Prestacop Poc

The typical architecture will have 5 component :
1) have small program simulating the drone and sending drone like data to your solution (see subject for details on a message). Your system will store message in a distributed estream making it available to the component 2 and 3.
2) handle alert message from stream
3) store message in a distributed storage (ex: HDFS/S3)
4) analyse stored data with a distributed processing component (like spark). As a proof of your system capacity to analyse the store data answer 4 questions of your choice. (ex: is there more violation during the week or during week-end?)
5) load the CSV file in you distributed storage component. This must be done (couple) of line by (couple) of line.

All component must be scalable and used in a scalable way.
For component 3) you may use kafka connect or its equivalent (kinesis firehose).
Any code must be written in functional scala (compile to jvm on javascript doesn't matter). Unless I accept it as an exception the keywords «for, while, return, var» are forbidden as well as importing anything mutable.
One exception for now : if you want to display a number of received/stored… message or alert you may use the keyword.
Some student may choose as an option to code the five component in another functional language (F#, Haskell…).
If you're interested you may write the drone simulator in Go.

For the project you should use a git repo, work of different members of the group should be visible in different commits.
For submission you should send me an email with your git repo and the last commit hash
Late submission are accepted, minus 2 point per late day(s).

Once those 5 parts you can work on the optional part :
The optional part is quite open, the goal is for every group to work on something there are curious about or they find interesting for there CV. They can be done in the language of your choice unless you re using spark.

Here are some suggestion :
1) project deployed on the cloud (azure, aws, gcp,…) using IaC like terraform.
2) website using its dedicated db/queue to display every received alert (instead of a basic email/log/console print)
3) using docker and docker compose for the 5 component of the project
4) once component 4 is done, using spark-notebook/zeppelin to generate charts
5) using some dataviz or custom website to present the result of the spark analysis
6) using an ml model and adding information in the message to achieve predictive maintenance of the drone.

7) Whole project code in haskell/F#…
8) any idea you find relevant for the project if I validate it

FAQ

- Except when doing the cloud option everything can be written and deploy locally on your own computer (not distributed). The recommendation regarding not using distributed kafka/spark was only for the distributed option.
- Most student should focus on the basic components. A group can achieve a good mark (up to 16/20) without the optional part. The optional part is for the curious group which want to do more.
- If you want the cloud option to be done 100% correct all the components apart from drone simulator and the CsvToStream should be running on the cloud.
- Student are quite free for the architecture. For instance one student ask to write a rest server by itself to receive the CSV data and it's fine as long as csv is send (bunch) of line per (bunch) of line, and that it's failproof. For the 2nd option (alert website) one would probably use extra streams or database…
- presentation should be done on the 7th/8th with slides for the context and the architecture, small demo. Given time for presentation is 10/12 minutes (without question)