Construction of a system that is able to check event compliance over streaming event data in Python

AA GmbH is a company that wants to monitor the execution of a business process, in a way that some alerts are triggered when some anomalous pattern is detected on the data.

It is supposed that the compliance checker runs on a dedicate server, and provides a web service (POST) to check compliance of events when they happen. This web service, for better integration with the company software, accepts and returns JSON. In particular, the process instance (case ID) and a classification of the event that happens (for example, the activity name) should be accepted from the web service.

To ease maintenance of the compliance checker, it should be written in the Python language, preferably using the Flask framework.

AA GmbH have evaluated some conformance checking techniques in the past (token-replay, alignments etc.) but due to the big amount of streaming event data related to the business process, they are not scalable.

In particular, the process event arrival throughput is of 1000000 (one million) events in 1 hour.

AA GmbH have found that a possible solution is about building some probabilistic automata (with different prefixes length) on a training event log and using it to classify new events.

The implementation choices should take into account the required process event arrival throughput. Moreover, the final deployment should be done on an enterprise-level application server (e.g. UWSGI).

The service should log all the requests and it should be possible, for a given prefix length, to obtain a representation of the probabilistic automata with some sort of decoration related to the fitness of the state (i.e. how many times the given state has triggered a deviation) in a off-line setting (i.e. a PNG file containing the model should be generated).

Required documentation:

- Project formalization, cost and time analysis, and time allocation
- Requirements analysis
- Design and P.o.C. of the web service and of an example client
- Documentation on the implemented code (for each sprint)
- Testing and assessment (proof of work) of the code
- Deployment on an enterprise-level application server