LOG RETURNS

VISUALISE CORRELATION IN THE STOCK MARKET

May 2022

CONTENT

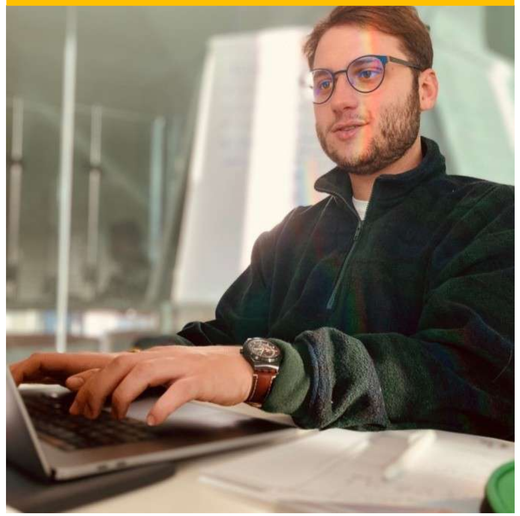
OUR TEAM

Etienne Salimbeni

Master student

Cybersecurity @EPFL

Carlo Musso



Master student

Computer Science @EPFL

Filippo Salmina

Master student

Financial Engineering @EPFL

How we chose our **Project.**

The three of us agreed on engaging in an ongoing real-world task rather than choosing a dataset for the sole purpose of the project. For this reason, we reached out to various labs and companies to understand their needs with a focus on data visualisation.

We finally decided to engage in the project proposed by Damien Challet: a professor at Centrale Supélec School in Paris (also EPFL associated).
Prof. Challet’s finance data-driven Finance lab is working on reviewing the state of the art of clustering financial time series.

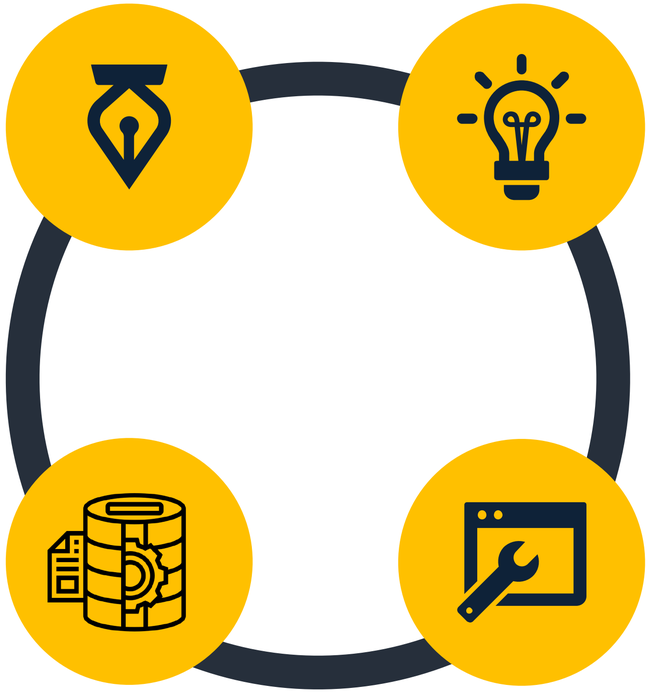
THE PROJECT IDEA

CONTEXT

Correlation between stock prices gives powerful insights into the evolution of financial systems in significant economic periods. Correlation tells us a lot about how industries are dependent on or independent of crisis, growth and industry sectors in a given time period.

DATA PROCESSING

To obtain the correlation values of each pair of stocks we select two stocks and a window of size n on the data set and compute the correlation between the two selected stocks. Then repeat this procedure for all pairs.



IDEA

The goal of this visualisation is to represent the evolution in time of correlation, hierarchies and clustering of various financial stocks. Visualisation is done through graphs and basic graph theory principles, focusing on per-sector analysis. In our visualisation, we represent stocks as nodes of a graph and their corresponding correlation as the weight of the edges connecting them.

WEB DEVELOPMENT

The graph will thus be constructed by nodes represented by stocks and edges with weights representing the distance co-variance between each pair of stocks. Our web application enables users to navigate the graph and gain further statistics on a selected node.

DATA PROCESSING

Filippo

THE PROCEDURE

In order to obtain the correlation values of each pair of stocks, we select two stocks (S1 and S2) and a window (W1) of size n on the data set (where n is the number of days which the window will represent) and compute the Pearson correlation in log returns between the two selected stocks in the window time-frame. If we repeat this process for each pair of stocks in our data set, we can structure our correlation data type as a correlation matrix. One time window is not that insightful. In order to represent the correlation evolving with time, we must compute 'k' of these matrices. The final graph will not directly represent the notion of correlation but its proportional inverse: distance, it’s more straightforward when visualising the graph.

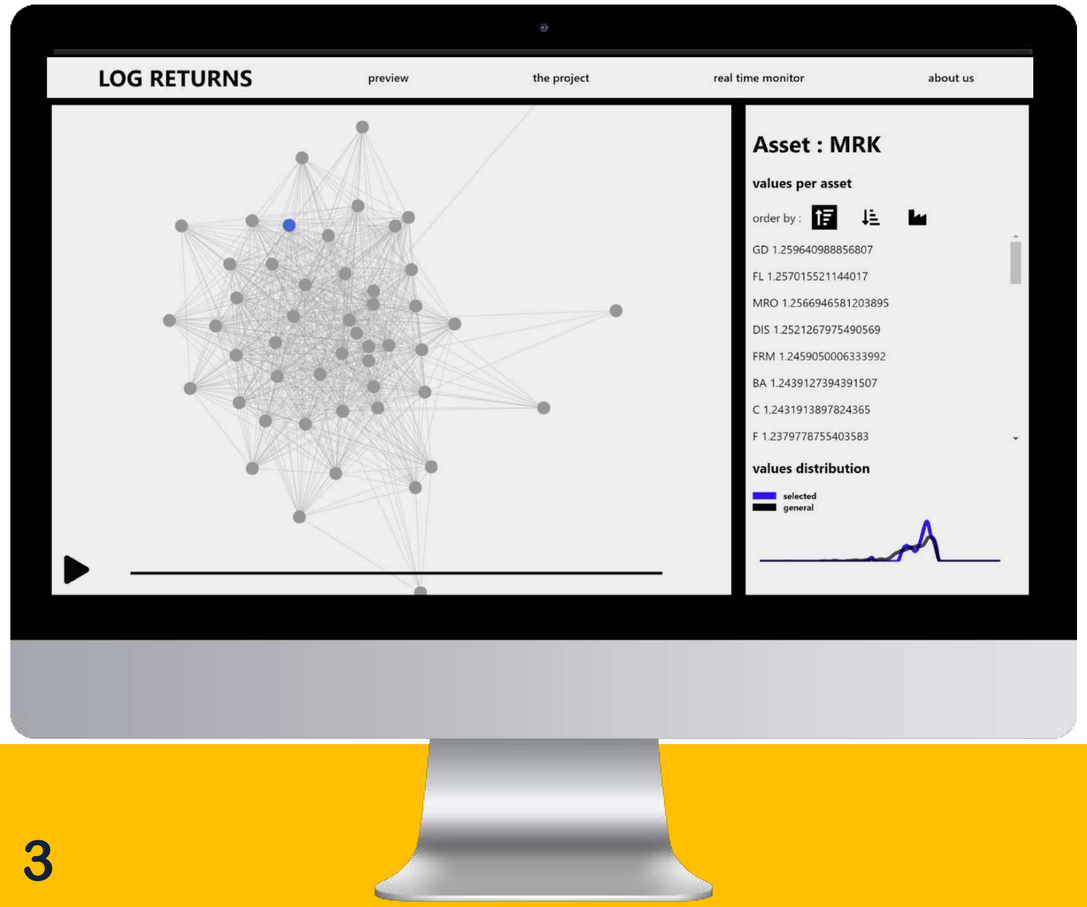
THE CHALLENGES

- Understand documentation for correlation and distance computation
- Implement python file for computing correlation and distance matrix
- Generate JSON data compatible with data received by the js graph library
- Scrape Yahoo Finance to retrieve more information to display on stocks

WEB DEVELOPMENT

FRONT-END

Etienne



THE TASK

Setup the core skeleton for the project. We opted for Next.js, which is a React framework. Then find a graph visualisation library which supports smooth graph animation: we opted for Cytoscape. Once the basics are done the goal is to enable the selection of a node and get an interactive asset tab with the list of correlation values which can be ordered by distance and sector.

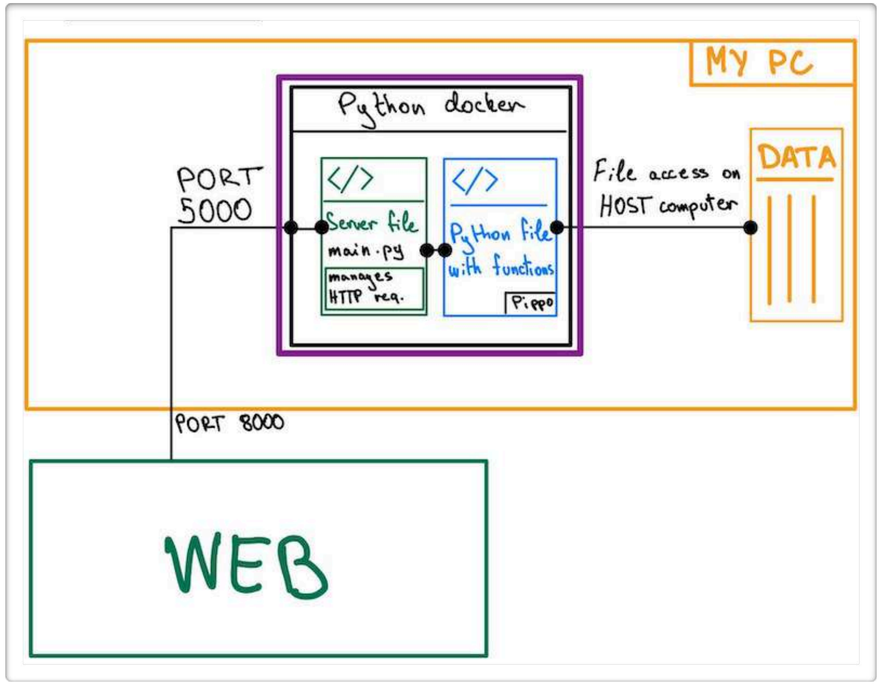
THE CHALLENGES

- Find graph library considering that the visualisation evolves over time.
- Library documentation to understand graph models, data, events, layouts and graph manipulation of Cytoscape.
- Because we have many correlated updates (select a node, node statistics, graph step update) it was not easy to merge Cytoscape with our react application: with Cytoscape directly changing DOM elements, and react updating their components based on react state updates.
- Because Cytoscape, setTimeout and React use 3 different ways to interact with the DOM , it was hard to keep everything synchronised.

WEB DEVELOPMENT

BACK-END

Carlo



THE TASK

Implement a docker server in python that manages all data requests coming from the web application. The idea is to make the server compute the correlation and distance matrix and then create the JSON file with correlation values to send to the application. Since data files in this domain can be very large, the docker should be able to access the host computer to read data from a specified file instead of writing the file in the docker on each build.

Note — at the moment the web application is not using the server as for the purpose of usability (not practical to open docker each time for a generic user) the files produced by the server (from 2007 to 2013) are directly stored in the application code. However, the working implementation of the server is there for future, more scalable, utilisation.

THE CHALLENGES

- Understand how Docker works.
- Create communication between the server and the javascript application in the 'connected' version of the graph component.
- Understand how docker-compose works in order to build a shared volume between the docker and the host where the docker accesses files on the host computer.

SMOOTH VISUALISATION & DESIGN REFACTOR

Filippo

THE PROBLEM OF SMOOTHNESS

One main problem with our web application was smoothness. At each time interval, because of the high number of edges changing value (also slightly), the whole graph was behaving in a glitchy manner making the evolution hard to follow. This problem was tackled by rounding and truncating values on edges and assigning them to value intervals so that variations were limited. This maked the evolution smoother, while still preserving correctness of the data.

Implementation of design refactor (on Etienne's template)
Refactor graph legend & time navigation bar design
Refactor stock information pallet design
Refactor top navigation bar design

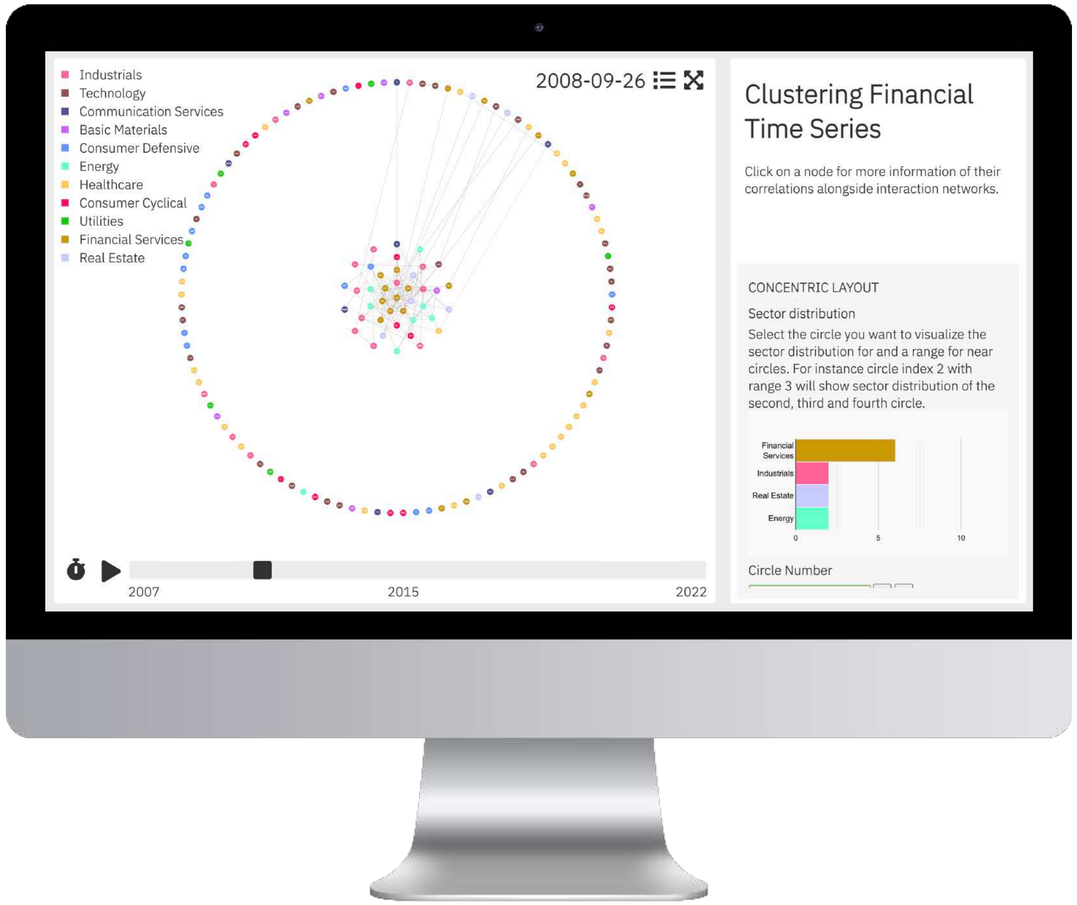
ONE MORE LAYOUT

Carlo

THE GOAL

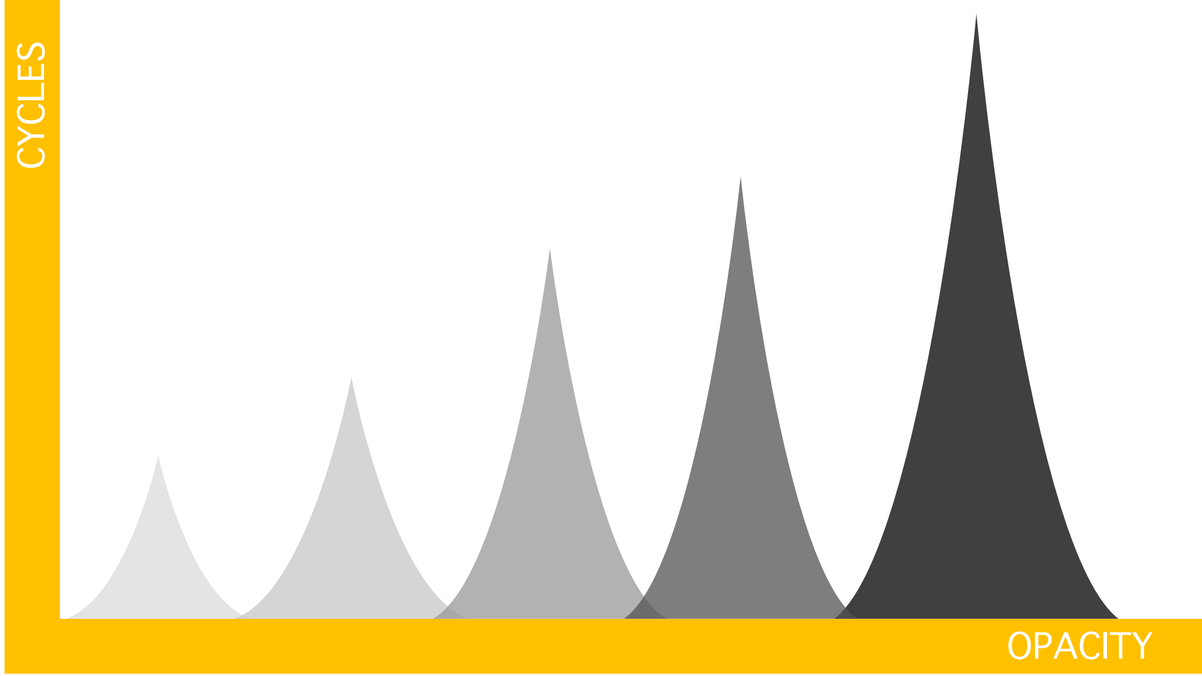
The concentric layout graph model assigns each node to a circular level around the center according to its degree. High-degree nodes are positioned in the middle, while low degree nodes are positioned in the outer circles. Namely, the number of edges attached to a node determines where the node is positioned. Thus in the periods where the stock market is generally highly correlated, all nodes tend to concentrate in the inner circles. In contrast, when the market is generally less correlated, nodes are mainly positioned in the outer circles of the concentric layout.

- Challenge — finding the layout which would best represent the above concept and adapt data to its parameters.



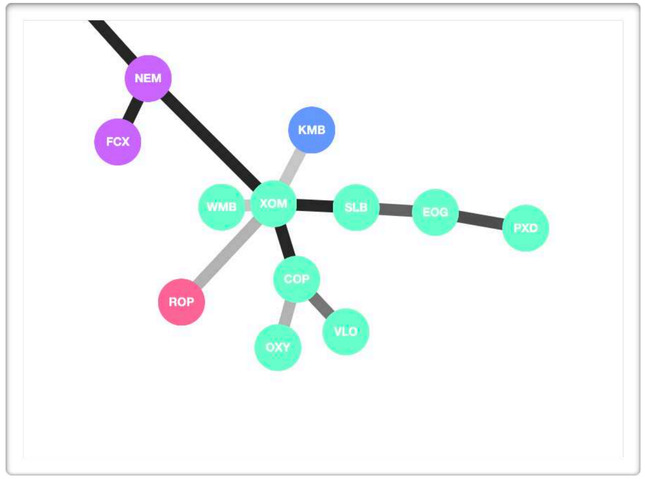
Monitoring
Node Lifetime.

Etienne



THE INSIGHT

To offer further insights on the evolution of correlation between stock nodes the edges connecting two nodes have a lifetime attribute. Namely, a number of cycles of the graph in which the edge is persistent. The more the edge is dark the more time passed in which the two nodes are correlated.

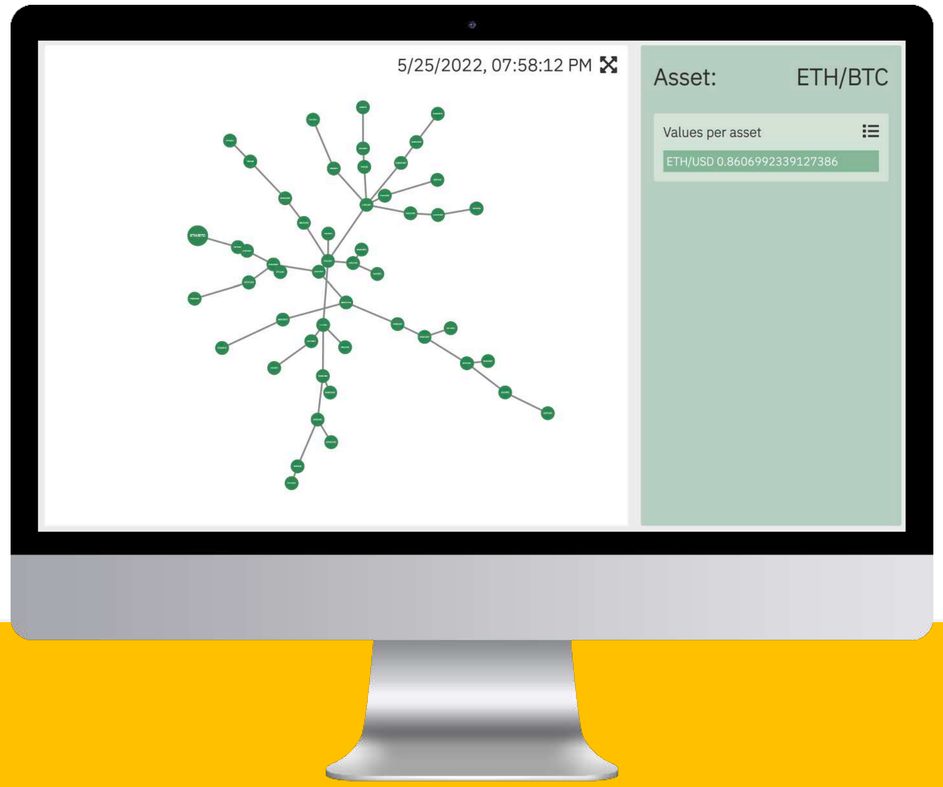


Straightforward insight: NEM and XOM have been correlated for a longer period then ROP and XOM.

REAL-TIME MONITOR

To make our dashboard a more interesting and exciting instrument, we looked into the mechanisms of a very highly-correlated financial market: cryptocurrencies.

We implemented on the LogReturns tool a real-time monitor fetching data from the FTX crypto exchange which gives live insight on the flow of cryptocurrencies' correlation state.



THE CHALLENGES

Filippo

RTM

- Understand how to fetch data from FTX API and communication with the web application.

Etienne

RTM

- Code optimisation on serverless function calls. Vercel deployment has a limit on API response time to 5 seconds.

Carlo

DELIVERABLES

- Write the project milestones and the process book.

LOG RETURNS LINK

https://log-returns.vercel.app/