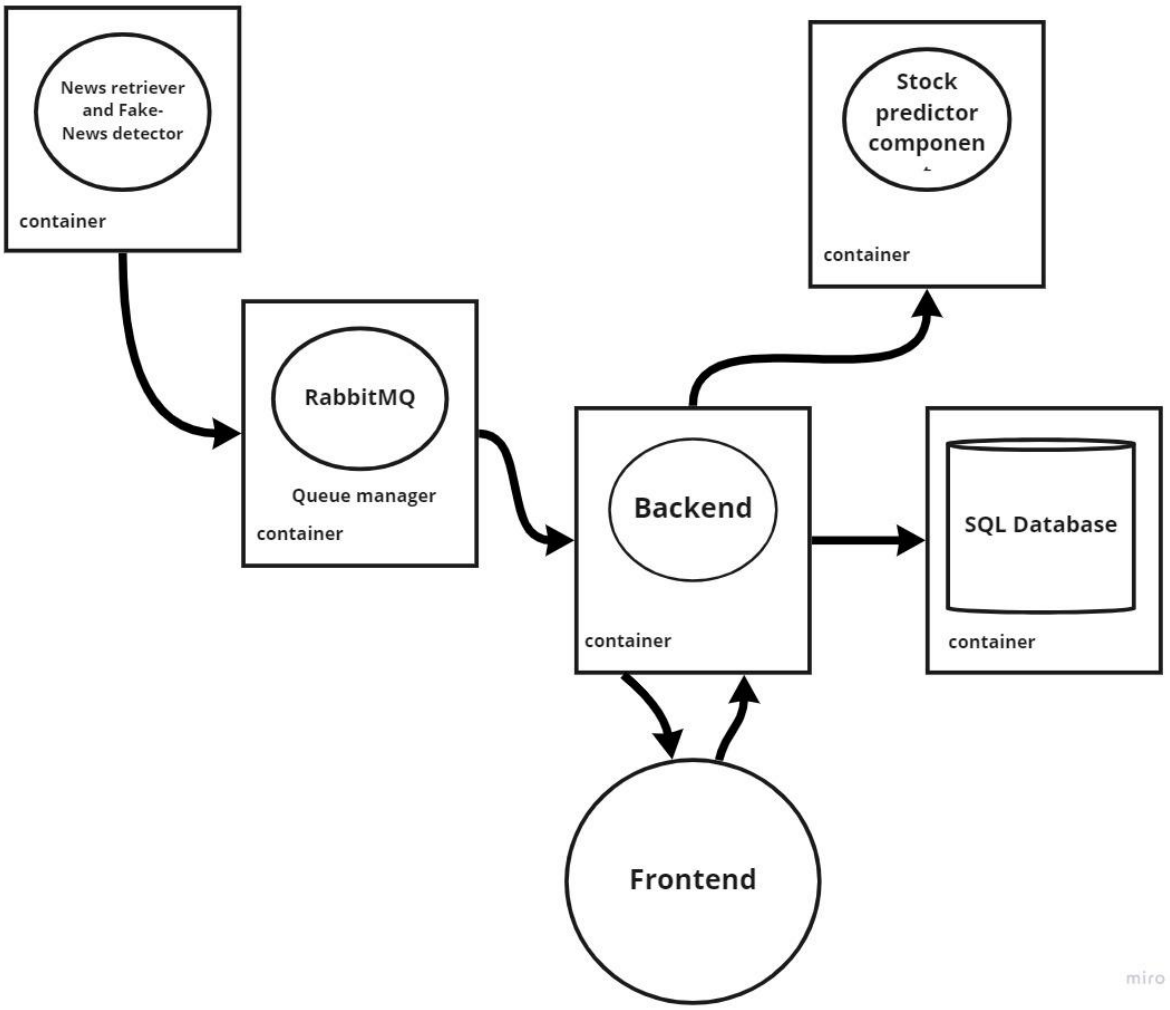
Design of the system:

• Front-end: login + stocks page + newspage
• SQL Based Database.
• RabbitMQ
• Backend : hosting the RESTful Web application.not
• News Component: retrieving news and sending those ones that pass a fake-news detector machine learning system. Interacting inter-application using rabbitMQ and queues.
• Stock prediction component: Component used to predict the future stock market trends. Exposed to the other components via REST API's.
• external service accessed via api to get the news
• external service accessed via api to get the stocks data



Software architecture:

Database

-mysql database to store news, users and various structures for the application functionalities(banned users ,…).

Backend

-we used node js with some utilities from the express framework for the backend that is linked to the frontend, the database , rabbitmq to connect with the fake-news detector and is also connected with the stock predictor via a rest interface.

Queue

-we used rabbitmq as an amqp that is interconnecting the fake news detector with the backend : the fake news detector pushes every fixed number of seconds news to the queue and the backend uses those pushed objects to obtain informations related to the news preprocessed by the fake news detector.

News fake detector

-it's composed by a retriever of news coming from the mediastack api filtered by some keywords,a machine learning model to distinguish the fake news from the real ones using a naive bayes classifier, a node js server to push those preprocessed news to the rabbitmq component.

Stock predictor

-we used prophet to forecast the trend values of the stocks taken from the yahoo finance api and a flask server to expose the functionalities of the component to the backend.

Front end

-we didn't adopt any frameworks , we sent html css js from the backend ,using the classical libraries for the frontend development (jquery,bootstrap…)

Docker

-we divided the application into 6 containers that were orchestrated via a docker compose file. All the containers were created using a dockerfile and were linked to a docker hub repository from which the docker compose took the images.