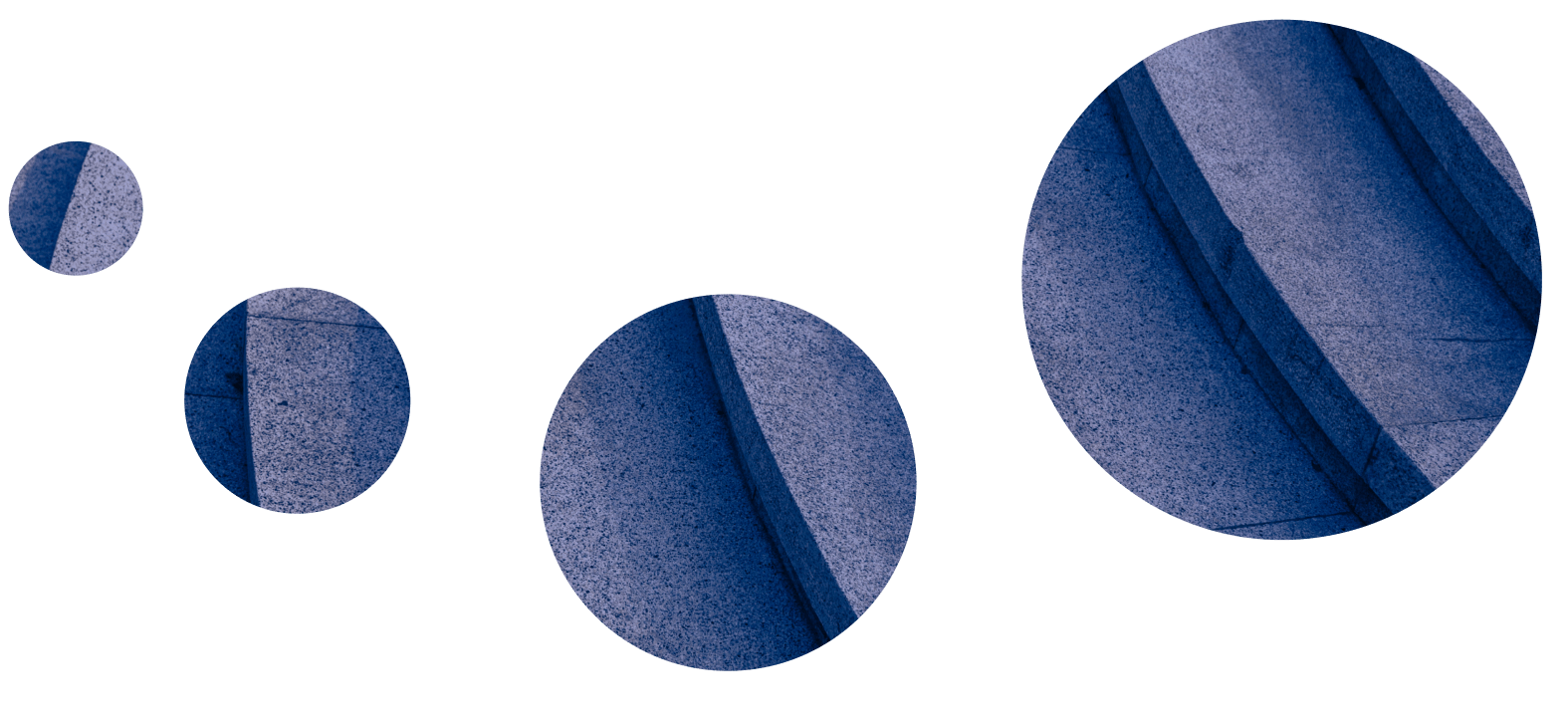
OPEN LPEN LAB – II:



Team: Technical writing and Documentation team (TWD)

**Glossary for Big Data Analytics**

**Big Data:** Refers to extremely large and complex data sets that are too large to be processed by traditional data processing applications.

**Hadoop:** An open-source framework for distributed storage and processing of big data on clusters of commodity hardware.

**MapReduce:** A programming model for processing big data that is widely used with Hadoop.

**HDFS (Hadoop Distributed File System):** The default file system used by Hadoop to store and process large data sets.

**NoSQL:** A class of database management systems that do not use traditional relational database management systems.

**Data Warehousing:** The process of collecting, storing, and organizing large data sets to be analyzed and mined for useful information.

**Data Mining:** The process of discovering patterns and knowledge from large data sets.

**Data Visualization:** The process of representing data in graphical or pictorial form for easier understanding and analysis.

**Machine Learning:** A field of computer science that focuses on the development of algorithms that can learn from and make predictions on data.

**Predictive Analytics:** The process of using data, statistical algorithms, and machine learning techniques to identify the likelihood of future outcomes based on historical data.

**Streaming Data:** Refers to data that is generated continuously, in real-time, and requires immediate processing.

**Spark:** An open-source, distributed computing system for processing big data that is designed to be fast and general-purpose.

**Hive:** An open-source data warehousing and querying tool for Hadoop that provides a SQL-like interface for data analysis.

**Pig:** An open-source platform for analyzing large data sets that provides a high-level language for expressing data analysis programs.

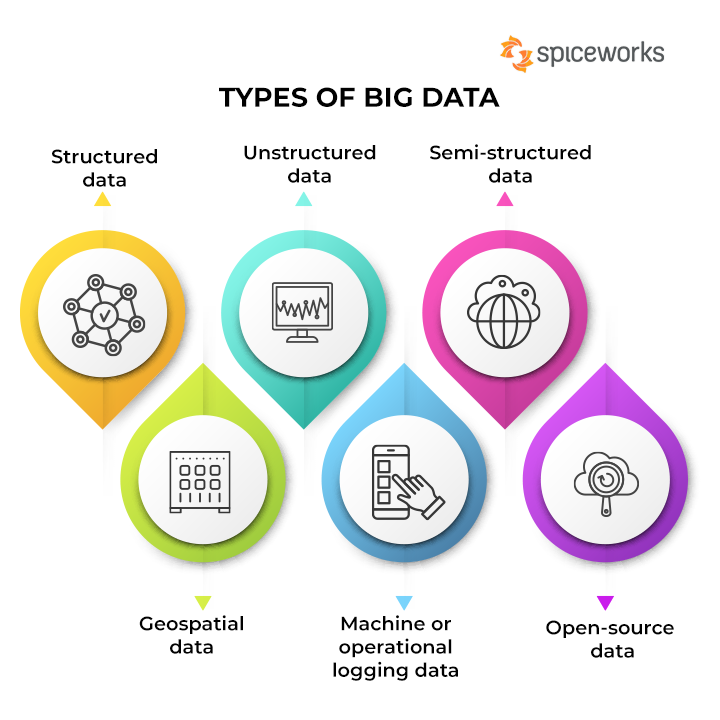
**NoSQL Databases:** Databases that do not use traditional relational database management systems, and are optimized for handling large amounts of unstructured data. Examples include MongoDB, Cassandra, and CouchDB.

**What is Big Data?**

Big data refers to the extremely large and complex data sets that are generated by organizations and individuals in increasing amounts. These data sets are typically too large and too complex to be processed and analyzed using traditional data processing applications and tools.

Big data often includes a mixture of structured and unstructured data, such as text, images, videos, and social media posts, and can come from a variety of sources, including social media, transactional systems, and machine logs. The growth of big data has been driven by the increasing volume of data generated by individuals, businesses, and society, as well as the declining cost of data storage and the increasing availability of powerful computing resources.

Big data presents both challenges and opportunities. On the one hand, big data can be difficult to process and analyze, requiring modern technologies and techniques to make sense of the vast amount of information. On the other hand, big data also provides organizations with new insights and opportunities to improve their operations, make better decisions, and gain a competitive advantage.



**Big Data in real-world**

Let us investigate a real time application of Big data.

The media and entertainment industry are one of the key sectors that is leveraging the power of big data to drive innovation and improve the customer experience. Here are some ways in which big data is being used in the media and entertainment industry:

1. Personalized recommendations: Streaming services such as Netflix and Amazon Prime use big data to provide personalized recommendations to users based on their viewing history and preferences. This helps improve the user experience by offering content that is more likely to be of interest to the user.
2. Customer analytics: Media companies are using big data to better understand their customers, including their behaviour, preferences, and buying habits. This information is used to develop targeted marketing campaigns, improve product offerings, and enhance the overall customer experience.
3. Content creation: Big data is being used to inform the creation of latest content in the media and entertainment industry. For example, movie studios use big data to analyze audience preferences and make decisions about the types of movies to produce.
4. Social media monitoring: Media companies are using big data to monitor and analyze social media activity, including user-generated content, to better understand audience sentiment and preferences. This information is used to inform content creation, marketing campaigns, and engagement strategies.
5. Ad targeting: Big data is being used to target advertisements more effectively by analyzing viewer behaviour, demographic information, and viewing history. This helps ensure that advertisements reach the right audience and are more likely to be effective.

Overall, big data is playing a key role in transforming the media and entertainment industry, providing companies with valuable insights and helping them better understand and serve their customers.

**Limitation of the traditional solutions for Big Data problem**

Traditional solutions for processing and analyzing big data have several limitations, including:

*Scalability:* Traditional data processing applications and tools are often not designed to handle the sheer volume and velocity of big data, making it difficult to scale processing and storage resources to meet growing demand.

*Complexity:* Big data often includes a mixture of structured and unstructured data, making it difficult to use traditional data processing tools and techniques to extract meaningful insights.

*Latency:* Traditional data processing solutions often involve batch processing, which can result in significant latency when analyzing big data. This can make it difficult to obtain real-time insights from big data.

*Cost:* Traditional data processing solutions can be expensive, especially when dealing with big data. This can make it difficult for organizations with limited resources to take advantage of the opportunities presented by big data.

*Inflexibility:* Traditional data processing solutions are often rigid and difficult to modify, making it difficult to quickly adapt to changing data requirements.

*Security:* Traditional solutions may not have adequate security measures in place to protect sensitive data.

*Integration:* Traditional solutions may have difficulty integrating with other systems and technologies, especially in a Big Data context.

These limitations are addressed in the modern technologies and solutions specifically designed for big data, such as Hadoop, Spark, and NoSQL databases, which offer a more flexible, scalable, and cost-effective way to process and analyze big data.

**How Hadoop solves Big Data problems?**

Hadoop addresses the challenges of big data by providing a scalable, distributed computing framework that can process large amounts of data in parallel across a cluster of commodity hardware.



*Scalability:* Hadoop can scale to handle petabyte-scale data sets by adding nodes to the cluster. The more nodes added, the more processing power and storage capacity is added to the cluster, allowing Hadoop to handle increasing amounts of data.

*Data Processing:* Hadoop uses the MapReduce programming model to process large data sets in parallel, dividing the input data into smaller chunks and processing each chunk on a separate node in the cluster. This allows for efficient processing of large data sets.

*Data Storage:* Hadoop's HDFS (Hadoop Distributed File System) provides a scalable, fault-tolerant solution for storing large amounts of data. HDFS stores data across multiple nodes in the Hadoop cluster, providing built-in replication and redundancy to ensure that data is not lost in case of node failure.

*Data Access:* Hadoop provides several data access and data processing tools, including Pig and Hive, that allow for data to be analyzed and processed in a variety of ways, including batch processing and real-time processing.

*Cost-Effective:* Hadoop runs on commodity hardware, reducing the cost of a large-scale data processing solution compared to traditional proprietary systems.

Hadoop provides a scalable, distributed computing framework for processing and storing large amounts of data, allowing organizations to address the challenges of big data by providing a cost-effective solution for handling large-scale data processing and analysis.

**Overview of Hadoop ecosystem**

The Hadoop ecosystem refers to the collection of open-source tools and technologies that work together to provide a comprehensive solution for Big Data processing and storage. The Hadoop ecosystem includes several different components, each of which serves a specific purpose:

*HDFS (Hadoop Distributed File System):* A distributed file system that stores data across multiple nodes in a Hadoop cluster.

*MapReduce:* A programming model for processing large data sets that is the foundation of Hadoop processing.

*YARN (Yet Another Resource Negotiator):* A resource management system in Hadoop that manages resources for the processing of data in a Hadoop cluster.

*Hive:* A data warehousing and SQL-like query language for Hadoop.

*Pig:* A high-level platform for creating MapReduce programs used with Hadoop.

*HBase:* A NoSQL database for Hadoop that provides real-time access to data.

*Spark:* An open-source, big data processing engine for use in Hadoop clusters.

*Mahout:* A machine learning library for Hadoop.

*Zookeeper:* A coordination service for distributed systems that is used in Hadoop to coordinate actions between nodes.

*Flume:* A service for collecting, aggregating, and moving large amounts of log data into Hadoop.

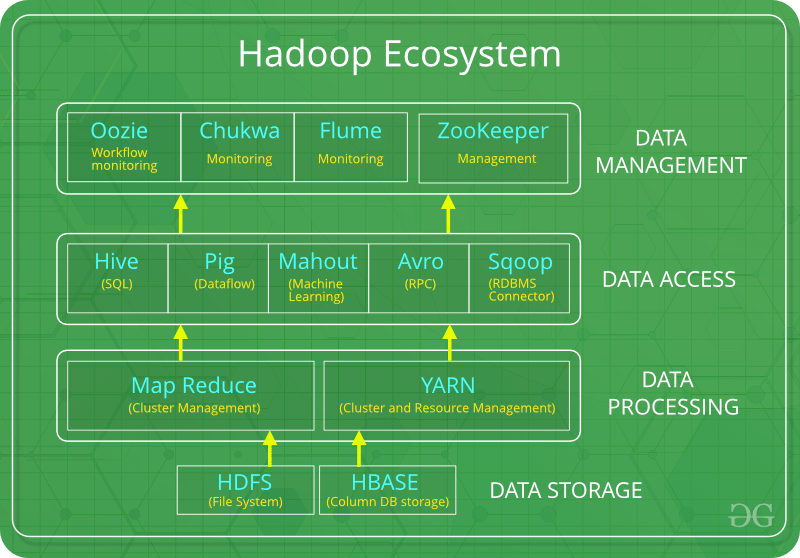
*Sqoop:* A tool for transferring data between Hadoop and structured data stores such as relational databases.

*Oozie:* A workflow management system for Hadoop that coordinates and manages Hadoop jobs.

*Ambari:* A web-based tool for managing and monitoring Hadoop clusters.

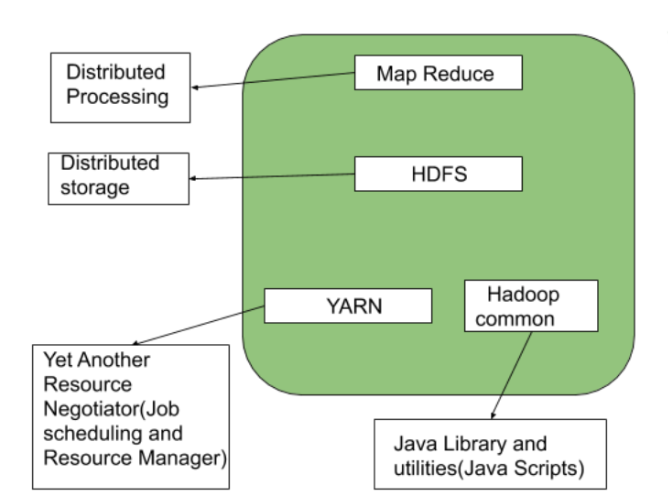
*Avro:* A data serialization system for Hadoop that provides a compact binary format for data and schema.

*Parquet:* A columnar storage format for Hadoop that provides efficient storage and processing for big data.



**Details of Hadoop Architecture**

The Hadoop architecture is based on a master/slave design, where the master node manages the overall system and the slave nodes perform the actual processing. The main components of the Hadoop architecture are:



*HDFS (Hadoop Distributed File System):* HDFS is a scalable and fault-tolerant storage system that is responsible for storing and managing large data sets in a distributed manner. It consists of a NameNode, which acts as the master node, and multiple DataNodes, which store the actual data.

*YARN (Yet Another Resource Negotiator):* YARN is the resource management system in Hadoop that enables multiple processing engines to run on the same data stored in HDFS. It consists of a ResourceManager, which acts as the master node, and multiple NodeManagers, which are responsible for managing resources on individual nodes.

*MapReduce:* MapReduce is the processing engine in Hadoop that performs the actual data processing. It consists of a JobTracker, which acts as the master node, and multiple TaskTrackers, which perform the processing tasks.

*HDFS Client:* The HDFS client is responsible for communicating with the HDFS NameNode and DataNodes to read and write data to the HDFS.

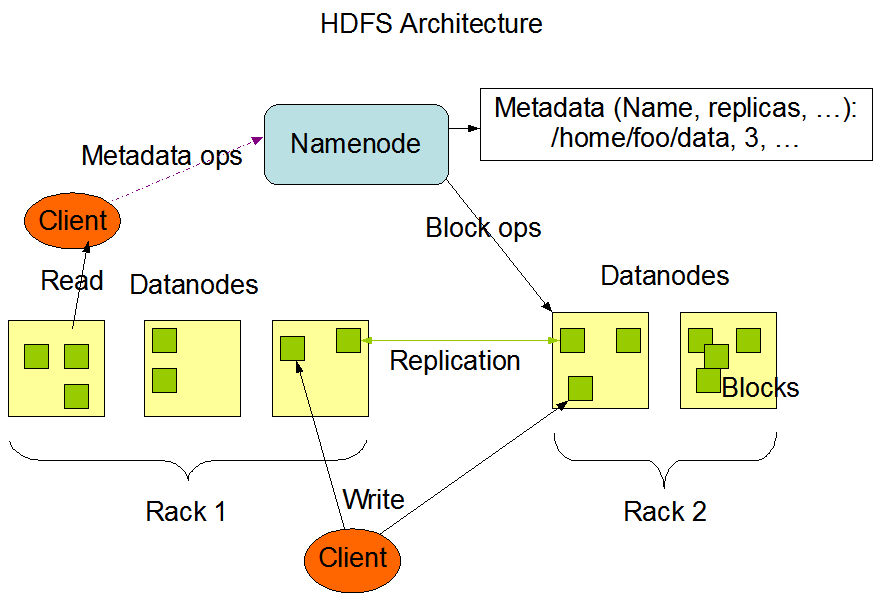
*Application Master:* The Application Master is responsible for negotiating resources with the ResourceManager and executing tasks on the NodeManagers.

*Task Tracker:* The TaskTracker is responsible for executing tasks assigned by the JobTracker and reporting the progress back to the JobTracker.

*DataNode:* The DataNode is responsible for storing the actual data in HDFS and serving read and write requests from the HDFS client.

This architecture allows Hadoop to process large amounts of data in parallel, providing high performance and scalability, while also ensuring fault tolerance through data replication and node redundancy.

**What is Hadoop File System?**



Hadoop Distributed File System (HDFS) is a scalable, fault-tolerant and distributed file system designed to store large data sets. It runs on commodity hardware and is part of the Apache Hadoop ecosystem.

HDFS stores data across multiple nodes in a cluster, allowing it to handle data much larger than a single machine can store. It also replicates data across multiple nodes, ensuring that the data is available even if a node fails.

The HDFS architecture consists of two main components: NameNode and DataNode. The NameNode is responsible for managing the file system metadata, while the DataNode stores the actual data blocks.

HDFS also supports the creation of multiple replicas of data blocks, ensuring data availability and reliability in the case of node failures.

Overall, HDFS is optimized for high throughput, write-once-read-many access patterns and is well suited for storing large files, such as images, audio and video files, and logs.

**What is Map** **reduce?**

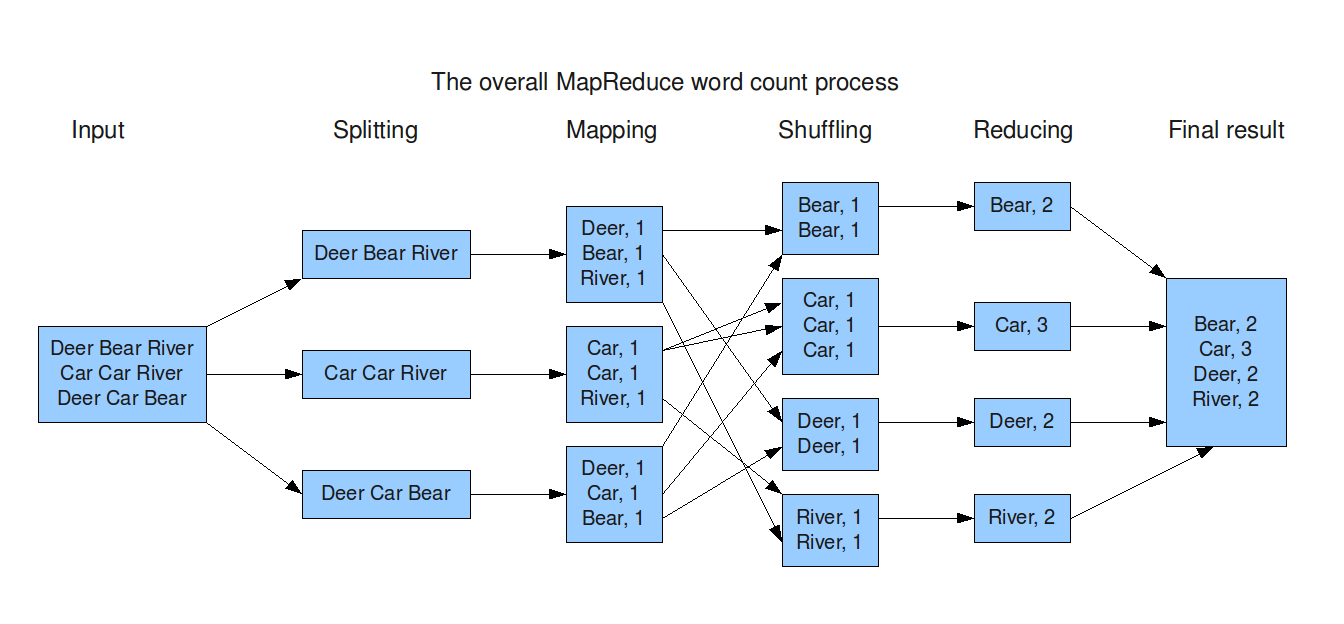
MapReduce is a programming model used to process large data sets in parallel and distribute processing across multiple nodes in a Hadoop cluster. MapReduce is well suited for processing large amounts of structured and semi-structured data, such as log files and other forms of data generated by user activity. It is often used in big data analytics applications, such as data warehousing, machine learning, and scientific simulations.

**Working of Map reduce**

The working of MapReduce can be explained in the following steps:

1. Input Splitting: The input data is split into chunks, called input splits, and each split is assigned to a mapper task.
2. Mapping: The mapper task processes the input split and produces intermediate key-value pairs, which are then passed to the shuffle and sort phase.
3. Shuffle and Sort: The intermediate key-value pairs are sorted and grouped by key. This phase is responsible for redistributing the data so that all the values associated with a given key are processed by the same reducer.
4. Reducing: The reducer task processes the intermediate key-value pairs and aggregates the values associated with the same key. The result of this phase is the final output of the MapReduce job.
5. Output Writing: The final output of the MapReduce job is written to the output file, which can be stored in a distributed file system like HDFS.

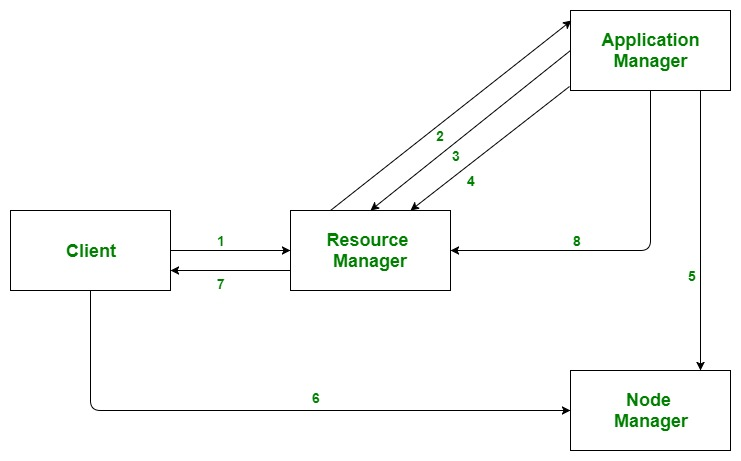
This process helps to handle large data sets efficiently and provides scalability by distributing the processing across multiple nodes.



**What is YARN?**

YARN (Yet Another Resource Negotiator) is a resource management system in Hadoop that enables multiple processing engines to run on the same data stored in HDFS (Hadoop Distributed File System). It acts as the central authority for resource management and job scheduling in Hadoop, providing a common platform for running various data processing applications.

**YARN Architecture**



1. Client**:** It submits map-reduce jobs.
2. Resource Manager: It is the master daemon of YARN and is responsible for resource assignment and management among all the applications. Whenever it receives a processing request, it forwards it to the corresponding node manager and allocates resources for the completion of the request accordingly. It has two major components:
   * Scheduler: It performs scheduling based on the allocated application and available resources.
   * Application manager: It is responsible for accepting the application and negotiating the first container from the resource manager.
3. Node Manager: It take care of individual node on Hadoop cluster and manages application and workflow and that particular node. Its primary job is to keep-up with the Resource Manager. It registers with the Resource Manager and sends heartbeats with the health status of the node.
4. Application Master: An application is a single job submitted to a framework. The application master is responsible for negotiating resources with the resource manager, tracking the status and monitoring progress of a single application. The application master requests the container from the node manager by sending a Container Launch Context(CLC) which includes everything an application needs to run. Once the application is started, it sends the health report to the resource manager from time-to-time.
5. Container: It is a collection of physical resources such as RAM, CPU cores and disk on a single node. The containers are invoked by Container Launch Context(CLC) which is a record that contains information such as environment variables, security tokens, dependencies etc.

Contributors:

1. [@Nikhil Raj P - [CB.SC.I5DAS20023], Amrita Vishwa Vidyapeetham.](mailto:cb.sc.i5das20023@cb.students.amrita.edu)
2. [@Navami Vipith Kumar - [CB.SC.I5DAS20022], Amrita Vishwa Vidyapeetham.](mailto:cb.sc.i5das20022@cb.students.amrita.edu)
3. [@Arya C Prajeesh - [CB.SC.I5DAS20006], Amrita Vishwa Vidyapeetham.](mailto:cb.sc.i5das20006@cb.students.amrita.edu)
4. [@Navaneeth P - [CB.SC.I5DAS20122], Amrita Vishwa Vidyapeetham.](mailto:cb.sc.i5das20122@cb.students.amrita.edu)
5. [@Meera T Ramesh - [CB.SC.I5DAS20119], Amrita Vishwa Vidyapeetham.](mailto:cb.sc.i5das20119@cb.students.amrita.edu)
6. [@Aarsha Nair K S - [CB.SC.I5DAS20001], Amrita Vishwa Vidyapeetham.](mailto:cb.sc.i5das20001@cb.students.amrita.edu)

Facilitator:

1. [@Guru Prakash J - [](mailto:j_guruprakash@cb.amrita.edu)[CSE]](mailto:j_guruprakash@cb.amrita.edu)