ORDER VISIBILTY DASHBOARD

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

BY
Ishika Hadpawat
EN18IT301048

Under the Guidance of
Prof. Prasanna Kapse

Department of Computer Science & Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

April 2022

ORDER VISIBILTY DASHBOARD

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

BY

Ishika Hadpawat
EN18IT301048

Under the Guidance of
Prof. Prasanna Kapse

Department of Computer Science & Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

April 2022

Report Approval

The project work **"Order Visibility Dashboard"** is hereby approved as a creditable study of an engineering/computer application subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approved any statement made, opinion expressed, or conclusion drawn there in; but approve the "Project Report" only for the purpose for which it has been submitted.

Internal Examiner

Name:

Designation

Affiliation

External Examiner

Name:

Designation

Affiliation

Declaration

I/We hereby declare that the project entitled **"Order Visibility Dashboard"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Computer Science' completed under the supervision of **Prof. Prasanna Kapse,** Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, I/we declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Signature and name of the student(s) with date

Ishika Hadpawat

EN18IT301048

DATE :

Certificate

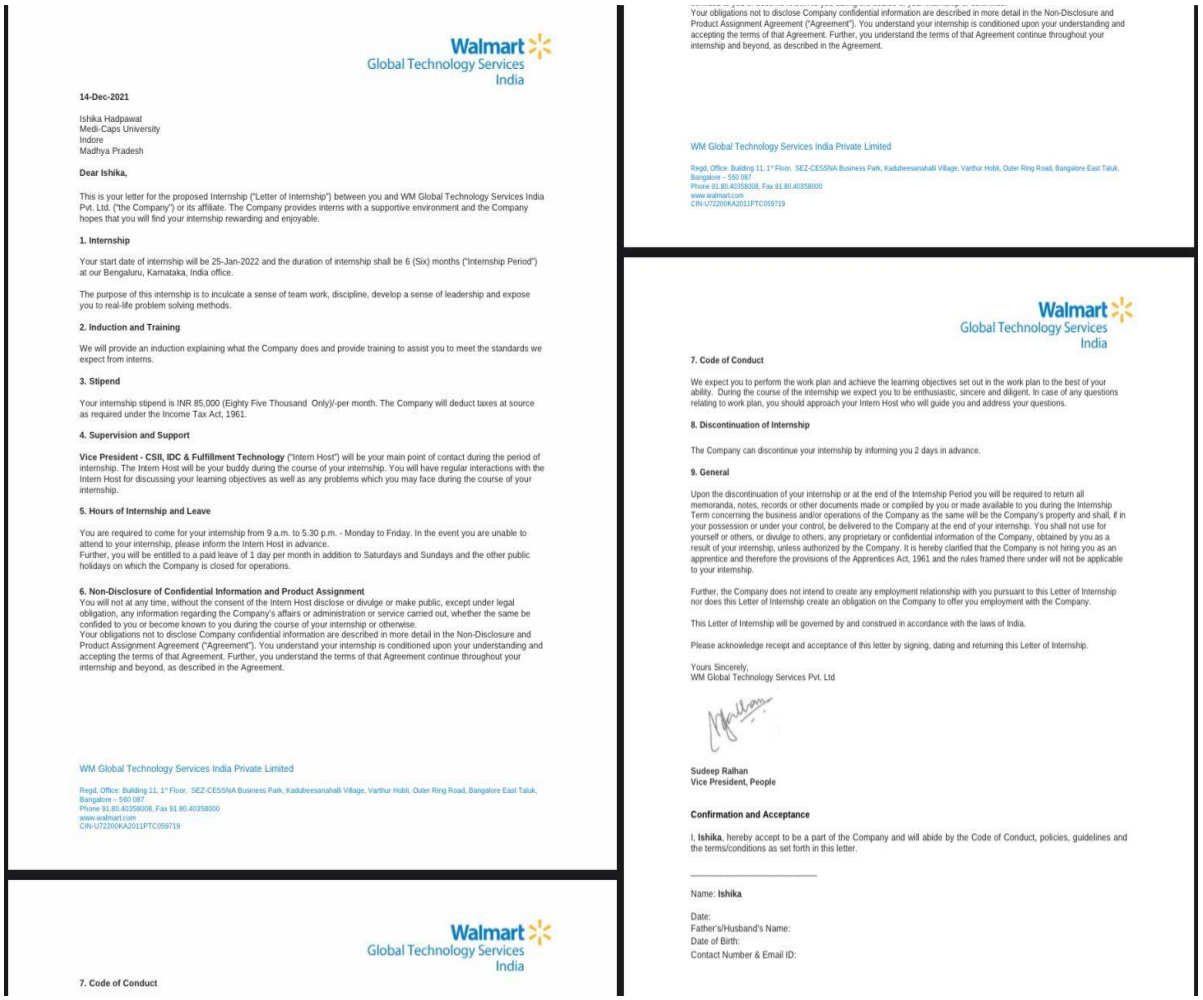
I/We, **Prof. Prasanna Kapse** certify that the project entitled **"Order Visibility Dashboard"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology by **Ishika Hadpawat** is the record carried out by him/them under my/our guidance and that the work has not formed the basis of award of any other degree elsewhere.

_____	_____
<Name of Internal Guide>	<Name of External Guide (If any)>
<Name of the Department>	<Name of the Department>
Medi-Caps University, Indore	Name of the Organization

_____

Dr. Pramod S. Nair
Head of the Department
Computer Science & Engineering
Medi-Caps University, Indore

Offer Letter of the Project work-II/Internship



Walmart
Global Technology Services
India

14-Dec-2021

Ishika Hadpawat
Medi-Caps University
Indore
Madhya Pradesh

Dear Ishika,

This is your letter for the proposed Internship ("Letter of Internship") between you and WM Global Technology Services India Pvt. Ltd. ("the Company") or its affiliate. The Company provides interns with a supportive environment and the Company hopes that you will find your internship rewarding and enjoyable.

1. Internship

Your start date of internship will be 25-Jan-2022 and the duration of internship shall be 6 (Six) months ("Internship Period") at our Bengaluru, Karnataka, India office.

The purpose of this internship is to inculcate a sense of team work, discipline, develop a sense of leadership and expose you to real-life problem solving methods.

2. Induction and Training

We will provide an induction explaining what the Company does and provide training to assist you to meet the standards we expect from interns.

3. Stipend

Your internship stipend is INR 85,000 (Eighty Five Thousand Only)/-per month. The Company will deduct taxes at source as required under the Income Tax Act, 1961.

4. Supervision and Support

Vice President - CSII, IDC & Fulfillment Technology ("Intern Host") will be your main point of contact during the period of internship. The Intern Host will be your buddy during the course of your internship. You will have regular interactions with the Intern Host for discussing your learning objectives as well as any problems which you may face during the course of your internship.

5. Hours of Internship and Leave

You are required to come for your internship from 9 a.m. to 5.30 p.m. - Monday to Friday. In the event you are unable to attend to your internship, please inform the Intern Host in advance.
Further, you will be entitled to a paid leave of 1 day per month in addition to Saturdays and Sundays and the other public holidays on which the Company is closed for operations.

6. Non-Disclosure of Confidential Information and Product Assignment
You will not at any time, without the consent of the Intern Host disclose or divulge or make public, except under legal obligation, any information regarding the Company's affairs or administration or service carried out, whether the same be confided to you or become known to you during the course of your internship or otherwise.
Your obligations not to disclose Company confidential information are described in more detail in the Non-Disclosure and Product Assignment Agreement ("Agreement"). You understand your internship is conditioned upon your understanding and accepting the terms of that Agreement. Further, you understand the terms of that Agreement continue throughout your internship and beyond, as described in the Agreement.

WM Global Technology Services India Private Limited

Regd. Office: Building 11, 1st Floor, SEZ-CESSNA Business Park, Kadubeesanahalli Village, Varthur Hobli, Outer Ring Road, Bangalore East Taluk, Bangalore – 560 087
Phone 91.80.40358008, Fax 91.80.40358000
www.walmart.com
CIN-U72200KA2011PTC059719

Your obligations not to disclose Company confidential information are described in more detail in the Non-Disclosure and Product Assignment Agreement ("Agreement"). You understand your internship is conditioned upon your understanding and accepting the terms of that Agreement. Further, you understand the terms of that Agreement continue throughout your internship and beyond, as described in the Agreement.

WM Global Technology Services India Private Limited

Regd. Office: Building 11, 1st Floor, SEZ-CESSNA Business Park, Kadubeesanahalli Village, Varthur Hobli, Outer Ring Road, Bangalore East Taluk, Bangalore – 560 087
Phone 91.80.40358008, Fax 91.80.40358000
www.walmart.com
CIN-U72200KA2011PTC059719

Walmart
Global Technology Services
India

7. Code of Conduct

Walmart
Global Technology Services
India

7. Code of Conduct

We expect you to perform the work plan and achieve the learning objectives set out in the work plan to the best of your ability. During the course of the internship we expect you to be enthusiastic, sincere and diligent. In case of any questions relating to work plan, you should approach your Intern Host who will guide you and address your questions.

8. Discontinuation of Internship

The Company can discontinue your internship by informing you 2 days in advance.

9. General

Upon the discontinuation of your internship or at the end of the Internship Period you will be required to return all memoranda, notes, records or other documents made or compiled by you or made available to you during the Internship Term concerning the business and/or operations of the Company as the same will be the Company's property and shall, if in your possession or under your control, be delivered to the Company at the end of your internship. You shall not use for yourself or others, or divulge to others, any proprietary or confidential information of the Company, obtained by you as a result of your internship, unless authorized by the Company. It is hereby clarified that the Company is not hiring you as an apprentice and therefore the provisions of the Apprentices Act, 1961 and the rules framed there under will not be applicable to your internship.

Further, the Company does not intend to create any employment relationship with you pursuant to this Letter of Internship nor does this Letter of Internship create an obligation on the Company to offer you employment with the Company.

This Letter of Internship will be governed by and construed in accordance with the laws of India.

Please acknowledge receipt and acceptance of this letter by signing, dating and returning this Letter of Internship.

Yours Sincerely,
WM Global Technology Services Pvt. Ltd

Sudeep Ralhan
Vice President, People

Confirmation and Acceptance

I, **Ishika**, hereby accept to be a part of the Company and will abide by the Code of Conduct, policies, guidelines and the terms/conditions as set forth in this letter.

_____

Name: **Ishika**

Date:
Father's/Husband's Name:
Date of Birth:
Contact Number & Email ID:

Completion certificate/Letter

Acknowledgements

I would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) Dileep K Patnayak,** Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) D K Panda,** Pro Vice Chancellor, **Dr. Suresh Jain,** Dean Faculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Dr. Pramod S. Nair** for his continuous encouragement for betterment of the project.

I express my heartfelt gratitude to my **External Guide, Mr. Manoj Kumar**, Project Lead, Honeywell Technology Solutions Lab Pvt. Ltd as well as to my Internal Guide, Dr**. Debendra Kumar Panda,** Professor, Department of Electronics Engineering, MU, without whose continuous help and support, this project would ever have reached to the completion.

I would also like to thank to my team at Honeywell Mr. Kishore Sandrana, Mr .Gaurav Vashisth, Mr. Jayaprakash Vendkadabathi, Ms. Shabna Vasudevan, Mr. Rajesh Kumar, Mr. Gopi Badarla, Ms. Asha Kuriakose who extended their kind support and help towards the completion of this project.
It is their help and support, due to which we became able to complete the design and technical report. Without their support this report would not have been possible.

Ishika Hadpawat
B.Tech. IV Year
Department of Computer Science & Engineering
Faculty of Engineering
Medi-Caps University, Indore

Abstract

An As a part of my under-graduate program, I joined Walmart Global Tech. Services Pvt. Ltd. on 25 January 2022 as a 6-month intern. In the initial stage of my internship, I went through several internal trainings and evaluations related to different technologies and software development. After our training we will be a part of the development team for the internal project of our company i.e., to make a website for our Engineer Team, which would consist of all the events associated with a particular order that we provide to our customers basis on the orders they place, including other functionalities as well.

Walmart, the technology arm of the US retail giant, is also one of the top retail tech companies in India. It is a product-based company, uses next generation technology to transform Walmart into retail destination of choice.

Walmart's main objective is to capture the brand's energy as a human-led, tech-empowered innovator that positively impacts the lives of millions of associates and customers globally.

Walmart stands on - SAVE MONEY, LIVE BETTER

Table of Contents

Abbreviations

Abbreviations	Full Form
CLI	Command Line Interface
IDE	Integrated Development Environment
SDK	Software Development Kit
DBMS	Database Management System
RDBMS	Relational Database Management System
NoSQL	Non- Structured Query Language
JS	JavaScript
JSX	JavaScript XML
DOM	Document Object Model
SDLC	Software Development LifeCycle

Chapter-1

1.1 Internal Training

SAVE MONEY, LIVE BETTER

The training portals we were allowed to access in Walmart Global Tech Services Pvt. Ltd are a set of virtual led sessions by professionals and a vibrant tech group, whose pursuit is progress for people everywhere. That's why we take a closer look at things, learn them, spread them and ask questions and think ahead.

We spend most of a significant portion of our time coding, documenting and training individuals. We find answers to the most pressing questions which also drive us to be lifelong learners. We work on live projects for clients of different domains. We keep using phrase that is our constant motivation, i.e., learning regularly is the most important thing for us.

The initial training was for about one and half months, which comprised of several important topics that are required for the understanding the pro1.1cess flow of software development and other required topics.

1.2 Job Responsibilities

Other than the technical training that I went through, I was also trained for some other responsibilities-

- ICODE Training: ICODE training is a training that every new individual go through after hiring in Walmart. It takes us basically through all the basic tech stack and technology on which company rely on and the team finally helps to set our devices for use.

- Professional Training: After we hire the candidates, we provide them with our Professional Training if required. We provide training in several technology domains such as Basics of Software Development, Python - Basics to Advance, Java – Basics to Advance, Web Development, on basis of the project they are assigned to, and, we arrange seminars and Bootcamps throughout the year to spread knowledge as much as possible.

- Remote Projects: We dedicatedly work on many remote projects of our company, which are assigned to us after we reach a certain amount of seniority. Before handling personal projects, we are assigned support projects, in which we support other seniors in their respective remote projects.

1.3 Components of the Initial Training

The initial training of one and half months comprised of mainly 4 basic components under which all the topics were covered. The components were:

- Kafka
- Elasticsearch
- Spring Boot
- React JS
- Technology Stack

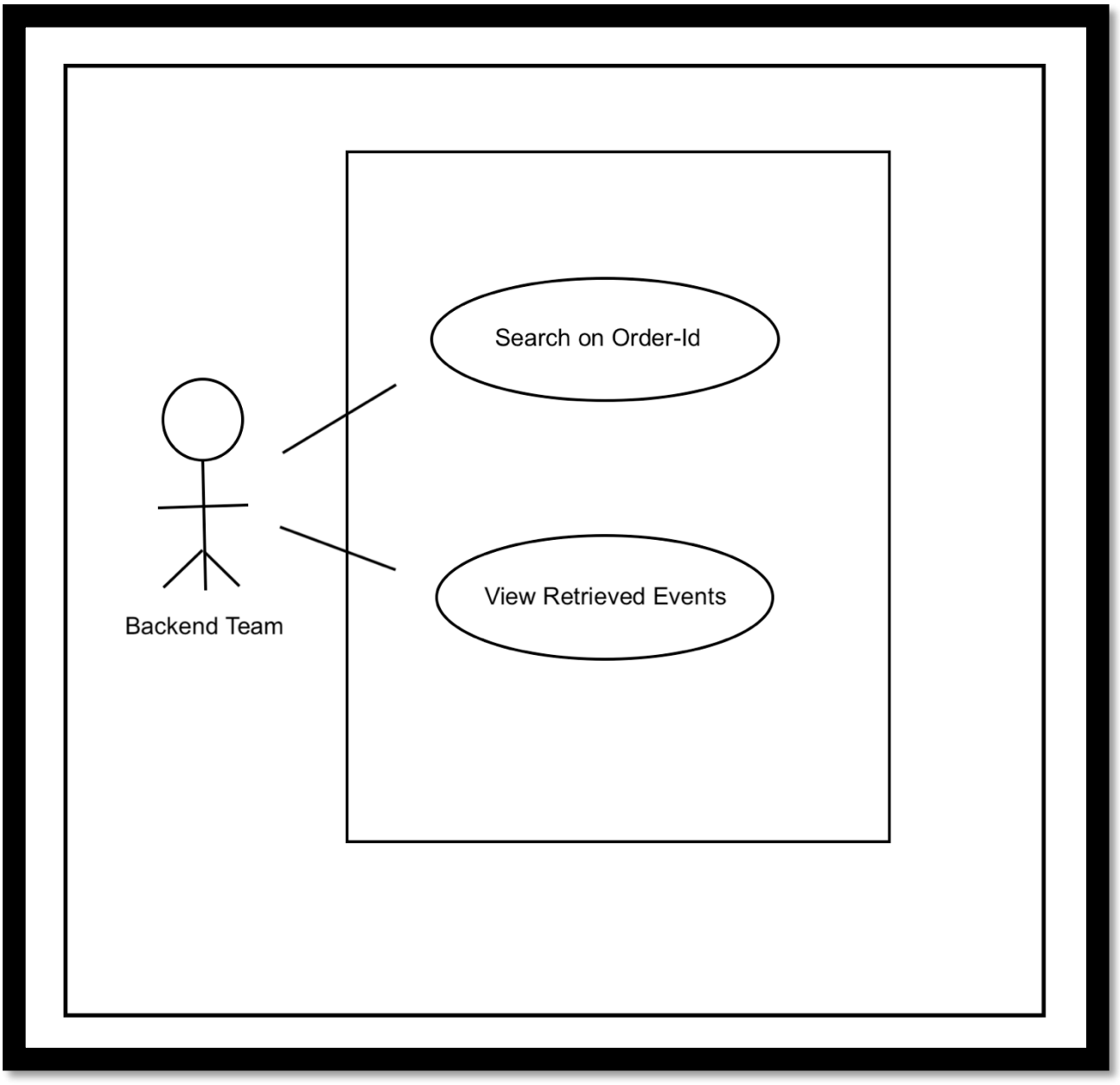
Other than these I also learned and extensively worked on JavaScript, HTML, CSS.

Chapter-2

System Analysis

2.1 Use Case Diagram

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors and their relationships. It models the task, services, and functions required by a system /subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system. The main purpose of a use case diagram is to portray the dynamic aspect of a system.



2.1.1 Use Case Description

Use Case ID : 1

Search on Order Id

- **Brief Description**

  This use case allows a developer from backend team to view all his requirements on basis of the order Id, for a particular order that moved into the system.

- **Flow Of Events**

- **Basic Flow**

  1. The user enters an order id, on which he/she wants to perform a search on.

  2. The system reads the entered order id and passes it to backend side to fetch results.

- **Alternative Flow**

  1. Invalid Order Id

     If, in the Basic Flow, the user enters an invalid order id that neither exist nor have any data associated with it, then the system displays an error message stating “Order Id Doesn’t Exist”. The user can either choose to return to the beginning of basic flow or can exit, at which point the use case ends.

- **Special Requirements**

  None

- **Pre-Conditions**

  The system is in the search state and has the search screen displayed.

- **Post-Conditions**

  If the use case is successful, the user was able to search his requirements. If not, the system state is unchanged.

- **Extension Points**

  None

Use Case Id : 2

View Retrieved Events

- **Brief Description**

  This use case allows a developer from backend team to view all his requirements that he searched for, on order id.

- **Flow Of Events**

- **Basic Flow**

  1. The system displays the required fetched result to the user.

- **Alternative Flows**

  None

- **Special Requirements**

  None

- **Pre-Conditions**

  The system is in the display state and has results displayed.

- **Post-Conditions**

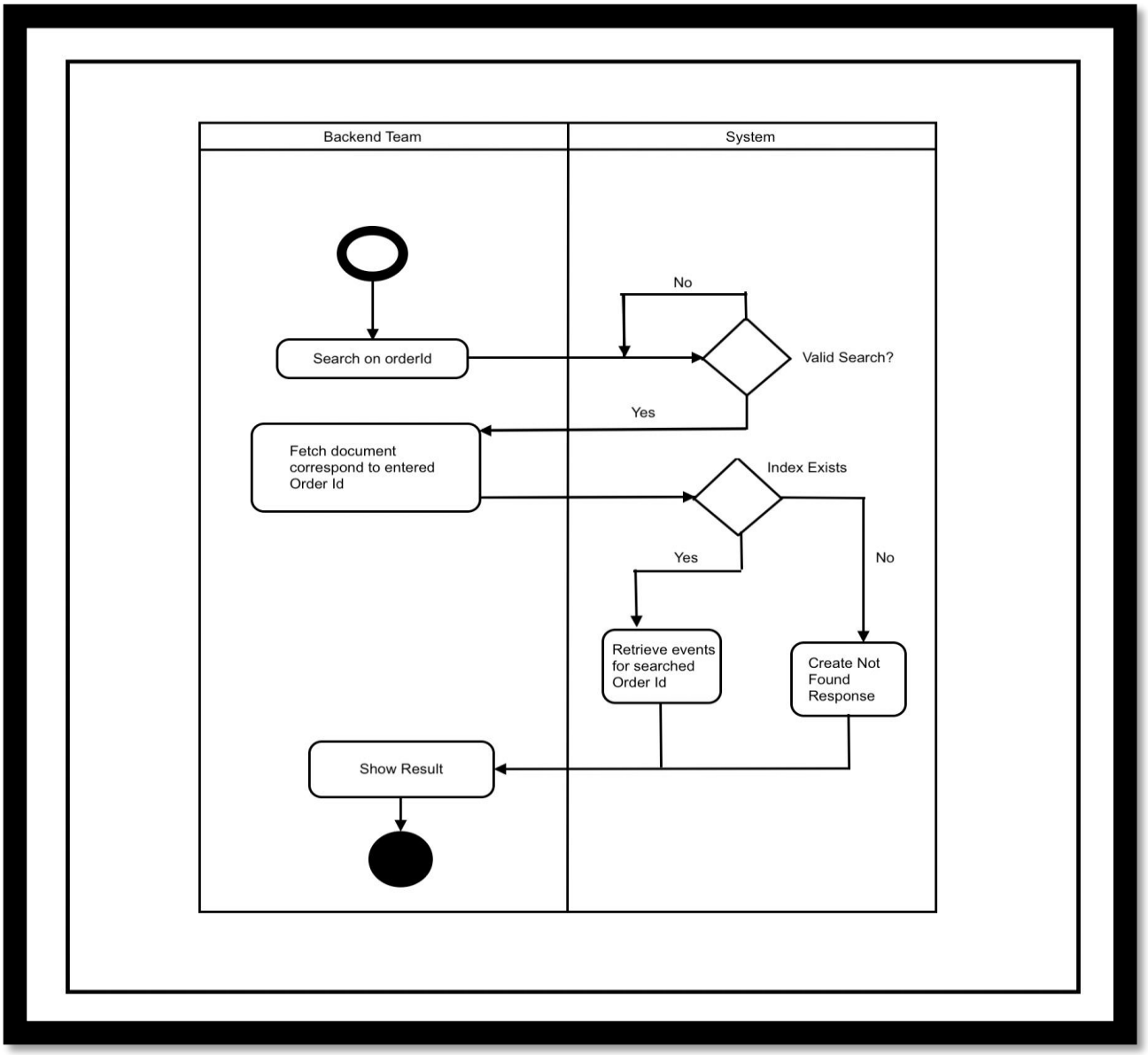
  If the use case is successful, the user was able to view the results. If not, the system state is unchanged.

- **Extension Points**

  None

2.2 Activity Diagram

The activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities. The activity diagram helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kind of flows, the activity diagram has come up with fork, join etc.

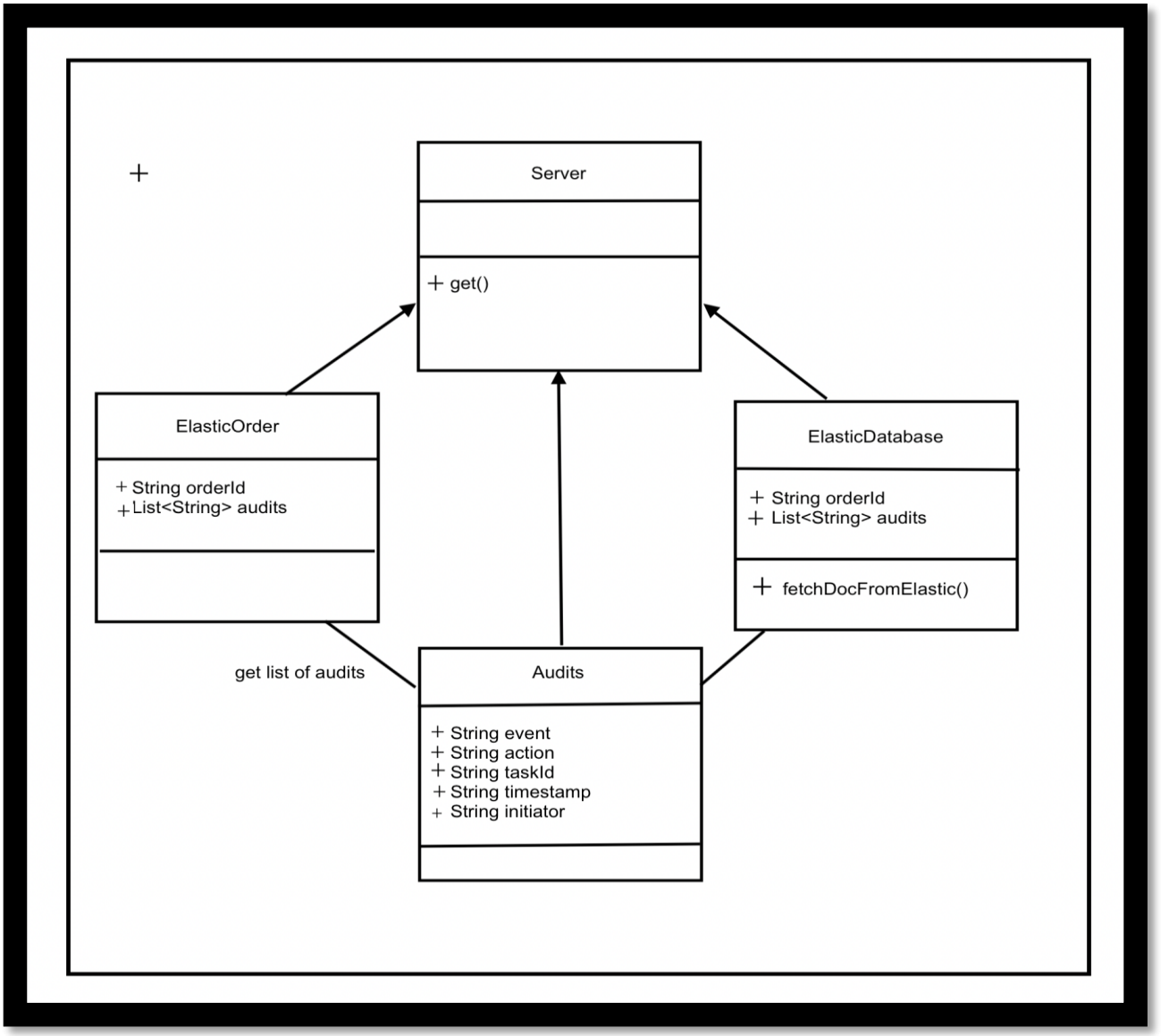


2.3 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

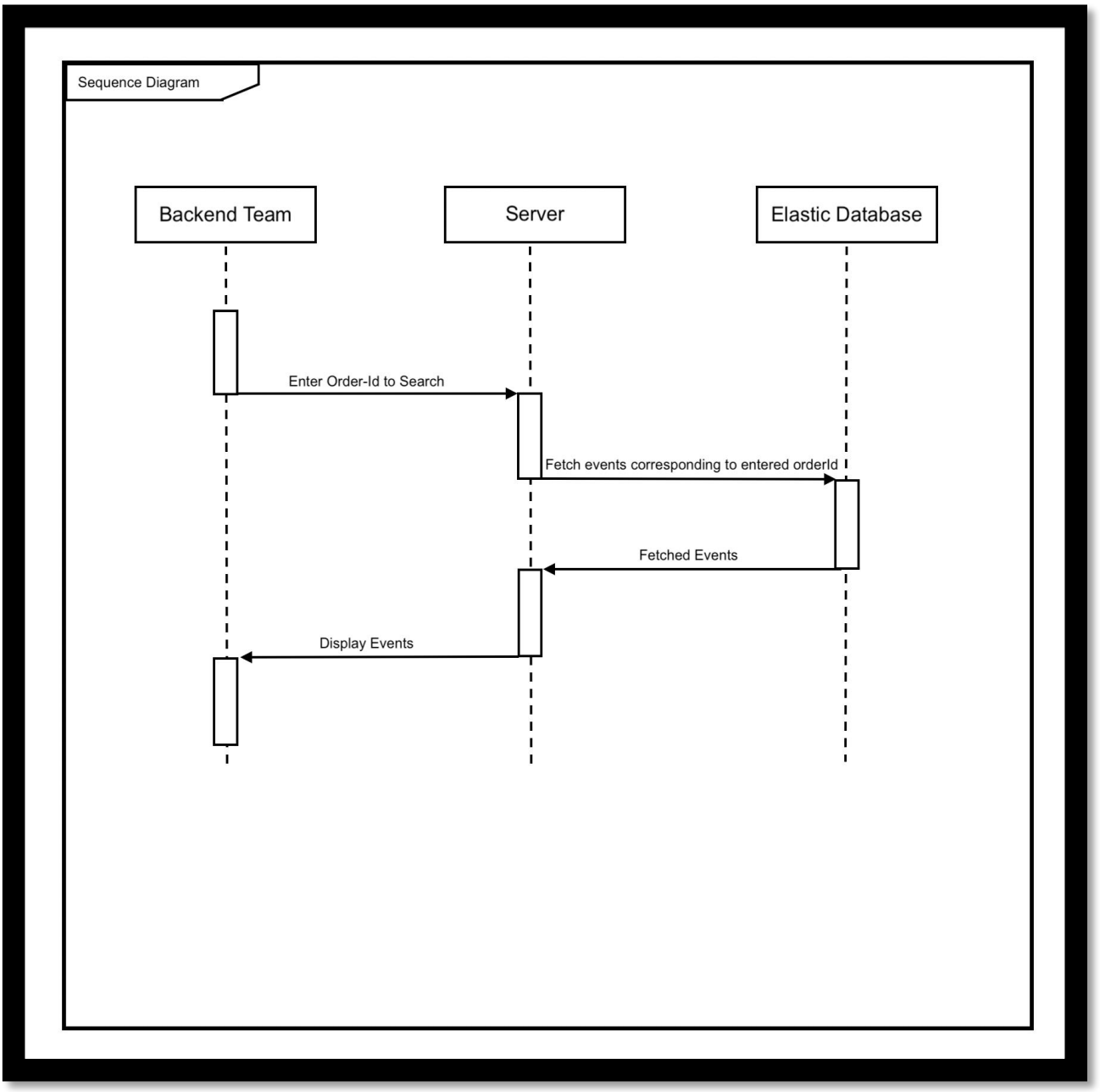
Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.



2.4 Sequence Diagram

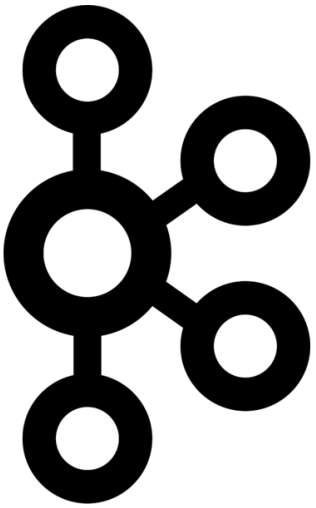
The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extend across the bottom of the page. It incorporates the iterations as well as branching.



Chapter-3

Kafka

2.1 Kafka - Introduction



Apache Kafka Tutorial provides the basic and advanced concepts of Apache Kafka. This tutorial is designed for both beginners and professionals.

Apache Kafka is an open-source stream-processing software platform which is used to handle the real-time data storage. It works as a broker between two parties, i.e., a sender and a receiver. It can handle about trillions of data events in a day.

Apache Kafka is a software platform which is based on a distributed streaming process. It is a publish-subscribe messaging system which let exchanging of data between applications, servers, and processors as well. Apache Kafka was originally developed by **LinkedIn**, and later it was donated to the Apache Software Foundation. Currently, it is maintained by **Confluent** under Apache Software Foundation. Apache Kafka has resolved the lethargic trouble of data communication between a sender and a receiver.

What is a messaging system?

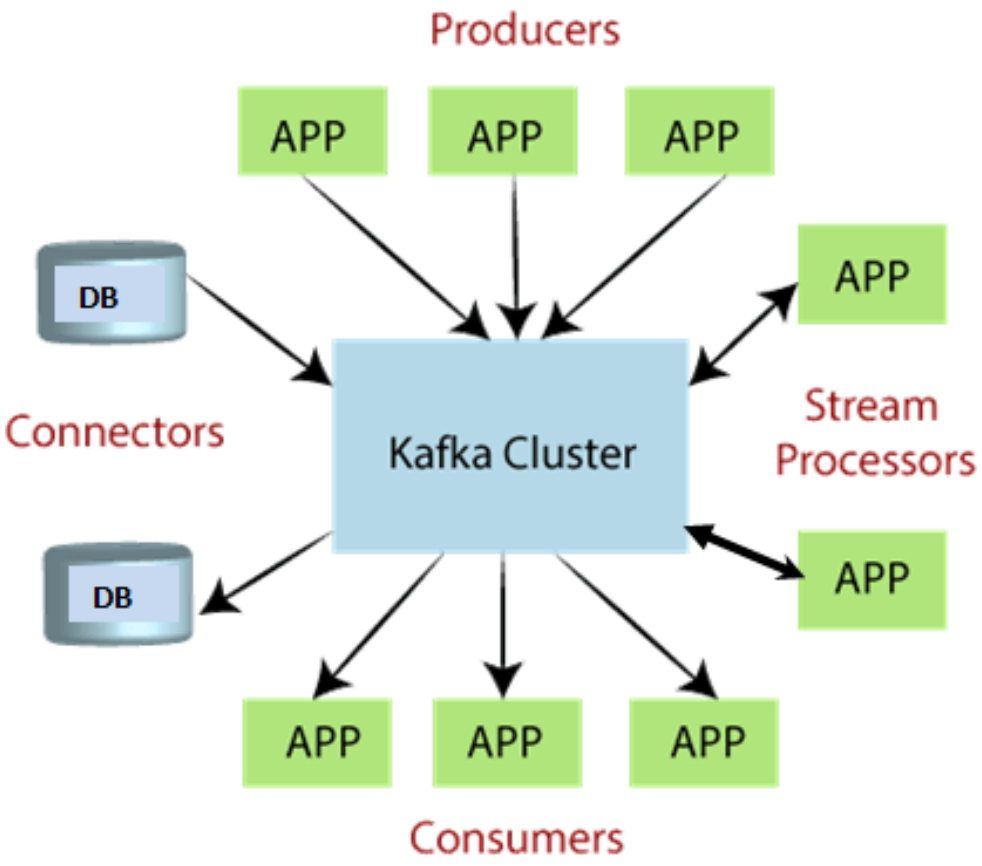
A messaging system is a simple exchange of messages between two or more persons, devices, etc. A publish-subscribe messaging system allows a sender to send/write the message and a receiver to read that message. In Apache Kafka, a sender is known as a **producer** who publishes messages, and a receiver is known as a **consumer** who consumes that message by subscribing it.

What is Streaming process

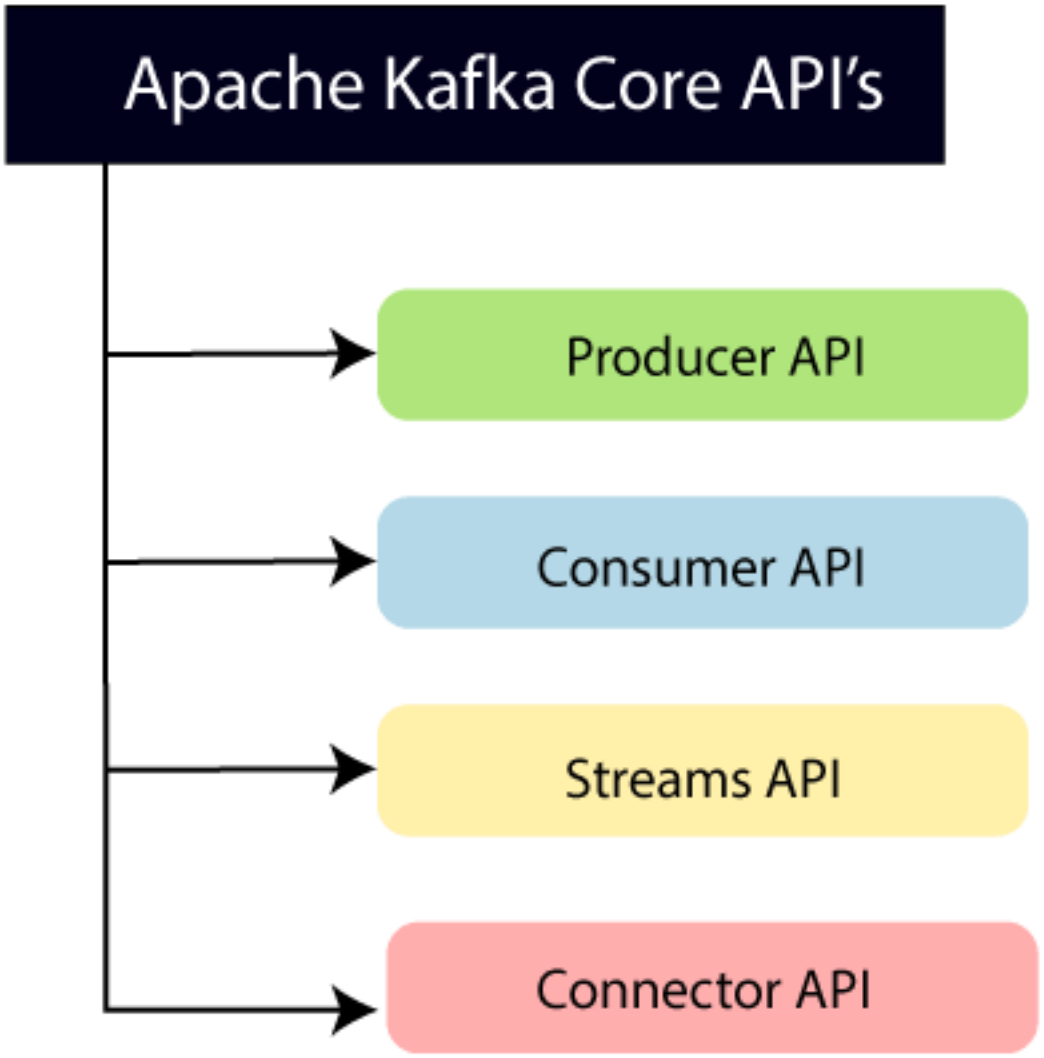
A streaming process is the processing of data in parallelly connected systems. This process allows different applications to limit the parallel execution of the data, where one record executes without waiting for the output of the previous record. Therefore, a distributed streaming platform

enables the user to simplify the task of the streaming process and parallel execution. Therefore, a streaming platform in Kafka has the following key capabilities:

- As soon as the streams of records occur, it processes it.
- It works similar to an enterprise messaging system where it publishes and subscribes streams of records.
- It stores the streams of records in a fault-tolerant durable way.



To learn and understand Apache Kafka, the aspirants should know the following four core APIs :



Producer API: This API allows/permits an application to publish streams of records to one or more topics. (discussed in later section)

Consumer API: This API allows an application to subscribe one or more topics and process the stream of records produced to them.

Streams API: This API allows an application to effectively transform the input streams to the output streams. It permits an application to act as a stream processor which consumes an input stream from one or more topics, and produce an output stream to one or more output topics.

Connector API: This API executes the reusable producer and consumer APIs with the existing data systems or applications.

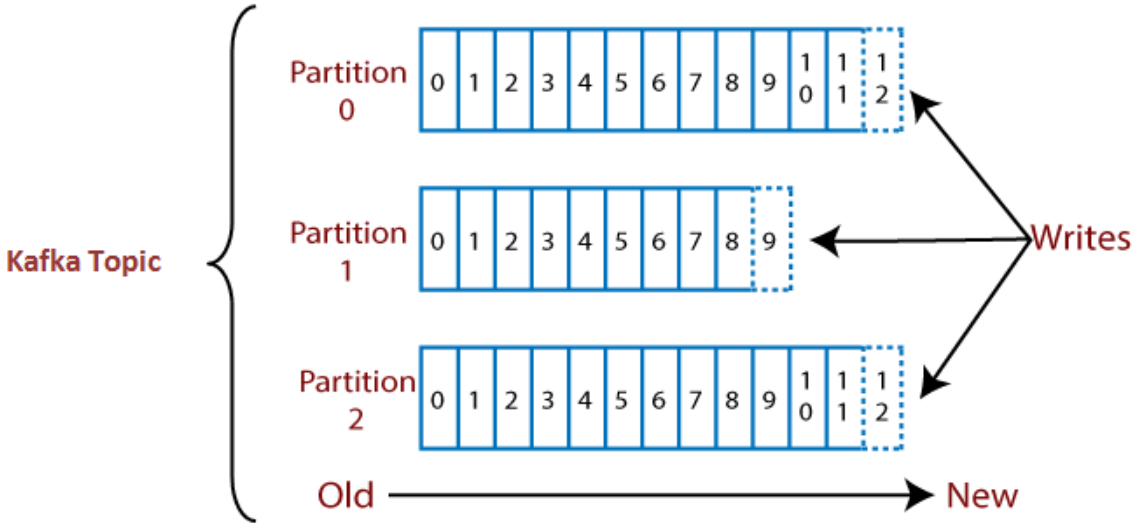
3.2 Kafka - Terminology

3.2.1 **Topics**

Generally, a topic refers to a particular heading or a name given to some specific inter-related ideas. In Kafka, the word topic refers to a category or a common name used to store and publish a particular stream of data. Basically, topics in Kafka are similar to tables in the database, but not containing all constraints. In Kafka, we can create n number of topics as we want. It is identified by its name, which depends on the user's choice. A producer publishes data to the topics, and a consumer reads that data from the topic by subscribing it.

3.2.2 **Partitions**

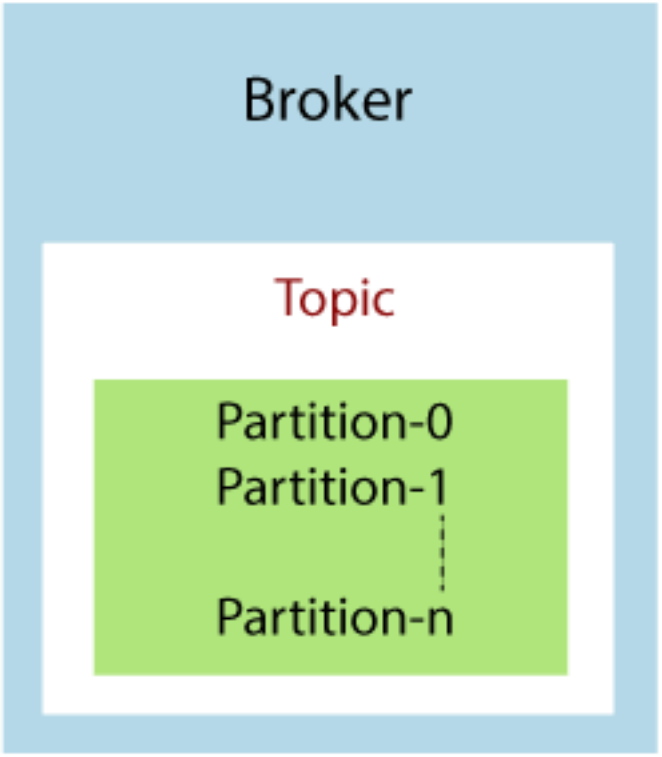
A topic is split into several parts which are known as the partitions of the topic. These partitions are separated in an order. The data content gets stored in the partitions within the topic. Therefore, while creating a topic, we need to specify the number of partitions(the number is arbitrary and can be changed later). Each message gets stored into partitions with an incremental id known as its Offset value. The order of the **offset value** is guaranteed within the partition only and not across the partition. The offsets for a partition are infinite.



Suppose, a topic containing three partitions 0,1 and 2. Each partition has different offset numbers. The data is distributed among each offset in each partition where data in offset 1 of Partition 0 does not have any relation with the data in offset 1 of Partition1. But, data in offset 1of Partition 0 is inter-related with the data contained in offset 2 of Partition0.

3.2.3 **Brokers**

A Kafka cluster is comprised of one or more servers which are known as brokers or Kafka brokers. A broker is a container that holds several topics with their multiple partitions. The brokers in the cluster are identified by an integer id only. Kafka brokers are also known as **Bootstrap brokers** because connection with any one broker means connection with the entire cluster. Although a broker does not contain whole data, but each broker in the cluster knows about all other brokers, partitions as well as topics.



3.2.4 **Producers**

A producer is the one which publishes or writes data to the topics within different partitions. Producers automatically know that, what data should be written to which partition and broker. The user does not require to specify the broker and the partition.

A producer uses following strategies to write data to the cluster:

- Message Keys
- Acknowledgment

Message Keys

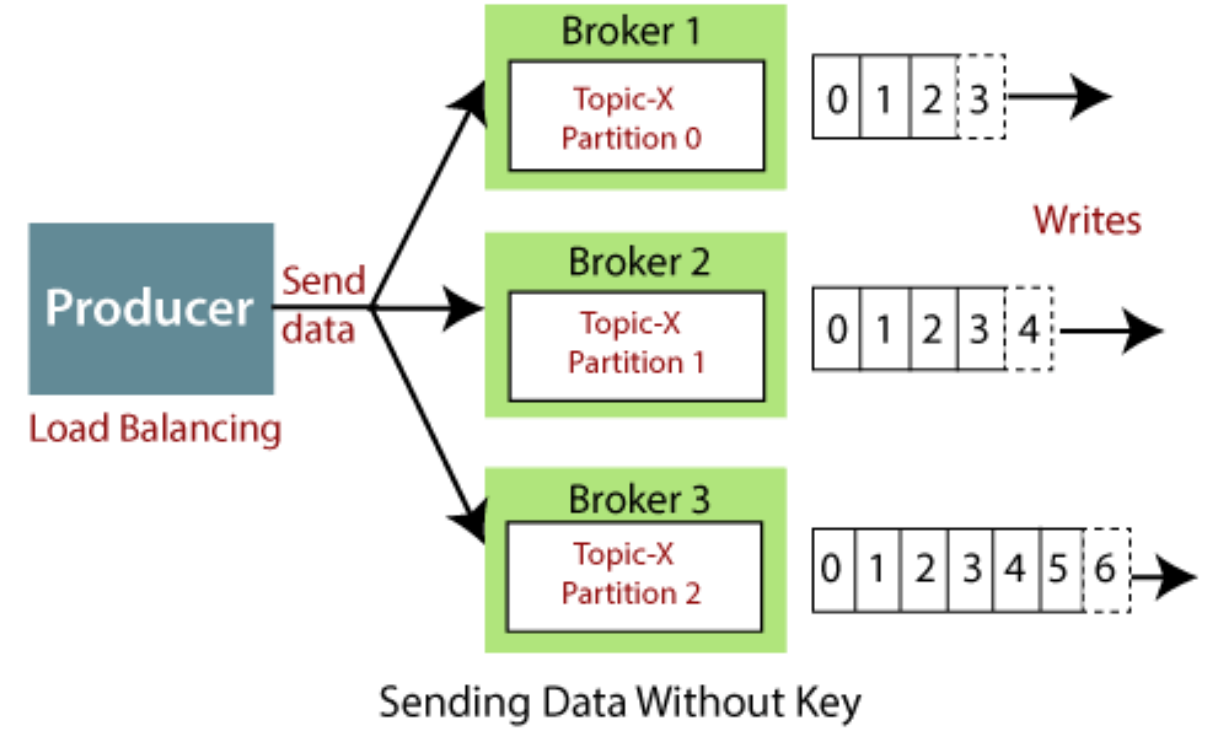
Apache Kafka enables the concept of the key to send the messages in a specific order. The key enables the producer with two choices, i.e., either to send data to each partition (automatically) or send data to a specific partition only. Sending data to some specific partitions is possible with the message keys. If the producers apply key over the data, that data will always be sent to the same partition always. But, if the producer does not apply the key while writing the data, it will be sent in a round-robin manner. This process is called **load balancing**. In Kafka, load balancing is done when the producer writes data to the Kafka topic without specifying any key, Kafka distributes little-little bit data to each partition.

Therefore, a message key can be a string, number, or anything as we wish.

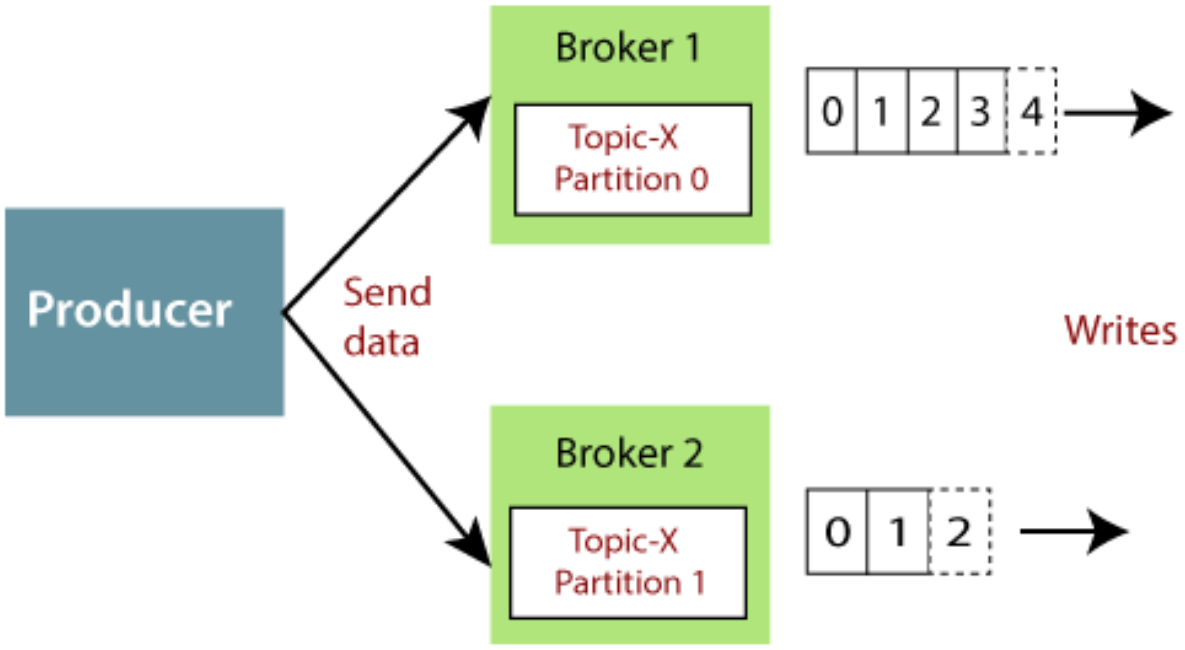
There are two ways to know that the data is sent with or without a key:

1. If the value of key=NULL, it means that the data is sent without a key. Thus, it will be distributed in a round-robin manner (i.e., distributed to each partition).

2. If the value of the key!=NULL, it means the key is attached with the data, and thus all messages will always be delivered to the same partition.



Sending Data Without Key

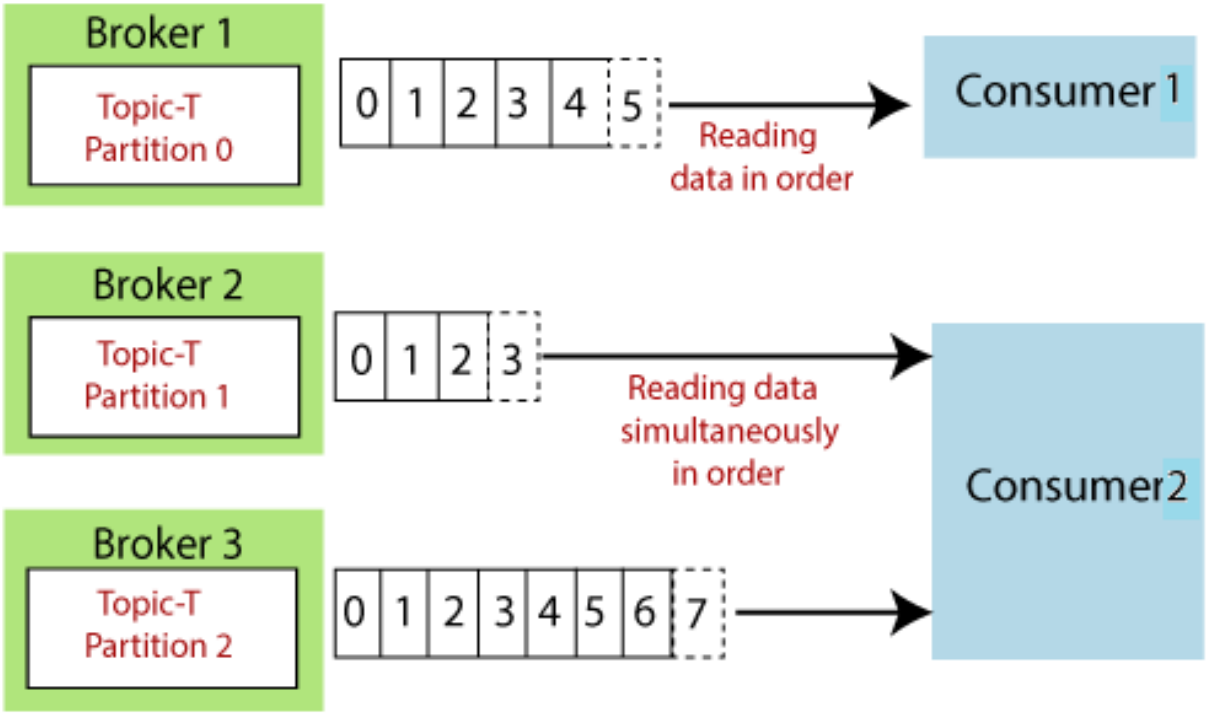


Sending data with key = Prod_id

3.2.5 **Consumers and Consumer Groups**

A consumer is the one that consumes or reads data from the Kafka cluster via a topic. A consumer also knows that from which broker, it should read the data. The consumer reads the data within each partition in an orderly manner. It means that the consumer is not supposed to read data from offset 1 before reading from offset 0. Also, a consumer can easily read data from multiple brokers at the same time

For example, two consumers namely, Consumer 1 and Consumer 2 are reading data. Consumer 1 is reading data from Broker 1 in sequential order. On the other hand, Consumer 2 is simultaneously reading data from Broker 2 as well as Broker 3 in order.



Consumer Groups

A consumer group is a group of multiple consumers which visions to an application basically. Each consumer present in a group reads data directly from the exclusive partitions. In case, the number of consumers are more than the number of partitions, some of the consumers will be in an inactive state. Somehow, if we lose any active consumer within the group then the inactive one can takeover and will come in an active state to read the data.

3.3 Kafka Commands

1. Start Zookeeper –

   bin/zookeeper-server-start.sh config/zookeeper.properties

2. Start Kafka Broker

   bin/kafka-server-start.sh config/server.properties

3. Creating Kafka Topics

   bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic topic-name

4. List of Topics

   bin/kafka-topics.sh --list --zookeeper localhost:2181

5. Start Producer to send Message

   bin/kafka-console-producer.sh --broker-list localhost:9092 --topic topic-name

6. Start Consumer to receive Message

   bin/kafka-console-consumer.sh --zookeeper localhost:2181 —topic topic-name --from-beginning

Chapter - 4

Version Control System

4.1 Version Control System - Introduction

Version control systems are a class of software tools that help a software team manage changes to source code over the period of time. Version control software keeps track of ever y modification to the code in a special kind of database. If a mistake is made, developer s can turn back the clock and compare earlier versions of the code to help fix the mistakes while minimizing disruption to all team members.

Benefits of version control systems:Developing software without using version control is risky, like not having backups. Version control can also enable developer s to move faster and it allows software teams to preserve efficiency and agility as the team scales to include more developers. The primary benefit s you should expect from version control are as follows.

- A complete long-term change history of every file.
- Branching and merging.
- Traceability. Version Control Systems (VCS) have seen great improvements over the past few decades and some are better than others. VCS are sometimes known as SCM (Source Code Management) tools or RCS (Revision Control System) **3.2 Git** • Introduction Git is a mature, actively maintained open-source project originally developed in 2005 by Linus Torvalds, the famous creator of the Linux operating system kernel. A staggering number of software projects rely on Git for version control, including commercial projects as well as open-source. Developers who have worked with Git are well represented in the pool of available software development talent and it works well on a wide range of operating systems and IDEs (Integrated Development Environments). Git is the best choice for most software teams today. While every team is different and should do their own analysis, here are the main reasons why version control with Git is preferred over alternative.

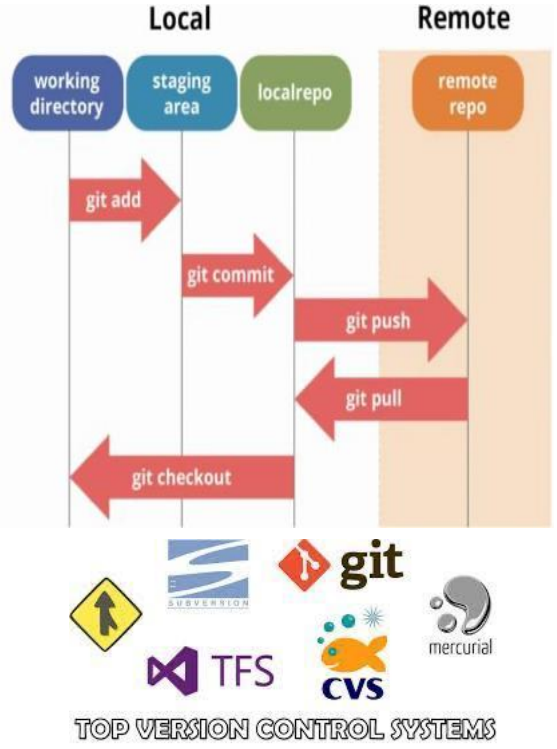
GIT

- It has the functionality, performance, security and flexibility that most teams and individual developers need.
- It is the most broadly adopted tool of its kind.
- It is a very well supported Test open-source project with over a decade of solid stewardship.
- It enjoys great community support and a vast user base.
- It has been designed with the integrity of managed source code as a top priority.
- It is not fooled by the names of the files when determining what the storage and version history of the file tree should be, instead, Git focuses on the file content itself.
- It keeps track of changes to the code.
- Revert back to old version of the code.

- Test changes to code without losing the original.
- It synchronizes code between different people.

• Useful Commands

- git init
- git clone
- got add
- git commit
- git status
- git push
- git pull
- merge conflict
- git log
- git reset
- branching
- git merge
- pull request



Git Hub

Git Hub is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management (SCM) functionality of Git, plus its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, continuously integration and wikis for every project.

Chapter – 5

Project Setup

5.1 Introduction to Project Setup

• IDE:

An integrated development environment (IDE) is software for building applications that combines common developer tools into a single graphical user interface (GUI). An IDE typically consists of:

• **Source code editor**: A text editor that can assist in writing software code with features such as syntax highlighting with visual cues, providing language specific auto-

completion, and checking for bugs as code is being written.

- **Local build automation**: Utilities that automate simple, repeatable tasks as part of creating a local build of the software for use by the developer, like compiling computer source code into binary code, packaging binary code, and running automated tests.

- **Debugger**: A program for testing other programs that can graphically display the location of a bug in the original code.

Why do developers use IDEs?

An IDE allows developers to start programming new applications quickly because multiple utilities don't need to be manually configured and integrated as part of the setup process. Developers also don't need to spend hours individually learning how to use different tools when every utility is represented in the same workbench. This can be especially useful for onboarding new developers who can rely on an IDE to get up to speed on a team's standard tools and workflows. In fact, most features of IDEs are meant to save time, like intelligent code completion and automated code generation, which removes the need to type out full character sequences.

Other common IDE features are meant to help developers organize their workflow and solve problems. IDEs parse code as it is written, so bugs caused by human error are identified in real-time. Because utilities are represented by a single GUI, developers can execute actions without switching between applications. Syntax highlighting is also common in most IDEs, which uses visual cues to distinguish grammar in the text editor. Some IDEs additionally include class and object browsers, as well as class hierarchy diagrams for certain languages.

• SDK:

SDK stands for software development kit or devkit for short. It’s a set of software tools and programs used by developers to create applications for specific platforms. SDK tools will include a range of things, including libraries, documentation, code samples, processes, and guides that developers can use and integrate into their own apps. SDKs are designed to be used for specific platforms or programming languages.

Thus, you would need an Android SDK toolkit to build an Android app, an iOS SDK to build an iOS app, a VMware SDK for integrating with the VMware platform, or a Nordic SDK for building Bluetooth or wireless products, and so on.

The Characteristics of a Good SDK:

Because your mobile SDK is meant to be used outside your organization, it has to provide value to other businesses and their developers. That value is dependent on your SDK having the following characteristics:

- Easy to use by other developers
- Thorough documentation to explain how your code works
- Enough functionality so it adds value to other apps
- Does not negatively impact a mobile device’s CPU, battery, or data consumption
- Plays well with other SDKs

In short, it just has to work. Ideally, it should work elegantly, but when time is of the essence, as long as it gets the job done, it should be good enough.

Chapter – 6

Java Framework: Spring

6.1 Introduction to Java:

Java is a programming language and computing platform first released by Sun Microsoft systems in 1995. There are lots of applications and websites that will not work unless you have Java installed, and more are created every day. Java is fast, secure, and reliable. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

Java platform is a collection of programs that help to develop and run programs written in the Java programming language. Java platform includes an execution engine, a compiler, and a set of libraries. JAVA is platform-independent language. It is not specific to any processor or operating system.

Java is a programming language created by James Gosling from Sun Microsystems (Sun) in 1991. The target of Java is to write a program once and then run this program on multiple operating systems. The first publicly available version of Java (Java 1.0) was released in 1995. Sun Microsystems was acquired by the Oracle Corporation in 2010. Oracle has now the steer Manship for Java. Oracle continues this project called OpenJDK.

NEW FEATURES OF JAVA 8

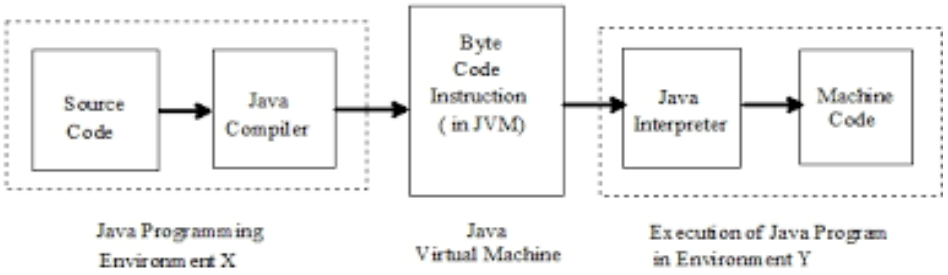
- Interface Default and Static Methods
- Method References
- Lambda Expressions
- Date and Time
- API Type Annotation
- Optional <T>

JDK: JDK provides the environment to develop and execute (run) the Java program. JDK is a kit (or package) which includes two things, Development Tools (to provide an environment to develop your java programs) and JRE (to execute your java program).

JRE: JRE is an installation package that provides environment to only run (not develop) the java program (or application) onto your machine. JRE is only used by them who only want to run the Java Programs i.e., end-users of your system.

JVM: Java Virtual Machine (JVM) is a very important part of both JDK and JRE because it is contained or inbuilt in both. Whatever Java program you run using JRE or JDK goes into JVM and JVM is responsible for executing the java program line by line hence it is also known as interpreter. Through the JavaVM, the same application is capable overrunning on multiple platforms and that is why the JAVA is said to be platform independent.

COMPILATION PROCESS: During the compilation phase, the Java compiler compiles the source code and generates bytecode. This intermediate bytecode is saved in the form of a class file. In the second phase, Java virtual machine (JVM) also called Java interpreter takes the class as input and generates output by executing the bytecode.



6.2 Framework: Spring

❖ **INTRODUCTION TO SPRING:**

Spring is a lightweight framework. It can be thought of as a framework of frameworks because it provides support to various frameworks The Spring framework comprises several modules such as IOC, AOP, DAO, Context, ORM, WEB MVC etc.

❖ INVERSION OF CONTROL:

IOC makes the code loosely coupled. In such a case, there is no need to modify the code if our logic is moved to a new environment. In the Spring framework, IOC container is responsible to inject the dependency. We provide metadata to the IOC container either by XML file or annotation. The IoC container is responsible to instantiate, configure, and assemble the objects.

The main tasks performed by the IoC container are:

- to instantiate the application class
- to configure the object
- to assemble the dependencies between the objects.
- There are two types of IoC containers. They are:

    1. BeanFactory

    2. ApplicationContext : The ApplicationContext interface is built on top of the BeanFactory interface. It adds some extra functionality than BeanFactory such as simple integration with Spring's AOP, message resource handling, event propagation, application-layer specific context (e.g., WebApplicationContext) for a web application. So, it is better to use ApplicationContext than BeanFactory.

❖ Dependency Injection:

Dependency Injection (DI) is a design pattern that removes the dependency from the programming code so that it can be easy to manage and test the application. Dependency Injection makes our programming code loosely coupled.

Spring framework provides two ways to inject dependency

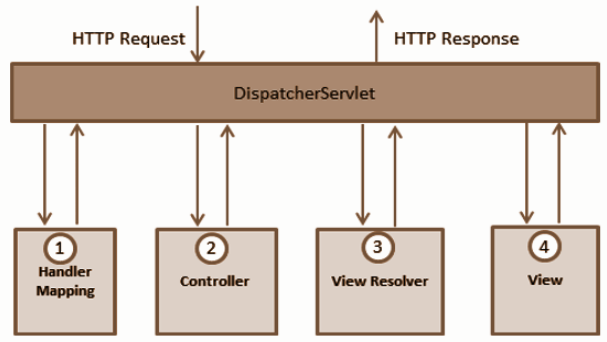
- By Constructor

By Setter method

❖ Autowiring:

Autowiring feature of the spring framework enables you to inject the object dependency implicitly. It internally uses setter or constructor injection. Autowiring can't be used to inject primitive and string values. It works with reference only.

```
class Employee{
Address address;
Employee(){
address = new Address();
}
}
```



EXAMPLE OF DEPENDENCY INJECTION:

class Employee{

Address address;
Employee(Address address){
address = new Address();
}}

❖ **ORM (OBJECT RELATIONAL MAPPING):**
Spring provides API to easily integrate Spring with ORM frameworks such as Hibernate, JPA (Java Persistence API), JDO (Java Data Objects), Oracle Toplink.

❖ **Spring MVC Tutorial**
A Spring MVC is a Java framework that is used to build web applications. It follows the Model- View-Controller design pattern. It implements all the basic features of a core spring framework like Inversion of Control, Dependency Injection. A Spring MVC provides an elegant solution to use MVC in the spring framework with the help of DispatcherServlet. Here, DispatcherServlet is a class that receives the incoming request and maps it to the right resource such as controllers, models, and views.

❖ **What is Spring Boot?**

• Spring Boot is a project that is built on the top of the Spring Framework. It provides an easier and faster way to set up, configure, and run both simple and web- based applications.• It is a Spring module that provides the **RAD (Rapid Application Development)** feature to the Spring Framework. It is used to create a stand-alone Spring-based application that you can just run because it needs minimal Spring configuration.

In short, Spring Boot is the combination of Spring Framework and Embedded Servers.

• In Spring Boot, there is no requirement for XML configuration (deployment descriptor). It uses convention over configuration software design paradigm that means it decreases the effort of the developer.

We should use Spring Boot Framework because:

- The dependency injection approach is used in Spring Boot.
- It contains powerful database transaction management capabilities.
- It simplifies integration with other Java frameworks like JPA/Hibernate ORM, Struts, etc.

• It reduces the cost and development time of the application.

❖ **Features:**

1. **Web Development**

It is a well-suited Spring module for web application development. We can easily create a self-contained HTTP application that uses embedded servers like Tomcat, Jetty, or Undertow. We can use the spring-boot-starter-web module to start and run the application quickly.

2. SpringApplication

The SpringApplication is a class that provides a convenient way to bootstrap a Spring application. It can be started from the main method. We can call the application just by calling a static run() method.

3. Admin Support

Spring Boot provides the facility to enable admin-related features for the application. It is used to access and manage applications remotely. We can enable it in the Spring Boot application by using spring. Application. admin. enabled property.

4. Externalized Configuration

Spring Boot allows us to externalize our configuration so that we can work with the same application in different environments. The application uses YAML files to externalize configuration.

5. **Properties Files**

Spring Boot provides a rich set of Application Properties. So, we can use that in the properties file of our project. The properties file is used to set properties like server-port =8082 and many others. It helps to organize application properties.

6. **YAML Support**

It provides a convenient way of specifying the hierarchical configuration. It is a superset of JSON. The SpringApplication class automatically supports YAML. It is an alternative of properties file.

7. Type-safe Configuration

The strong type-safe configuration is provided to govern and validate the configuration of the application. Application configuration is always a crucial task which should be type- safe. We can also use annotation provided by this library.

8. Logging

Spring Boot uses Common logging for all internal logging. Logging dependencies are managed by default. We should not change logging dependencies if no customization is needed.

9. **Security**

Spring Boot applications are spring bases web applications. So, it is secure by default with basic authentication on all HTTP endpoints. A rich set of Endpoints is available to develop a secure Spring Boot application.

Chapter – 7

SDLC

7.1 Introduction to Software Development

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software's. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

- SDLC is the acronym of Software Development Life Cycle.
- It is also called as Software Development Process.
- SDLC is a framework defining tasks performed at each step in the software development process.
- ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

❖ What is SDLC?

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.

A typical Software Development Life Cycle consists of the following stages − Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

7.3 Agile Methodology

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like −

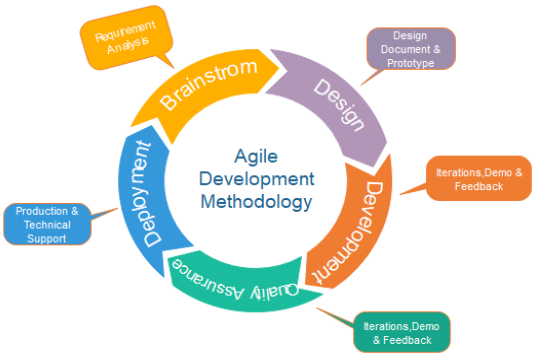


Fig. Agile Model

- Planning
- Requirements Analysis
- Design
- Coding
- Unit Testing and
- Acceptance Testing.

At the end of the iteration, a working product is displayed to the customer and important stakeholders.

❖ What is Agile?

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In Agile, the tasks are divided to time boxes (small time frames) to deliver specific features for a release.

Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

Here is a graphical illustration of the Agile Model −

The Agile thought process had started early in the software development and started becoming popular with time due to its flexibility and adaptability.

The most popular Agile methods include Rational Unified Process (1994), Scrum (1995), Crystal Clear, Extreme Programming (1996), Adaptive Software Development, Feature Driven Development, and Dynamic Systems Development Method (DSDM) (1995). These are now collectively referred to as **Agile Methodologies**, after the Agile Manifesto was published in 2001.

Following are the Agile Manifesto principles −

Individuals and interactions − In Agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.

Working software− Demo working software is considered the best means of communication with the customers to understand their requirements, instead of just depending on documentation.

Customer collaboration − As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.

Responding to change − Agile Development is focused on quick responses to change and continuous development.

Agile Vs Traditional SDLC Models

Agile is based on the **adaptive software development methods**, whereas the traditional SDLC models like the waterfall model is based on a predictive approach. Predictive teams in the traditional SDLC models usually work with detailed planning and have a complete forecast of the exact tasks and features to be delivered in the next few months or during the product life cycle.

Predictive methods entirely depend on the **requirement analysis and planning** done in the beginning of cycle. Any changes to be incorporated go through a strict change control management and prioritization.

Agile uses an **adaptive approach** where there is no detailed planning and there is clarity on future tasks only in respect of what features need to be developed. There is feature driven development and the team adapts to the changing product requirements dynamically. The product is tested very frequently, through the release iterations, minimizing the risk of any major failures in future.

Customer Interaction is the backbone of this Agile methodology, and open communication with minimum documentation are the typical features of Agile development environment. The agile teams work in close collaboration with each other and are most often located in the same geographical location.

❖ Agile Model - Pros and Cons

Agile methods are being widely accepted in the software world recently. However, this method may not always be suitable for all products. Here are some pros and cons of the Agile model.

The advantages of the Agile Model are as follows −

- Is a very realistic approach to software development.
- Promotes teamwork and cross training.
- Functionality can be developed rapidly and demonstrated.
- Resource requirements are minimum.
- Suitable for fixed or changing requirements
- Delivers early partial working solutions.
- Good model for environments that change steadily.
- Minimal rules, documentation easily employed.
- Enables concurrent development and delivery within an overall planned context.
- Little or no planning required.
- Easy to manage.
- Gives flexibility to developers.

The disadvantages of the Agile Model are as follows −

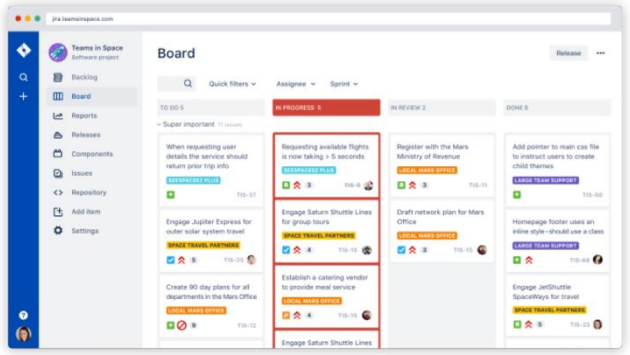
- Not suitable for handling complex dependencies.
- More risk of sustainability, maintainability and extensibility.

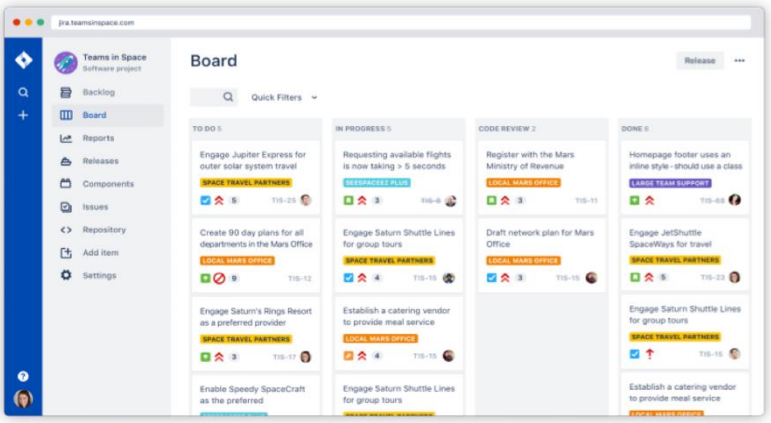
- An overall plan, an agile leader and agile PM practice is a must without which it will not work.
- Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
- Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
- There is a very high individual dependency, since there is minimum documentation generated.
- Transfer of technology to new team members may be quite challenging due to lack of documentation.

7.4 Other Software Development Tools

1. **JIRA:**

Jira Software is an agile project management tool that supports any agile methodology, be it scrum, Kanban, or your own unique flavor. From agile boards, backlogs, roadmaps reports to integrations and add-ons you can plan, track, and manage all your agile software development projects from a single tool. Pick a framework to see how Jira Software can help your team release higher quality software, faster.





Chapter – 8

Database Management System

8.1 DBMS

❖ **What is a Database?**

A database is a collection of related data which represents some aspect of the real world. A

database system is designed to be built and populated with data for a certain task.

❖ **What is DBMS?**

Database Management System (DBMS) is a software for storing and retrieving users' data while considering appropriate security measures. It consists of a group of programs which manipulate the database. The DBMS accepts the request for data from an application and instructs the operating system to provide the specific data. In large systems, a DBMS helps users and other third-party software to store and retrieve data.

DBMS allows users to create their own databases as per their requirement. The term "DBMS" includes the user of the database and other application programs. It provides an interface between the data and the software application.

❖ **Characteristics of Database Management System**

- Provides security and removes redundancy
- Self-describing nature of a database system
- Insulation between programs and data abstraction
- Support of multiple views of the data
- Sharing of data and multiuser transaction processing
- DBMS allows entities and relations among them to form tables.
- It follows the ACID concept (Atomicity, Consistency, Isolation, and Durability).
- DBMS supports multi-user environment that allows users to access and manipulate data in parallel.

❖ **DBMS vs. Flat File DBMS**

DBMS	Flat File DBMS
Multi-user access	It does not support multi-user access
Design to fulfil the need for small and large business.	It is only limited to smaller DBMS system.
Remove redundancy and Integrity	Redundancy and Integrity issues
Expensive. But in the long-term Total Cost of Ownership is cheap	It's cheaper

❖ **Users in a DBMS environment**

Following are the various category of users of a DBMS system:

Component Name	**Task**
Application Programmers	The Application programmers write programs in various programming languages to interact with databases.
Database Administrators	Database Admin is responsible for managing the entire DBMS system. He/She is called Database admin or DBA.
End-Users	The end users are the people who interact with the database management system. They conduct various operations on database like retrieving, updating, deleting, etc.

❖ **Application of DBMS**

Sector	Use of DBMS
Banking	For customer information, account activities, payments, deposits, loans, etc.
Airlines	For reservations and schedule information.
Universities	For student information, course registrations, colleges and grades.
Telecommunication	It helps to keep call records, monthly bills, maintaining balances, etc.
Finance	For storing information about stock, sales, and purchases of financial instruments like stocks and bonds.

Sales	Use for storing customer, product & sales information.
Manufacturing	It is used for the management of supply chain and for tracking production of items. Inventories status in warehouses.
HR Management	For information about employees, salaries, payroll, deduction, generation of pay checks, etc.

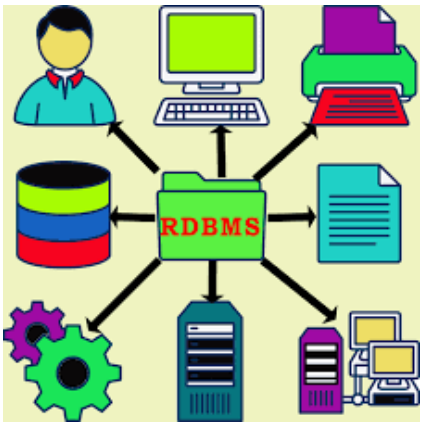
❖ **Type of DBMS Data Structure**

When it comes to choosing a database the biggest decisions is picking a relational (SQL) or non-relational (NoSQL) data structure. While both the databases are viable options still there are certain key differences between the two that users must keep in mind when deciding.

Type – SQL databases are primarily called as Relational Databases (RDBMS); whereas NoSQL database are primarily called as non-relational or distributed database.

1. RDBMS

Stands for "Relational Database Management System."



An RDBMS is a DBMS designed specifically for relational databases. Therefore, RDBMSes are a subset of DBMSes.

A relational database refers to a database that stores data in a structured format, using rows and columns. This makes it easy to locate and access specific values within the database. It is "relational" because the values within each table are related to each other. Tables may also be related to other tables. The relational structure makes it possible to run queries across multiple tables at once.

While a relational database describes the type of database an RDMBS manages, the RDBMS refers to the database program itself. It is the software that executes queries on the data, including adding, updating, and searching for values. An RDBMS may also provide a visual representation of the data. For example, it may display data in a table like a spreadsheet, allowing you to view and even edit individual values in the table. Some RDMBS programs allow you to create forms that can streamline entering, editing, and deleting data.
Most well-known DBMS applications fall into the RDBMS category.

Examples include Oracle Database, MySQL, Microsoft SQL Server, and IBM DB2. Some of these programs support non- relational databases, but they are primarily used for relational database management.
Examples of non-relational databases include Apache HBase, Elasticsearch, IBM Domino, and Oracle NoSQL Database. These types of databases are managed by other DMBS programs that support NoSQL, which do not fall into the RDBMS category.

SQL

SQL (*Structured Query Language*) is used to perform operations on the records stored in the database such as updating records, deleting records, creating and modifying tables, views, etc.

SQL is just a query language; it is not a database. To perform SQL queries, you need to install any database, for example, Oracle, MySQL, MongoDB, PostGre SQL, SQL Server, DB2, etc.

❖ What is SQL?

- SQL stands for **Structured Query Language**.
  It is designed for managing data in a relational database management system (RDBMS).
- It is pronounced as S-Q-L or sometime **See-Qwell**.
- SQL is a database language, it is used for database creation, deletion, fetching rows, and modifying rows, etc.
- SQL is based on relational algebra and tuple relational calculus.

All DBMS like MySQL, Oracle, MS Access, Sybase, Informix, PostgreSQL, and SQL Server use SQL as standard database language.

❖ Why SQL is required

SQL is required:

- To create new databases, tables and views.
- To insert records in a database
- To update records in a database
- To delete records from a database
- To retrieve data from a database

❖ What SQL Does?

- With SQL, we can query our database in several ways, using English-like statements.
- With SQL, a user can access data from a relational database management system.
- It allows the user to describe the data.
- It allows the user to define the data in the database and manipulate it when needed.
- It allows the user to create and drop database and table.
- It allows the user to create a view, stored procedure, function in a database.
- It allows the user to set permission on tables, procedures, and views.

2. NON-RDBMS

Non-relational databases are sometimes referred to as “NoSQL,” which stands for Not Only SQL. The main difference between these is how they store their information.

A non-relational database stores data in a non-tabular form, and tends to be more flexible than the traditional, SQL-based, relational database structures. It does not follow the relational model provided by traditional relational database management systems. Non-relational databases (often called NoSQL Databases) are different from traditional relational databases in that they store their data in a non-tabular form. Instead, non-relational databases might be based on data structures like documents. A document can be highly detailed while containing a range of different types of information in different formats. This ability to digest and organize various types of information side-by-side makes non-relational databases much more flexible than relational databases.

Non-relational databases are often used when large quantities of complex and diverse data need to be organized. For example, a large store might have a database in which each customer has their own document containing all of their information, from name and address to order history and credit card information. Despite their differing formats, each of these pieces of information can be stored in the same document.

Non-relational databases often perform faster because a query doesn’t have to view several tables in order to deliver an answer, as relational datasets often do. Non-relational databases are therefore ideal for storing data that may be changed frequently or for applications that handle many different kinds of data. They can support rapidly developing applications requiring a dynamic database able to change quickly and to accommodate large amounts of complex, unstructured data.

❖ Advantages of Non-Relational Databases

There are several advantages to using non-relational databases, including:

- Massive dataset organization

  In the age of Big Data, non-relational databases can not only store massive quantities of information, but they can also query these datasets with ease. Scale and speed are crucial advantages of non-relational databases.

- Flexible database expansion

  Data is not static. As more information is collected, a non-relational database can absorb these new data points, enriching the existing database with new levels of granular value even if they don’t fit the data types of previously existing information.

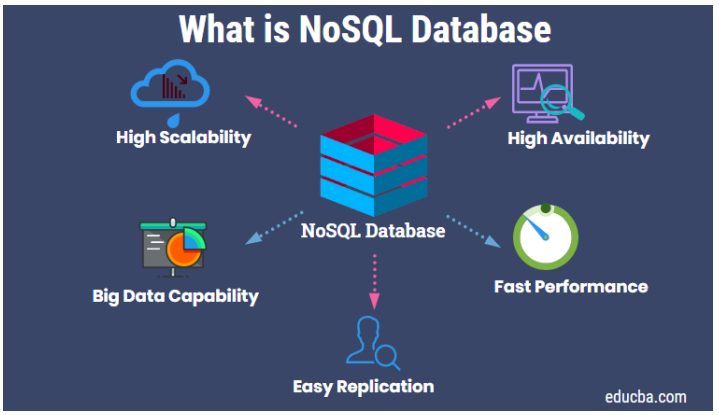
- Multiple data structures

  The data now collected from users takes on a myriad of forms, from numbers and strings, to photo and video content, to message histories. A database needs the ability to store these various information formats, understand relationships between them, and perform detailed queries. No matter what format your information is in, non-relational databases can collate different information types together in the same document.

- Built for the cloud

  A non-relational database can be massive. And as they can, in some cases, grow exponentially, they need a hosting environment that can grow and expand with them. The cloud's inherent scalability makes it an ideal home for non-relational databases.

NoSQL



NoSQL databases (aka "not only SQL") are non-tabular databases and store data differently than relational tables. NoSQL databases come in a variety of types based on their data model. The main types are document, key-value, wide-column, and graph. They provide flexible schemas and scale easily with large amounts of data and high user loads.

NoSQL Database Features

Each NoSQL database has its own unique features. At a high level, many NoSQL databases have the following features:

- Flexible Schemes
- Horizontal Scaling
- Fast queries due to data model
- Ease of use for developers

Types of NoSQL databases

Over time, four major NoSql databases emerged: document databases, key-value databases, wide-column stores, and graph databases.

- Document databases store data in documents similar to JSON (JavaScript Object Notation) objects. Each document contains pairs of fields and values. The values can typically be a variety of types including things like strings, numbers, booleans, arrays, or objects.
- Key-value databases are a simpler type of database where each item contains keys and values.
- Wide-column stores store data in tables, rows, and dynamic columns.
- Graph databases store data in nodes and edges. Nodes typically store information about people, places, and things, while edges store information about the relationships between the nodes.

8.2 ELASTICSEARCH

Elasticsearch is a **NoSQL Database**, which is developed in Java programming language It is a real-time, distributed, and analysis engine that is designed for storing logs. It is a highly scalable document storage engine. Similar to the MongoDB, it stores the data in document format. It enables the users to execute the advanced queries to perform detailed analysis and store all data centrally.

Elasticsearch database is licensed under the Apache version 2.0 and based on Apache Lucene search engine. It is built-in Restful APIs that help in fulfilling the request and responding to the request. It is an essential part of Elastic Stack or we can also say that it is a **heart of Elastic Stack**. It is open-source, which means that it is freely available. So, anyone can download it without paying any cost.

Elasticsearch mostly used in Single Page Application (SPM) projects. Many large organizations across the world use it. It supports full-text search that is completely document-based instead of schemas and tables. There are some more other search-based engines available, but they all are based on tables and schemas.

A typical elasticsearch document looks like :-

```
{
   "first_name": "Alex",
   "last_name": "Batson",
   "phone_no": "987654321",
   "email": abc@gmail.com,
   "city": "New York",
   "country": "USA",
   "occupation": "Software Developer",
}
```

Why Elasticsearch?

With large datasets, relational database comparatively works slow and leads to slow search results from the database when queries are executed. RDBMS can be optimized but also brings a set of limitations like every field cannot be indexed and updating rows for heavily indexed tables is a long and annoying process.

Elasticsearch is a NoSQL distributed database, which is a solution for quick retrieval and storing data.

There are some other reasons for using Elasticsearch NoSQL database –

- Elasticsearch allows you to perform and combine various types of searches, like structured as well as unstructured. It also helps in working upon the data, which is based on geography as well as on matrix.
- You can retrieve the result from the data which you import in anyway you want. It is all based on structured query sets.
- It allows the users to ask the query anyway they want.
- Elasticsearch provides aggregations that help us to explore trends and patterns in our data.
- Elasticsearch takes care of both query and analysis on data.
- Elasticsearch database helps to complete the search query based on the previous searches automatically.

Uses of Elasticsearch

Textual Search

Elasticsearch is useful for searching of pure text. It is mainly used where there is a lot of text, but we want to search the data with a specific phrase for the best match. In other words, we search for pure text.

Product Search

Elasticsearch uses properties and name, which offers faster product searches.

Geo Search

Elasticsearch is also used for geo-localized any product. For example - A search query like "All institutes that offer PGDM courses in India" can be used to display relevant information of institute by Elasticsearch, which offers PGDM courses across India.

Data Aggregation

Aggregation's framework provides aggregated data based on search queries. It allows to group and performs calculations and statistics on your data using simple search queries. An aggregation can be

Auto-Suggest

Elasticsearch has an auto-suggest feature, which provides several suggestions to complete an incomplete query. This allows users to type a few characters, and then it will automatically display several suggestions to complete the query.

AutoComplete

Based on the previous searches, the Elasticsearch database helps to complete the search query automatically.

JSON Document Storage

Elasticsearch stores the data in the form of document. The documents are JSON objects that are stored in Elasticsearch index. In other words, the document is considered as a base unit of storage that can be indexed.

Metrics and Analysis

It analyses a dashboard that consists of several emails, logs, syslogs, and databases, which helps the businesses to understand their data and provides actionable insights.

Chapter – 9

JavaScript

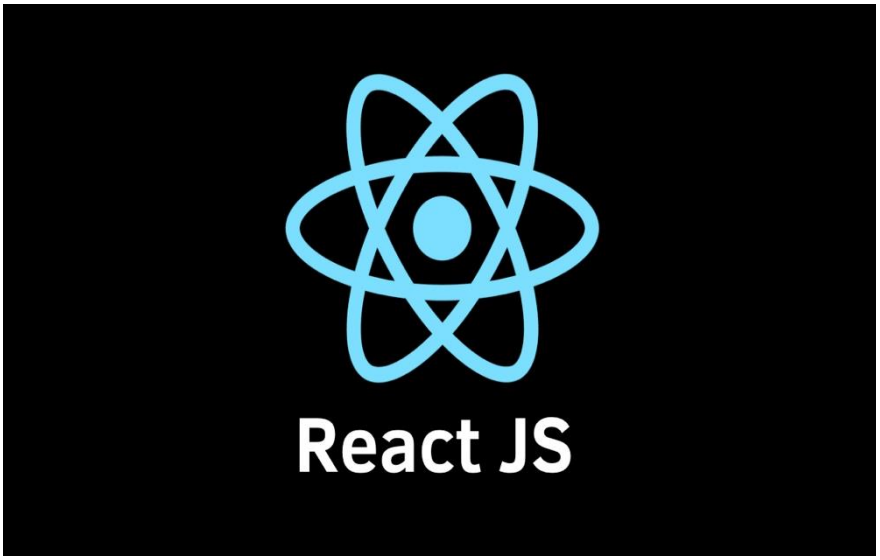
9.1 JavaScript Introduction

JavaScript(often shortened to **JS**) is a lightweight, interpreted, object-oriented language with first-class functions, and is best known as the scripting language for Web pages, but it's used in many non-browser environments as well. It is a prototype-based, multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles.
JavaScript runs on the client side of the web, which can be used to design / program how the web pages behave on the occurrence of an event. JavaScript is an easy to learn and also powerful scripting language, widely used for controlling web page behaviour.
Contrary to popular misconception, **JavaScript is *not* "Interpreted Java"**. In a nutshell, JavaScript is a dynamic scripting language supporting prototype-based object construction. The basic syntax is intentionally similar to both Java and C++ to reduce the number of new concepts required to learn the language. Language constructs, such as if statements, for and while loops, and switch and try ... catch blocks function the same as in these languages (or nearly so). JavaScript can function as both a procedural and an object-oriented language. Objects are created programmatically in JavaScript, by attaching methods and properties to otherwise empty
objects **at run time**, as opposed to the syntactic class definitions common in compiled languages like C++ and Java. Once an object has been constructed it can be used as a blueprint (or prototype) for creating similar objects.
JavaScript's dynamic capabilities include runtime object construction, variable parameter lists, function variables, dynamic script creation (via eval), object introspection (via for ... in), and source code recovery (JavaScript programs can decompile function bodies back into their source text).

The standards for JavaScript are the ECMAScript Language Specification (ECMA-262) and the ECMAScript Internationalization API specification (ECMA-402). The JavaScript documentation throughout MDN is based on the latest draft versions of ECMA-262 and ECMA- 402. And in cases where some proposals for new ECMAScript features have already been implemented in browsers, documentation and examples in MDN articles may use some of those new features. Do not confuse JavaScript with the Java programming

language. Both "Java" and "JavaScript" are trademarks or registered trademarks of Oracle in the U.S. and other countries. However, the two programming languages have very different syntax, semantic, and use.

9.1 JavaScript Library – ReactJS Introduction



ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front end library responsible only for the view layer of the application. It was created by **Jordan Walke,** who was a software engineer at **Facebook.** It was initially developed and maintained by Facebook and was later used in its products like **WhatsApp** & **Instagram.** Facebook developed ReactJS in **2011** in its newsfeed section, but it was released to the public in the month of **May 2013.**

Today, most of the websites are built using MVC (model view controller) architecture. In MVC architecture, React is the 'V' which stands for view, whereas the architecture is provided by the Redux or Flux.

A ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code. The components are the heart of all React applications. These Components can be nested with other components to allow complex applications to be built of simple building blocks. ReactJS uses virtual DOM based mechanism to fill data in HTML DOM. The virtual DOM works fast as it only changes individual DOM elements instead of reloading complete DOM every time.

To create React app, we write React components that correspond to various elements. We organize these components inside higher level components which define the application structure. For example, we take a form that consists of many elements like input fields, labels, or buttons. We can write each element of the form as React components, and then we combine it into a higher-level component, i.e., the form component itself. The form components would specify the structure of the form along with elements inside of it.

Today, many JavaScript frameworks are available in the market(like angular, node), but still, React came into the market and gained popularity amongst them. The previous frameworks follow the traditional data flow structure, which uses the DOM (Document Object Model).

DOM is an object which is created by the browser each time a web page is loaded. It dynamically adds or removes the data at the back end and when any modifications were done, then each time a new DOM is created for the same page. This repeated creation of DOM makes unnecessary memory wastage and reduces the performance of the application.

Therefore, a new technology ReactJS framework invented which remove this drawback. ReactJS allows you to divide your entire application into various components. ReactJS still used the same traditional data flow, but it is not directly operating on the browser's Document Object Model (DOM) immediately; instead, it operates on a virtual DOM. It means rather than manipulating the document in a browser after changes to our data, it resolves changes on a DOM built and run entirely in memory. After the virtual DOM has been updated, React determines what changes made to the actual browser's DOM. The React Virtual DOM exists entirely in memory and is a representation of the web browser's DOM. Due to this, when we write a React component, we did not write directly to the DOM; instead, we are writing virtual components that react will turn into the DOM.

9.1.1 ReactJS Features

Currently, ReactJS gaining quick popularity as the best JavaScript framework among web developers. It is playing an essential role in the front-end ecosystem. The important features of ReactJS are as following.

- JSX
- Components
- One-way Data Binding
- Virtual DOM
- Simplicity
- Performance

JSX

JSX stands for JavaScript XML. It is a JavaScript syntax extension. Its an XML or HTML like syntax used by ReactJS. This syntax is processed into JavaScript calls of React Framework. It extends the ES6 so that HTML like text can co-exist with JavaScript react code. It is not necessary to use JSX, but it is recommended to use in ReactJS.

Components

ReactJS is all about components. ReactJS application is made up of multiple components, and each component has its own logic and controls. These components can be reusable which help you to maintain the code when working on larger scale projects.

One-way Data Binding

ReactJS is designed in such a manner that follows unidirectional data flow or one-way data binding. The benefits of one-way data binding give you better control throughout the application. If the data flow is in another direction, then it requires additional features. It is because components are supposed to be immutable and the data within them cannot be changed. Flux is a pattern that helps to keep your data unidirectional. This makes the application more flexible that leads to increase efficiency.

Virtual DOM

A virtual DOM object is a representation of the original DOM object. It works like a one-way data binding. Whenever any modifications happen in the web application, the entire UI is re-rendered in virtual DOM representation. Then it checks the difference between the previous

DOM representation and new DOM. Once it has done, the real DOM will update only the things that have actually changed. This makes the application faster, and there is no wastage of memory.

Simplicity

ReactJS uses JSX file which makes the application simple and to code as well as understand. We know that ReactJS is a component-based approach which makes the code reusable as your need. This makes it simple to use and learn.

Performance

ReactJS is known to be a great performer. This feature makes it much better than other frameworks out there today. The reason behind this is that it manages a virtual DOM. The DOM is a cross-platform and programming API which deals with HTML, XML or XHTML. The DOM exists entirely in memory. Due to this, when we create a component, we did not write directly to the DOM. Instead, we are writing virtual components that will turn into the DOM leading to smoother and faster performance.

9.1.2 ReactJS Components

A Component is considered as the core building blocks of a React application. It makes the task of building UIs much easier. Each component exists in the same space, but they work independently from one another and merge all in a parent component, which will be the final UI of your application.

Every React component have their own structure, methods as well as APIs. They can be reusable as per your need. For better understanding, consider the entire UI as a tree. Here, the root is the starting component, and each of the other pieces becomes branches, which are further divided into sub-branches.

In ReactJS, we have mainly two types of components. They are

1. Functional Components
2. Class Components

Functional Components

In React, function components are a way to write components that only contain a render method and don't have their own state. They are simply JavaScript functions that may or may not receive data as parameters. We can create a function that takes props(properties) as input and returns what should be rendered. A valid functional component can be shown in the below example.

```
function WelcomeMessage(props) {
    return <h1>Welcome to the , {props.name}</h1>;
}
```

Class Components

Class components are more complex than functional components. It requires you to extend from React. Component and create a render function which returns a React element. You can pass data from one class to other class components. You can create a class by defining a class that extends Component and has a render function. Valid class component is shown in the below example.

```
class MyComponent extends React.Component {
      render() {
        return (
          <div>This is main component.</div>
        );
      }
    }
```

9.1.3 ReactJS State

The state is an updatable structure that is used to contain data or information about the component. The state in a component can change over time. The change in state over time can happen as a response to user action or system event. A component with the state is known as stateful components. It is the heart of the react component which determines the behavior of the component and how it will render. They are also responsible for making a component dynamic and interactive.

A state must be kept as simple as possible. It can be set by using the setState() method and calling setState() method triggers UI updates. A state represents the component's local state or information. It can only be accessed or modified inside the component or by the component directly. To set an initial state before any interaction occurs, we need to use the getInitialState() method.

For example, if we have five components that need data or information from the state, then we need to create one container component that will keep the state for all of them.

9.1.4 ReactJS Hooks

Hooks are the new feature introduced in the React 16.8 version. It allows you to use state and other React features without writing a class. Hooks are the functions which "hook into" React state and lifecycle features from function components. It does not work inside classes.

Hooks are backward-compatible, which means it does not contain any breaking changes. Also, it does not replace your knowledge of React concepts.

When to use a Hooks

If you write a function component, and then you want to add some state to it, previously you do this by converting it to a class. But, now you can do it by using a Hook inside the existing function component.

Chapter – 10

Project

“ Order Visibility Dashboard ”

10.1 Technology Used

The technology stack that will be used are as follows:

❖ **FRONTEND:** HTML, CSS JS – REACT

❖ **BACKEND:** JAVA

❖ **DATABASE:** ELASTICSEARCH

10.2 Planning and Analysis

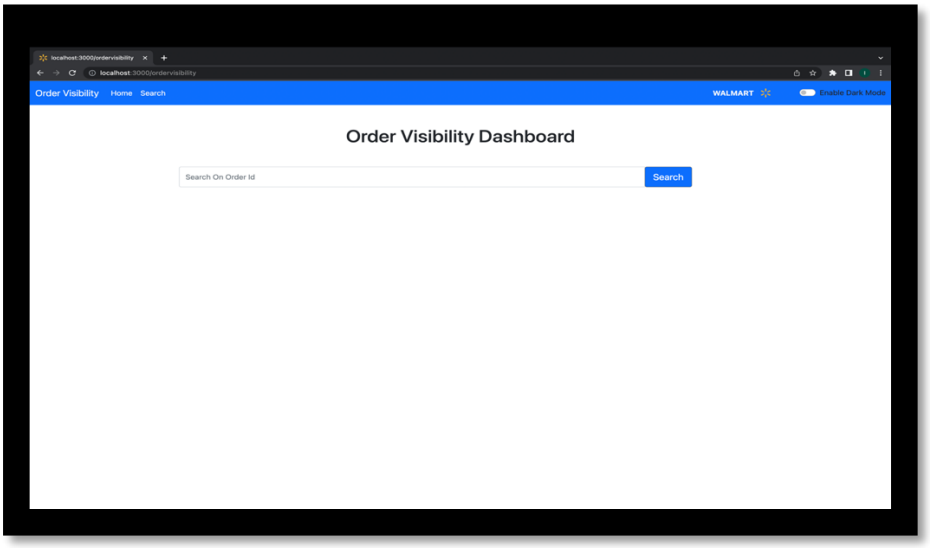
❖ The project is still in its planning phase.

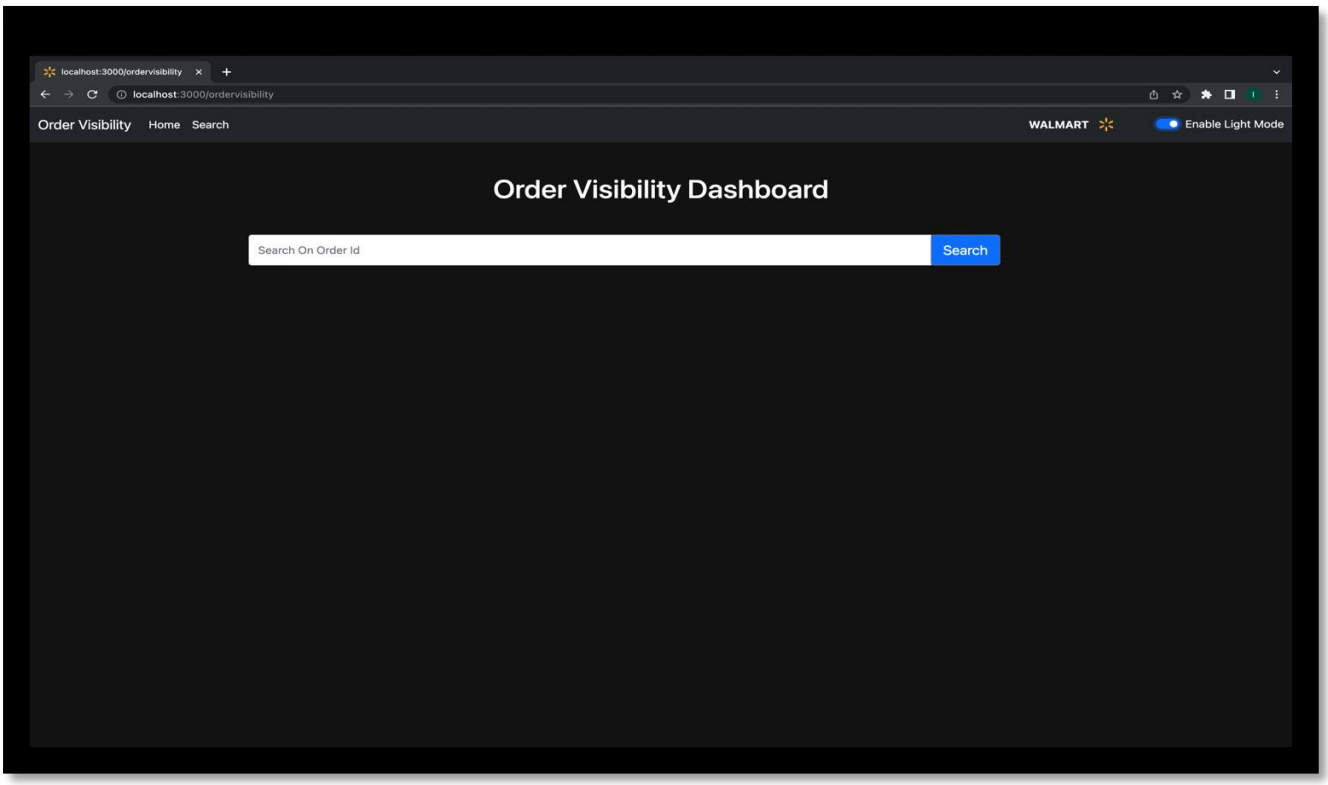
❖ Currently the analysis of the problem statement and requirements are being discussed.

❖ The project might be ready to use till September 2022!

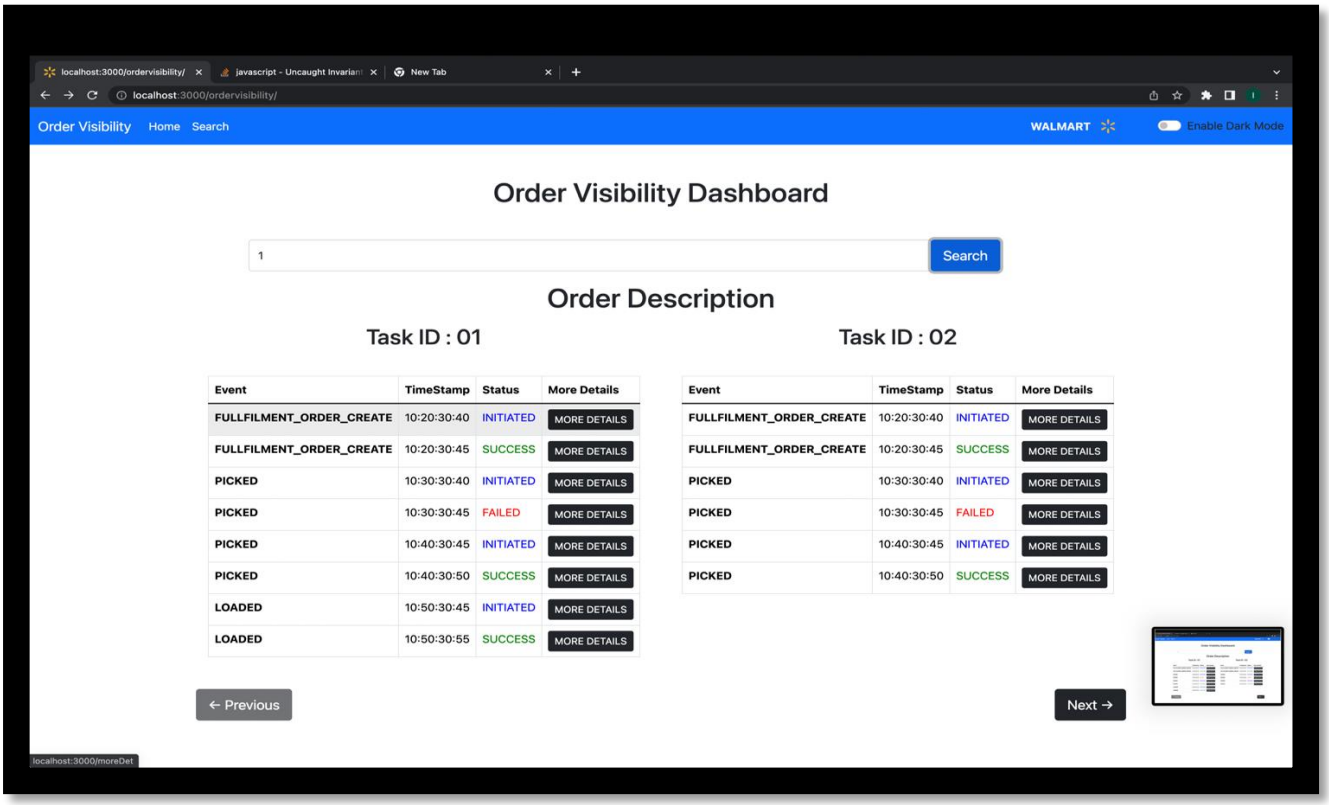
10.3 Snapshots from UI side

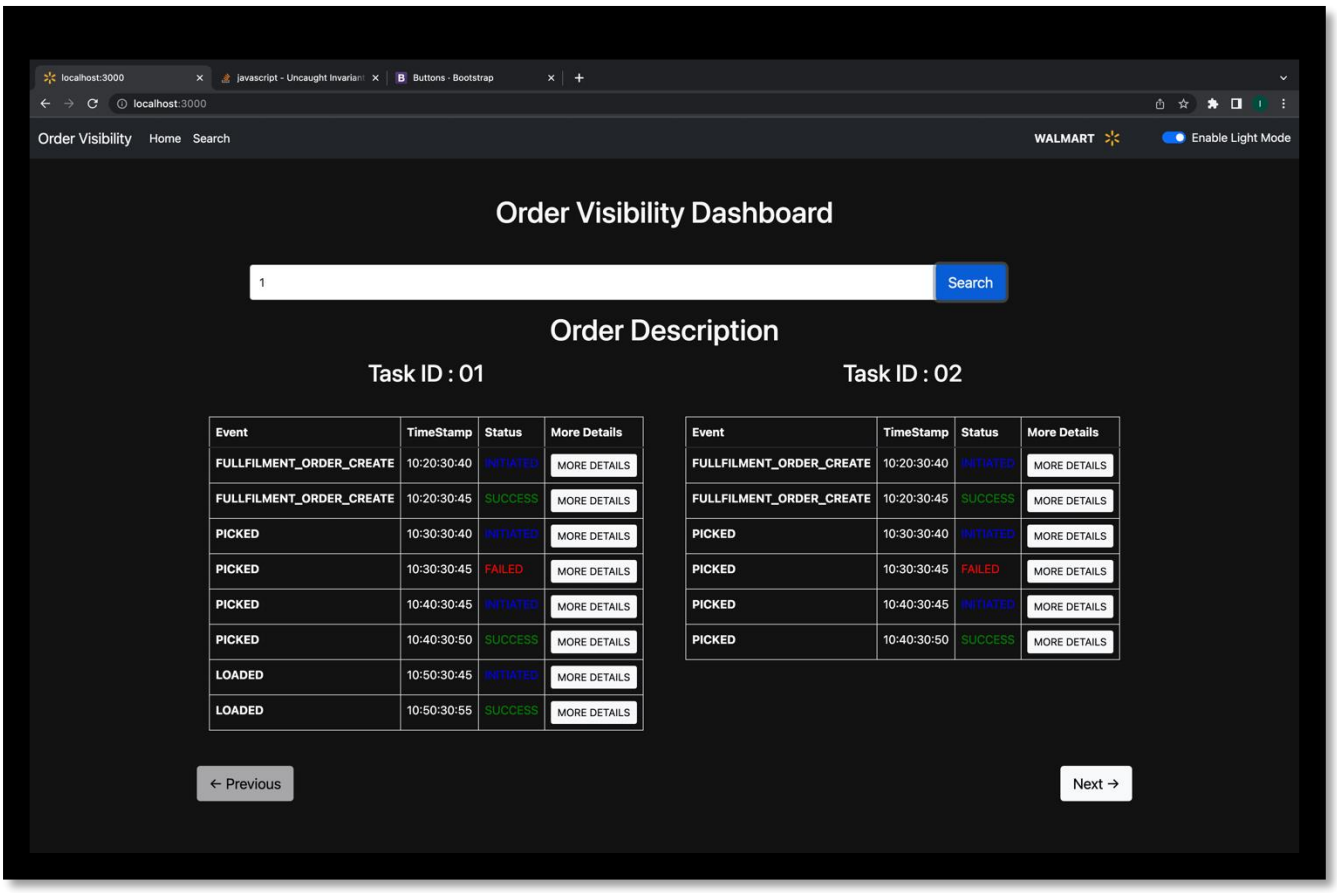
10.3.1 Home page



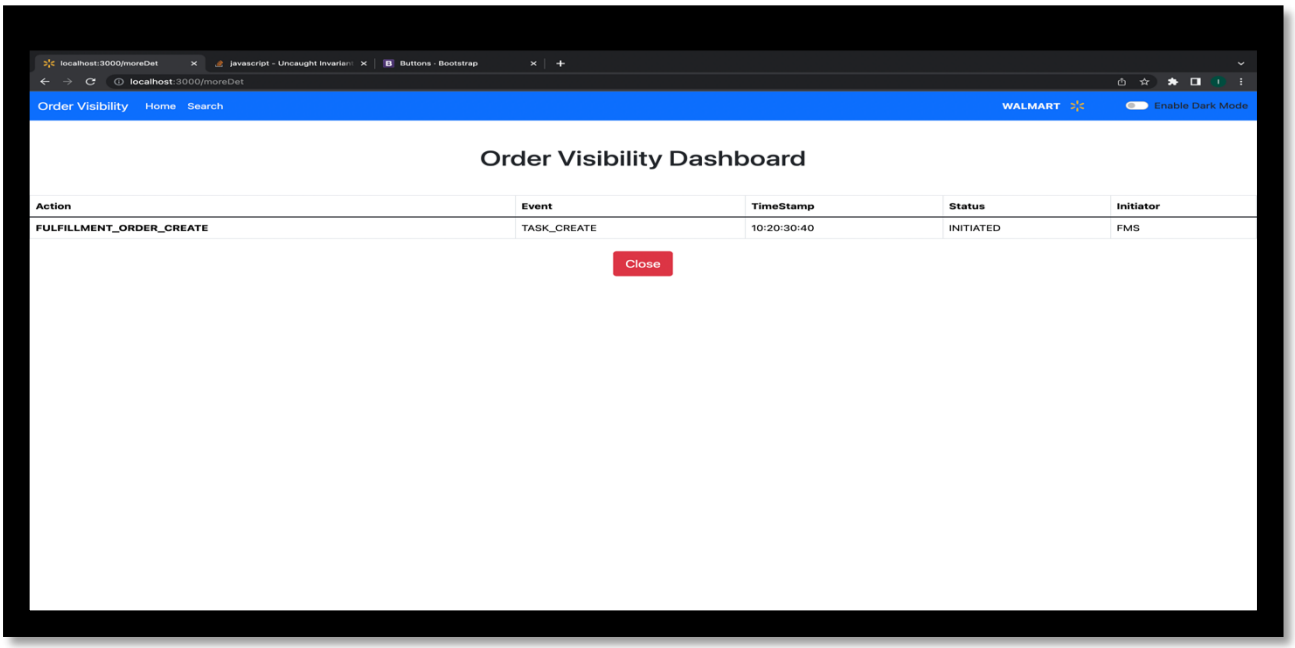


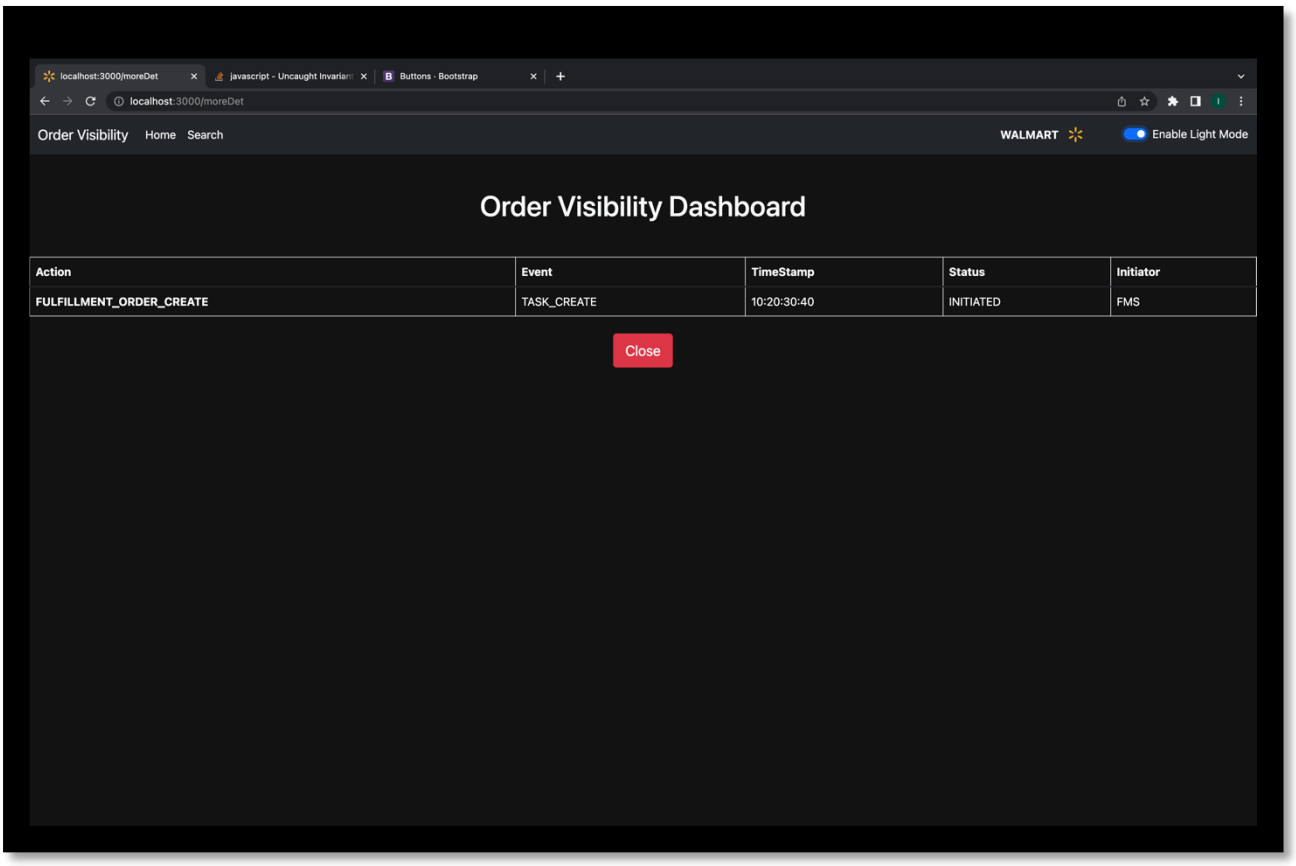
10.3.2 Displaying all the fetched events in a tabular form after performing search



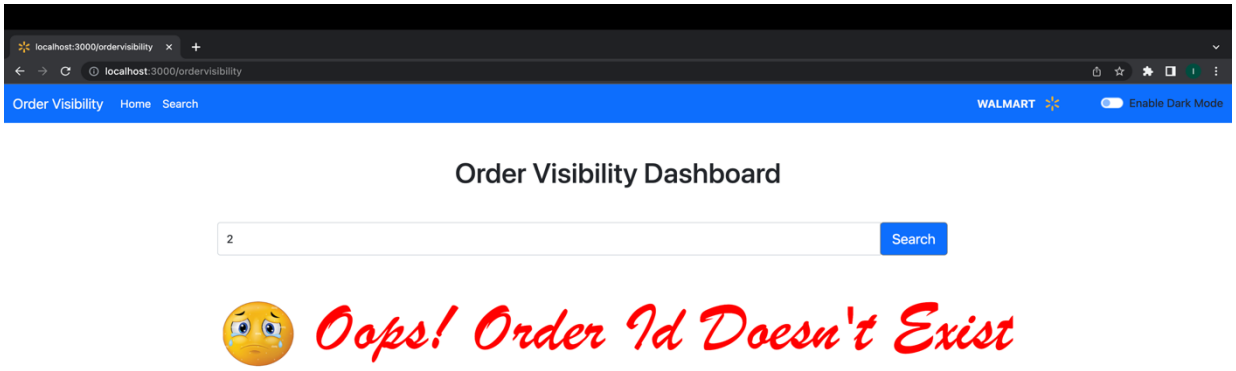


10.3.3 Displaying detailed description of a particular event





10.3.4 Displaying error message for invalid Order Id



Chapter – 11

Summary & Conclusion

Training and development are considered as a strategy for growth in the field of Computer Science and Information Technology. It is adopted by the organization to fill the gap between skills and future opportunities. These initial and advance training programs definitely enhance skills, improve efficiency, and productivity and growth opportunities for employees, mainly freshers. Skills, knowledge and attitudes are the basics for efficient running of a business through the human resources of an organization

By improving the capabilities of employees, organizational capabilities can also be improved. In result, the structure of organization become flatter, in which there will be fewer levels between the top and the bottom employees. Many organizations provide coaching to their high-potential employees to develop their skills in creativity, thinking, innovation, vision, motivating others and brainstorming. Rather than putting the employees into management and expecting that they will develop their knowledge and skills on their own, organizations can systematically develop their skills through combination of these technical training and development programs.

Evaluation of training must be appropriate for the persons and situations. The feedback from learners is important not only for instructors but also for confidence of the learner.

As a result of this internal/initial training, I am now aware of all these technologies mentioned, and also am ready to use that knowledge in building the upcoming project.

I was also given a series of evaluations which I passed with utmost accuracy to provide the proof of learning and the ability to use that knowledge practically.

Chapter – 12

References

1) **www.google.com**

2) **https://en.wikipedia.org/wiki/**

3) **www.tutorialspoint.com**

4) **www.javatpoint.com**

5) **www.w3schools.com**

6) **https://developer.mozilla.org/en-US/docs/Web/JavaScript**

7) https://reactjs.org/

8) https://getbootstrap.com/

9) https://www.elastic.co/