A Scalable Infrastructure Proposal for Hypatialabs (Draft)

Objectives

Hypatialabs is developing a data-intensive machine learning prediction project. This proposal is from standard building blocks of open source, container based and cloud service deployment. The goals are to propose an infrastructure of reliability, scalability and maintainability.

In summary, we propose the high-level infrastructure in those areas:

Data Lake

REST base dashboard

Docker Container Runtime

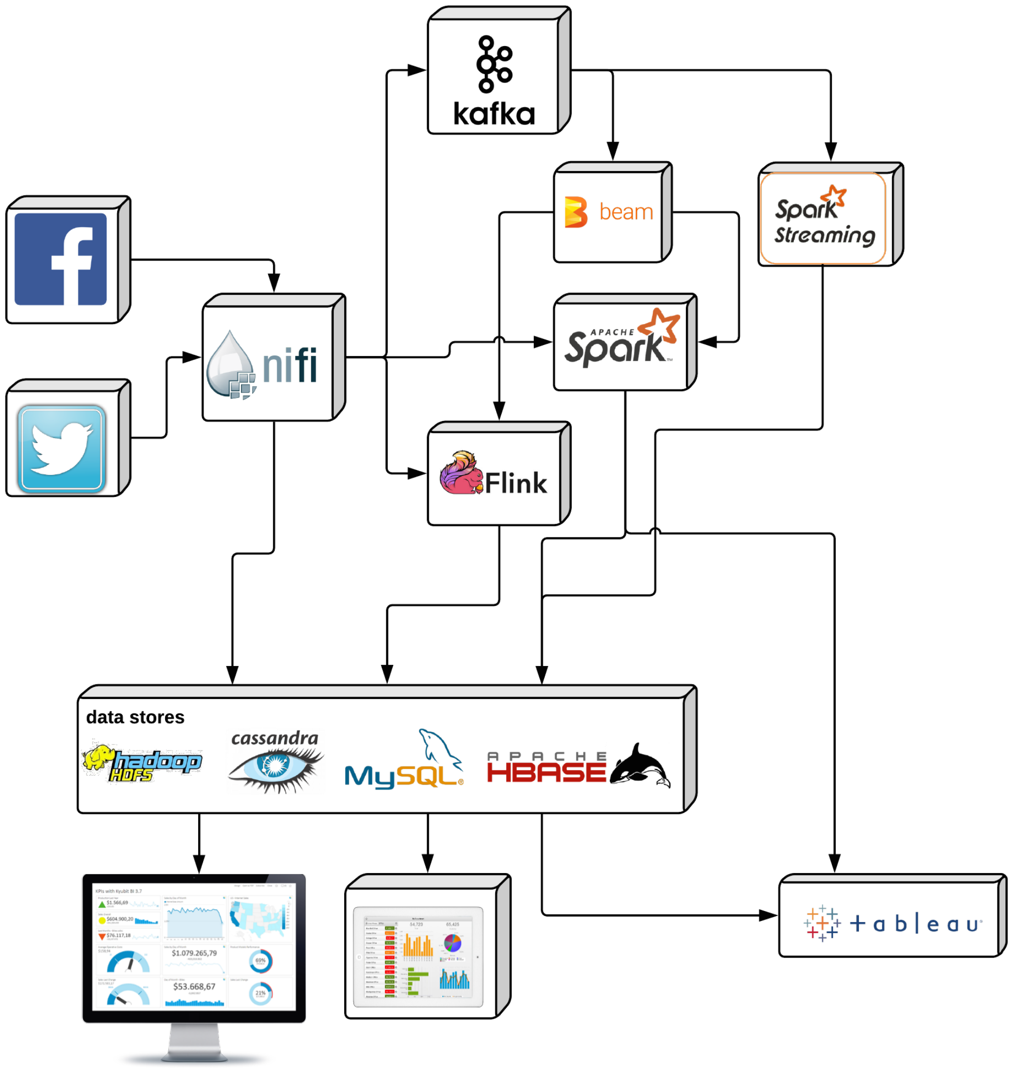
Amazon Deployment

The proposal provides a big picture of development and deployment architecture for a big data based machine learning project. It is intended to build each component gradually and extensively while the company grows. As for now, we should narrow down the scope of the infrastructure to fit into the quick rollout requirement for demonstration purpose. Since the entire design is well modularized and container based, it is easy to convert them into smaller scope design which will be based on what we have done now and what we plan to deliver for the prototype presentation.

Data Lake

The concept of data lake is to build a storage repository to hold the vast amount of raw data in original format for use while needed. Unlike the traditional data warehouse, data lakes retain all data from all sources in all data types. It is treated as central repository of data to support all application needs. It is accessible before being transformed, cleansed and structured such that applications can get to their results faster than the traditional data warehouse approach.

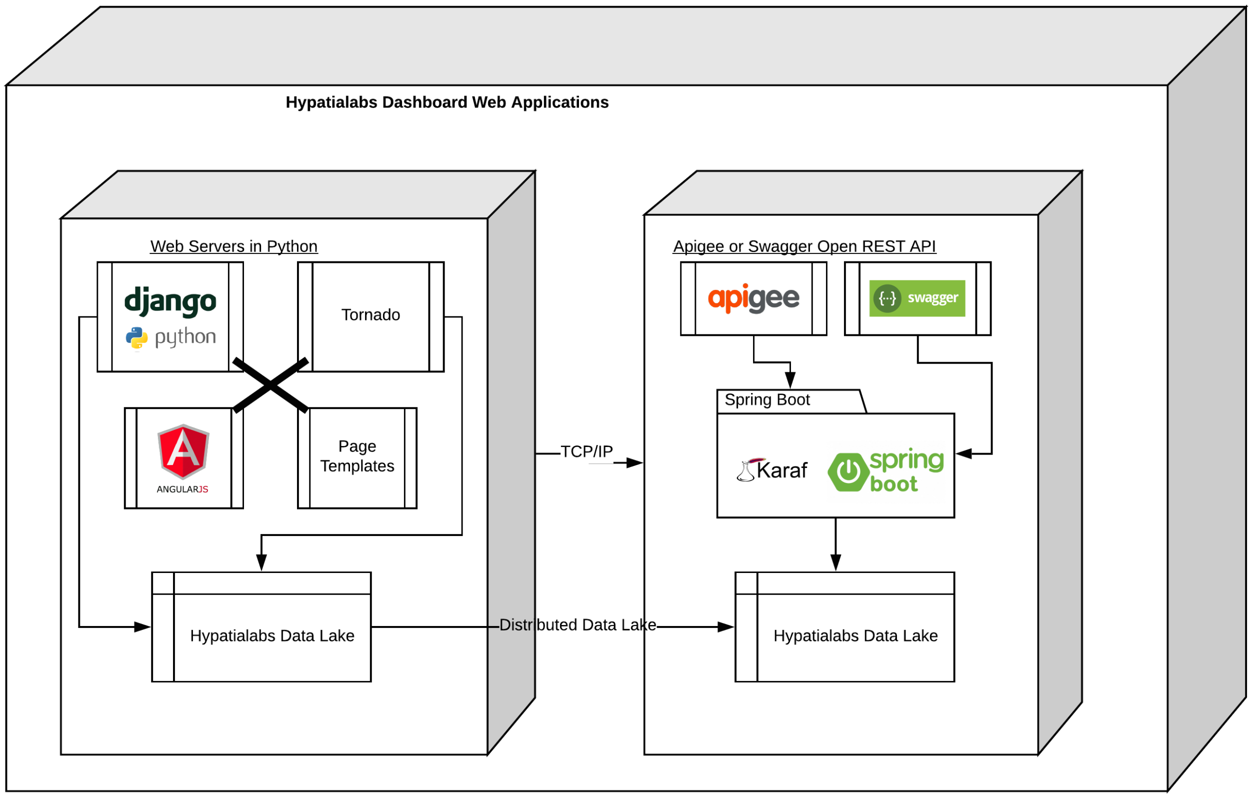
In this proposal, Apache NiFi is the center of data ingestion to deliver an easy to use, powerful and reliable system for processing and distributing the data over all resources. In other words, NiFi is used for routing and processing data from any source to any destination and with data transformation if needed.



Hortonworks (www.hortonworks.com) has provided a product called Hortonworks Data Platform that can be adapted to build Enterprise Level Data Lake and Data Ingestion. We may either consider leveraging it or build one on our own.

RESTful Dashboard

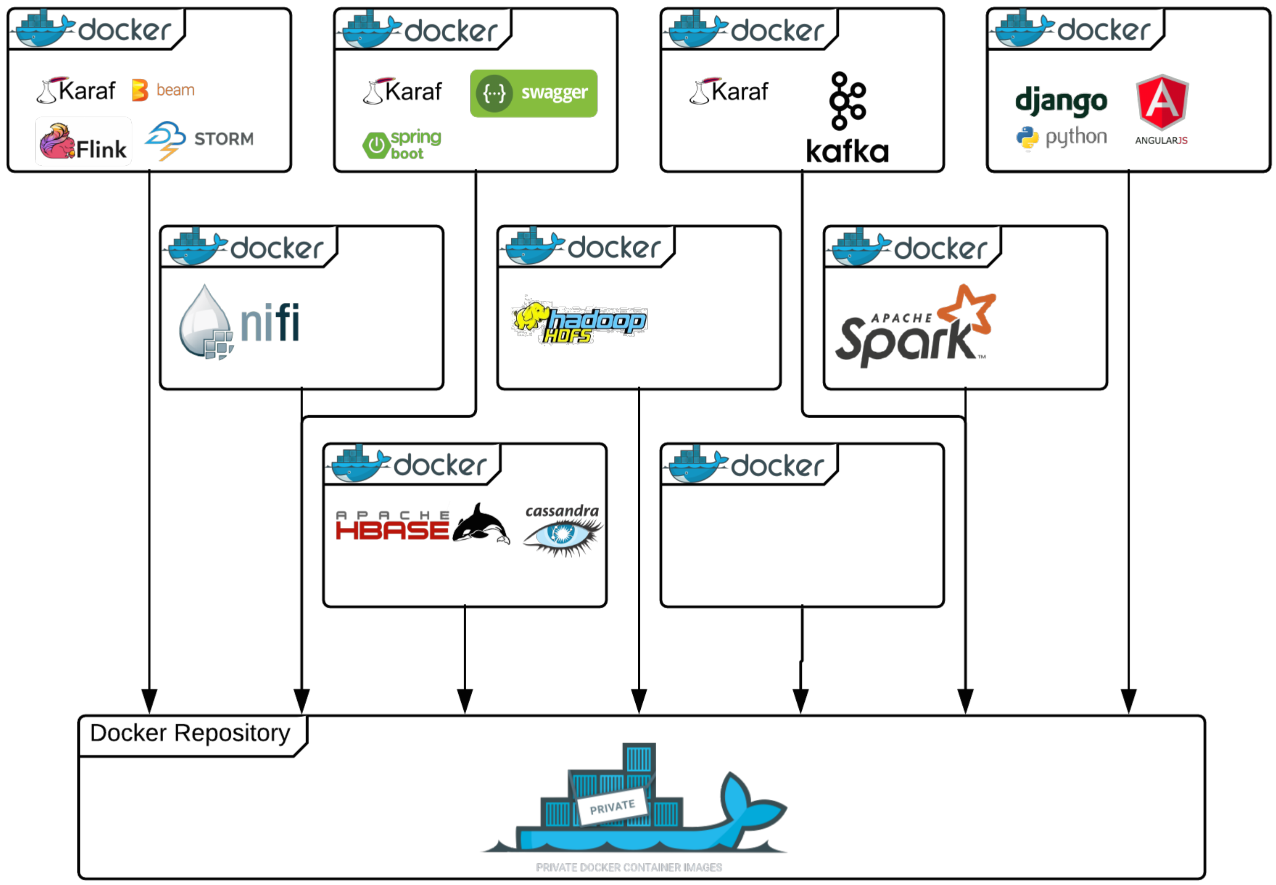
REST based APIs architecture has grown to be an industrial standard for data integration backbone of all collections of resources that are adhered to. Apigee and Swagger are the two popular REST specifications. Combining with Python and JavaScript based web applications accessing data of Data Lake, we can achieve the goals of Uniform Interface, Layer System based on stateless client/server protocol in that we got all the advantages of separation of responsibility, visibility, reliability and scalability and independent of platform and language in integration.



Docker Imagines, Repository and Registry

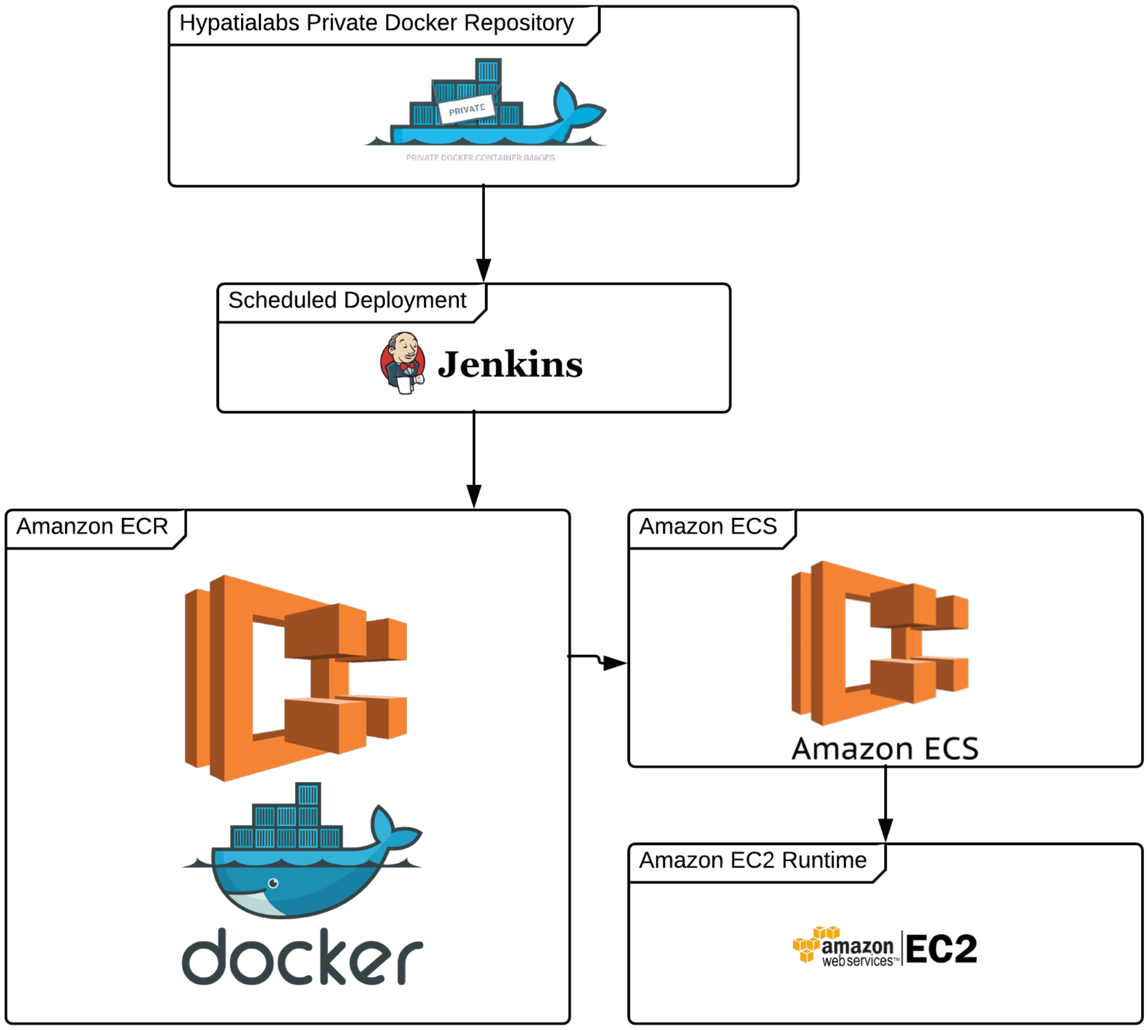
Docker is one of the hottest open source projects that allows to deploy applications inside containers, adding a layer of abstraction and easy management. With Docker containers, we can ensure the Continuous Deployment and Testing, Cloud Platforms support, Environment Standardization and Version Control, and furthermore the Isolation, Segregation and Security.

The container image Repository and Registry on the other hand is a collection of container images to provide the version control of images and easy to retrieve, install and deploy Docker images to various environments and platforms.



ECS/EC2 Deployment

Amazon Elastic Container Service (Amazon ECS) is a highly scalable, high-performance container orchestration service that supports Docker containers and allows you to easily run and scale containerized applications on AWS. Amazon Elastic Container Registry (Amazon ECR) is a fully-managed Docker container registry that makes it easy for developers to store, manage, and deploy Docker container images.



From Hypatialabs Private Docker Repository and Registry, we may directly deploy our applications to Amazon ECS/ECR and then deploy to runtime environment like EC2 seamlessly. On the other hand, we may leverage Amazon ECR to build Hypatialabs Private Docker Registry as well.

Conclusion

This proposal has covered the Storage and data ingestion design, programming architecture, application organization and deployment. With careful planning and design, we segregate the responsibility of each modules in that it largely reduces the dependency among each other. Thus, it is easy to deploy to Amazon EC2 and seamless to integrate with third-party resources such as Twitter etc. Overall it is well designed with scalable, extensible and reliable features.