

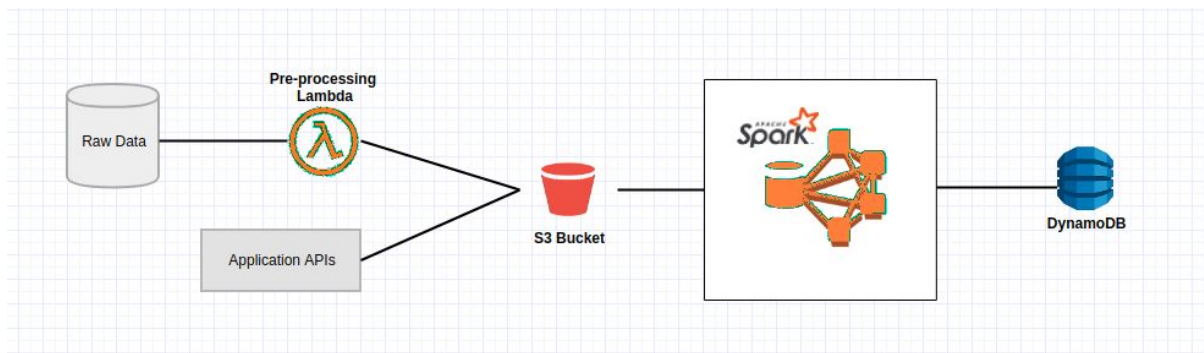
PROJECT PROPOSAL

DATA CENTER SCALE COMPUTING

PROBLEM DESCRIPTION

- Through this project, we aim to focus on scalability. In an application like Uber or any taxi service app, it is pertinent that the user requests of a cab are met in an acceptable amount of time. If the application delays a user's response a lot or fails to return a response, this could lead to a reduced popularity for the application.
- The application should be able to handle a large number of requests at the same time, to promote user satisfaction and also for monetary purposes.
- We plan to explore **Kubernetes/Docker services (EKS/ECS) to handle multiple requests parallelly.** We also plan to split independent requirements of a request and delegate them to different microservices so that they are computed parallelly and independently.

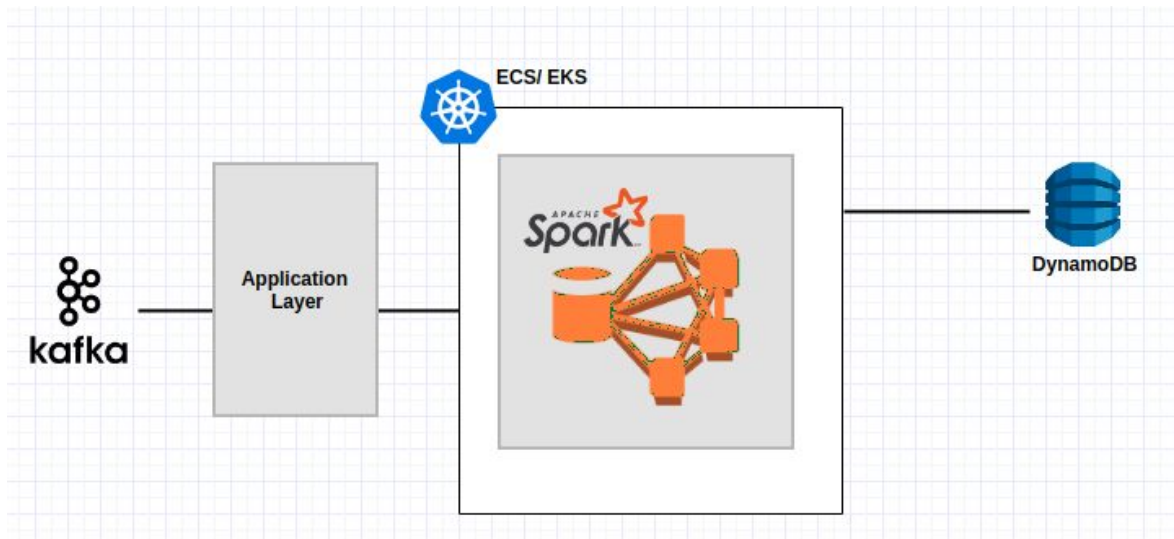
HIGH-LEVEL ARCHITECTURE



Pre-Processing data and storing the structured data in DynamoDB database

i would come up with a more specific use-case

- **Raw data** would be put into some kind of database (Maybe S3)
- Pre-processing lambda function takes data from this database, preprocesses it and stores it in S3.
- Spark then takes care of creating tables and columns in DynamoDB.



Kafka to process "real time" requests

- Kafka would contain user requests. For example, a request to search for a cab at a particular location.
- This request would then go to the application layer which would structure the user query, maybe add additional context and send it to a node with Spark capabilities in EKS/ECS (Elastic Kubernetes Service or Elastic Container Service).
- EKS is Elastic Kubernetes Service which provides an option to configure a Kubernetes cluster. ECS is Elastic Container Service which provides an option of creating multiple dockers.
- **We then configure some nodes in EKS/ECS to have Spark functionalities.**
- When a request reaches the EKS/ECS a new node might be created on the fly or use an existing node with spark functionalities.
- Spark takes care of interacting with the DynamoDB and querying for results which it sends back to the application layer.
- Extended Goal: Show some kind of visualization of the result.

so y'all will manually provision a spark cluster w/in kubernetes? email me if i'm not understanding this correctly. Does EKS/ECS automatically scale the spark cluster, or do y'all have to configure that? please email me the answer

DATASET

- We will be using NYC taxi trip data from Yellow Taxi, Green Taxi and Uber.
- Description:
 - Format: .csv
 - The data requires minimal preprocessing. (Removal of unnecessary metadata)
 - The dataset is static.

- The data can be exported as a CSV. There is no need of a developer account.
- The data will be stored in S3 buckets.
- Sample dataset - (Next Page)

Column Name	Description	Type	
vendorid	A code indicating the TPEP provider that provided the reco...	Plain Text	T
pickup_datetime	The date and time when the meter was engaged.	Date & Time	
dropoff_datetime	The date and time when the meter was disengaged.	Date & Time	
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehi...	Plain Text	T
rate_code	The final rate code in effect at the end of the trip. 1= Stand...	Number	#
Pickup_longitude		Number	#
Pickup_latitude	Latitude where the meter was engaged.	Number	#
Dropoff_longitude	Longitude where the meter was disengaged.	Number	#
Dropoff_latitude	Latitude where the meter was disengaged.	Number	#
Passenger_count	The number of passengers in the vehicle. This is a driver-e...	Number	#

Some Columns that are present in the [database](#)

CHALLENGES

- One of the challenges is to process the incoming requests fast and gain a major boost in the processing time through our architecture.
- Our architecture consists of multiple technologies like Kafka, EKS/ECS, ElasticSearch alongwith Microservices. The biggest challenge would be to integrate all the components and run the queries.
- There are technologies like Kubernetes, ECS, EKS, Kafka which our group is not familiar with. It would be a challenge to gain holistic conceptual understanding and recognize appropriate implementation strategies.
- Our metrics of measuring performance of our project would be to calculate the average running times of 100 different requests which requires a considerable processing power and time and compare the running times with and without using ECS/EKS etc.

GENERAL TASKS AND TIMELINE

Given that we are to present two checkpoints for the given project, we have divided our timeline accordingly.

Weeks	Tasks planned
Oct 23 - Oct 30	Project Proposal <ul style="list-style-type: none">● Teaming up to discuss architecture and write proposal
Nov 1 - Nov 13	<ul style="list-style-type: none">● Finalise project Design● Start setting up EKS/ECS● Preprocessing, data cleaning and structuring● Setting up Kafka● Designing the Application Layer
Nov 14- Nov 29	Project Implementation
	<ul style="list-style-type: none">● Integrating Spark with EKS/ECS● Structuring the queries from Spark on DynamoDB● Kafka processing● Kafka-Application Layer integration● Application Layer● Application Layer-EKS/ECS Integration● Writing Unit test cases
Nov 30 - Dec 10	<ul style="list-style-type: none">● Project Completion● Documentation● Integration testing● Performance Measure

TASK DIVISION AND TIMELINE

S. No.	Name	Tasks	Timeline
1	Akriti Kapur	<ul style="list-style-type: none">• Defining Architecture• Overlooking preprocessing, data cleaning and structuring• Kafka processing• Structure the queries• Creating test cases to produce “real-time” data	<ul style="list-style-type: none">• Week 0• Week 1• Week 2• Week 2• Week 3
2	Amith Gopal	<ul style="list-style-type: none">• Identifying Challenges• Containerization using EKS/ECS• Integration of Spark with EKS/ECS• Integration of Application Layer with EKS/ECS• Simple queries to test Spark-dynamoDB	<ul style="list-style-type: none">• Week 0• Week 1• Week 2• Week 3• Week 3
3	Sowmya	<ul style="list-style-type: none">• Identifying the AWS services• Containerization using EKS/ECS• Integration of Spark with EKS/ECS• Integration of Application Layer with EKS/ECS• Documentation	<ul style="list-style-type: none">• Week 0• Week 1• Week 2• Week 3• Week 3
4	Tarunianand	<ul style="list-style-type: none">• Finding datasets• Working with Akriti on Kafka processing• Developing application layer (Django etc.)• Creating test cases to produce “real-time” data• Documentation and Demonstration prep	<ul style="list-style-type: none">• Week 0• Week 1• Week 2• Week 3• Week 3

CONCERN

- Since our project is heavily dependent on the AWS services, we are concerned about the potential cost incurred by the time we complete the project.