Financial Risk Management with Apache Spark

Design Component

CS Capstone Project
Saint Louis University
March 2, 2018

Kamran Madatov
Bao Thai
John Mitton

Under Guidance from:
Dr. Goldwasser & Dr. Chambers & Dr. Ferry

Mentor:
Dr. Ahn

A written document that describes a detailed design for achieving the formal requirements. A design document should include a description of the major components, their interfaces and how they interact to form the whole. Figures should be included for clarity, such as a UML diagram of the software design or an ER-diagram for a database. This document should also contain a discussion of any third-party technologies or software packages that will be used in meeting the project goals. Teams should demonstrate that they have already evaluated and familiarized themselves with any such technologies. Finally, this document must include a proposed timeline for the remainder of the project life cycle, making sure to include specific sub-goals for the development, implementation, and testing phases of the project.

I. Project Description

Value at Risk (VaR) has been widely adopted in the financial industry to measure risk. It is used for regulatory compliance, understanding the risk characteristics of large portfolios, and and making informed trading decisions. Three common methods of calculating Value at Risk are variance-covariance, historical simulation, and Monte Carlo simulation. Monte Carlo simulation can be more accurate than the simple models, but it requires more computational power. Fortunately, Apache Spark provides an easy way to scale statistical problems beyond what a single server can handle. Using Spark and historical stock data, we will calculate VaR of stocks with Monte Carlo Simulation in less time.

II. Functional Component

Monte Carlo

Process

1. Historical data on the desired stock is collected from online via a web crawler.
2. The data is read from the csv file and stored in appropriate lists while the simulation runs. E.g. opening, closing, date etc.
3. An expected value and standard deviation is calculated and used to generate the normal random variables.
4. Those variables are the increase or decrease for a day. 1000 of them are generated and the mean of them is the expected increase or decrease for the day.
5. Step for is repeated for the desired number of days.

Spark Testing

```
[mittonjw@hopper csvget]$ time python monteCarlo.py
(229.60866234094658, 232.37271069552997, 235.13675905011337)

real    0m1.839s
user    0m2.271s
sys     0m3.609s
[mittonjw@hopper csvget]$ time python monteCarlo.py
(127.36229985967023, 129.10873267725515, 130.85516549484007)

real    0m1.440s
user    0m2.000s
sys     0m3.612s
[mittonjw@hopper csvget]$ time python monteCarlo.py
(9.2076127787427762, 9.6533378882849412, 10.099062997827106)

real    0m1.825s
user    0m2.259s
sys     0m3.540s
[mittonjw@hopper csvget]$ time python monteCarlo.py
(510.1790187071482, 520.28121949368722, 530.38342028022623)

real    0m1.611s
user    0m2.079s
sys     0m3.608s
```

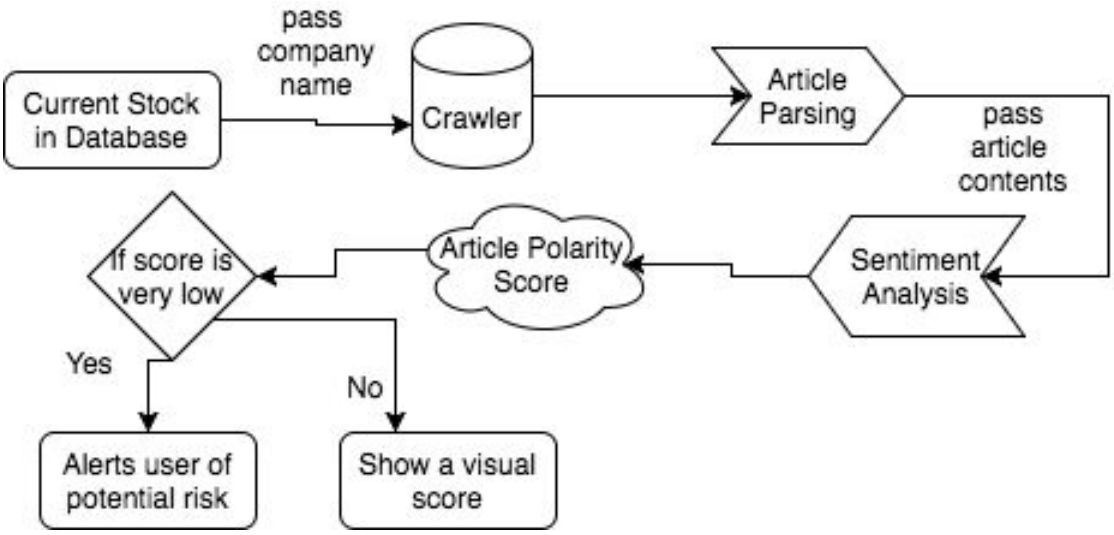
Figure X.X- Time to run one simulation on Hopper

```
[hadoop@ip-172-31-41-182 ~]$ time spark-submit monteCarlo.py
350.341495613

real    0m2.689s
user    0m3.380s
sys     0m0.220s
[hadoop@ip-172-31-41-182 ~]$ time spark-submit monteCarlo.py
234.367435337

real    0m3.171s
user    0m3.712s
sys     0m0.232s
[hadoop@ip-172-31-41-182 ~]$ time spark-submit monteCarlo.py
39.0466213559

real    0m2.812s
user    0m3.724s
sys     0m0.228s
[hadoop@ip-172-31-41-182 ~]$ time spark-submit monteCarlo.py
1026.76373186

real    0m2.900s
user    0m4.016s
sys     0m0.196s
```

Figure X.X- Time to run one simulation in Spark on an AWS cluster

The system time was significantly faster on the AWS cluster using Spark then on Hopper. However, more real time was taken. I believe that is due to having to send data over the connection to AWS. Which leads me to the conclusion that running many simulations in Spark on AWS will be faster then on Hopper, since the system time is much faster the real time discrepancies will not make a difference.

Articles - Sentiment Analysis



Process

1. An automated script containing the whole process of the web crawler (Crawler, Article Parsing, Sentiment Analysis) will run continuously on an instance (hopefully will be EC2 instance)
2. The inputs for the scripts are from the database with columns - **Company, Last Updated**
    a. Last updated tells when the article was last crawled to let the script knows don't crawl before that date (assuming we have some data already)
3. Crawled data stored in a different database with the front end will use to display onto the front end

Visualization

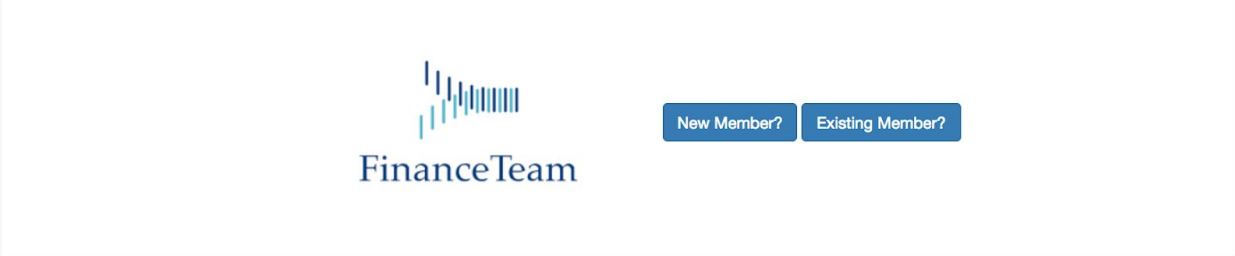
- Back End Database, InnoDB

	id	articleURL	company	domain	date	sentScore
Edit Copy Delete	1	http://www.fool.com/investing/2018/02/28/2-best-dr...	Nvidia	fool.com	2018-02-28	0.77
Edit Copy Delete	2	http://www.fool.com/investing/2018/02/28/is-advanc...	Nvidia	fool.com	2018-02-28	0.64
Edit Copy Delete	3	http://www.fool.com/investing/2018/02/27/3-stocks-...	Nvidia	fool.com	2018-02-27	2.24
Edit Copy Delete	4	http://www.fool.com/investing/2018/02/27/wall-stre...	Nvidia	fool.com	2018-02-27	0.28
Edit Copy Delete	5	http://www.fool.com/investing/2018/02/26/why-is-th...	Nvidia	fool.com	2018-02-26	0.13
Edit Copy Delete	6	http://www.fool.com/investing/2018/02/23/amazon-an...	Nvidia	fool.com	2018-02-23	-0.32
Edit Copy Delete	7	http://www.fool.com/investing/2018/02/23/dont-wast...	Nvidia	fool.com	2018-02-23	0.32
Edit Copy Delete	8	http://www.fool.com/investing/2018/02/22/3-stocks-...	Nvidia	fool.com	2018-02-22	3.17
Edit Copy Delete	9	http://www.fool.com/investing/2018/02/22/forget-cr...	Nvidia	fool.com	2018-02-22	0.50
Edit Copy Delete	10	http://www.fool.com/investing/2018/02/21/3-stocks-...	Nvidia	fool.com	2018-02-21	1.38
Edit Copy Delete	11	http://www.fool.com/investing/2018/02/28/why-i-wil...	Apple	fool.com	2018-02-28	-3.09
Edit Copy Delete	12	http://www.fool.com/investing/2018/02/28/apple-inc...	Apple	fool.com	2018-02-28	2.39
Edit Copy Delete	13	http://www.fool.com/investing/2018/02/28/4-reasons...	Apple	fool.com	2018-02-28	1.30
Edit Copy Delete	14	http://www.fool.com/investing/2018/02/28/you-gotta...	Apple	fool.com	2018-02-28	0.74

Figure X.X - Database of article parse

FROND END

- **Homepage, Registration, Login**



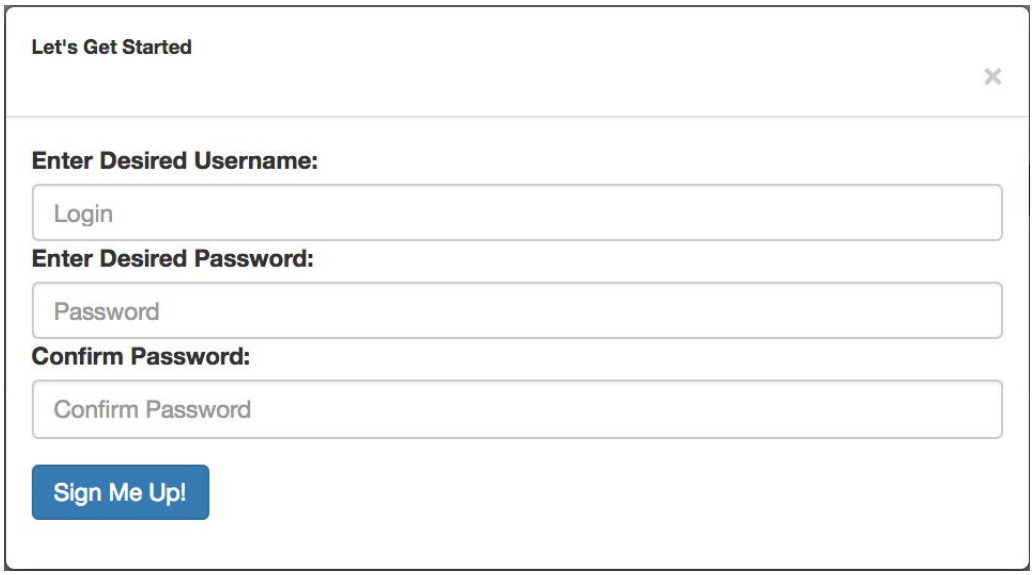


Figure X.X - Home Page and Registration & Login Modals

- **User searching for stocks**

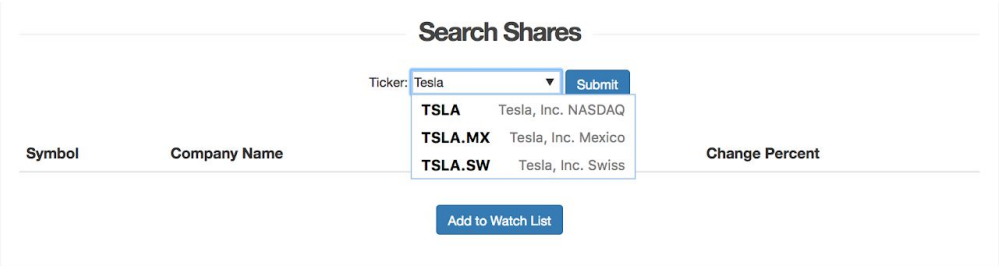


Figure X.X - Searching stock through Yahoo Search Ticker API

- **User obtaining and extracting stock data**

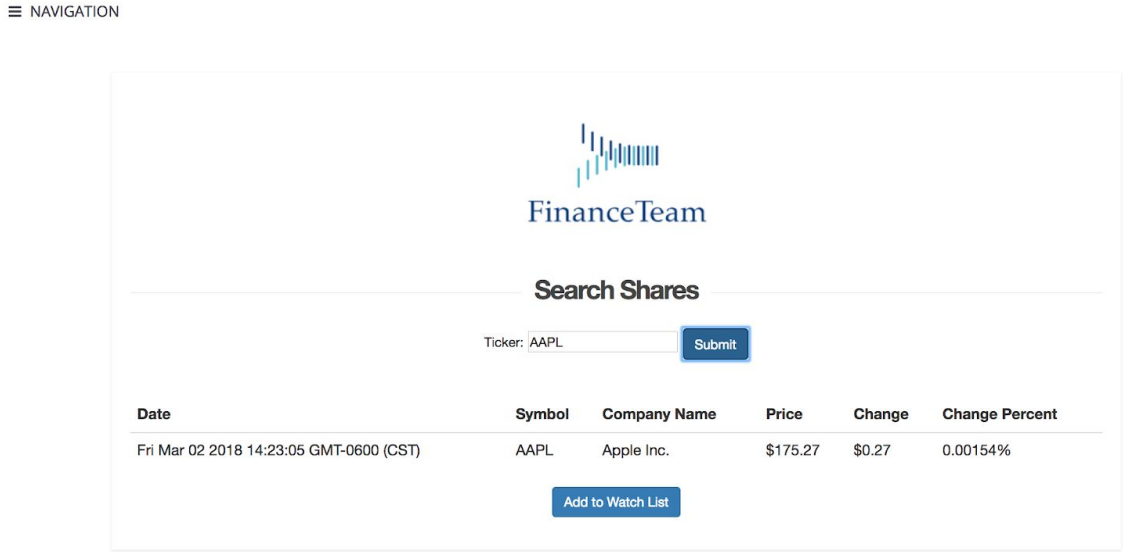


Figure X.X - Extracting stock information through IEX API

- **Profile**

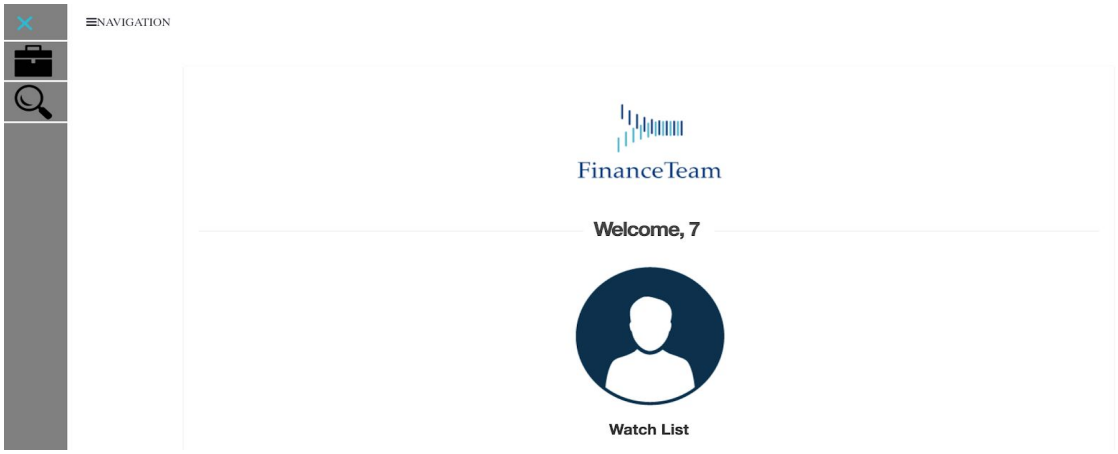


Figure X.X - Profile Welcome Page and Navigation

- **An interactive plot by Javascript (amCharts)**
  - Allows to view date and score of article (today - 3 weeks prior)
  - Clicking on data points will jump to the article link
  - E.g. : Tesla graph

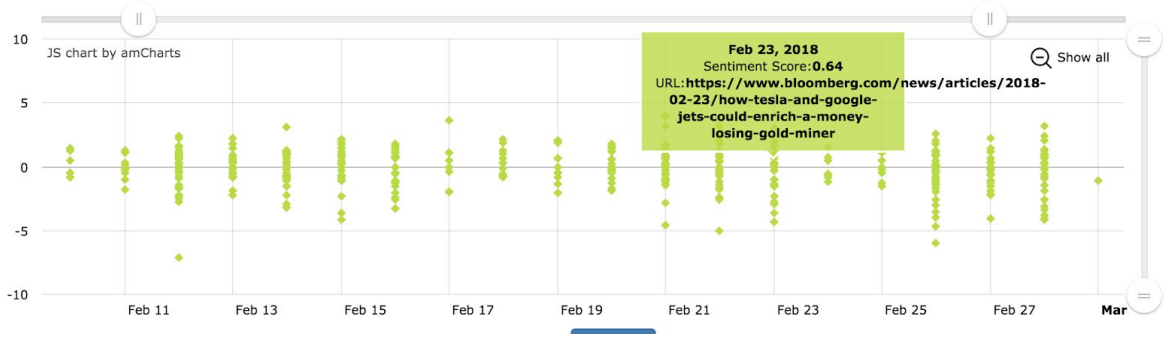


Figure X.X - sentiment scoring plot

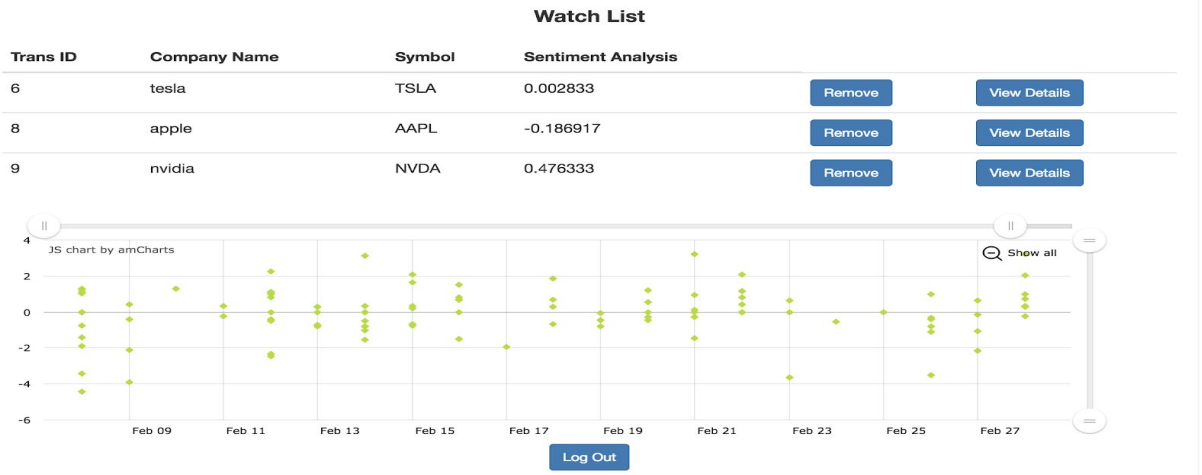


Figure X.X - Personal Stock Portfolio Watch List

- General interpretation of article score
    - **[-1,1]** - Stable in growth
        - Long term investment opportunity?
    - **> 1** - High risk,
        - High chance for a short term growth (Short term long buys)
    - **< -1** - High risk
        - High chance stock continue decreases (good for shorting)

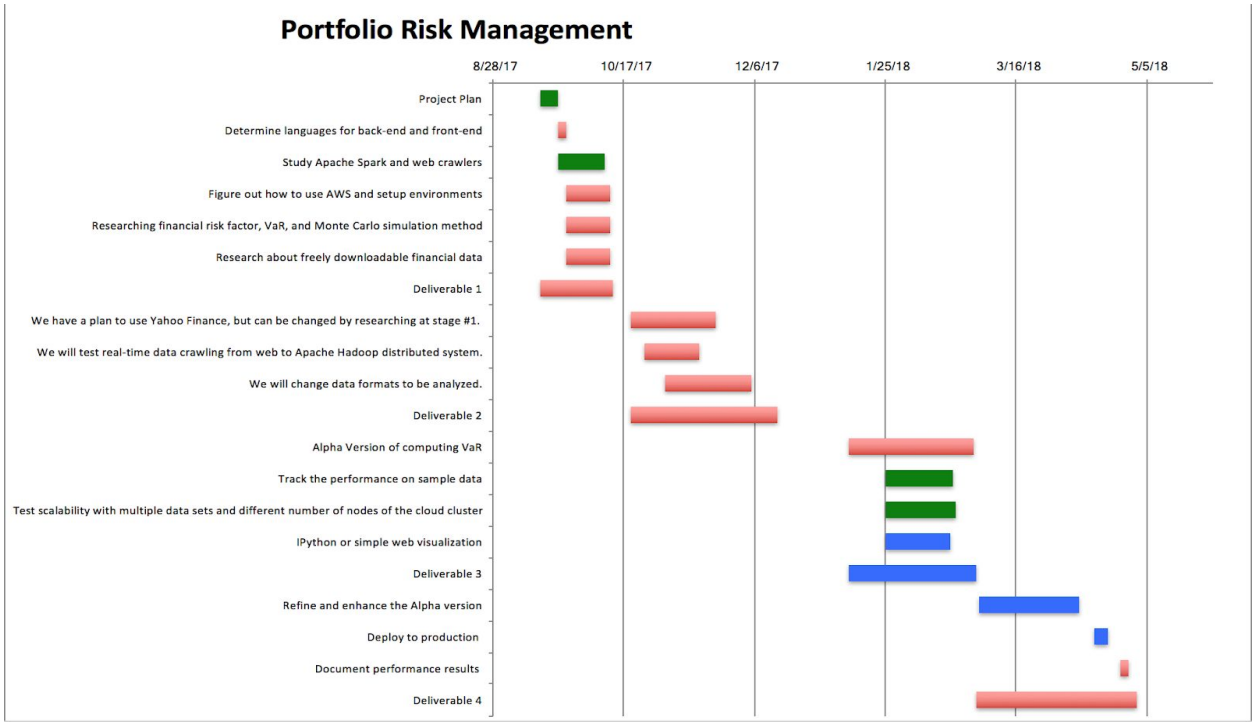
Future Plans and Improvement

- Future Planning
    - Apache Spark Computing
        - These web crawler and monte carlo requires a machine with high processing power and computing, so we will look into Apache Spark to set up our cloud computation system
        - Apache Spark will hold our python functions and will run continuously with Hadoop's cloud server to generate the monte carlo data as well as sentiment scoring.
        - We will set constraints to run and collect certain data only to prevent collection of the same data
    - Hadoop and Data Management
        - We are generating a lot of data and we want to figure out how to better manage it and store it for later purposes, such as the risk analysis and sentiment scoring
        - We are planning to use Hadoop as a cloud storage system. And this will be the middleware for storing articles that needs to be parse, and then passed to Apache Spark (sentiment scoring parsing) to obtain a scored result. And then it will be passed back again to Hadoop to store the result

- Rooms for Improvement
    - Filtering Articles
        - Some articles are irrelevant to the company, but the crawler picked it up due to having one related keyword which could cause deviation to our sentiment scoring
    - Front End
        - Minor Bug Fixes
    - Newer Dictionary
        - Loughran Dictionary is from 2014, could be new changes that is more correct or better score articles
    - Monte Carlo
        - Study other models to find room for improvement
        - Look into the possibility of a neural network or other deep learning
    - Apache Spark

- Better utilise the parallelization capabilities
- Optimize to run on multiple nodes

Remainder Timeline and Planning



Overview

The backend (monte carlo, webcrawler, database, and sentiment score) and frontend (stock ticker search api, stock data extraction api, login, registration, portfolio, add stock to portfolio) is currently standing strong with minor needs of improvement. Moving forward, apache spark/hadoop computation/data-management will be implemented for efficiency, performance, and high traffic management. And potentially move and base and run the entire project with AWS Services.

- Alpha Model:
  - Functionality of Back-End: Monte Carlo (Long Term Analysis) & WebCrawler/Sentiment Scoring (Immediate Analysis), Database
  - Functionality of Front-End: Login, Register, Personal Portfolio Stock Watch List, search & add stock to portfolio, Interactive Chart Analysis of Sentiment Scoring
- Beta Goals:
  - Merging Sentiment Scoring and Monte Carlo
  - Scale with Apache Spark & Hadoop and
  - Base and run the project with AWS