

STOCK MONITOR WITH REDIS PROJECT

AIM: Stock Monitor with Redis Project Requirement: In a flask, the user will be provided stock, and once provided current value, date and time has to be recorded to the Redis store. The counter should check every 5 mins once the stock value and store it. A table format of the Redis store with all the timestamps has to be displayed. A line graph will be an option but nice to have on the same page if you have time.

What is Flask?

Flask is a web framework, it's a python module that lets you develop web applications easily. It is developed by **Armin Ronacher** who leads an international group of Python enthusiasts(POCCO). It has a small and easy-to-extend core: it's a microframework that doesn't include an ORM(Object Relational Manager) or such features.

It does have many cool features like URL routing, and a template engine. It is a WSGI(Web Server Gateway Interface) web app framework.

What is WSGI?

It is an acronym for web server gateway interface which is a standard for Python Web Application development. It is considered the specifications for the universal interface between the Web Server and Web Application.

What is Redis?

Redis stands for Remote Dictionary server and it is a fast and open source in memory, it is a key-value data store. Redis is an open-source (BSD licensed), in-memory data store used as a database, cache, message, and streaming engine.

Redis supports most leading programming languages and protocols, including Python, Java, PHP, Go, Ruby, C/C#/C++, JavaScript, Node.js, etc.

- First we need to install some packages.
 - Flask
 - Jinja2
 - Redis
 - Yfinance
 - Pydantic
 - JsonModel
 - PositiveInt

What is Jinja2?

Jinja2 is a web template engine that combines templates with a certain data source to render the dynamic web pages.

Flask Environment Setup:

To install a flask on the system, we need to have python 2.7 or higher installed on our system.

What is yfinance?

The yfinance is one of the famous modules in python, which is used to collect online data, and with it, we can collect the financial data of Yahoo. With the help of the yfinance module, we retrieve and collect the company's financial information(such as financial ratios, etc.)as well as the histories of marketing data by using its functions. We have to install the yfinance module in our system (as it is not a built-in module in python). Once the installation process is done, we will proceed to the implementation part of the yfinance module.

APPLICATIONS:

- Linux
- Pycharm
- Notepad
- Terminal
- Microsoft Edge

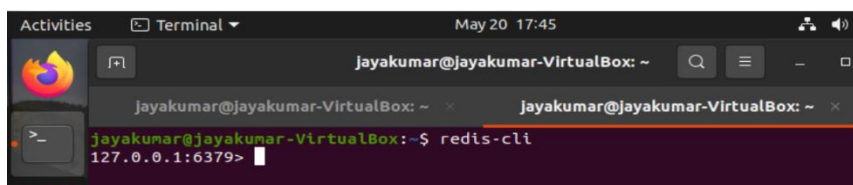
Steps to Install Required Packages:

- There are two ways for installing packages
 1. By using Terminal: Pip install **Required package name**
 2. By using IDLE:
 - (i) First create a project in IDLE
 - (ii) Click on File>>> Go to Settings>>> Open Project>>> Open project interpreter and install packages

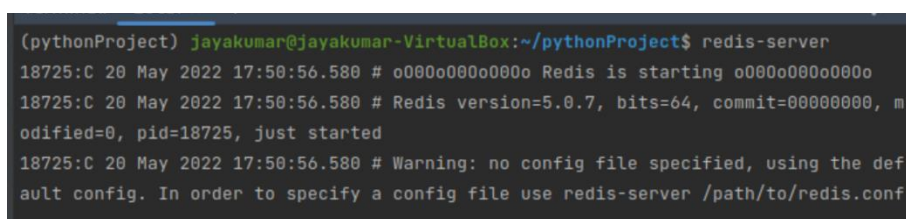
How to install Redis:

By using Terminal: Pip Install Redis

After installing the Redis, activate Redis by using this command: Redis-cli

A screenshot of a terminal window titled 'Terminal' with a timestamp of 'May 20 17:45'. The terminal shows the user 'jayakumar@jayakumar-VirtualBox: ~' and the command 'redis-cli' being executed. The prompt '127.0.0.1:6379>' is visible below the command.

How to connect to the server: Redis-server

A screenshot of a terminal window showing the output of the 'redis-server' command. The output includes the Redis version (5.0.7), the PID (18725), and a warning that no config file was specified, so the default config is being used. The text is as follows:
(pythonProject) jayakumar@jayakumar-VirtualBox:~/pythonProject\$ redis-server
18725:C 20 May 2022 17:50:56.580 # 000000000000 Redis is starting 000000000000
18725:C 20 May 2022 17:50:56.580 # Redis version=5.0.7, bits=64, commit=00000000, modified=0, pid=18725, just started
18725:C 20 May 2022 17:50:56.580 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf

Python Script:

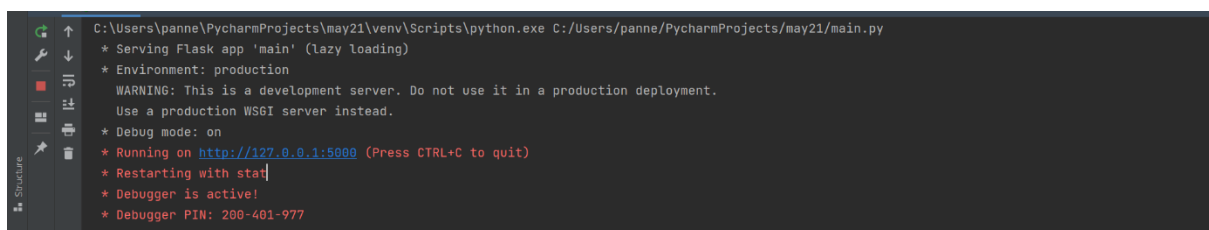
Input:

```
#create Stock Market Web App using Flask
```

```
#import Required packages
```

```
import redis
from flask import Flask, render_template
db = redis.StrictRedis(host='localhost', port='6379', db=0, charset='utf-8', decode_responses=True)
flaskapp = Flask(__name__)
@flaskapp.route('/')
def home():
    return render_template("base.html")
from pydantic import PositiveInt
from redis_om import JsonModel, Field
class Item(JsonModel):
    name: str = Field(index=True)
    price: PositiveInt = Field(index=True)
    barcode: PositiveInt = Field(index=True)
    description: str = Field(index=True)
@flaskapp.route("/stocks")
def stocks_page():
    item = Item.find().all()
    return render_template('stocks.html', items=item)
if __name__ == "__main__":
    flaskapp.run(debug=True)
```

Output:



```
C:\Users\panne\PycharmProjects\may21\venv\Scripts\python.exe C:/Users/panne/PycharmProjects/may21/main.py
* Serving Flask app 'main' (Lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 200-401-977
```

```
C:\Users\panne\PycharmProjects\pythonProject6\venv\Scripts\python.exe C:/Users/panne/PycharmProjects/pythonProject6/main.py
```

```
* Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
```

```
* Serving Flask app 'main' (lazy loading)
```

```
* Environment: production WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
```

```
* Debug mode: off
```

```
URL / Server URL : http://127.0.0.1:5000 ( Local link )
```

127.0.0.1 = host, 5000 = port

Stocks.json code:

```
[{"name": "vodafone", "price": 500, "barcode": "789789789789", "description": "none"}, {"name": "SBI", "price": 1000, "barcode": "152152152152", "description": "none"}, {"name": "JIO", "price": 2000, "barcode": "426426426426", "description": "none"}]
```

Templates folder:

In the templates folder we create HTML codes.

- Base.html

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">

    <!-- CSS -->
    <link rel="stylesheet" type="text/css" href="static/css/myapp.css">

    <title>
      {% block title %}

      {% endblock %}
    </title>
  </head>
  <body>
    <p style="text-align:center;"></p>

    <div class="navbar">
      <a href="{% url_for('home') %}"><button style="display:block; margin: 0 auto; height: 30px;
      width: 400px; left: 250; top: 250; font-size:20px">Home</button></a>
      <a href="{% url_for('stocks_page') %}"><button style="display:block; margin: 0 auto; height:
      30px; width: 400px; left: 250; top: 250; font-size:20px">Stocks</button></a>

    </div>

    {% block content %}

    {% endblock %}

  </body>
</html>
```

- Stocks.html

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

{% block title %}
    stocks
{% endblock %}

{% block content %}

<footer>
<font color="black"><u> contact address:</u><br>
</font>

<font color="green">
<<span>${x1F4E7:pannelajayakumar@gmail.com}</span><br>
</font>

<font color="green"><span>&#x1F4F1:+91-9700158638</span></span><br></font>
</footer>
<div>
<table>
    <thead>
        <tr>
            <th scope="col">ID</th>
            <th scope="col">Name</th>
            <th scope="col">Barcode</th>
            <th scope="col">Price</th>

            <th scope="col">options</th>
        </tr>
    </thead>

    <tbody>
        {% for item in items %}
            <tr>
                <td>{{ item.id }}</td>
                <td>{{ item.name }}</td>
                <td>{{ item.barcode }}</td>
                <td>{{ item.price }}</td>
                <td>
                    <button class="btn btn-outline btn-info">More Info</button>
                    <button class="btn btn-outline btn-success">Purchase this Item</button>
                </td>
            </tr>
        {% endfor %}
    </tbody>
```

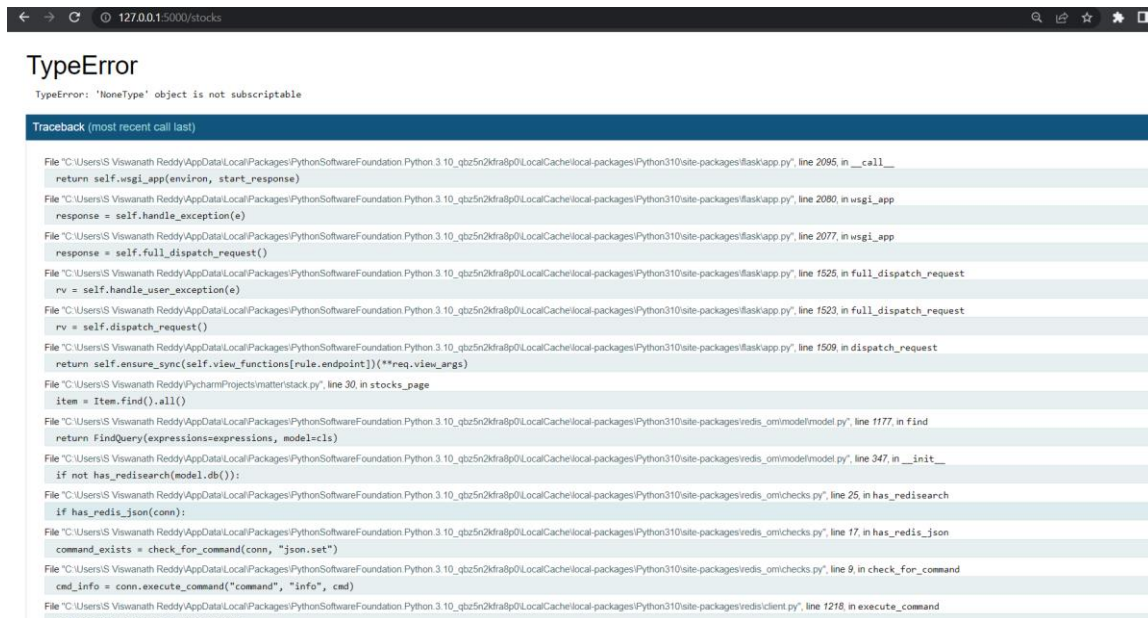
</table>

</div>

{% endblock % }



Click on Stocks and enter the company name. it shows total info like today's price, company history, and all company details. But I am getting error, that error is Type Error.



Conclusion:

In this task, I am monitoring the web app by using Redis. There are many monitoring tools, But it is open source, and fast. But it is difficult to understand for beginners.