COURSE	Application Development Frameworks (2CSDE86)
PROJECT TITLE	**Django Web App for Stock Portfolio and Visualization**
ROLL NO 1	18BCE150
NAME 1	Parth Desai
ROLL NO 2	18BCE135
NAME 2	Nirav Madhani
ROLL NO 3	18BCE147
NAME 3	Parth Panchal

PROJECT DESCRIPTION

It is a web application for managing your investment portfolio. Users can get real-time stock data and market news via Yahoo Finance, IEX Cloud, and Quandl APIs. They can also compare the performance of the stocks with charts and predict the future behaviour for the same. Users are given the ability to create a customizable stock portfolio and watchlist to track both current and prospective holdings. The application uses RESTful Web Services to pull live/historical stock data.

You can easily search tickers through the integrated API platform provided by iexcloud.io as well as easily add, edit or delete tickers based on your preferences.

Displays accurate and real-time stock market data for each individual stock in the portfolio. Individual stock details include:

- Basic information includes open price, market cap, volume, EPS, etc.
- Financial Ratios include P/E ratio, Price/Book ratio, short ratio, etc.
- Real Time Data include real time market cap, ask price, bid price, etc.
- Changes and Trends include 50 days, 200 days, and yearly moving average

- **Authentication**: System can handle multiple users and provide reasonable level of security
- **Multiple Portfolios**: Users can own multiple portfolios

Technology used:

- Django
- Django Rest Framework
- Back end language: Python
- SQLite
- Bootstrap
- HTML/CSS

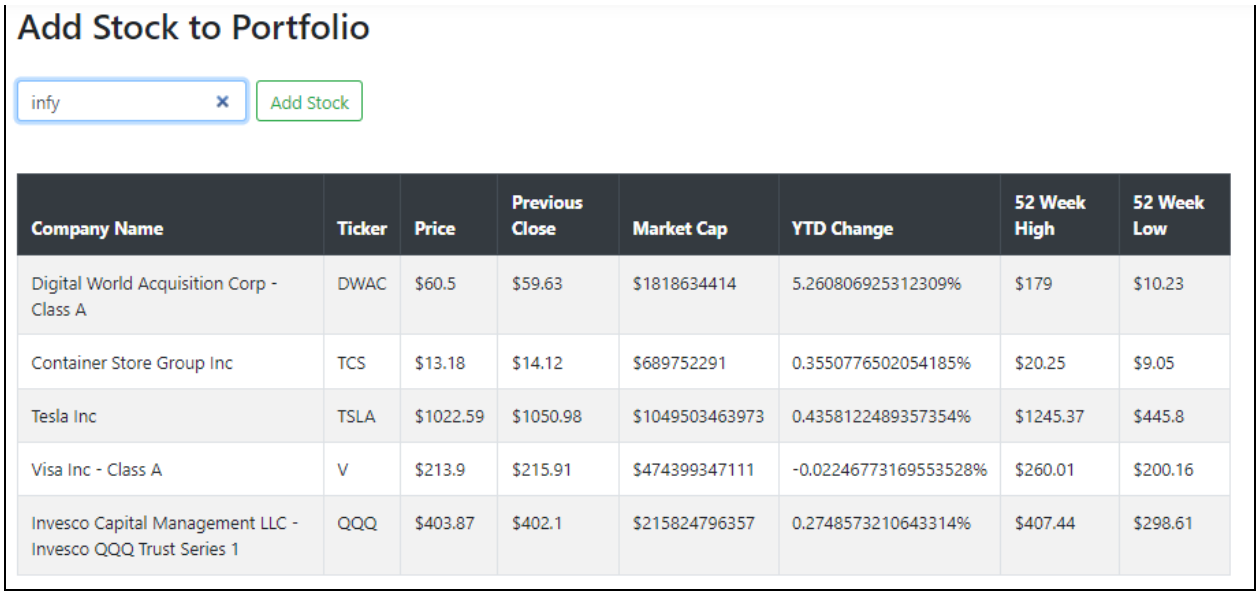
PROJECT FEATURES IN DETAILS WITH CODE FROM EACH FILE

We have explained the features of our web app in the form of navigation bar titles that are displayed on our site for the sake of simplicity.

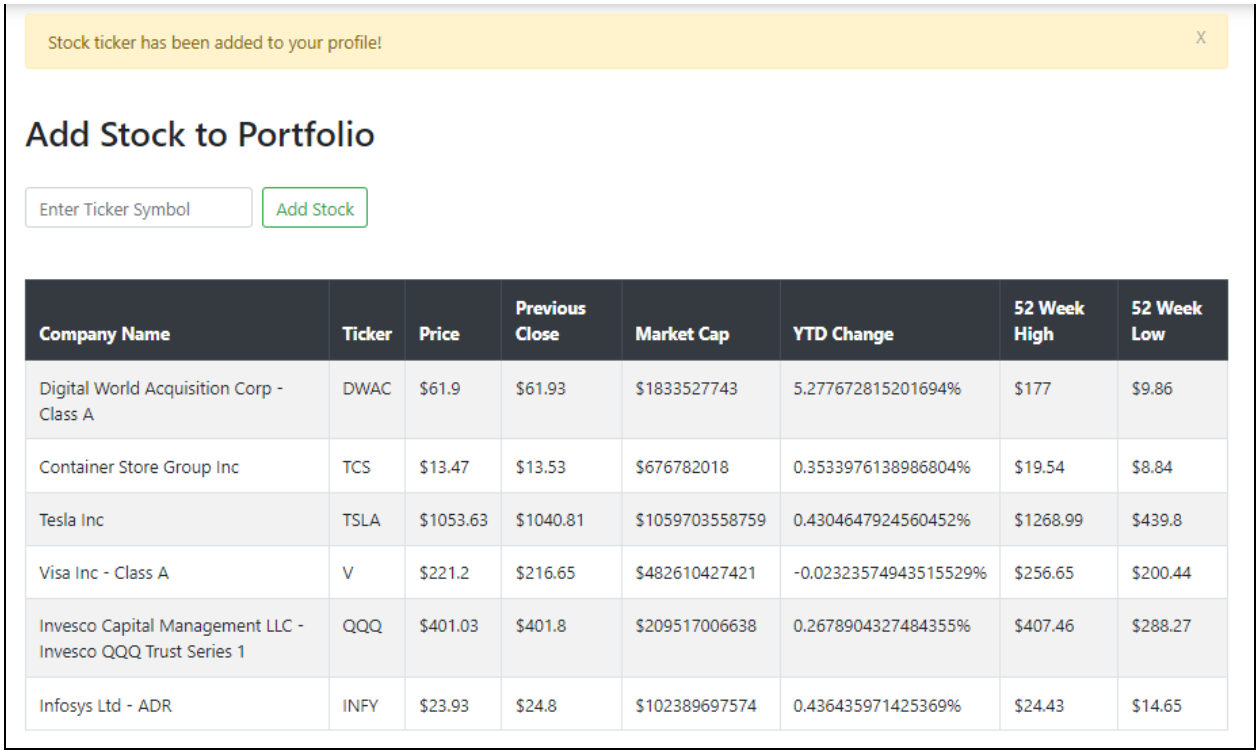
Add Stock

The add stock page enables the user to add a stock ticker of his/her choice to their account portfolio. It utilizes a database function to store the ticker symbols when requested as a way of developing a personal portfolio. It is a page designed to show the tickers in the user's portfolio and allows them to delete them individually as the investment strategy changes over time.

Attempting to add stock with ticker symbol 'infy' to the portfolio



Stock for Infosys Ltd was successfully added to the portfolio.



Code – add_stock.html

```
{% extends 'base.html' %}
{% block content %}
```

```
<h2> Add Stock to Portfolio </h2>
<br/>
<form action="{% url 'add_stock' %}" class="form-inline my-2 my-lg-0"
method="POST">
    {% csrf_token %}
  <input id="input1" class="form-control mr-sm-2" type="search"
placeholder="Enter Ticker Symbol" aria-label="Search" name="ticker"
  onClick='document.getElementById("myDropdown1").classList.toggle("show");'
onkeyup="filter('myDropdown1','input1')">

  <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Add
Stock</button>

</form>
<div class="dropdown-content" id="myDropdown1">

      </div>
<br/><br/>

<table class="table table-striped table-bordered table-hover">
  <thead class="thead-dark">
    <tr>
      <th scope="col">Company Name</th>
      <th scope="col">Ticker</th>
      <th scope="col">Price</th>
      <th scope="col">Previous Close</th>
      <th scope="col">Market Cap</th>
      <th scope="col">YTD Change</th>
      <th scope="col">52 Week High</th>
      <th scope="col">52 Week Low</th>
    </tr>
  </thead>
  <tbody>
{% if ticker %}
            {% for list_item in output %}
              <tr>
              <td scope="row">{{ list_item.companyName }}</th>
              <td>{{ list_item.symbol }}</td>
              <td>${{ list_item.latestPrice }}</td>
```

```
              <td>${{ list_item.previousClose }}</td>
              <td>${{ list_item.marketCap }}</td>
              <td>{{ list_item.ytdChange }}%</td>
              <td>${{ list_item.week52High }}</td>
              <td>${{ list_item.week52Low }}</td>
              </tr>
            {% endfor %}
{% endif %}
  </tbody>
</table>
<br><br>

{% for item in ticker.keys %}
    <a href="{% url 'delete' item %}" class="btn btn-danger">Delete {{ item
}}</a>
         
{% endfor %}
{% endblock %}
```

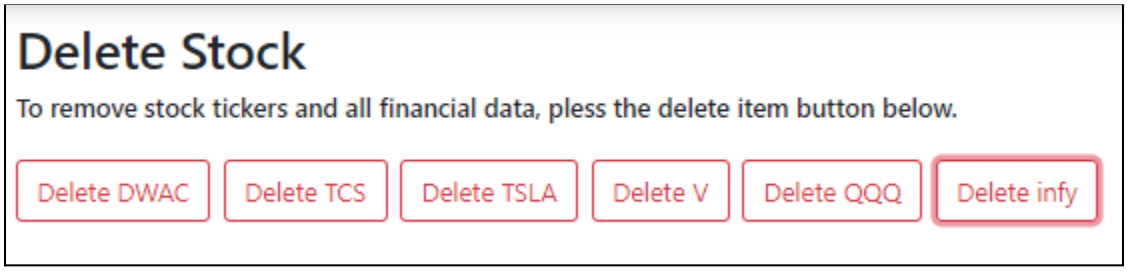
Code - add_stock() in views.py

```
@login_required
def add_stock(request):
    if request.method == 'POST':
        form = StockForm(request.POST or None)
        if form.is_valid():
            obj = form.save(commit=False)
            obj.user = request.user
            obj.save()
            messages.success(request, ("Stock ticker has been added to your
profile!"))
            return redirect('add_stock')
    else:
        if request.user.is_anonymous:
            ticker = []
        else:
            ticker = Stock.objects.filter(user=request.user)
        output = []
```

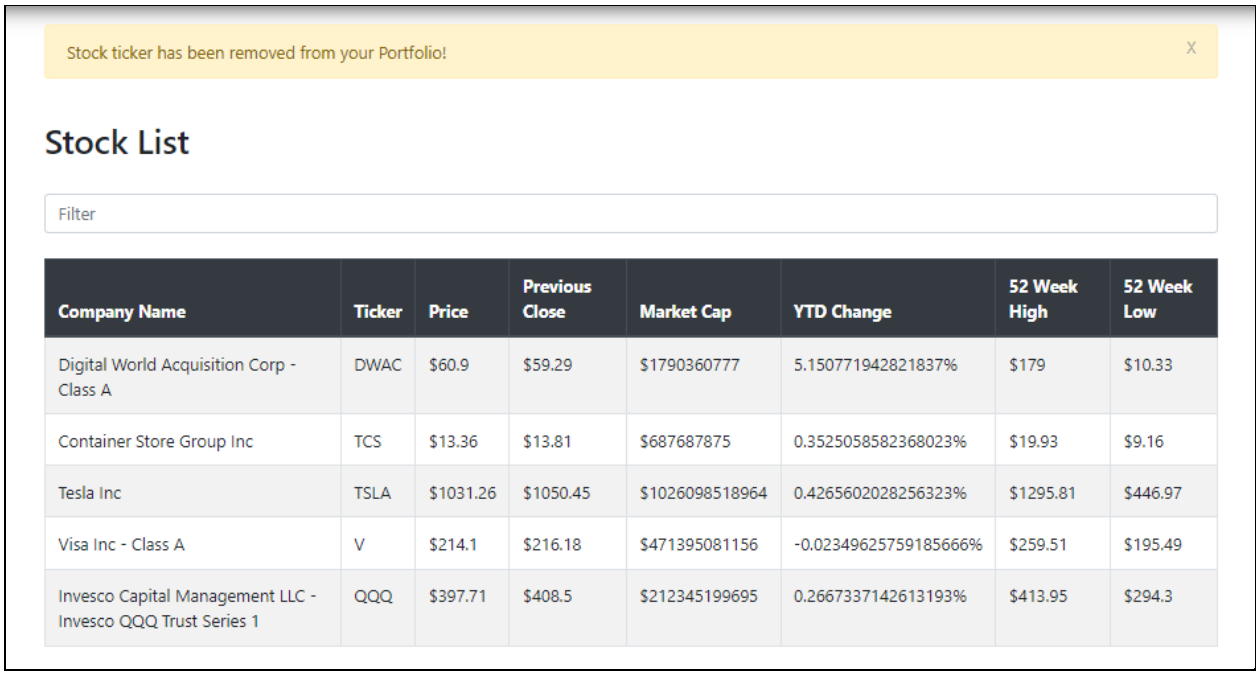
```
        for ticker_item in ticker:
            api_request =
requests.get("https://sandbox.iexapis.com/stable/stock/" + str(ticker_item) +
"/quote?token=Tpk_c46f4087296c43358402984f3b26ed2f")
            # for error handling
            try:
                api = json.loads(api_request.content)
                output.append(api)
            except Exception as e:
                api = "Sorry there is an error"
        return render(request, 'add_stock.html', {'ticker': ticker, 'output':
output})
```

Delete Stock

The delete stock function is used to remove any stock from the user's portfolio in case he/she is no longer interested in trading that particular stock. It also includes the functionality which helps return the rendered webpage which enables the user to delete the stock.



Stock with ticker symbol infy deleted successfully.



Code – delete_stock.html

```
{% extends 'base.html' %}
{% block content %}
<h2> Delete Stock </h2>
<h6> To remove stock tickers and all financial data, pless the delete item button
below.</h6>
<div class="row container">
  {% if ticker %}
      {% for item in ticker %}
          <a href="{% url 'delete' item.ticker %}" class="btn btn-outline-danger
mr-2 my-3">Delete {{ item }}</a><br><br>
      {% endfor %}
  {% else %}
      Your portfolio appears to be empty.
  {% endif %}
</div>
{% endblock %}
```

Code – delete() in views.py

```
@login_required
```

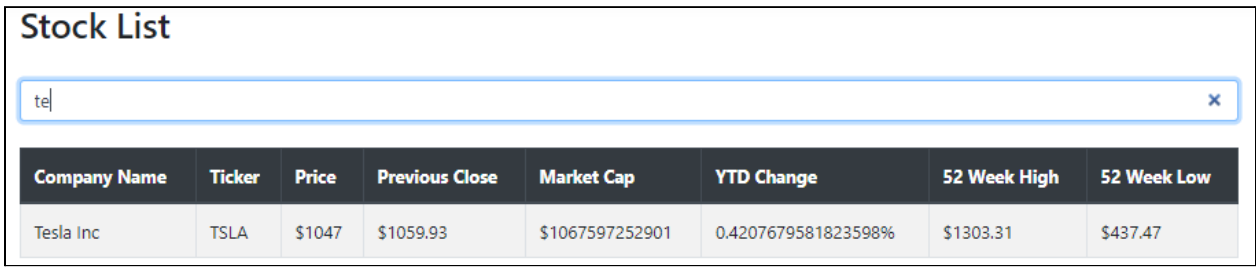
```
def delete(request, stock_id):
    item = Stock.objects.get(ticker=stock_id,user=request.user)
    item.delete()
    messages.success(request, ("Stock ticker has been removed from your
Portfolio!"))
    return redirect(list_stock)
```

Code - delete_stock() in views.py

```
@login_required
def delete_stock(request):
    ticker = Stock.objects.filter(user=request.user)
    return render(request, 'delete_stock.html', {'ticker': ticker})
```

List Stock

The list section shows the different stocks that the user has added to his portfolio. It gives a tabular representation of all the important details related to each stock, like the symbol, company name, its price, previous close, market cap, return YTD, PE Ratio, 52Wk High, and 52 week low. In addition to this, it also allows the user to search for stocks he/she has added, and filter the view accordingly.



Code - stockList.html

```
{% extends 'base.html' %}

{% block content %}
<h2>Stock List</h2>
<br>
<div>
```

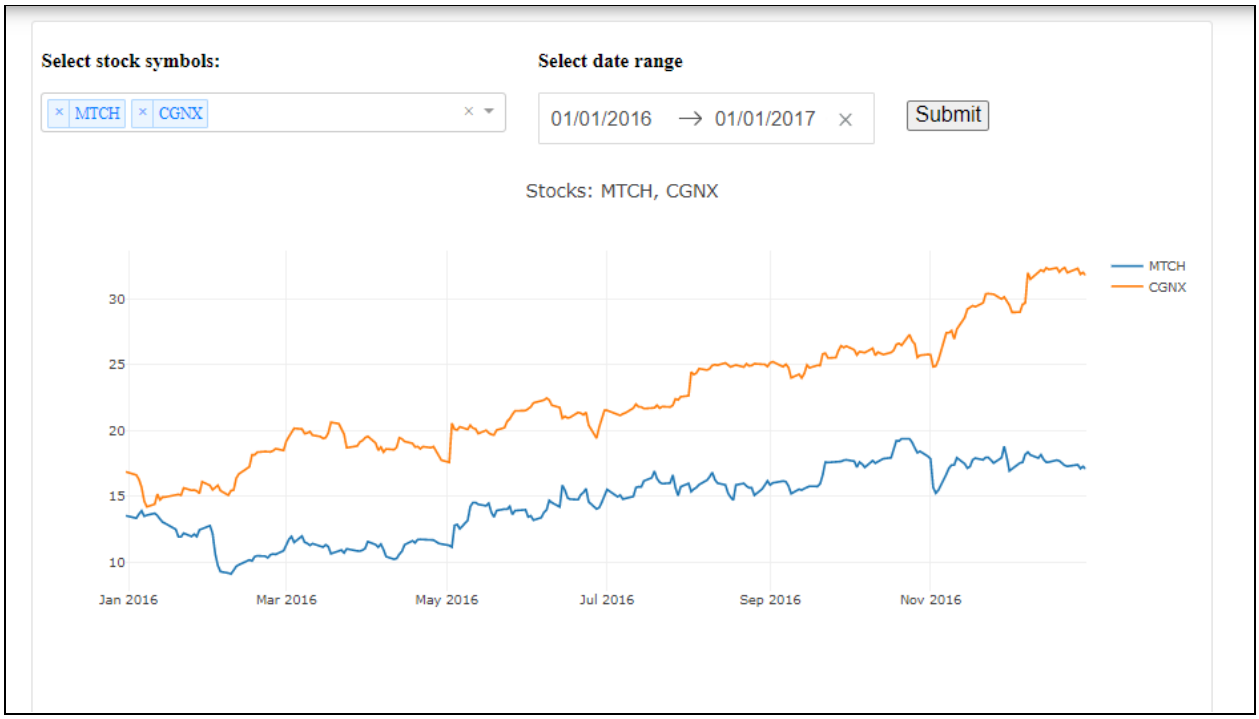
```
<input id="listSearch" class="form-control mr-sm-2" type="search"
placeholder="Filter" aria-label="Search" name="Filter"
onkeyup="myFunction()"
  ></div>
<br>
<table class="table table-striped table-bordered table-hover" id="myTable">
  <thead class="thead-dark">
    <tr>
      <th scope="col">Company Name</th>
      <th scope="col">Ticker</th>
      <th scope="col">Price</th>
      <th scope="col">Previous Close</th>
      <th scope="col">Market Cap</th>
      <th scope="col">YTD Change</th>
      <th scope="col">52 Week High</th>
      <th scope="col">52 Week Low</th>
    </tr>
  </thead>
  <tbody>
    {% if ticker %}
      {% for list_item in output %}
        <tr>
        <td scope="row">{{ list_item.companyName }}</th>
        <td>{{ list_item.symbol }}</td>
        <td>${{ list_item.latestPrice }}</td>
        <td>${{ list_item.previousClose }}</td>
        <td>${{ list_item.marketCap }}</td>
        <td>{{ list_item.ytdChange }}%</td>
        <td>${{ list_item.week52High }}</td>
        <td>${{ list_item.week52Low }}</td>
        </tr>
      {% endfor %}
  </tbody>
</table>
{% endif %}

{% endblock %}
```

Code - list_stock in views.py

```
@login_required
def list_stock(request):
    if request.user.is_anonymous:
        ticker = []
    else:
        ticker = Stock.objects.filter(user=request.user)
    output = []
    for ticker_item in ticker:
        api_request = requests.get("https://sandbox.iexapis.com/stable/stock/" +
str(ticker_item) + "/quote?token=Tpk_c46f4087296c43358402984f3b26ed2f")
        # for error handling
        try:
            api = json.loads(api_request.content)
            output.append(api)
        except Exception as e:
            api = "Sorry there is an error"
    return render(request, 'stockList.html', {'ticker': ticker, 'output':
output})
```

View Graph

The viewgraph functionality is a very useful tool for the user. It enables the user to view and analyze trends for any stock right here in the app, without the need to go to another site for the same. It is a very extensive tool and we have enabled this with the help of a plotly graph called stock-graphic, which is a stock visualization graph. It allows the user to view trends of multiple stocks at the same time, within any given date range.



Code – stocks.html

```
{% extends 'base.html' %}
{% load static %}
{% block content %}
    {% load plotly_dash %}
    <div class="{% plotly_class name='stock-graphic' %} card" style="height:
100%; width: 100%">
    {% plotly_app name='stock-graphic' ratio=1 %}
    </div>
<br>
{% endblock %}
```

Code – view_stock in views.py

```
@csrf_exempt
def view_stock(request):
    return render(request, 'stocks.html', {})
```

Code – stock_graphic.py

```
import dash_core_components as dcc
```

```
import dash_html_components as html
from dash.dependencies import Input, Output, State
import pandas as pd
from datetime import date, datetime
import pandas_datareader.data as web
from django_plotly_dash import DjangoDash

# File contains Stock tickers for all NASDAQ symbols
nasdaq = pd.read_csv('data/NASDAQcompanylist.csv')

# Plotly Dash App applied within Django

app = DjangoDash('stock-graphic')

app.layout = html.Div([

    # First row with inputs and button

    html.Div([
        html.H3('Select stock symbols:', style={'paddingRight': '50px'}),
        dcc.Dropdown(id='stock-dropdown',
                     options=[dict(label=ticker, value=ticker) for ticker in
nasdaq['Symbol']],
                     multi=True,),
    ], style=dict(display='inline-block',
                  verticalAlign='top',
                  width='40%',)),

    html.Div([
        html.H3('Select date range'),
        dcc.DatePickerRange(
            id='date-range',
            min_date_allowed=datetime(2015, 1, 1),
            max_date_allowed=datetime.today(),
            minimum_nights=30,
            clearable=True,
            with_portal=True,
            start_date=date(2016, 1, 1),
            end_date=datetime.today()
```

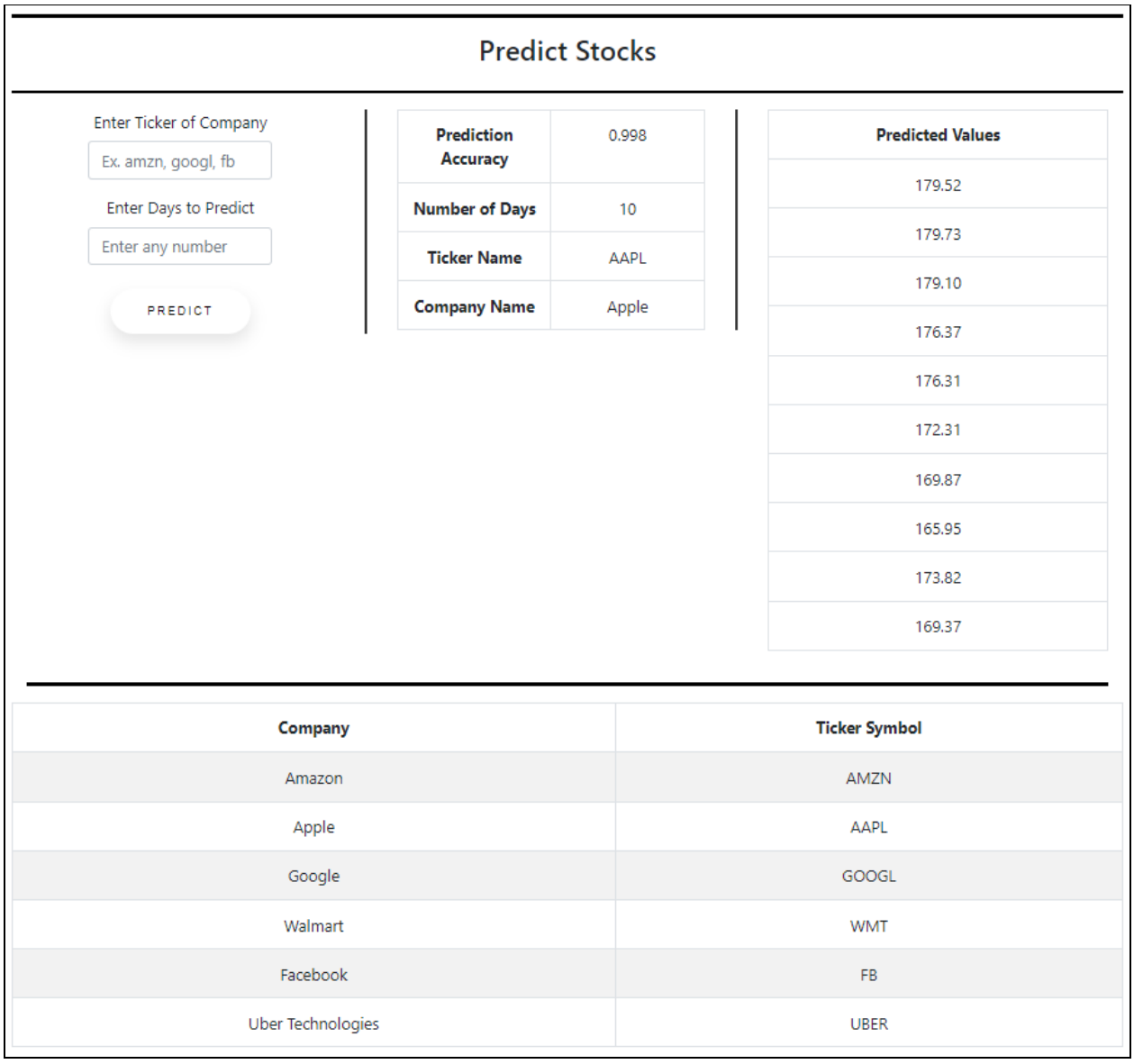
```
        )], style={'display':'inline-block', 'marginLeft': '30px'}),

    html.Div([
        html.Button(
            id='submit-button',
            n_clicks=0,
            children='Submit',
            style={'fontSize': 20, 'marginLeft': '30px'}
        ),
    ], style={'display': 'inline-block'}),


    # Second row contains share graphic only

    html.Div(
        dcc.Graph(id='display-graphic',
                  figure=dict(data=[dict(x=[0, 1],
                                         y=[0, 1])],
                              layout=dict(
                                  height=500,
                                  # title='Default',
                                  markers='closest')
                              ))
    )
])


@app.callback(Output('display-graphic', 'figure'),
              [Input('submit-button', 'n_clicks')],
              [State('stock-dropdown', 'value'),
               State('date-range', 'start_date'),
               State('date-range', 'end_date')])
def update_graph(n_clicks, ticker, start_date, end_date):

    """Retrieves stock price history for all tickers from Yahoo using the Pandas
module."""

    data = []
    graph_title = "Stocks: "
```

```
for tick in ticker:
    graph_title = graph_title + tick + ", "
    df = (web.DataReader(tick,
                         start=start_date,
                        end=end_date,
                        data_source='yahoo'))

    data.append(dict(x=df.index,
                     y=df['Close'],
                    name=tick))

graph_title = graph_title[:-2]

figure = dict(data=data,
              layout=dict(title=f"{graph_title}"))
return figure
```

Predict Stock

The predict stock page is a tool that serves as a means for the user to predict the future prices of the required stocks. It uses a simple linear regression algorithm that makes predictions based on the previous prices of the said stock. The tool takes the ticker of the stock and the number of days to predict for as the input, and then gives out the predicted values and the accuracy of those values, as the output.

Predicting the prices of AAPL for the next 10 days:



Code – predict_stock.html

```
<body>
    <div class="row">
        <div class="col text-center">
            <hr />
            <h3>
                Predict Stocks
            </h3>
        </div>
    </div>
```

```
    <hr>
    <div class="container">
        <div class="row">
            <div class="col text-center">
                <div class="mycontent-left">
                    <form action="/predict/" method="post">{% csrf_token %}
                        <div class="form-group">
                            <div>
                                <label for="exampleInputEmail1">Enter Ticker of
Company</label>
                                <div align="center">
                                    <input type="text" class="form-control"
id="exampleInputEmail1" placeholder="Ex. amzn, googl, fb" name="ticker"
style="width:60%" required>
                                </div>
                            </div>
                        </div>
                        <div class="form-group">
                            <label for="exampleInputEmail2">Enter Days to
Predict</label>
                            <div>
                                <div align="center">
                                    <input type="text" class="form-control"
id="exampleInputEmail2" placeholder="Enter any number" name="days"
style="width:60%" required>
                                </div>
                            </div>
                            <br>
                            <div class="wrap">
                                <button type="submit" value="Predict"
class="button">Predict</button>
                            </div>
                        </div>
                    </form>
                </div>
            </div>
            <div class="col text-center">
                <div class="mycontent-left">
```

```
                    <table align='center' class="table table-bordered
table-hover">
                        <tbody>
                            <tr>
                                <td style="width:50%"><strong>Prediction
Accuracy</strong></td>
                                <td style="width:50%">{{
confidence|floatformat:-3 }}</td>
                            </tr>
                            <tr>
                                <td style="width:50%"><strong>Number of
Days</strong></td>
                                <td style="width:50%">{{ number_of_days }}</td>
                            </tr>
                            <tr>
                                <td style="width:50%"><strong>Ticker
Name</strong></td>
                                <td style="width:50%">{% if ticker_value ==
"aapl" %}AAPL{% endif %}{% if ticker_value == "amzn" %}AMZN{% endif %}{% if
ticker_value == "googl" %}GOOGL{% endif %}{% if ticker_value == "wmt" %}WMT{%
endif %}{% if ticker_value == "fb" %}FB{% endif %}{% if ticker_value == "uber"
%}UBER{% endif %}</td>
                            </tr>
                            <tr>
                                <td style="width:50%"><strong>Company
Name</strong></td>
                                <td style="width:50%">{% if ticker_value ==
"aapl" %}Apple{% endif %}{% if ticker_value == "amzn" %}Amazon{% endif %}{% if
ticker_value == "googl" %}Google{% endif %}{% if ticker_value == "wmt"
%}Walmart{% endif %}{% if ticker_value == "fb" %}Meta{% endif %}{% if
ticker_value == "uber" %}Uber{% endif %}</td>
                            </tr>
                        </tbody>
                    </table>
                </div>
            </div>
            <div class="col text-center">
                <table align='center' class="table table-bordered table-hover">
                    <tbody>
```

```
                        <tr>
                            <td style="width:50%"><strong>Predicted
Values</strong></td>
                        </tr>
                        {% for forecast in forecast %}
                        <tr>
                            <td style="width:50%">{{ forecast|floatformat:-2
}}</td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
        <hr />
        <div class="row">
            <table class="table table-striped table-bordered table-hover">
                <thead>
                    <tr>
                        <th>Company</th>
                        <th>Ticker Symbol</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>Amazon</td>
                        <td>AMZN</td>
                    </tr>
                    <tr>
                        <td>Apple</td>
                        <td>AAPL</td>
                    </tr>
                    <tr>
                        <td>Google</td>
                        <td>GOOGL</td>
                    </tr>
                    <tr>
                        <td>Walmart</td>
                        <td>WMT</td>
```

```
                    </tr>
                    <tr>
                        <td>Facebook</td>
                        <td>FB</td>
                    </tr>
                    <tr>
                        <td>Uber Technologies</td>
                        <td>UBER</td>
                    </tr>
                </tbody>
            </table>
        </div>
    </div>
</body>
{% endblock %}
```

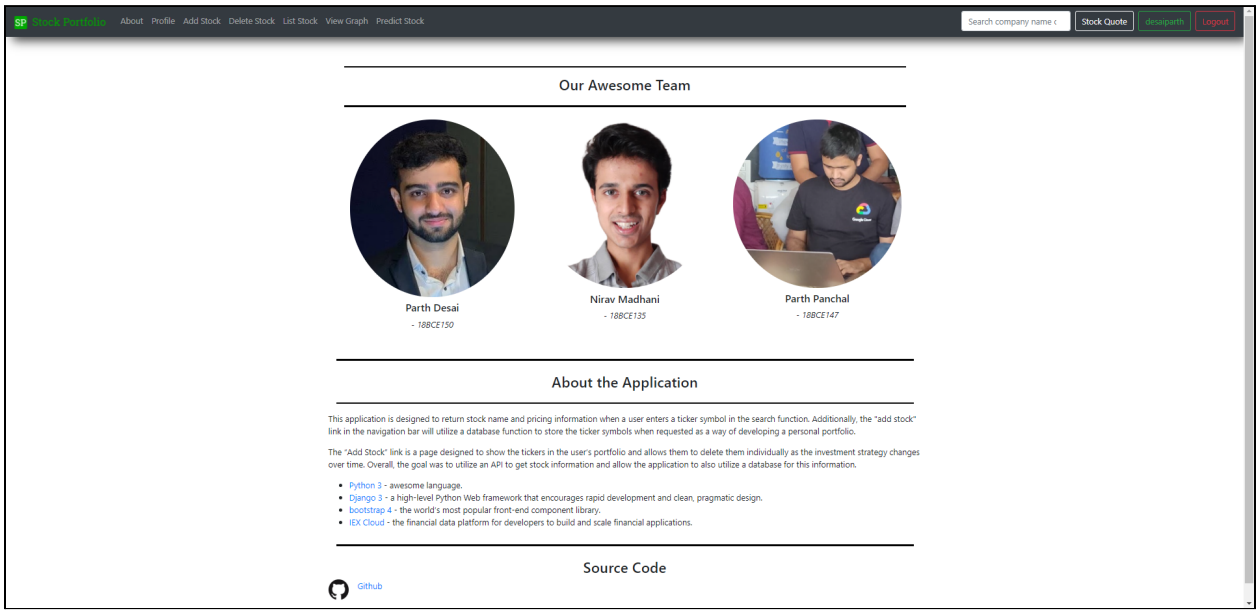
Code - predict() in views.py

```
def predict(request):
    # Quandl API key. Create your own key via registering at quandl.com
    quandl.ApiConfig.api_key = "RHVBxuQQR_xxy8SPBDGV"
    # Getting input from Templates for ticker_value and number_of_days
    ticker_value = request.POST.get('ticker')
    number_of_days = request.POST.get('days')
    number_of_days = int(number_of_days)
    # Fetching ticker values from Quandl API
    df = quandl.get("WIKI/"+ticker_value+"")
    df = df[['Adj. Close']]
    forecast_out = int(number_of_days)
    df['Prediction'] = df[['Adj. Close']].shift(-forecast_out)
    # Splitting data for Test and Train
    X = np.array(df.drop(['Prediction'],1))
    X = preprocessing.scale(X)
    X_forecast = X[-forecast_out:]
    X = X[:-forecast_out]
    y = np.array(df['Prediction'])
    y = y[:-forecast_out]
    X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y,
test_size = 0.2)
```

```
    # Applying Linear Regression
    clf = LinearRegression()
    clf.fit(X_train,y_train)
    # Prediction Score
    confidence = clf.score(X_test, y_test)
    # Predicting for 'n' days stock data
    forecast_prediction = clf.predict(X_forecast)
    forecast = forecast_prediction.tolist()
    return render(request,'predict_stock.html',{'confidence' :
confidence,'forecast':
forecast,'ticker_value':ticker_value,'number_of_days':number_of_days})
```

About

This page displays the details of our team members and gives a brief description of our project and its functionalities.



Code - about.html

```
{% extends 'base.html' %}
{% block title %}Our Team{% endblock %}
{% block content %}
<style>
  hr {
        position: relative;
```

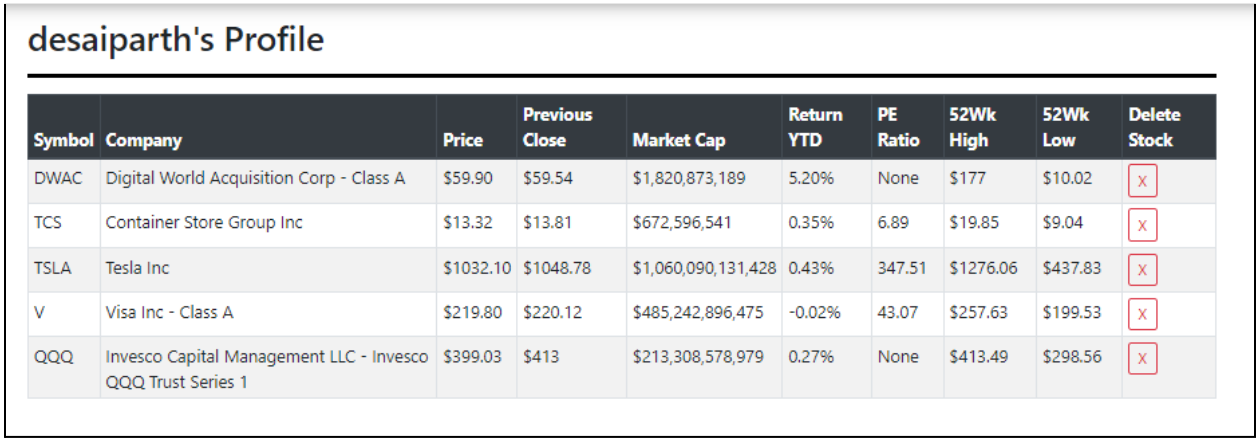
```
        border: none;
        height: 3px;
        background: black;
    }
</style>
<section class="pt-3 pb-4">

      <div class="container">
          <div class="row">
            <div class="col text-center">
            <hr />
              <h3>
                Our Awesome Team
              </h3>
            </div>
          </div>
          <hr>
          <div class="row">
            <div class="col-md-4 text-center">
            {% load static %}
                <img class = "img-fluid feature-img p-2 rounded-circle" src="{%
static 'quotes/profile.jpg' %}" alt="Parth Desai">
              <h5>
                Parth Desai
              </h5>
              <p>
                <i>- 18BCE150</i>
              </p>

            </div>
            <div class="col-md-4 text-center">
              {% load static %}
                <img class = "img-fluid feature-img p-2 rounded-circle" src="{%
static 'quotes/2.png' %}" alt="Nirav Madhani">
              <h5>
                Nirav Madhani
              </h5>
              <p>
                <i>- 18BCE135</i>
              </p>

            </div>
            <div class="col-md-4 text-center">

                {% load static %}
                <img class = "img-fluid feature-img p-2 rounded-circle" src="{%
static 'quotes/3.jpeg' %}" alt="Parth Panchal">
                <h5>
                  Parth Panchal
                </h5>
                <p>
                  <i>- 18BCE147</i>
```

```
              </p>
            </div>
          </div>
  </section>
  <hr />
  <div class="row mt-4">
    <div class="col text-center">
      <h3>
        About the Application
      </h3>
    </div>
  </div>
  <hr />
  <div class="row">
    <p>This application is designed to return stock name and pricing information
when a user enters a ticker symbol in the search function. Additionally, the "add
stock" link in the navigation bar will utilize a database function to store the
ticker symbols when requested as a way of developing a personal portfolio.</p>

    <p>The “Add Stock” link is a page designed to show the tickers in the user’s
portfolio and allows them to delete them individually as the investment strategy
changes over time. Overall, the goal was to utilize an API to get stock
information and allow the application to also utilize a database for this
information.</p>
    <br />
    <ul>
    <li><a href="https://www.python.org/">Python 3</a> - awesome language.</li>
    <li><a href="https://www.djangoproject.com/">Django 3</a> - a high-level
Python Web framework that encourages rapid development and clean, pragmatic
design.</li>
    <li><a href="https://getbootstrap.com/">bootstrap 4</a> - the world’s most
popular front-end component library.</li>
    <li><a href="https://iexcloud.io/">IEX Cloud</a> - the financial data
platform for developers to build and scale financial applications.</li>
    </ul>
    <br />
  </div>
  <hr />
  <div class="row mt-4">
    <div class="col text-center">
      <h3>
        Source Code
      </h3>
    </div>
  </div>
  <div class="row">
  <a href="https://github.com/Nirav-Madhani/ADF_Project"><img style="width: 35%;
float: left;" src="{% static 'quotes/GitHub-Mark-64px.png' %}" alt="Github
Profile"><p style="width: 50%; float: right;">Github</p></a>
  </div>
  <br>
  <br>
```

```
</section>
{% endblock %}
```

Profile

The profile section shows the different stocks that the user has added to his portfolio. It gives a tabular representation of all the important details related to each stock, like the symbol, company name, its price, previous close, market cap, return YTD, PE Ratio, 52Wk High, and 52 week low. It also gives the user the easy option of a quick-delete button, where he/she can remove a stock from the profile page itself, if required.



desaiparth's Profile

Symbol	Company	Price	Previous Close	Market Cap	Return YTD	PE Ratio	52Wk High	52Wk Low	Delete Stock
DWAC	Digital World Acquisition Corp - Class A	$59.90	$59.54	$1,820,873,189	5.20%	None	$177	$10.02	X
TCS	Container Store Group Inc	$13.32	$13.81	$672,596,541	0.35%	6.89	$19.85	$9.04	X
TSLA	Tesla Inc	$1032.10	$1048.78	$1,060,090,131,428	0.43%	347.51	$1276.06	$437.83	X
V	Visa Inc - Class A	$219.80	$220.12	$485,242,896,475	-0.02%	43.07	$257.63	$199.53	X
QQQ	Invesco Capital Management LLC - Invesco QQQ Trust Series 1	$399.03	$413	$213,308,578,979	0.27%	None	$413.49	$298.56	X

Code - profile.html

```
{% extends 'base.html' %}
{% load humanize %}

{% block title %}Profile{% endblock %}

{% block content %}
<style>
hr {
        position: relative;
        border: none;
        height: 3px;
        background: black;
    }
</style>
```

```
<h2>{{ user.username }}'s Profile</h2>
<hr />
<table class="table table-striped table-bordered table-hover table-sm">
  <thead class="thead-dark">
    <tr>
        <th scope="col">Symbol</th>
        <th scope="col">Company</th>
        <th scope="col">Price</th>
        <th scope="col">Previous Close</th>
        <th scope="col">Market Cap</th>
        <th scope="col">Return YTD</th>
        <th scope="col">PE Ratio</th>
        <th scope="col">52Wk High</th>
        <th scope="col">52Wk Low</th>
        <th scope="col">Delete Stock</th>
    </tr>
  </thead>
  <tbody>
    {% if ticker %}
        {% for list_item in output %}
            <tr>
            <td scope="row">{{ list_item.symbol }}</td>
            <td>{{ list_item.companyName }}</th>
            <td>${{ list_item.latestPrice |floatformat:-2 }}</td>
            <td>${{ list_item.previousClose |floatformat:-2 }}</td>
            <td>${{ list_item.marketCap |intcomma }}</td>
            <td>{{ list_item.ytdChange | floatformat:-2 }}%</td>
            <td>{{ list_item.peRatio }}</td>
            <td>${{ list_item.week52High }}</td>
            <td>${{ list_item.week52Low }}</td>
            <td><a href="/delete/{{list_item.symbol}}" class="btn
btn-outline-danger btn-sm">X</a></td>
            </tr>
        {% endfor %}
    {% endif %}
  </tbody>
</table>
<br><br>
{% endblock %}
```

Code - profile() in views.py

```
@login_required
def profile(request):
    if request.method == 'POST':
        form = StockForm(request.POST or None)
        if form.is_valid():
            form.save()
            messages.success(request, ("Stock ticker has been added to your
Portfolio!"))
            return redirect('add_stock')
    else:
        ticker = Stock.objects.filter(user=request.user)
        output = []
        for ticker_item in ticker:
            api_request =
requests.get("https://sandbox.iexapis.com/stable/stock/" + str(ticker_item) +
"/quote?token=Tpk_c46f4087296c43358402984f3b26ed2f")
            # for error handling
            try:
                api = json.loads(api_request.content)
                output.append(api)
            except Exception as e:
                api = "Sorry there is an error"
        return render(request, 'profile.html', {'ticker': ticker, 'output':
output})
```

DJANGO CONCEPTS UTILIZED

- Created different Django views
- URL Mapping
- Implemented different templates
- Utilized built-in and custom tags and filters
- Using templates in views
- Template Inheritance
- Assets handing

- Crispy Forms
- Form rendering process
- Used build-in and custom widgets
- Installation and Configuration of Database
- Defining personal model
- Database CRUD operations
- Rendering model in Admin interface
- Fields validation
- Customizing Authentication
- Admin Interface and its control
- User and group creation
- Permission Handling
- Utilized Django Rest Framework
- Created graphs in Django with the help of Plotly Dash

CHALLENGES

- Researching about different investment strategies and gaining the required financial knowledge to carry out the project
- Exploring the free APIs that can provide historical and real time stock data.
- Preparing the chart data to give to Django Plotly Dash