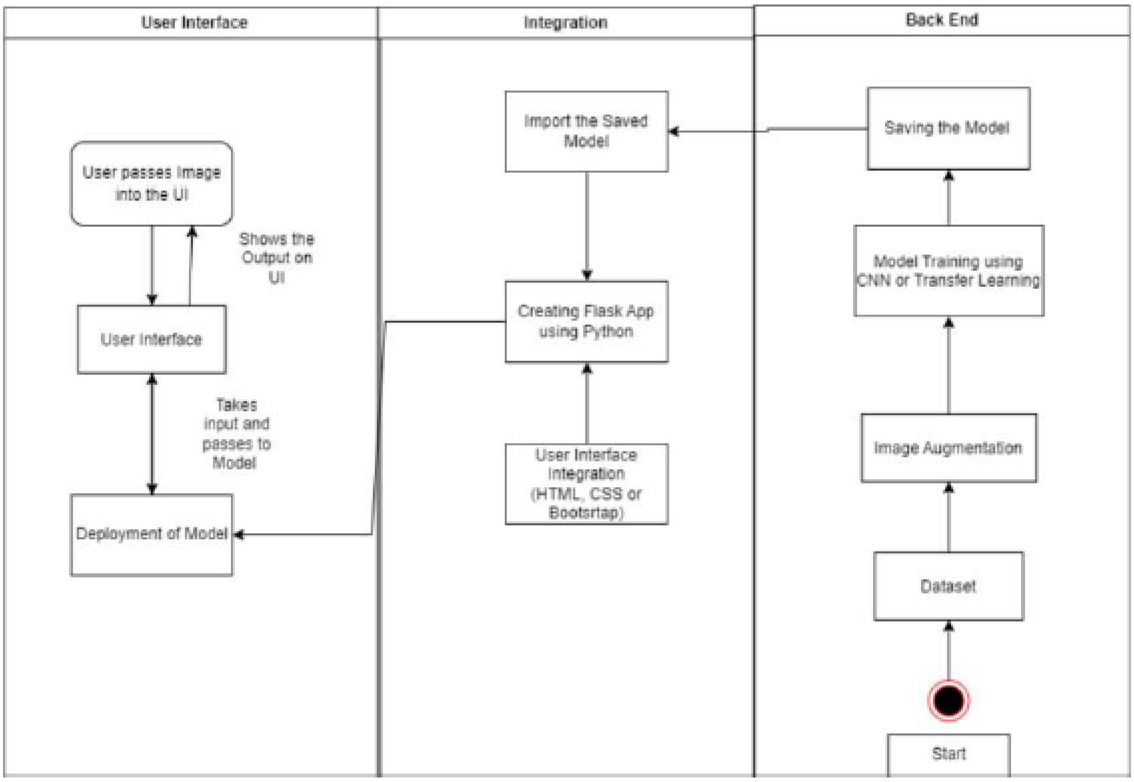
DSN-5	As a data scientist, I can train the selected model on the training dataset.	I can fit the model to the training data and adjust its parameters.	High	Sprint-3
DSN-6	As a data scientist, I can evaluate the performance of the trained model.	I can assess how well the model predicts share prices using appropriate metrics.	High	Sprint-3
DSN-7	As a data scientist, I can optimize the performance of the model if necessary.	I can improve the model's performance by tuning its parameters or using advanced techniques.	Medium	Sprint-4

5.2 Solution Architecture



6. Project Planning and Scheduling

6.1 Technical Architecture

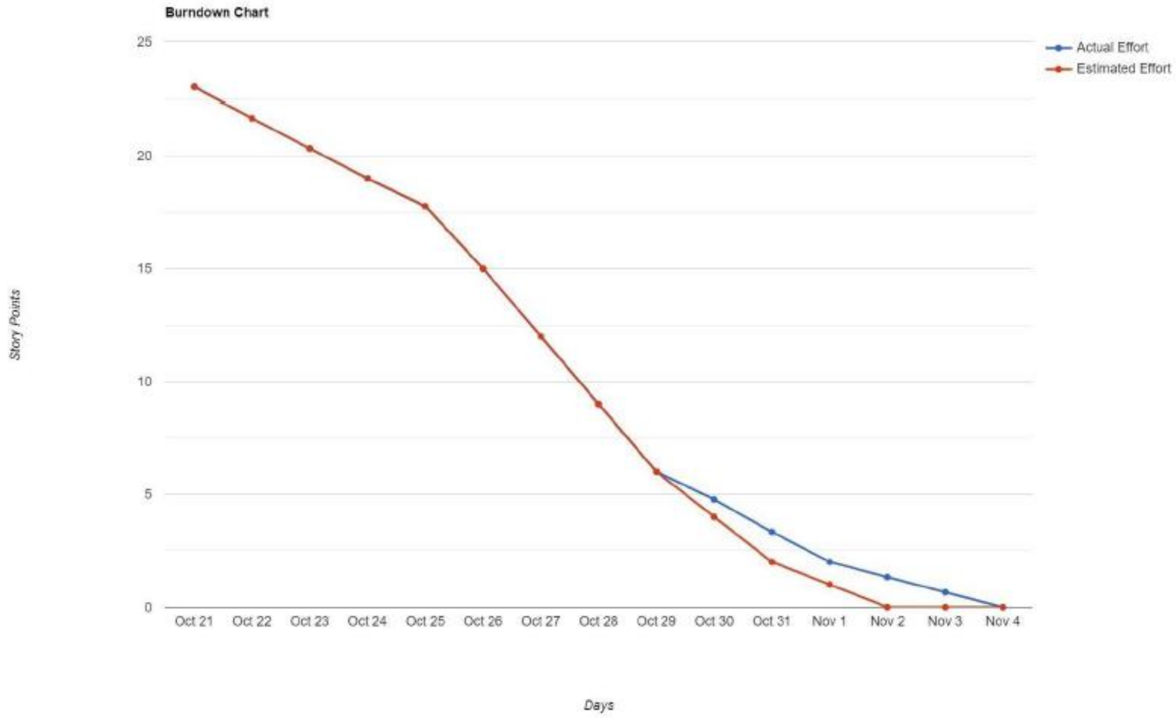


6.2 Sprint Planning and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Historical Stock Prices	USN-1	Include historical daily or minute-by-minute stock prices for the target GPU companies.	5	High	Rishika
Sprint-1	Financial Indicators	USN-2	Integrate financial indicators such as earnings per share (EPS), price-to-earnings ratio (P/E), and return on equity (ROE).	8	High	Aayushi
Sprint-2	Market Indices	USN-3	Include relevant market indices, such as the S&P 500 or industry-specific indices, to capture broader market trends.	3	Medium	Rishika
Sprint-2	Technical Indicators	USN-4	Incorporate technical indicators like moving averages, Relative Strength Index (RSI), and Bollinger Bands.	5	Medium	Aayushi
Sprint-1	News and Sentiments Analysis	USN-5	Integrate sentiment analysis of news articles and social media related to the GPU companies.	8	High	Rishika
Sprint -1	Macroeconomic factors	USN-6	Include macroeconomic indicators such as GDP growth, inflation rates, and interest rates.	5	High	Aayushi
Sprint-2	Dividend yields	USN-7	Consider the dividend yields of the GPU companies	8	Medium	Rishika
Sprint 2	Volatility measures	USN-8	Incorporate measures of stock volatility, such as historical volatility or implied volatility	3	Medium	Aayushi

6.3 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	4	3 Days	21 Oct 2023	23 Oct 2023	4	23 Oct 2023
Sprint-2	4	3 Days	24 Oct 2023	26 Oct 2023	4	26 Oct 2023
Sprint-3	10	3 Days	27 Oct 2023	29 Oct 2023	9	29 Oct 2023
Sprint-4	4	3 Days	30 Oct 2023	1 November 2023	4	31 Oct 2023
Sprint -5	2	3 Days	2 Nov 2023	4 Nov 2023	2	2 Nov 2023



7. Coding and Solutioning:

IMPORTING ALL THE REQUIRED LIBRARIES

```
[5] import matplotlib.pyplot as plt
    import seaborn as sns
    import tensorflow as tf
    from sklearn.model_selection import train_test_split
    from statsmodels.tsa.arima.model import ARIMA

    import sklearn
    from sklearn.metrics import mean_squared_error, mean_absolute_error,r2_score
    import math
    from tqdm import tqdm_notebook
    import numpy as np
    import pandas as pd
    from itertools import product
    from sklearn.linear_model import LinearRegression
    import warnings
    warnings.filterwarnings( "ignore" )
```

```
[2] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

READING THE DATASETS

```
[6] amd=pd.read_csv("/content/drive/MyDrive/AMD (1980 -11.07.2023).csv")
    asus=pd.read_csv("/content/drive/MyDrive/ASUS (2000 - 11.07.2023).csv")
    intel=pd.read_csv("/content/drive/MyDrive/INTEL (1980 - 11.07.2023).csv")
    msi=pd.read_csv("/content/drive/MyDrive/Motorola Solutions (MSI) (1962 -11.07.2023).csv")
    nvidia=pd.read_csv("/content/drive/MyDrive/NVIDIA (1999 -11.07.2023).csv")
```

Displaying first 5 rows of all the dataframes in order to ensure that the data is loaded correctly

```
[ ] amd.head()
```

```
[ ] asus.head()
```

```
[ ] intel.head()
```

```
[ ] msi.head()
```

```
[ ] nvidia.head()
```

DATA PREPARATION

Checking for null values

```
[ ] amd.isnull().sum()
```

```
[ ] asus.isnull().sum()
```

```
[ ] msi.isnull().sum()
```

```
[ ] intel.isnull().sum()
```

```
nvidia.isnull().sum()
```

DATA MANIPULATION

```
[ ]  asus=asus.dropna()
```

```
[ ]  data_list=[amd, asus, intel, msi, nvidia]
     for data in data_list:
         data["Date"]=pd.to_datetime(data["Date"])
```

RESAMPLING THE DATA

```
[ ]  data_list = [amd,asus, intel, msi, nvidia]

     names=[0,1,2,3,4]
     index=0
     for data in data_list:
       dates=data["Date"]
       data["Company"]=np.repeat(names[index],len(data))
       data["Year"]=dates.dt.year
       data["Month"]=dates.dt.month
       data["Day"]=dates.dt.day
       index+=1
```

TRAIN TEST SPLITTING

```
[7]  data_list=[amd,asus,msi,intel,nvidia]
     test_data=[]
     train_data=[]
     for data in data_list:
       train=data[:int(0.8*len(data))]
       test=data[int(0.8*len(data)):]
       train_data.append(train)
       test_data.append(test)
       print(test.shape,train.shape)
```

```
[ ]  train_data=pd.concat(train_data)
     test_data=pd.concat(test_data)
     print(train_data.shape,test_data.shape)
```

```
[ ]  x_train=train_data[["Open","High","Low","Volume","Year","Month","Day","Company"]]
     x_test=test_data[["Open","High","Low","Volume","Year","Month","Day","Company"]]
     y_train=train_data["Close"]
     y_test=test_data["Close"]
     print(x_train.shape)
     print(x_test.shape)
     print(y_train.shape,y_test.shape)
```

EXPLORATORY DATA ANALYSIS

Descriptive Statistical

```
[ ]  amd.describe(include="all")
```

```
[ ]  asus.describe(include="all")
```

```
[ ]  intel.describe(include="all")
```

```
[ ]  msi.describe(include="all")
```

```
[ ]  nvidia.describe(include="all")
```

Visual Analysis

```
import matplotlib.dates as mdates

df_plot=amd.drop(columns=['Date'])
ncols=2
nrows=int(round(df_plot.shape[1]/ncols,0))

fig,ax=plt.subplots(nrows=nrows,ncols=ncols,sharex=True,figsize=(14,7))
for i,ax in enumerate(fig.axes):
  sns.lineplot(data=df_plot.iloc[:,i],ax=ax)
  ax.tick_params(axis="x",rotation=30,labelsize=10,length=0)
  ax.xaxis.set_major_locator(mdates.AutoDateLocator())
fig.tight_layout()
plt.show()
```

MODEL BUILDING

```
lr=LinearRegression()
lr.fit(x_train,y_train)
```

```
print('Test Score:',lr.score(x_test,y_test))
print('Train Score:',lr.score(x_train,y_train))
```

```
y_pred=lr.predict(x_test)
print("r2_score:",r2_score(y_test,y_pred))
print("MAE:",mean_absolute_error(y_test,y_pred))
```

Double-click (or enter) to edit

```
amd_dates=test_data[test_data["Company"]==0]["Date"]
amd_pred=lr.predict(x_test[x_test["Company"]==0])
amd_orig=test_data[test_data["Company"]==0]["Close"]

asus_dates=test_data[test_data["Company"]==1]["Date"]
asus_pred=lr.predict(x_test[x_test["Company"]==1])
asus_orig=test_data[test_data["Company"]==1]["Close"]

intel_dates=test_data[test_data["Company"]==2]["Date"]
intel_pred=lr.predict(x_test[x_test["Company"]==2])
intel_orig=test_data[test_data["Company"]==2]["Close"]

msi_dates=test_data[test_data["Company"]==3]["Date"]
msi_pred=lr.predict(x_test[x_test["Company"]==3])
msi_orig=test_data[test_data["Company"]==3]["Close"]

nvidia_dates=test_data[test_data["Company"]==4]["Date"]
nvidia_pred=lr.predict(x_test[x_test["Company"]==4])
nvidia_orig=test_data[test_data["Company"]==4]["Close"]
```

```python
# Combine data for AMD
amd_data = {'Date': amd_dates, 'Original': amd_orig, 'Predicted': amd_pred}
amd_df = pd.DataFrame(amd_data)

# Combine data for ASUS
asus_data = {'Date': asus_dates, 'Original': asus_orig, 'Predicted': asus_pred}
asus_df = pd.DataFrame(asus_data)

# Combine data for INTEL
intel_data = {'Date': intel_dates, 'Original': intel_orig, 'Predicted': intel_pred}
intel_df = pd.DataFrame(intel_data)

# Combine data for MSI
msi_data = {'Date': msi_dates, 'Original': msi_orig, 'Predicted': msi_pred}
msi_df = pd.DataFrame(msi_data)

# Combine data for NVIDIA
nvidia_data = {'Date': nvidia_dates, 'Original': nvidia_orig, 'Predicted': nvidia_pred}
nvidia_df = pd.DataFrame(nvidia_data)

# Plotting
plt.figure(figsize=(15, 8))

sns.lineplot(x='Date', y='Original', data=amd_df, label='AMD Original')
sns.lineplot(x='Date', y='Predicted', data=amd_df, label='AMD Predicted')

sns.lineplot(x='Date', y='Original', data=asus_df, label='ASUS Original')
sns.lineplot(x='Date', y='Predicted', data=asus_df, label='ASUS Predicted')

sns.lineplot(x='Date', y='Original', data=intel_df, label='INTEL Original')
sns.lineplot(x='Date', y='Predicted', data=intel_df, label='INTEL Predicted')

sns.lineplot(x='Date', y='Original', data=msi_df, label='MSI Original')
sns.lineplot(x='Date', y='Predicted', data=msi_df, label='MSI Predicted')

sns.lineplot(x='Date', y='Original', data=nvidia_df, label='NVIDIA Original')
sns.lineplot(x='Date', y='Predicted', data=nvidia_df, label='NVIDIA Predicted')

plt.legend()
plt.show()
```

SAVING MODEL

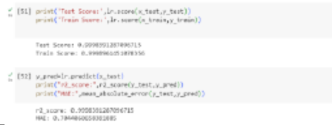
```python
[ ] import pickle as pkl
    pkl.dump(lr,open("lr.pkl","wb"))
```

We cannot retrieve the summary as it is for a linear regression model as Linear Regression does not have a summary method, rather we can do this:

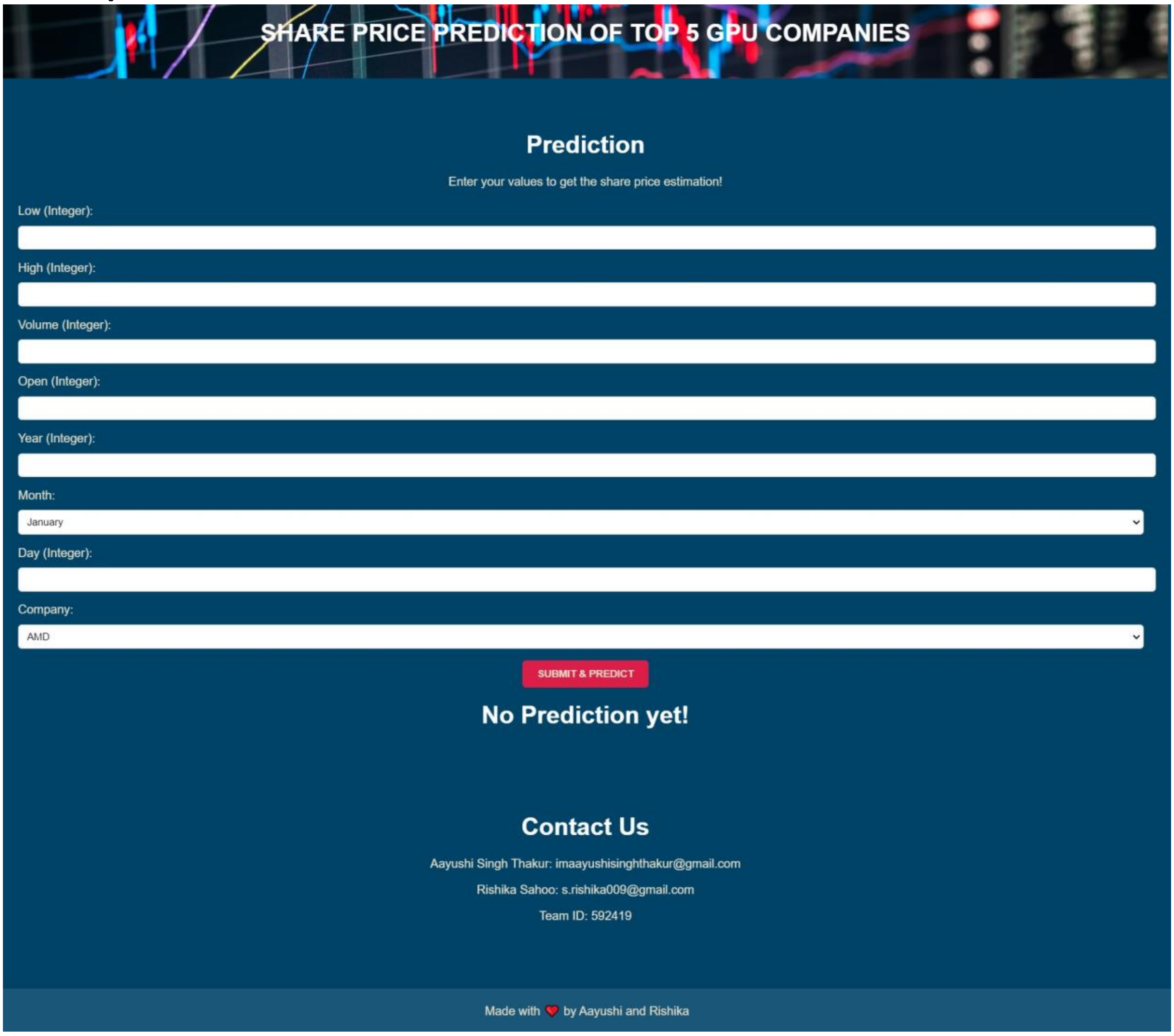
```python
# Retrieve the coefficients and intercept
coefficients = lr.coef_
intercept = lr.intercept_

# Print the coefficients and intercept
print("Coefficients:", coefficients)
print("Intercept:", intercept)
```

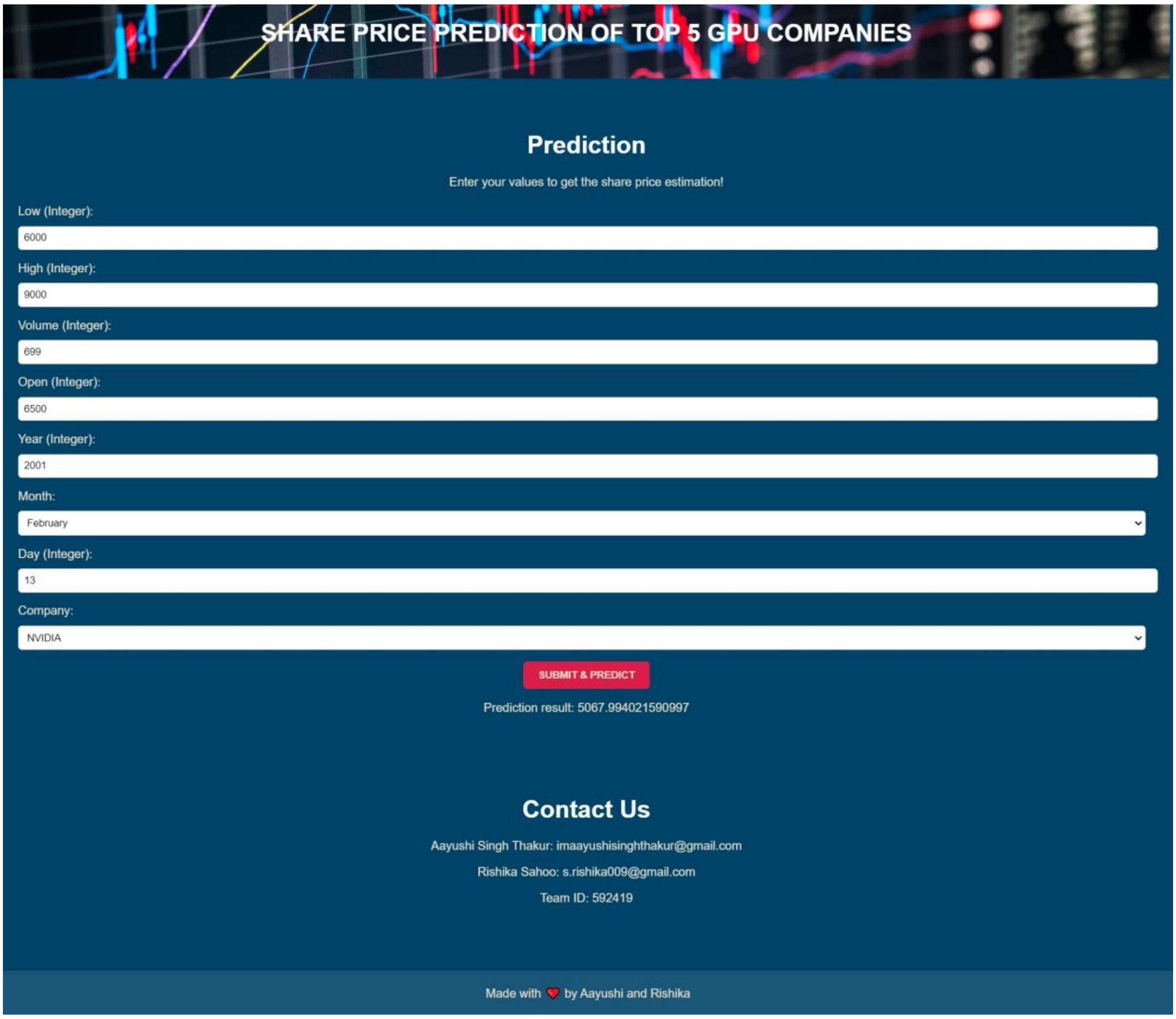
Features: Taking in input from the user about the low, high, open, date and company name, we predict the closing price using the historical data that we have in the form of a dataset.

8. Performance testing:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	We cannot retrieve the summary as it is for a linear regression model as Linear Regression does not have a summary method, rather we did this.	
2.	Accuracy	Training Accuracy - Validation Accuracy -	
3.	Confidence Score (Only Yolo Projects)	Class Detected - Confidence Score -	NOT APPLICABLE

9. Output Screenshots:





10. Advantages:

Informed Decision-Making: The project empowers investors with accurate share price predictions, facilitating informed decision-making and strategic planning for investments in the GPU industry.

Market Stability: By contributing to market predictability and risk management, the project supports overall market stability, reducing uncertainties and promoting a rational investment environment.

Technological Innovation: Leveraging advanced machine learning technologies, the project sets a precedent for technological innovation within financial analysis, showcasing the potential for future advancements.

Disadvantages:

Data Sensitivity: Reliance on historical and real-time data introduces a sensitivity to data accuracy, and the model's predictions are contingent on the quality and timeliness of the input data.

Market Volatility: The inherent volatility of stock markets, especially in the technology sector, may pose challenges to the model's accuracy during periods of rapid and unpredictable market fluctuations.

Dependency on External Factors: External factors, such as sudden geopolitical events or unforeseen economic shifts, can significantly impact share prices, presenting a challenge for the model to adapt quickly to unforeseen circumstances.

11.Conclusion:

In this project, we successfully developed a web application for predicting share prices of the top GPU companies. Leveraging Flask for the backend, our application seamlessly integrates machine learning predictions from a pre-trained model using a pickle file. With a user-friendly interface, the website allows users to input data, receive accurate share price forecasts, and navigate through informative pages. This project demonstrates the fusion of data science and web development, offering a valuable tool for investors and enthusiasts in the dynamic GPU market.

12.Future Scope:

The future scope of this project includes enhancing prediction accuracy by incorporating real-time market data, implementing user authentication for personalized experiences, integrating additional machine learning models for comprehensive insights, expanding company coverage, and developing a mobile application to provide on-the-go share price predictions, making it a versatile and robust tool for investors and financial enthusiasts.

13. Appendix:

The below drive link contains the source file for the frontend as well as the backend and the video description of our project:

https://drive.google.com/drive/folders/19GODohrumjTJpMYbkbWKoHJgh1GCD2BA?usp=sharing