-: JAVA REALTIME PROJECT :-

****

Java Realtime Project (Fullstack Development) => 20-JRTP

What is the main aim of project training ?

-> People completing these courses in the institutes

Core Java
Adv Java
Spring
Hibernate
Spring Boot & Microservices

-> AFter completion of theses courses are you able to develop one project end to end ?

1) Can you tell me where to use OOPS in the Project ?

2) Can you tell me where to use I/O streams in the project?

3) Can you tell me where to use Collections in the Project?

4) Can you tell me where to use Multi Threading the project?

5) Where to use lambdas in the project ?

6) Where to use Streams in the project?

7) How to implement webservice calls in the project ?

8) How to handle exceptions the project ?

9) How to implement security in the project?

==========

In this course we are going to develop applications with below tech stack

JDK 1.8
Spring Boot
REST APIs
Microservices
Angular 13
AWS Cloud
20 + Realtime Tools

Java Realtime-Project (Fullstack Development) - Online Training

(20-JRTP)

Why we need to do this java realtime-project training ?
Course Details (Pre-requisites & Course Content)

-> Now a days companies are looking for Fullstack Developer

Fullstack = Front End Development + Backend Development

-> Front end means user interface which will be displayed on web pages

-> Backend means the business logic of our application which will not be displayed for end user

Note: When user performs some operation on the front end then backend logic will execute

-> Now a days Fullstack developers are having lot of demand in the market (high packages)

Java Fullstack Developer

Backend Technologies

Core Java
Adv Java
Hibernate
Spring Boot
REST APIs
Microservices

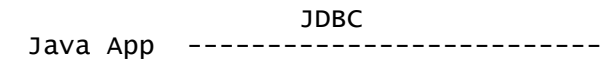
Core Java : Fundamental concepts of java

(syntaxes, data types, variables, methods, constructors, arrays, strings, oops, io streams, threads, exception handling, generics, annotations, lambdas, streams, collections)

-> In core java we will learn how to write programs and how to build logics

Adv Java : Advanced concepts in java programming

JDBC : Connecting to database with Java program to perform DB operations



Servlets : To make our application as web application
(Everybody can access our application using internet)

JSP : Java Server Pages (User interface) --- Outdatd in the market
Note: People are using JS technologies (Angular & React) For Frontend Development.

Hibernate

-> Hibernate is an ORM framework
-> Hibernate Framework is developed on top of JDBC
-> Hibernate framework providing few common logic to perform DB operations
   (driver loading, connections, cpool, transcations, exceptions etcc)

      Java App -----------> Hibernate --------> JDBC ----------> Database

Spring Framework

-> Spring is an application development framework
-> Using Spring we can develop applications with loosely coupling
-> Standalone, web, distributed applications can be developed using Spring
-> In Spring Framework we have to take of configurations

Spring Boot

-> Spring Boot is one approach to develop Spring Based applications
-> The main advantage of Spring Boot is Auto-Configuration
-> Spring Boot is trending in the market

REST APIs

-> REST APIs are used to develop distributed applications

-> If one application is communicating with another application then we will call it as distributed application

-> Application Backeng logics are implementing using REST APIs

Microservices

-> Microservices is a design Pattern to develop applications with loosely coupled functionalities

Note: Now a days we are using Spring Boot to develop Back End Rest APIs by following Microservices Design Pattern

Front End Technologies

HTML
CSS
Boot Strap
Java Script
Angular or React JS

Java Fullstack Developer

Backend Tech Stack : Core Java + Adv Java + Hibernate + Spring Boot + REST APIs + Microservices

Front End Tech Stack : HTML + CSS + Boot Strap + Java Script + Angular or React JS

Database : Oracle / MySQL/ PostgreSql / MongoDB

Tools : Maven, Git Hub, Jenkins, Sonar Qube , JUnit, Jacocco, JMETER, Jenkins,

Docker, K8S

Cloud : AWS / Azure / GCP

Java Real-Time Project (20-JRTP)

Pre-Requisites :

Core Java, Adv Java, SQL, Spring Core, Hibernate Basics

Note: Spring Boot & Microservices (Paralelly You can learn)

Course Content

Part-1 : Software Industry Details

Part-2 : Realtime Tools (20+)

Part-3 : AWS Cloud

Part-4 : Angular 13

Part-5 : Mini Projects (3)

Part-6 : Major Project (1)

Part-7 : Interview Guide(FAQs, Resume preparation, Mock interviews, Placements)

Daily class timings : 9:00 AM - 10:30 AM (IST) (Mon-Sat)
Duration : 3.5 Months

Types of software Companies

We can categorize IT companies into 3 types

1) Product Based Companies

2) Service Based Companies

3) Outsourcing Companies

Product Based Companies

-> The companies which are developing projects/products and selling to customers in the market are called as Product Based Companies.

Ex: Microsoft, Apple, HP, DELL, Oracle, Sony etc.....

-> Product Based Companies will focus mainley on below topics in the interview

a) Data Structures
b) Algorithms
c) Problems Solving
d) Design Patterns
e) Coding Test

-> Product Based Companies will provide high packages

Avg Package = no.of.years.of.exp * 8 lakhs

Service Based Companies

-> The companies which are developing projects as per client given rquirements are called as Service Based Companies.

Ex : TCS, Infy, Wipro, TechM, Cognizent, Capgemeni, Deloitte, Accenture, HCL etc..

-> Service Based Companies will focus mainly on below topics in interview

a) Coding Test
b) Fullstack Developer (Back end + Front End)
c) Microservice
d) Spring Boot
c) REST API
d) Angular / React JS
e) Cloud Platforms

-> Service Based Companies package structure will be like this

Package = no.of.years.of.exp * 4 Lakhs

Note : In service based companies, ONSITE opportunities will be available
(Going to client location based on project requirement)

Outsourcing Companies

-> The companies which are providing employees for other companies on contract basis are called as Outsouring Companies.

Ex: sourceone, magnainfotech, a2zresources etc..

Note: Now a days most of service based companies are also doing outsourcing business

Parent	Client
HCL	Microsoft
Capgemini	Oracle
CTS	JPMC
TCS	DELL

-> You will do work in client company and you will get salary from parent company

---
Types of Projects In IT

Q) What is Software Project ?

Ans) Set of programs are called as Project

Q) Why we need to develop projects?
Ans)

Software projects are used to reduce human efforts
Software projects are used to simplify humans life by automating the work

Ex: Ticket Booking, online shopping, banking operations, food orders, money transfer etc...

In IT companies we can see below types of Projects

1) Scratch Development Projects

2) Maintenence / Support Projects

3) Migration Projects

-> Scratch development means developing the project from very beginning (brand new)

-> Maintenence / Support means making changes to existing projects

a) Change Request (CR)
b) Enhancement
c) Bug Fixing

-> Change Request means modifying existing functionality (CR)
-> Enhancement means adding new functionality in the existing project
-> Bug Fixing means resolving the exceptions occuring in the project

-> Migration Project means converting project from old technology to lastest technology

Mainframework project ----------> Java Project

Struts ----------> spring boot

java 1.6 v ----------> Java 11

Types of companies
What is project
Why Project
Types of Projects

Types of Jobs in Software Industry

-> IT companies are providing 2 types of jobs

a) Permanent Job
b) Contract Job

-> In permanent job more benefits will be available for the employee... those are

a) Profident Fund (PF) (12% of Basic Salary)
(Employee Contribution + Employer Contribution)

b) Health Insurance (For You + For Your Family)

c) Notice Period

Note: If you want to leave the company you have to serve notice period. If company want to terminate you from your job then company should pay notice period salary.

Note: Company to Company Notice period duration will be different

Deloitte -- 2 Months
Accenture -- 3 Months
Infy -- 3 Months

Note: Some companies will provide BuyOut option. That means we can buy our notice period by paying amount to company. (Its depends on Company decision)

Note: We can negotiate our notice period with manager.

-> Contract jobs are temporary jobs. When the work is there in project then company will take you and once the work is completed company will terminate you from the job ...

-> For contract jobs notice period will be 1 week or 10 days.

-> Employee benefits will be less when you go for Contract jobs

Note: Sometimes companies will take contract employees and based on their performance they will convert contract job to permanent job.

Types of Teams in the project

---

1) Onshore team
2) Offshore team

-> The team which is working in client location is called as Onshore team.

-> The team which is working in non-client location is called as Offshore team.

-> Mainly Business Analyst or Functional Team members will work in client location.
-> Development, Testing & Deployments will done by offshore team members.

-> Onshore & offshore team members will communicate daily through bridge calls to discuss project related details.

-> For bridge calls companies will use below softwares

Zoom, Skype Business, Microsoft Teams etc....

-> For sending emails in the company we will use Microsoft Outlook

-> Company will provide official email for us

email : ashok.b@tcs.com

Types of Jobs

Permanent Job

Contract Job
Types of Teams
Onshore Team
Offshore Team
Bridge Call

Designations in IT industry (Roles in IT Companies)

-> Company to company Designations will be different

-> Software Trainee / Associate Analyst (0 - 2 years Exp)

-> Software Engineer / Business Technology Analyst / System Engineer (2 - 4 Years)

-> Consultant / System Engineer Level - 4 (4 - 6 Years)

-> Sr. Consultant / Team Lead / Tech Lead / Sr.S/w Engineer - (6 - 9 years)

-> Manager (9 - 12 Years)

-> Sr. Manager (12 - 16 Years)

-> Director (16 - 20 Years)

-> Vice Pricedent (20+ Years)

-> Based on our performance company will give promotion from one role to another role.
-> Based on the Roles company will decide salary structure

Types of Software Companies
1) Product Based
2) Service Based
3) Outsourcing

Types of Projects
1) Scratch Development
2) Maintenence/Support
3) Migration

Types Of Jobs
1) Permanent Job
2) Contract Job

Types of Teams
1) Onshore Team
2) Offshore Team

Bridge Call

Roles in IT Companies

How to cover Gap as experience

-> After this course completion, you can keep 3 to 4 years exp and you can attend interviews

Scenario-1

Gradudated in 2019
From 2019 to 2022 (3 Years Gap)
You can show that 3 years gap as exp

Scenario-2

Graduated in 2018
From 2018 to 2022 (4 Years Gap)
You can say Attended trainings For 5 Months
You can show 3.5 Years exp

Scenario-3

Graduated in 2017
From 2017 to 2022 (5 Years Gap)
2 years i have done non-it job (2017 - 2019)
You can show 3 Years exp in IT Job (2019 - 2022)

Scenario-4

Graduated in 2016
From 2016 to 2022 (6 Years gap)
2.5 years worked as Lab assistent in Engineering College (2016 - 2019)
3.5 Years Exp in IT Field (2019 - 2022)

Scenario-4

Graduated in 2015
From 2015 to 2022 (7 Years gap)
3.5 Years Gap you can show as Non-IT job
3.5 Years gap you can show as IT exp

Scenario-5

I am already having pf account, will it become a problem?

-> If you have pf number or if you don't have pf number that will not become issue
-> Every Company will provide you new pf account.

Scenario-6

I am working in IT company from 2 years as a core java developer.
Can i get the job in other IT company with this experience ?

In this course you will learn Angular, AWS, Fullstack Development
After this course you prepare your resume as fullstack developer with Spring Boot, REST API, Microservices, Angular & AWS.

To show the gap as experience you need to submit below documents to company
---
1) Offer letter
2) Hike Letter
3) Last 3 Months Payslips
4) Bank Stmts
5) Form-16
6) Experience Letter

7) Reliving Letter

Note: All the above documents consultancy people will provide for you (20k)

-> After Joining in the company with fake exp, if our Background Verification Got Failed no need worry because they will not kill us. They will just terminate us from the job. We can't go to that company in future.

Note: 70-80 % of people in IT are fake experienced people only.

-> IT companies also knows that people are coming to interviews with fake exp. Companies are having requirement for Skilled People. If you have strong technical skills they will select you in the interview.

After completion of our Java Realtime-Project Training how much package we can expect ?

With 4 Years Exp one lady got 23 Lakhs Package
With 3 Years Exp one boy got 19.8 lakhs package

Module-1 : Software Industry Details --- Completed

Module-2 : Java Realtime Tools

-> As part of project development we will use several tools

1) Build Tools : Ant, Maven and Gradle

- Project Creation
- Dependencies/Jars downloading
- Project Compilation
- Execute Unit Test Cases
- Project Packaging (jar or war)

2) Code Repository Tools : SVN, GIT Hub and BitBucket

- Code Integration at one place
- Monitored Access (who, when, why and what)

3) Project Management Tool : JIRA

- Project work planning
- Work allocation to team members
- Work status tracking
- Bug Reporting

4) Logging Tools : Log4J, Log4J2, Logback, Logstash & SLF4J

- To generate log messages of application
- Tracking Application execution details
- Tracking Exceptions Occured in application

5) Log Monitoring Tools : Putty, WinScp and Splunk

- To see log messages of our application

6) Code Review Tools : PMD and Sonar Qube

- To find developer mistakes in the code

7) Unit Testing Tool : JUNIT with Mocking

- To test unit amount of work
- Junit is used to automate unit testing
- Mocking is used for isolated unit testing

8) Code Coverage : Jacocco and Cobertura

- To identify how many lines executed in unit testing
- It will provide lines executed and lines not executed
- Code Coverage report will help to improve unit testing scenarios

9) CI CD Deployment Tools : Jenkins

- To automate application deployment process
- Deployment Schedules

10) API Testing tool : Postman and Swagger UI

- To test backend rest apis
- Swagger can generate rest api documentation also

11) Containerization Tool : Docker

- To generate application and its dependencies as one container

12) Orchestration Tool : Kubernetes (K8S)

- To deploy docker containers on cluster

13) Reports Generation : Apache POI (Excel) & IText API (PDF)

- Excel Report generation
- Pdf report

14) Message Queue : Apache Kafka

- To process stream of data

15) Distributed Cache : Redis Cache

- Cache is used to reduce no.of db calls
- Cache is used to improve performance of the application

16) Performance Testing : JMETER and HP Load Runner

- To test application stability and responsiveness

=> Along with coding we should have practical knowledge on above tools also

Build Tools
++++++++++++++
-> Build tools are used to automate application build process

-> Build tools are going to perform below operations

1) Create Project Structure
2) Download Required Dependencies (jars)
3) Compile Source Code
4) Execute Unit Test cases
5) Package our project (jar or war)

-> To Automate build process of java application we have 3 build tools

1) Ant (Outdated)
2) Maven
3) Gradle

Note: Spring Boot supports both Maven and Gradle to create boot applications.

Apache Maven
++++++++++++

-> Maven is an open source software given by Apache organization

-> Maven is acting as build tool for java applicatons

-> Maven objectivies

a) Making the build process easy
b) Providing an uniform build system
c) Providing quality project information
d) Encouraging better development practices

Maven Environment Setup
++++++++++++++++++++++

1) Download maven software from official website (zip file)

URL : https://maven.apache.org/download.cgi
File Name: apache-maven-3.8.5-bin.zip

2) Extract zip file and set MAVEN_HOME in environment variable

3) Set Path for Maven in Environment variables

4) Set JAVA_HOME also in environment variables

JAVA_HOME = C:\Program Files\Java\jdk1.8.0_201
MAVEN_HOME = C:\apache-maven-3.6.0

Note: Maven Path we have to set upto maven bin directory

C:\apache-maven-3.6.0\bin

=> To verify maven setup, open command prompt and execute below command

$ mvn -v

Note: It should display version of the maven. If it is displaying version that means maven setup is successfull.

-> In last session we discussed about What is Maven and Maven Setup in our machine

-> Maven is a build tool which is used to automate application build process.

-> We will use Spring Tool Suite (STS) IDE to develop our applications

Note: Every IDE will have Maven support

-> We can create maven projects in the IDE directley

-> Create Workspace folder (20-JRTP/Workspace)

-> Open Tool Suite IDE and choose the workspace...

Note: The projects we create in IDE will stored in workspace folder.

Creating Maven Project

-> Open IDE -> File -> New -> Maven Project -> Next -> Select quick start archetype
-> Enter groupId, artifactId click on Finish

Note: Archetype represents type of the project

maven-archetype-quick-start : Standalone Project

maven-archetype-webapp : Web Application

groupId : We can give company domain name or project name

artifactId : We can give project name or project module name

Note: After project is created check project build.

Note: By default JRE will be availablein build path, add JDK to build Path.

Maven Project Folder Structure
---
src/main/java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
target
pom.xml

-> src/main/java folder is used to create application classes. The main source code of the application will be availabe here only.

-> src/main/resources folder is used to store application configuration files like .properties | .yml | .xml etc...

-> src/test/java folder is used to create Unit Test classes of our application (Junit classes)

-> src/test/resources folder is used to created config files required for Unit Testing classes execution.

-> Maven Dependencies folder represents the jars which are downloaded and added to our project by Maven tool

-> target folder contains .class files (it is equal to bin folder)

-> pom.xml file is the main file in Maven tool

-> pom stands for Project object model. Maven relatd configurations we will do in pom.xml file.

-> pom.xml acts as mediator between programmer and maven software.

```
                              pom.xml
                programmer -----------------> Maven Software
```

Adding Dependencies in the Maven

-> Dependencies means the jars which we want to use in our project

-> If we want to develop our project using Spring framework then we have to add Spring Jars as dependencies in our project.

-> Dependencies we will add in our projct pom.xml file

-> We can get dependencies from "www.mvnrepository.com"

Note: When we add dependency in pom.xml file, maven will download that dependency and it will add that to our project build path.

```
        <dependencies>
                <dependency>
                        <groupId>junit</groupId>
                        <artifactId>junit</artifactId>
                        <version>3.8.1</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-context</artifactId>
                        <version>5.3.18</version>
                </dependency>
        </dependencies>
```

-> When we add 'spring-context' dependency, maven downloading spring-core, spring-beans, spring-aop, spring-expression. This is called as Transtivite Dependency management.

-> Maven download dependencies from Repository.

-> Repository means collection of 'jars'.

Maven Repositories

Maven supports 3 types of repositories

1) Central Repository

2) Remote Repository

3) Local Repository

-> Local Repository will be created in our machine
   (Local Repo Location : C://users/<username>/.m2)

-> Every IT Company will maintain their own Remote Repositories for managing jars.

-> Central Repository is maintaining by Apache organizaation

-> When we add dependency in pom.xml file then maven will search for that dependency in Local Repository first. If it is not available then it will download that dependency from Maven Central Repository and it will store in Local Repository. From Local Repository dependency will be added to project build path.

-> When we are working in the realtime, we will configure Remote Repository in our IDE. When we add dependency then maven will download dependency from Remote Repository to Local Repository. From Local Repository dependency will be added to project build path. To connect with Remote Repository our team lead will provide one "settings.xml" file and we need to import that file into our IDE. That file contains configuaration to connect with Remote Repository.

How to import settings.xml file in our IDE

Open IDE -> In Main Menu select Window -> Preferences -> Type Maven -> Select User Settings -> Import settings.xml -> Re-Start IDE.

Maven Goals

-> In maven we have several goals to perform build process

-> Every Maven goal is associated with one maven plugin

-> Plugin is going to perform actual task of the goal.

clean   : It is used to delete target folder in our project
compile : It is used to compile all the java clases in our project (generates .class files)
test   : It is used to execute junit classes (unit test execution)
package : It is used to package our application as jar or war file (artifact generation)
install : To make our project as dependency in maven repository. Other apps can use this   dependency

Executing Maven Goals

Right Click On Project -> Run as -> Maven Build -> Enter Goal Name -> Apply -> Finish

Note : We can give multiple goals at a time with space as delimiter

```
clean
clean compile
clean compile test
clean compile test package
```

Build Tools
++++++++++++++
-> Build tools are used to automate application build process

-> Build tools are going to perform below operations

1) Create Project Structure
2) Download Required Dependencies (jars)
3) Compile Source Code
4) Execute Unit Test cases
5) Package our project (jar or war)

-> To Automate build process of java application we have 3 build tools

1) Ant (Outdated)
2) Maven
3) Gradle

Note: Spring Boot supports both Maven and Gradle to create boot applications.

Apache Maven
++++++++++++

-> Maven is an open source software given by Apache organization

-> Maven is acting as build tool for java applicatons

-> Maven objectivies

a) Making the build process easy
b) Providing an uniform build system
c) Providing quality project information
d) Encouraging better development practices

Maven Environment Setup
++++++++++++++++++++++++

1) Download maven software from official website (zip file)

URL : https://maven.apache.org/download.cgi
File Name: apache-maven-3.8.5-bin.zip

2) Extract zip file and set MAVEN_HOME in environment variable

3) Set Path for Maven in Environment variables

4) Set JAVA_HOME also in environment variables

JAVA_HOME = C:\Program Files\Java\jdk1.8.0_201
MAVEN_HOME = C:\apache-maven-3.6.0

Note: Maven Path we have to set upto maven bin directory

C:\apache-maven-3.6.0\bin

=> To verify maven setup, open command prompt and execute below command

$ mvn -v

Note: It should display version of the maven. If it is displaying version that means maven setup is successfull.

-> In last session we discussed about What is Maven and Maven Setup in our machine

-> Maven is a build tool which is used to automate application build process.

-> We will use Spring Tool Suite (STS) IDE to develop our applications

Note: Every IDE will have Maven support

-> We can create maven projects in the IDE directley

-> Create Workspace folder (20-JRTP/Workspace)

-> Open Tool Suite IDE and choose the workspace...

Note: The projects we create in IDE will stored in workspace folder.

Creating Maven Project

-> Open IDE -> File -> New -> Maven Project -> Next -> Select quick start archetype
-> Enter groupId, artifactId click on Finish

Note: Archetype represents type of the project

maven-archetype-quick-start : Standalone Project

maven-archetype-webapp : Web Application

groupId : We can give company domain name or project name

artifactId : We can give project name or project module name

Note: After project is created check project build.

Note: By default JRE will be availablein build path, add JDK to build Path.

Maven Project Folder Structure

---

src/main/java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
target
pom.xml

-> src/main/java folder is used to create application classes. The main source code of the application will be availabe here only.

-> src/main/resources folder is used to store application configuration files like .properties | .yml | .xml etc...

-> src/test/java folder is used to create Unit Test classes of our application (Junit classes)

-> src/test/resources folder is used to created config files required for Unit Testing classes execution.

-> Maven Dependencies folder represents the jars which are downloaded and added to our project by Maven tool

-> target folder contains .class files (it is equal to bin folder)

-> pom.xml file is the main file in Maven tool

-> pom stands for Project object model. Maven relatd configurations we will do in pom.xml file.

-> pom.xml acts as mediator between programmer and maven software.

```
                              pom.xml
               programmer -----------------> Maven Software
```

Adding Dependencies in the Maven

-> Dependencies means the jars which we want to use in our project

-> If we want to develop our project using Spring framework then we have to add Spring Jars as dependencies in our project.

-> Dependencies we will add in our projct pom.xml file

-> We can get dependencies from "www.mvnrepository.com"

Note: When we add dependency in pom.xml file, maven will download that dependency and it will add that to our project build path.

```
        <dependencies>
                <dependency>
                        <groupId>junit</groupId>
                        <artifactId>junit</artifactId>
                        <version>3.8.1</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-context</artifactId>
                        <version>5.3.18</version>
                </dependency>
        </dependencies>
```

-> When we add 'spring-context' dependency, maven downloading spring-core, spring-beans, spring-aop, spring-expression. This is called as Transtivite Dependency management.

-> Maven download dependencies from Repository.

-> Repository means collection of 'jars'.

Maven Repositories

Maven supports 3 types of repositories

1) Central Repository

2) Remote Repository

3) Local Repository

-> Local Repository will be created in our machine
   (Local Repo Location : C://users/<username>/.m2)

-> Every IT Company will maintain their own Remote Repositories for managing jars.

-> Central Repository is maintaining by Apache organizaation

-> When we add dependency in pom.xml file then maven will search for that dependency in Local Repository first. If it is not available then it will download that dependency from Maven Central Repository and it will store in Local Repository. From Local Repository dependency will be added to project build path.

-> When we are working in the realtime, we will configure Remote Repository in our IDE. When we add dependency then maven will download dependency from Remote Repository to Local Repository. From Local Repository dependency will be added to project build path. To connect with Remote Repository our team lead will provide one "settings.xml" file and we need to import that file into our IDE. That file contains configuaration to connect with Remote Repository.

How to import settings.xml file in our IDE

Open IDE -> In Main Menu select Window -> Preferences -> Type Maven -> Select User Settings -> Import settings.xml -> Re-Start IDE.

Maven Goals

-> In maven we have several goals to perform build process

-> Every Maven goal is associated with one maven plugin

-> Plugin is going to perform actual task of the goal.

clean   : It is used to delete target folder in our project
compile : It is used to compile all the java clases in our project (generates .class files)
test   : It is used to execute junit classes (unit test execution)
package : It is used to package our application as jar or war file (artifact generation)
install : To make our project as dependency in maven repository. Other apps can use this dependency

Executing Maven Goals

Right Click On Project -> Run as -> Maven Build -> Enter Goal Name -> Apply -> Finish

Note : We can give multiple goals at a time with space as delimiter

```
clean
clean compile
clean compile test
clean compile test package
```

-> When we execute package goal, project will be packaged and it will be stored in target folder

-> By default maven will consider, Project ArtifactID and Version for package name

-> We can configure our name for packaging file using <finalName> tag in pom.xml like below

```
	<build>
		<finalName>Maven-App-1.0.RELEASE</finalName>
	</build>
```

-> When we execute Maven install goal, it will create as dependency in Local Repository. That dependency we can use in another project.

Working with Maven Install Goal

PART-1
++++++
1) Create Maven Project (02-Pwd-Utils)

2) Create PasswordService class with encode ( ) and decode ( ) methods

- String encode (String txt) => Takes plain text as input and returns encoded text
- String decode (String encodeTxt) => Takes encoded txt as input and returns decoded text

Note: We are using java.util.Base64 class to perform Encoding and Decoding

```
package in.ashokit.security;

import java.util.Base64;
import java.util.Base64.Decoder;
import java.util.Base64.Encoder;

public class PasswordService {

	public static String encode(String txt) {
		Encoder encoder = Base64.getEncoder();
		return encoder.encodeToString(txt.getBytes());
	}

	public static String decode(String encodeTxt) {
		Decoder decoder = Base64.getDecoder();
		byte[] decode = decoder.decode(encodeTxt);
		return new String(decode);
	}
}
```

3) Configure <finalName> like below in pom.xml file

```
	<build>
	    <finalName>ashokit-pwd-security</finalName>
	</build>
```

4) Make changes to artifactId and version like below

```
	<artifactId>ashokit-pwd-security</artifactId>
	<version>1.0.RELEASE</version>
```

5) Right Click On project and Execute Maven install goal (It will install as dependency in local repo)

Note: Go to maven local repository and verify installed dependency

(Location :
C://Users/<uname>/.m2/repository/in/ashokit/ashokit-pwd-security/1.0.RELEASE)

6) Open the pom file available in above location and copy the dependency details
(groupId, artifactId and version)

```
<groupId>in.ashokit</groupId>
<artifactId>ashokit-pwd-security</artifactId>
<version>1.0.RELEASE</version>
```

++

Part-2
++++++

1) Create New Maven Project (03-Admin-Mgmt)

2) Add 'ashokit-pwd-security' project dependency in pom.xml file

3) Create a Demo class and use 'PasswordService' class to perform Encoding and Decoding.

Note: PasswordService class which is available in another project getting used in our 03-Admin-Mgmt project.

```
package in.ashokit;

import in.ashokit.security.PasswordService;

public class Demo {

	public static void main(String[] args) {
		String encodedTxt = PasswordService.encode("india@123");
		System.out.println(encodedTxt);
		String decodedTxt = PasswordService.decode(encodedTxt);
		System.out.println(decodedTxt);
	}
}
```

++
Creating Web application Using Maven
++

1) Create maven project using 'maven-archetype-webapp'

2) Configure 'jdk' in project build path (Right Click -> Build Path -> Libraries)

3) Add servlet-api dependency in project pom.xml file

4) Configure Tomcat Server in IDE (Servers section)

5) Right CLick on the project and Run as -> Run on Server

6) Right Click On project -> Run As -> Maven Build -> clean package -> Apply -> Finish

	(For web app 'war' file will be created inside 'target' folder)

Dependency Scopes In Maven

-> Dependency scope will represent when to use that depedency in our project

(when to include that dependency in the class path)

-> We have below 6 scopes for dependencies in maven

compile (it is default scope)
runtime
test
provided
system
import

compile

This is the default scope, used if none is specified. Compile dependencies are available in all classpaths of a project.

provided

This is much like compile, but indicates you expect the JDK or a container to provide the dependency at runtime.

-> For example, when building a web application for the Java Enterprise Edition, you would set the dependency on the Servlet API to scope provided because the web container provides those classes.

-> A dependency with this scope is added to the classpath used for compilation and test, but not the runtime classpath.

Ex: servlet-api depdency

runtime

This scope indicates that the dependency is not required for compilation, but is for execution. Maven includes a dependency with this scope in the runtime and test classpaths, but not the compile classpath.

Ex: jackson dependency in spring boot

test

This scope indicates that the dependency is not required for normal use of the application, and is only available for the test compilation and execution phases.

Typically this scope is used for test libraries such as JUnit and Mockito.

system

This scope is similar to provided except that you have to provide the JAR which contains it explicitly. The artifact is always available and is not looked up in a repository.

import

This scope is only supported on a dependency of type pom in the <dependencyManagement> section.

It indicates the dependency is to be replaced with the effective list of dependencies in the specified POM's <dependencyManagement> section.

Since they are replaced, dependencies with a scope of import do not actually participate in limiting the transitivity of a dependency.

Dependency Exclusion In Maven

-> When we configure dependency in pom.xml file then maven will download all realted dependencies for that. It is called as transitive dependency management.

-> For example, when we add 'spring-context' dependency it is downloading below dependencies

spring-beans
spring-core
spring-context
spring-aop
spring-expression
spring-jcl

-> With this transitive dependency management, we have advantages and dis-advantages

-> Advantage is with less configuration we are getting more dependencies

-> Dis-Advantage is all those dependencies will be loaded then memory will be wasted.

-> To overcome this problem maven provided 'dependency exclusion' concept

-> Using this exclusion concept we can remove unwanted dependencies to save memory

Syntax To Exclude

```
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.3.18</version>
        <exclusions>
                <exclusion>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-aop</artifactId>
                </exclusion>
        </exclusions>
</dependency>
```

Maven Multi Module Project

-> Multi Module Project is used to create projects with Parent and Child Relation.

-> With Multi Module project, we can achieve loosely coupling (For every module seperate project will be created hence maintenence of the code will become easy)

-> Create Maven Project with packaging type as 'pom' (It is called as parent project)

-> Create maven module
   (File -> New -> Other -> Maven -> Maven Module ->
                        Enter Module name & Select Parent Project -> Finish)

        Note : We can create multiple modules like this

-> Verify parent project pom.xml file (Modules will be added)

-> Verify Module project pom.xml file (Parent project info will be available)

Parent POM Looks Like Below

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>in.ashokit</groupId>
  <artifactId>05-Maven-Parent-App</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>pom</packaging>
  <modules>
        <module>Admin</module>
        <module>Reports</module>
  </modules>
</project>
```

Maven Module Project POM looks like below

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>in.ashokit</groupId>
    <artifactId>05-Maven-Parent-App</artifactId>
    <version>0.0.1-SNAPSHOT</version>
  </parent>
  <artifactId>Admin</artifactId>
</project>
```

=> When we execute Maven Goal for parent project, then it will reflect on all
modules also.

Version Control Tools
+++++++++++++++++++++

-> In realtime, multiple people will get involved in project development

-> Developers will be working from different locations

Below are two common problems we will face in project development
----
1) Code Integration
2) Monitoring code changes (Who, When, Why and What)

-> Version Control Tools providing solution for above two problems

-> We have several version control tools in the market

1) SVN (Outdated)
2) GIT Hub
3) Bit Bucket

-> By using version control tool/sofware we will create a Repository

-> Repository is a location which is used to store project source code. For Every Project one Repository will be created in realtime. Every Repository will have its own URL.

-> Team Members will connect to Repository using Repository URL to perform Code Integrations.

-> Repository will have monitored access
(It will track all the changes like who, when, why and what)

Note: Once we join in the project, team lead or team member will share project Repository URL with us.

Note: We will perform Push and Pull operations in the Repository

Push means sending code from our system to Repository
Pull means taking latest code from Repository to our System

GitHub
++++++
-> Git Hub is a cloud provider which is giving hosting, storage services over the web

-> We can create source code repositories in GitHub to store our project code

-> Git Hub providing both free & licensed accounts

Free Accounts : Limited Functionality
Licensed Account : All Functionalities

-> Git Hub will internally uses git as a version control software

Environment Setup
----
1) Create account in github.com

URL : https://github.com/

2) Download Git Client software & install it

URL : https://git-scm.com/downloads/win

-> After installation, right click on the mouse and check it
(Git GUI and Git Bash options will be displayed that means installation success)

Git Hub Repositories

-> Git hub providing 2 types of Repositories

1) public repository
2) private repository

-> public repository means everybody can access and we will choose who can commit to repo

-> private repository means we will choose who can access and who can commit to repo

Creating Git Repository

-> Login into your github account

-> Go to Repositories tab and click on 'Create New Repository'

-> Enter Repo name and select as 'Public Repo' and create it.

Note: URL will be generated for the repository

Repo URL : https://github.com/ashokitschool/JRTP20-App1.git

-> It will display commands to push project into repo

```
$ git init
$ git commit -m "first commit"
$ git branch -M main
$ git remote add origin https://github.com/ashokitschool/JRTP20-App1.git
$ git push -u origin main
```

Push Maven Project into above Git repo by following given commands

-> Go to Maven Project folder and open Git Bash
(Right click on Mouse -> Open Gitbash here)

-> Execute below commands in git bash to introduce yourself to git

```
$ git config --global user.name "your-uname"
$ git config --global user.email "your-email"
```

-> Execute below git commands to push project into repo

```
$ git init
$ git add --a
$ git commit -m "first commit"
$ git branch -M main
$ git remote add origin <git-repo-url>
$ git push -u origin main
```

Note: It will ask credentials for first time...

git status

-> This command is used to display the files which are staged and un-stage

-> The files which are newly created/modified will be in un-staged state. These files are not eligible for commit.

-> The files which are added to staging elgible for commit.

-> To add file to staging area we will use 'git add' command

git add

-> This command is used to add file/files to staging area

```
        $ git add <filename>  (It will add given file)
        $ git add --a   (It will add all modified/newly created files)


git restore
-----------
-> This command is used to unstage the file

        $ git restore --staged <filename>

git commit
----------
-> This command is used to commit staged files to local repository

        $ git commit -m 'commit-msg'


git push
--------
-> This command is used to push changes from local repository to central repository

        $ git push


Note: If we want to push code to brand new repository then we should configure
remote repo url using below command.

        $ git remote add origin <git-repo-url>


git log
-------
-> This command is used to display commits history with commit-id and
commit-message

                $ git log


git clone
---------
-> This command is used to take project from repository to local machine

                $ git clone <repo-url>

Note: When we join in the project, we have to clone the project from repo to start
our work. We can take repo url from team members.


git pull
--------
-> This command is used to take latest changes from repository

                $ git pull


Note:
-> clone command we will use for first time to take entire project from repository.
-> pull command we will use daily, to take latest change from repository.


-> We should be very careful while executing 'pull' command because it may give
conflicts if we have different changes in central reposiotyr and local repository.

Best Practise : Before making any changes, first take latest changes from
repository using pull command. Before pushing changes, take latest changes then
```

merge your changes and push it.

Note: Getting conflicts is common problem, we have to resolve conflicts manually and we have to push the files.

=> When we do commit, it will generate unique commit id with 40 characters length. From that commit id it will display first 7 characters in git repo.

Original Commit Id : ea69d669c8cfe9d6381868ff7ea2cf63e0e185dc

---

=> After making code changes, if we want to remove those channes from working tree (local file) then we can use 'restore' command

```
$ git restore <filename>
$ git restore .
```

Note: When the files are un-staged then 'git restore' is used to discard the changes in working directory. When files are staged then 'git restore' is used to un-stage the files.

-> Today morning @ 7 AM , manager assigned me one bug (BUG-ID : 101) and asked me to fix it.

-> I have started bug fixing by making code changes
(i am working on it, modified few java files - work status is IN-PROGRESS)

-> Manager called me @11:00 AM and told me, Ashok 101-BUG is not priority now... please work on BUG_ID : 121 which is high priority. After 121 bug got fixed then you can continue fixing
BUG-ID : 101

The problem is , i have modifed few files for BUG-101 fix. As issue fix is not comleted, i can't commit the modified files and i can't delete the changes bcz half work is completed.

********** In this scenario we will use 'git stash' command**********

-> 'git stash' command is used to save our changes to temporay storage and make working tree clean.

-> After main work is completed then we can get our stashed changes back using 'stash apply'

```
$ git stash
$ git stash apply
```

```
git config
git init
git status
git add <filename>
git add --a
git commit -m 'msg'
git restore
git remote add orgin <repo-url>
git push
git pull
```

git log
git clone
git stash
git stash apply
git diff

GIT Branches

-> Git branches are used to maintain multiple code bases for the project

-> When we create git repository by default 'main' branch will be created

-> All the source code of the project will be kept in 'main' branch

Note: It is not recommended to do development/ R & D works on 'main' branch directly because existing functionality may break.

-> When we want to work on new task or new experiment then create a branch and work on that branch. Once your code is fully tested and then you can merge your changes to main branch using 'Pull Request'.

-> Working with branch is always safe.

-> In realtime, we can see below branches in the repository

main
feature
develop
sit
uat
release

-> main branch will contains final source code of the project

-> feature branch is used for R & D purpose

-> develop branch is used for development activies

-> SIT branch is used for System Integration Testing (it will done by testing team)

-> UAT branch is used for USER Acceptance Testing (it will done by client)

-> Release branch is used for production deployments (Live deployment)

=> In GIT Repo we have option to create branch

(Select source branch and target branch)

//to clone default branch we will use below command
$ git clone <repo-url>

//to clone specific branch we will use below command
$ git clone -b <branch-name> <repo-url>

=> We can merge changes from one branch to another branch using 'Pull Request'

Workflow
++++++++++

-> Create Git Repo and push your project (main branch will be created)

-> Create new branch (feature) from 'main' branch

-> Clone 'feature' branch using below command

        $ git clone -b feature <repo-url>

-> Make some changes in feature branch code and push those changes to repository

-> Go to git hub repository it will ask to create 'pull request' to merge 'feature' branch changes to 'main' branch

-> Open Pull Request and Merge the changes from 'feature' branch to main branch
   (If you want then you can delete feature branch)

Note: When we are merging changes from one branch to another branch there is a chance of getting conflicts. We have to resolve those conflicts and we have to merge.

-> 'git checkout' command is used to switch from one branch to another branch

                $ git checkout <branch-name>

workflow
+++++++++
-> clone feature branch from repo

-> Navigate to project and check branch name (it will display as feature)

                $ git branch

-> Execute 'git checkout main' then you will be switched to main branch

                $ git checkout main

                $ git branch

Branches in Git Hub
Branch Creation
Cloning specific branch
Merging branches
Pull Request
What is conflict
How to resolve conflicts

Today's agenda
+++++++++++++++
git merge vs git rebase
source tree (git client)
Bitbucket
working with repositories from IDE

-> git merge & git rebase commands are used to merge changes from one branch to another branch

git merge vs git rebase

These commands are used to merge changes from 1 branch to another branch.

git merge
git rebase

//merge demo
-> Create a new folder (merge-demo) and navtigate into that folder using git bash and execute below commands

```
git status
git init
touch m1.txt
git status
git add .
git status
git commit -m "added m1.txt"
git status
git log

git branch feature
git checkout feature
git status
```

Note: From master we have created feature branch

```
touch f1.txt
git status
git add .
git commit -m 'added f1.txt"
git status
git log
```

Feature contains m1.txt and f1.txt

if you swtich to master you can see only m1.txt

```
git checkout master
```

//i will make few more changes to master

```
git status
git touch m2.txt
git add .
git commit -m 'added m2.txt"
git log
```

The changes happend in master branch are relevant to feature. I want to get master branch changes to feature branch.

we have two options

git merge
git rebase

```
git checkout feature  (i have m1.txt and f1.txt but m2.txt not available)
git merge master (m2.txt also we got into feature)
git log
```

Note: git merge command, commited all master branch changes to feature branch.

Note: See commit history we can see that at what time merge happend it is displaying that.

//rebase demo
=> Create rebase-demo folder and navigate to that folder and open cmd and execute below commands

```
git init
git status
touch m1.txt
git add .
git commit -m 'added m1.txt'
git log


git branch feature
git checkout feature
git status
git log
touch f1.txt
git add .
git commit -m 'added f1.txt'
git status
git log

//making few more changes to master
git checkout master
touch m2.txt
git add .
git commit -m 'added m2.txt'
git log
git checkout feature
git status

git rebase master (master branch changes will come to feature brach)
git log
```

Note: In commit history at that time we have done rebase will not displayed.

Note: Git merge will maintain commits history and it will log when merge happend
Note: Git rebase will make commit history linear.

Bitbucket

Bitbucket is a Git-based source code repository hosting service owned by Atlassian. Bitbucket offers both commercial plans and free accounts with an unlimited number of private repositories.

-> Create free account in bitbucket

URL : https://bitbucket.org/

-> Create Repository in bitbucket (You can choose public or private)

-> clone that repository and perform git operations using 'git bash' client

Note: GIT Bash commands are same for both 'Git Hub repository' and 'BitBucket Repository'

-> In Bit Bucket also we have public & private repositories

- We have branches
- We have pull requests

=> As of now we have used to 'git bash' to execute git commands. Git commands are used to perform Git operations like clone, commit, push, pull etc..

=> We have some GUI softwares also to perform GIT operations

- Tortoise GIT
- Source TREE
- GIT GUI

Note: We have flexibility to use any client software to perform git operations.

-> Download Source tree software and install it

URL : https://www.sourcetreeapp.com/

-> Open source tree software there we can see all the options

-> Click on clone option and choose repo and url and desitnation path

Git Hub Repo URL : https://github.com/ashokitschool/JRTP20-App1.git

Note: It will clone git repo to given destination path

-> Modify the files then you can see those modified files in source tree under 'unstaged' section

-> Click on 'Stage All' button then those files will be staged

-> Click on Commit button and enter commit msg and select 'Push' immediately checkbox and click on 'Submit' button.

-> Our changes will be committed and pushed to git hub repository.

---

git merge vs git rebase
bitbucket
source tree

Q) What is .gitignore file in git repository ?

-> .gitignore file is used configure files/folders which we don't want to consider for commits

(Ex: target, .jar, .war, .settings, .classpath etc)

-> We will keep this file in project folder.

-> The .gitignore file itself is a plain text document.

Note: If .gitignore is not working execute below command to remove cached data

$ git rm -r --cached .

----------.gitignore file----------

```
HELP.md
target/

### STS ###
.apt_generated
.classpath
.factorypath
.project
.settings
.springBeans
.sts4-cache

### IntelliJ IDEA ###
.idea
*.iws
*.iml
*.ipr

### NetBeans ###
/nbproject/private/
/nbbuild/
/dist/
/nbdist/
/.nb-gradle/
build/
!**/src/main/**/build/
!**/src/test/**/build/

### VS Code ###
.vscode/
```

Working with IDE

-> Clone project from 'git hub repository' using clone command

$ git clone <repo-url>

-> Open IDE and import clone project into IDE

(File -> import -> Maven -> Existing Maven Project -> Select Project Location -> Finish)

-> Once Project imported, right click on project select 'Team' option then we can see all git related options

---

1) Who will create git repository for project ?

Ans) Git admin will create repository for the project

2) How team members will get project from repository ?

Ans) Admin team will share repository URL then we will use 'git clone' command

3) In Realtime , we will use public repositories or private repositoires ?

Ans) We will use only private repositories. GIT Admin will configure who can access project private repository.

4) Who will create branches in repository ?
Ans) All the team members who are having access for Repository can create branches.

5) How to merge changes from one branch to another branch ?
Ans) We have 3 options to merge

1) git merge
2) git rebase
3) Pull Request

6) Why we need git branches ?
Ans) To main seperate code bases
(without distrubing existing functioanlity we can work on new implementations)

7) From which branch code will be deployed to production ?
Ans) Release branch

8) What is branch locking?
Ans) Before deployments git admin team will lock branch that means no body can commit to that branch.

Note: admin will remove write access for all team members

9) When we will get 'git conflicts' ?

Ans) In 2 scenarios we can get conflicts

1) When we do 'git pull' by keeping local changes

2) When we are doing branches merging

Note: If you have any doubt in pushing or pulling ask team member help. Don't push or pull without having full knowledge. Because of your push others code shouldn't distrub

---

++++++++++++++++++++
SDLC
++++++++++++++++++++

-> SDLC stands for software development life cycle

-> SDLC represents life cycle of software application (end to end procedure)

-> SDLC contains several phases like below

1) Requirements Gathering
2) Requirements Analysis
3) Designing / Planning
4) Development / Implementation
5) Testing
6) Deployment
7) Maintenence / Support

-> There are several SDLC methodologies available in the market

1) Waterfall Methodology
2) V-Methodology
3) Spiral Model
4) Agile Methodology (Trending)

Waterfall Methodology

-> This is the first methodology which introduced in the market
-> Waterfall is a linear methodology (step by step process)
-> In this methodology after one phase is completed then we will go for next phase
-> Waterfall methodology will move only in forward direction
-> Changing requirements in the middle is not possible
-> Client involvement will be very less in this model
-> Client will see the whole project at the last stage

Note: Requirements & Budget is fixed in waterfall model.

-> Waterfall Model is not suitable in this competitive world because now a days as per business demands requirements are keep on changing.

-> To overcome this problems of waterfall model we are using Agile Model

Agile Methodology
+++
-> Agile is a iterative model which encourages development and testing activities parallely

-> Development and Testing will happen through out development life cycle of the project

-> In Agile methodology, Requirements & Budget is not fixed

-> Here we don't deliver entire project in single shot

-> We will divide project into multiple releases

Note: Delivering the project to client is called as Release.

Example

-> Client has given 100 requirements

-> We will divide requirements into multiple releases

RELEASE - 1 (20 REQUIREMENTS WE WILL DELIVER)
RELEASE - 2 (30 REQUIREMENTS WE WILL DELIVER)
RELEASE - 3 (25 REQUIREMENTS WE WILL DELIVER)
RELEASE - 4 (25 REQUIREMENTS WE WILL DELIVER)

Note: For every release, we will do planning, designing, implementation, testing and launch. It is a iterative approach.

Note: For every release client involvement will be there and we will take client

feedback.

Note: Requirements & Budget is not fixed here.

Agile Terminology
+++++++++++++++++
Product Owner : He is responsible to deliver project to client

Scrum Master : He is responsible to manage agile teams (Scheduling meetings, Work priorities)

Tech Lead : Seniour Team Member and responsible for resolving technial issues facing in project

Team Members : Developers and Testers will be available in the Agile team.
Agile team size is 7 to 10 members only.

Story :
+++++++
-> Story means a task
-> Every story contains story points
-> Story points represents duration to complete the task

Backlog
++++++++
-> The stories which are pending to complete are called as Backlog Stories
-> To identify pending works Scrum master will conduct Backlog Grooming meeting
-> All the team members should participate in backlog grooming and create stories in backlog.

Sprint Planning
+++++++++++++++
-> Sprint planning meeting will be conducted by scrum master
-> This is used to prioritize the stories for release which are in backlog

Sprint
+++++++
-> Sprint means the no.of stories that we are going to implement
-> Every Sprint will have 2 weeks duration
-> With in the sprint duration we have to complete stories which are added to sprint

Scrum
+++++
-> Scrum is a daily meeting which will be conducted by scrum master
-> In Scrum Meeting every team member should provide work updates to scrum master
(Which story you have taken, what is the status, any issues you are facing)

Retrospective
++++++++++++
-> Once Sprint is completed scrum master will conduct RetroSpective meeting
-> In this meeting we will discuss about our work in the sprint

- Achievements
- Mistakes
- Lessons Learnt
- Improvements
- New Ideas

Note: Agile project work will be managed using JIRA Software.

-> JIRA software is developed by Atlasian

-> JIRA is providing free tier accounts and commercial accounts also

---

-> Login into jira software

URL : https://www.atlassian.com/software/jira/free

- Enter Your Email & Unique Name for site
- Enter Project Name (JRTP29)
- Click on Change Template and Choose Scrum Template

-> Go to backlog section in JIRA and Create Stories in backlog

-> When we create story then it will generate unique id for every story

-> For every story assign story points

3 Points => 1 day time to complete
5 Points => 2 days time to complete
8 Points => 3 days time to complete

Note: All the team members will have access to create stories in the backlog.

-> Once backlog stories are completed we can do sprint planning

-> Select the stories which are priority then drop them in Sprint.

-> Once stories are finalized for sprint then scrum master will start the sprint by selecting sprint duration.

-> Once sprint is started, we have to work on stories available in the sprint

-> Identify stories which are in pending and assign one story to your name and complete it.
(Keep your name in assignee field and change status to In-Progress)

-> Once story completed move the status from in-progress to done status.

-> If your story is taking more time to complete then you should inform to Your Scrum Master regarding extra time.

Note: Any particular day if you can't join scrum call then you need to send your work status through email to scrum master.

Note: Once sprint started, we shouldn't change story points.

Note: If any story not completed with in sprint then you need to give justification for not completing. Pending stories will be moved to next sprint.

10 days sprint

1 developer should complete atlest 30 story points in sprint

-> Login into JIRA

-> Choose Scrum Template

-> Create Project In JIRA (It will generate a key based on Project Name)

-> Create Stories in backlog (pending items)

-> Assign Story Points For Every Story (Story Points Represents Duration To Complete)

-> Do Sprint Planning (Work Prioritization)

-> Create Sprint with Priority Tasks

-> Identify Pending Tasks in Sprint, Assign it to your name and change status to In-Progress

-> Work on the story and once Story Implementation got completed, update comments in Story and
	Change Story Status to DONE (Completed)

-> Join Daily Scrum Meeting and update your work status to Scrum Master.

-> If you are not able to join scrum call then inform to team members or scrum master.

-> If you are facing some issues in Story Completion you can ask your team members help.

-> If you stuck in the story then inform to SCRUM Master in scrum call. Scrum Master will guide us on how to proceed further.

-> If you have any planned/un-planned leaves then inform to Scrum Master

---

++++++++++++++
Project Lombok
++++++++++++++

-> Project Lombok is used to avoid boiler plate code in our model classes

-> In every model class we will write below code

	1) variables
	2) setter methods
	3) getter methods
	4) no-args constructor
	5) All Args Constructor
	6) equals ( ) method
	7) hashcode( ) method
	8) toString( ) method

-> If we are writing same code in multiple classes then it is called as boiler plate code	(duplicate code / redudant code)

-> By Using project lombok we can avoid boiler plate code in our project

-> Project Lombok provided annotations for generating generic code for model classes

@Setter : It will generate setter methods for all variables in the class

@Getter : It will generate getter methods for all variables in the class

@NoArgsConstructor : It will generate zero param constructor for class

@Equals : Generates equals( ) method

@HashCode : It generates hashCode( ) Method

@ToString : Generates toString ( ) method for our class

@Data : It is equal to all the above annotations

@AllArgsConstructor : It generates Args constructor for our class

---

Working with Project Lombok

++++++++++++++++++++++++++

Step-1) Add Lombok Dependency in Project pom.xml file

```
<dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>1.18.22</version>
        <scope>provided</scope>
</dependency>
```

Step-2) Go to Lombok Jar Location (In Maven Local Repo you can find)

Location : C:\Users\<uname>\.m2\repository\org\projectlombok\lombok\1.18.22

Step-3) Execute Lombok Jar in cmd with below command

cmd > java -jar <jar-file-name> (it will open lombok installer popup)

-> Specify IDE location upto STS.exe/eclipse.exe
-> Click on Install/Update
-> Once installation completed then close installer

Step-4) Re-Start IDE

Step-5) Use Lombok annotations in model classes (@Data)

Step-6) Check lombok working or not by opening outline of our Model class

---

Note: Step-2, Step-3 and Step-4 required only for first time.

Watch Eclipse Shortcuts Video : https://youtu.be/TvYMey5SYa8

Watch Eclipse Code Debugging Video : https://youtu.be/2WxsClYhreE

Project Lombok
Eclipse Shortcuts
Code Debugging in IDE

1st Min project

==========

In last session we discussed about 'Project Lombok'

Eclipse Top 20 shourtcut keys video shared

Java Code Debugging In IDE video shared

---

maven
github
bitbucket
source tree
agile with jira

Mini Projects Development

-> Real-environment culture

-> To improve your analysis skills

-> To understand project development procedure

-> To improve coding skills

-> To improve problem solving skills

Note: In this part, we will develop 3 mini projects. These 3 mini projects are equal to 3 modules in our Major Project.

01 : Crud Operations (fullstack development)

02 : User Management (fullstack development)

03 : Dynamic Search & Reports Generation (fullstack development)

Note: If you do these 3 mini projects then you will get full clarity on projects development
(where to start, how to start, how to code, how to test and how to deploy)

-> For these projects i will do live coding and i want you also to do these projects on your own

-> If you do these 3 projects then defentley your confidence levels will increase to attend interview.

-> The main aim of developing these mini projects to make our students as
Independent Programmers

Note : Assignements Checking will be there for all 3 mini projects. You guys should submit your assignment within given timeline.

Note: If you don't complete your assingnmet, then i will not allow you for the classes.

-> Assignment completion is mandatory in JRTP batches

Next 30 days very crucial for 20-JRTP students
(complete Programming => backend + frontend)

-> For every project one Functional Team will be available

-> Functional Team will collect Requirements from the client

-> Functional Team will prepare BRD

BRD -> Business Requirements Document

Note: BRD contains high level requirements

-> BRD will be submitted to client for review

-> Client will check BRD and will approve BRD (if all requirements are added in BRD)

-> Based on Approved BRD, functional team will prepare FDD

FDD -> Functional Design Document

Note: FDD contains detailed information about requirements

-> FDD will be submitted to client for review

-> Client will review FDD and will provide approval (if client satisfied)

-> Once FDD approved then Functional Team will share FDD to Development & Testing Team

-> Devlopers and Testers should read FDD and should understand Requirements in FDD

-> If Developer/Testers are having any doubts in FDD then we need to discuss with Functional Team regarding doubts and get clarifications

(Daily Meetings will be there to discuss requirements and doubts)

-> Once all your doubts are clarified then you should start your work

---

1) Analyze requirements given in FDD

2) If any doubts, discuss with Functional team and get clarifications

Note: Do reverse KT (explain your understanding to Functional Team)

3) Analyze DB part (No. of tables required and columns required in each table)

4) Analyze classes & methods required to implement

5) Implement Coding

6) Test the functionality

7) Push the code to repository

8) Deploy the code to server

Uploaded 01-MiniProject-FDD in portal. Download that and Read that FDD and ask your doubts in Tomorrow's class.

02-Mini Project Workflow
++++++++++++++++++++++++
1) Read FDD and Understand the requirements

2) If you have any questions in requirements, notedown and ask functional team in daily meeting

3) Design Database Tables (Transactional & Non-Transactional)

4) Prepare INSERT QUERIES TO INSERT Data into static tables (manually we have to insert in DB)

5) Analyze components (classes & methods)

6) Create Project For Backend, and implement code for backend (REST API)

7) Test Backend api using Swagger UI / POSTMAN

8) Deploy backend app in CLOUD (HEROKU / AWS)

DB ANALYSIS
+++++++++++
=> For 02-Mini Project we need total 4 tables

1) USER_DTLS
2) COUNTRY_MASTER
3) STATE_MASTER
4) CITY_MASTER

-> USER_DTLS table is called as transactional table that means application will perform insert/update/delete operations on the table.

-> COUNTRY_MASTER, STATE_MASTER, CITY_MASTER tables are called as non-transactional tables or static tables. Application will only read the data from these tables. We need to insert the data into these tables manually.

USER_DTLS
+++++++++
USER_ID PRIMARY KEY
FIRST_NAME
LAST_NAME
USER_EMAIL
USER_PWD
USER_MOBILE
DOB
GENDER
CITY_ID
STATE_ID

COUNTRY_ID
ACC_STATUS
CREATED_DATE
UPDATED_DATE

COUNTRY_MASTER
++++++++++++++
COUNTRY_ID PRIMARY KEY
COUNTRY_NAME

STATE_MASTER
++++++++++++
STATE_ID PRIMARY KEY
STATE_NAME
COUNTRY_ID

CITY_MASTER
++++++++++
CITY_ID PRIMARY KEY
CITY_NAME
STATE_ID


```
//SQL QUERIES TO INSERT DATA INTO COUNTRY_MASTER
Insert Into COUNTRY_MASTER (COUNTRY_ID, COUNTRY_NAME) Values(1,'India');
Insert Into COUNTRY_MASTER (COUNTRY_ID, COUNTRY_NAME) Values(2,'USA');

//SQL QUERIES TO INSERT DATA INTO STATE_MASTER
Insert Into STATES_MASTER (STATE_ID, COUNTRY_ID, STATE_NAME) Values(1,1,'Andhra
Pradesh');
Insert Into STATES_MASTER (STATE_ID, COUNTRY_ID, STATE_NAME)
Values(2,1,'Karnataka');
Insert Into STATES_MASTER (STATE_ID, COUNTRY_ID, STATE_NAME) Values(3,2,'New
Jersy');
insert into STATES_MASTER (STATE_ID, COUNTRY_ID, STATE_NAME) values(4,2,'Ohio');

//SQL QUERIES TO INSERT DATA INTO CITY_MASTER
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(1,'Vizag',1);
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(2,'Guntur',1);
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(3,'Banglore',2);
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(4,'Mysore',2);
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(5,'Maywood',3);
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(6,'Westwood',3);
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(7,'Oakwood',4);
Insert Into CITIES_MASTER (CITY_ID, CITY_NAME, STATE_ID) Values(8,'Cuyahoga
County',4);
```

---

1) Create a project with below dependencies

   a) starter-data-jpa
   b) h2
   c) web-starter
   d) mail-starter
   e) project-lombok
   f) devtools
   g) swaager & swagger-ui

2) Configure properties in application.properties file or application.yml file

   a) datasource properties
   b) ORM properties (ddl_auto and show_sql)
   c) SMTP properties (For email sending)

d) application messages

Note: We have to enable less-secure-apps for the email which we configure in SMTP Properties
(URL : https://myaccount.google.com/lesssecureapps)

-> Switch ON "Less Secure Apps"

3) Create required packages & classes in our project

in.ashokit (base package) (boot start class will be available in this)

in.ashokit.entity
- CountryMasterEntity.java
- StateMasterEntity.java
- CityMasterEntity.java
- UserDtlsEntity.java

in.ashokit.repository
- CountryMasterRepo.java (I)
- StateMasterRepo.java (I)
- CityMasterRepo.java (I)
- UserDtlsRepo.java (I)

in.ashokit.bindings
- LoginForm.java
- UserRegForm.java
- UnlockAccForm.java

in.ashokit.service
- UserMgmtService.java (I)
- UserMgmtServiceImpl.java

in.ashokit.rest
- LoginRestController.java
- RegistrationRestController.java
- UnlockAccRestController.java
- ForgotPwdRestController.java

in.ashokit.util
- EmailUtils.java

in.ashokit.constants
- AppConstants.java

in.ashokit.props
- AppProperties.java

in.ashokit.config
- SwaggerConfig.java

03-Mini Project
++++++++++++++++

-> Develop an application to perform dynamic search for citizen insurance plans data and export that data into Excel and Pdf files

ELIGIBILITY_DTLS (DB TBL)

PLAN_ID (PK)

CASE_NUM
PLAN_NAME
PLAN_STATUS
HOLDER_NAME
HOLDER_SSN
BENEFIT_AMT
START_DATE
END_DATE
DENIAL_REASN

NOTE: INSERT 20 RECORDS INTO DB TABLE

Plan names : SNAP, CCAP, Medicaid, Medicare and QHP

Plan Status : Approved and Denied

(Inser 4 records for every plan name => 2 records with Approved status + 2 records with Denied status)

Note: Plans which are approved will contains benefit_amt, start_date and end_date (denial_reason will be null)

Note: Plans which are denied will contain denial_reason (start_date, end_date and benefit_amt will be null)

Functionality

Frontend Screen should contain below fields

Plan Name (dropdown)
Plan Status (dropdown)
Start Date (date picker)
End Date (date picker)
Search button
Excel icon
Pdf icon

-> Plan Name dropdown should display unique plan names available in the db table (ELIGIBILITY_DTLS)

-> Plan Status dropdown should display unique plan statuses available in the db table (ELIGIBILITY_DTLS)

-> Plan Start Date field should display date picker to select the date

-> Plan End Date field should display date picker to select the date

Note: All the fields are optional. If plan start date is selected then plan end date is mandatory to select.

Note: User can search records by selecting drop downs values and can search without selecting dropdown values (This is dynamic search)

-> When user click on 'Search' button without selecting dropdowns then retrieve all records from DB table and display

-> When user select 'Plan Name' in dropdown and click on 'Search' button then retrieve only Selected Plan Name records from DB table and display.

-> When user select 'Plan Status' in dropdown and click on 'Search' button then retrieve records based on Selected Plan Status from DB table and display.

-> When user select both 'Plan Name' and 'Plan Status' and click on 'Search' button then retrieve records based on selected 'Plan Name' and 'Plan Status' from DB table and display

-> When use select Start Date and ENd Date we need to apply those dates in the condition to retrieve the data

-> When user click on 'Excel' icon then export total table data into Excel file

-> When use click on 'Pdf' icon then export total table data into PDF file

---

1) Create spring starter application with below dependencies

a) web-starter
b) data-jpa-starter
c) h2
d) apache poi
e) openpdf
f) project lombok
g) swaggerfox
h) swagger-ui

2) Configure application properties in application.properties file

    a) data-source
    b) server port (if required)

3) Create Request & Response Binding Classes

4) Create Entity class & Repository Interface

5) Create Service interface with Implement class

6) Create Report Generator Classes (Excel Report Generator & Pdf Report Generator)

7) Create RestController with Required methods

8) Create SwaggerConfig class for Documentation

9) Insert static data into DB table to test our application (data.sql)

10) Run the application and test it using swagger-ui

11) Deploy the application in cloud

12) Develop Front End Application using Angular

```
<dependency>
        <groupId>org.apache.poi</groupId>
        <artifactId>poi-ooxml</artifactId>
        <version>4.1.0</version>
</dependency>

<dependency>
        <groupId>com.github.librepdf</groupId>
        <artifactId>openpdf</artifactId>
        <version>1.3.8</version>
</dependency>
```

```
		<dependency>
			<groupId>io.springfox</groupId>
			<artifactId>springfox-swagger2</artifactId>
			<version>2.8.0</version>
		</dependency>
		<dependency>
			<groupId>io.springfox</groupId>
			<artifactId>springfox-swagger-ui</artifactId>
			<version>2.8.0</version>
		</dependency>

--------------------------------------------------------------------------------
------------------------------------------------------------

Monolith vs Microservices
++++++++++++++++++++++++

-> We can develop one application in 2 ways

			1) Monolith Architecture
			2) Microservices Architecture

-> Monolith Architecture means all the functionalities will be developed as single
aplication

Monolith Advantages
-------------------
1) Easy to develop
2) Total source code at one place
3) Easily We understand project functionality
4) Easy to test

Monolith Dis-Advantages
-----------------------
1) Maintence is very difficult
2) If any change in code, whole project should be re-deployed
3) Single Point Of Failure

=> To overcome the problems of Monolith Architecture people are using Microservices
Architecture

-> Microservices is not a programming language
-> Microservices is not a framework
-> Microservices is not an api
-> Microservices is an "Architectural Design Pattern"
-> Using Microservices Architecture pattern we can develop application with loosely
coupling
-> In Microservices architecture we will develop several REST APIs


Advantages of Microservices
---------------------------
1) Funcationlity dividing into multiple apis
2) Faster Development
3) Easy Maintenence
4) Quick Releases
5) Highly Scalable
6) Bug Fixing Easy
7) Technology independence

Challenges with Microservices
-----------------------------
1) Bounded Context (Deciding no.of apis required and deciding api boundary)

2) Lot of configurations
```

Ex: DataSource, SMTP, Kafka, Redis, Actuators etc...

3) Visibility (We may not get change to work with all apis)

Microservices Architecture

-> There is no fixed architecture for Microservices
-> People are customizing Microservices Architecture based on their requirements

1) Backend apis
2) service registry
3) api gateway

Guys,

Tomorrow we will implement a POC on Microservices with

1) Service Registry
2) Inter Service Communication
3) Clod Gateway
4) Load Balancing

I am planning long session to complete our POC

Note : Class Timing 7:45 AM To 11 AM (This timing for next 2 to 3 days)

Note: Join class from portal @7:45 AM IST

Don't miss live class, bcz it is very important

---Microservices
++++++++++++

What is Monolith Architecture

-> If we develop all the functionalities in single project then it is called as Monolith architecture based application

-> We will package our application as a jar/war to deploy into server

-> As monolith application contains all functionalities, it will become fat jar/war

Advantages
---
1) Simple to develop

2) Everything is available at once place

3) Configuration required only once

Dis-Advantages
---
1) Difficult to maintain

2) Dependencies among the functionalites

3) Single Point Of Failure

4) Entire Project Deployment

****** To overcome the problems of Monolith, Microservices architecture came into market******

-> Microservices is not a programming language

-> Microservices is not a framework

-> Microservices is not an Specification API

-> Microservices is an architectural design pattern

-> Microservices suggesting to develop application functionalities with loosely coupling

-> In Microservices architecture we don't develop all the functionalities in single project. We will divide project functionalities into several REST APIs

**********Note: One REST API is called as one Microservice**********

-> Microservices architecture based project means collection of REST APIs.

-> Microservices is not related to only java. Any programming language specific project can use Microservices Architecture.

Advantages
----

1) Loosely Coupling

2) Easy To maintain

3) Faster Development

4) Quick Deployment

5) Faster Releases

6) Less Downtime

7) Technology Independence

Dis-Advantages
----

1) Bounded Context

2) Lot of configurations

3) Visibility

4) Pack of cards

Microservices Architecture

-> We don't have any fixed architecture for Microservices

-> People are customizing microservices architecture according to their requirement

-> Most of the projects will use below components in Microservices Architecture

1) Service Registry (Eureka Server)

2) Services (REST APIs)

3) Interservice Communication (FeginClient)

4) API Gateway (Zuul Proxy)

Service Registry
++++++++++++++++
-> Service Registry acts as DB of services available in the project
-> It provides the details of all the services which are registered with Service Registry
-> We can identify how many services available in the project
-> We can identify how many instances available for each service
-> We can use "Eureka Server" as service registry
-> Eureka Server provided by "Spring Cloud Netflix" library

Services
+++++++++
-> Services means REST APIs / Microservices
-> Services contains backend business logic
-> In the project, some services will interact with DB
-> In the project, some services will interact with third party REST API (external communication)
-> In the project, some services will interact with another services with in the project
   (inter-service communication)
-> For inter-service communication we will use feign-client
-> To distribute the load, we can run one service with Multiple Instances (Load Balancing)

Note: We will register every service with Service Registry

API Gateway
++++++++++++
-> API Gateway is used to manage our backend apis of the project
-> API Gateway acts as mediator between end users and backend apis
-> API Gateway can filter logic to decide request processing
-> API Gateway will contain Routing logic (which request should go to which REST API)
-> API Gateway also will be registered with Service Registry

==========
Mini Project Implementation using Microservices Architecture
++++++++++++==++

1) Service Registry (Eureka Server)

2) Spring Boot Admin Server (To monitor & manage boot applications)

3) Zipkin Server (Distributed Log Tracing)
(https://zipkin.io/pages/quickstart.html)

Steps to develop Service Registry Application (Eureka Server)
++
1) Create Service Registry application with below dependency

a) EurekaServer (spring-cloud-starter-netflix-eureka-server)
b) web-starter
c) devtools

2) Configure @EnableEurekaServer annotation in boot start class

3) Configure below properties in application.yml file

```
server:
  port: 8761

eureka:
  client:
    register-with-eureka: false
```

Note: If Service-Registry project port is 8761 then clients can discover service-registry and will register automatically with service-registry. If service-registry project running on any other port number then we have to register clients with service-registry manually.

4) Once application started we can access Eureka Dashboard using below URL

URL : http://localhost:8761/

Steps to develop Spring Boot Admin Server Project
++

1) Create Boot application with below dependencies

a) web-starter
b) devtools
c) admin-server (codecentric)

2) Configure @EnableAdminServer annotation at boot start class

3) Configure the port number and run the application (port : 8080)

4) After application started, access Admin Server UI using app-url

URL : http://localhost:8080/

Steps to work with Zipkin Server
++++++++++++++++++++++++++++++++

1) Download Zipkin server jar from website

URL : https://zipkin.io/pages/quickstart.html

2) Run the zipkin server jar from command prompot

Cmd : java -jar <jar-file-name>

Note: Zipkin server will run on 9411 port number

3) Access Zipkin server dashboard in browser

URL : http://localhost:9411/

Steps to develop GREET-API
++
1) Create Spring Boot application with below dependencies

- eureka-discovery-client
- starter-web
- devtools
- actuator
- sleuth
- zipkin
- admin-client

2) Configure @EnableDiscoveryClient annotation at start class

3) Create RestController with required method

4) Configure below properties in application.yml file

----------application.yml----------

```
server:
  port: 9090
spring:
  application:
    name: GREET-API
  boot:
    admin:
      client:
        url: http://localhost:8080/
eureka:
  client:
    serviceUrl:
      defaultZone: http://localhost:8761/eureka
management:
  endpoints:
    web:
      exposure:
        include: '*'
```

---

5) Run the application and check in Eureka Dashboard (It should display in eureka dashboard)

6) Check Admin Server Dashboard (It should display) (we can access application details from here)

Ex: Beans, loggers, heap dump, thred dump, metrics, mappings etc...

7) Send Request to REST API method

8) Check Zipkin Server UI and click on Run Query button
(it will display trace-id with details)

Steps To Develop WELCOME-API
++
1) Create Spring Boot application with below dependencies

- web-starter
- devtools
- eureka-discovery-client
- fegin-client
- admin-client
- zipkin-client
- sleuth
- actuator

2) Configure @EnableDiscoveryClient & @EnableFeignClients annotations at boot start class

3) Create FeignClient to access GREET-API

```
@FeignClient(name = "GREET-API")
public interface GreetApiClient {

	@GetMapping("/greet")
	public String invokeGreetApi();

}
```

4) Create RestController with required method

Note: In Rest Controller we should have logic to access another REST API (GREET-API)

-> For Interservice Communication we will use FeignClient

-> Using FeginClient we can make rest call to another service using name of the service (no need of url)

-> FeginClient will get service URL from service-registry based on service-name

```
@RestController
public class WelcomeRestController {

	private Logger logger = LoggerFactory.getLogger(WelcomeRestController.class);

	@Autowired
	private GreetApiClient greetClient;

	@GetMapping("/welcome")
	public String welcomeMsg() {

		logger.info("welcomeMsg() execution - start");

		String welcomeMsg = "Welcome to Ashok IT..!!";

		String greetMsg = greetClient.invokeGreetApi();

		logger.info("welcomeMsg() execution - end ");

		return greetMsg + ", " + welcomeMsg;
	}
}
```

5) Configure below properties in application.yml file

```
server:
  port: 9091
spring:
  application:
    name: WELCOME-API
  boot:
    admin:
      client:
        url: http://localhost:8080/
eureka:
  client:
    serviceUrl:
      defaultZone: http://localhost:8761/eureka
management:
  endpoints:
    web:
      exposure:
        include: '*'
```

6) Run WELCOME-API project (it should register in Eureka and Admin server)

7) Send Request to welcome-api (it should final response)

8) Verify Zipkin Server Dashboard for log tracing

-> We are running Service Registry project with Eureka Server on 8761 port number

-> Eureka Discovery Client applications are auto-registering with Eureka Server when port is 8761

-> If we change Eureka Server port number then we have to register Eureka Client application with Eureka Server using below property in application.yml file

```
eureka:
   client:
      serviceUrl:
         defaultZone: http://localhost:9090/eureka
```

Note: We should configure this property in eureka client application yml file

---

GREET API URL : DESKTOP-BDG00U7:GREET-API:9090/

WELCOME API URL : DESKTOP-BDG00U7:WELCOME-API:9091/

API Gateway

-> API Gateway will act as mediator between client requests & backend apis

-> API Gateway will provide single entrypoint to access our backend apis

-> In Api Gateway we will write mainley below 2 types of logics

1) Filters

2) Routing

-> Filters are used to execute some logic before request processing and after request processing

-> Routing is used to tell which request should go to which REST API

-> In Spring Cloud, we have 2 options to create API Gateway

1) Zuul Proxy (old approach)
2) Spring Cloud Gateway (latest approach)

Note: Zuul Proxy is not supported by latest versions of spring boot

---

Working with Spring Cloud API Gateway

---

1) Create Spring boot application with below dependencies

-> web-stater
-> eureka-client
-> cloud-gateway
-> devtools

2) Configure @EnableDiscoveryClient annotation at boot start class

3) Configure API Gateway Routings in application.yml file like below

----------application.yml file----------

```
spring:
  cloud:
    gateway:
      discovery.locator:
        enabled: true
        lowerCaseServiceId: true
      routes:
      - id: welcome-api
        uri: lb://WELCOME-API
        predicates:
        - Path=/welcome
      - id: greet-api
        uri: lb://GREET-API
        predicates:
        - Path=/greet
  application:
    name: CLOUD-API-GATEWAY
server:
  port: 3333
```

In API gateway we will have 3 types of logics

1) Route

2) Predicate

3) Filters

-> Routing is used to defined which request should be processed by which REST API in backend. Routes will be configured using Predicate

-> Predicate : This is a Java 8 Function Predicate. The input type is a Spring Framework ServerWebExchange. This lets you match on anything from the HTTP request, such as headers or parameters.

-> Filters are used to manipulate incoming request and outgoing response of our application

Note: Using Filters we can implement security also for our application.

```
@Component
public class MyPreFilter implements GlobalFilter {

	private Logger logger = LoggerFactory.getLogger(MyPreFilter.class);

	@Override
	public Mono<Void> filter(ServerWebExchange exchange, GatewayFilterChain chain) {

		logger.info("MyPreFilter :: filter () method executed...");

		// Accessing HTTP Request information
		ServerHttpRequest request = exchange.getRequest();

		HttpHeaders headers = request.getHeaders();
		Set<String> keySet = headers.keySet();

		keySet.forEach(key -> {
			List<String> values = headers.get(key);
			System.out.println(key +" :: "+values);
		});

		return chain.filter(exchange);
	}
}
```

-> We can validate client given token in the request using Filter for security purpose

-> We can write request and response tracking logic in Filter

-> Filters are used to manipulate request & response of our application

-> Any cross-cutting logics like security, logging, moniroing can be implemented using Filters

---

Sleuth & Zipkin

-> Microservices application means several REST APIs will be available

-> As part of application execution one Rest API can communicate another REST API

-> When we send request from UI, it will process by Multiple REST APIs with Interservice communication

*** How we can understand which rest api is taking more time to process request ? ***

-> If we add Sleuth dependency in REST API then it will add span-id and trace-id for log messages

-> For every request once span-id will be generated by Sleuth

-> If one request is processing multiple REST API then Sleuth will use same span-id for REST APIs to generate log message

-> Trace-id is specific to one REST API

-> By using span-id and trace-id we can understand which REST api has taken more time process request

-> To monitor span-id and trace-id details we will use ZipKin server

-> Zipkin server is providing user interface (UI) to monitor all the details

Note: The REST APIs which are having sleuth dependency should register with Zipkin server

Note: By using Sleuth and Zipkin we achieve Distributed Log Tracing

Steps to work with Sleuth and Zipkin
++++++++++++++++++++++++++++++++++

1) create spring-boot application with below dependencies

a) web-starter
b) sleuth
c) zipkin
d) devtools

2) Create a REST Controller with required methods

3) Download zipkin-server jar file (https://zipkin.io/pages/quickstart)

4) Run zipkin-server using "java -jar <zipkin-jar-filename"

Note: Zipkin server runs on 9411 port

5) Run spring boot application and send a request to rest controller method

6) Verify boot application logs display in console (span-id and trace-id will be attached to logs)

7) Go to Zipkin server dashboard and monitor event details

(URL : http://localhost:9411)

Code Review
++++++++++++

-> In project development multiple team members will be involved

-> Some developers will be freshers, some are junior developers & some are senior developers

-> As a developer me might do some mistakes in coding or we may not follow proper coding standards

-> We will perform Code Review to identify the mistakes of the developers in coding

-> With the help of code review we can provide quality code and bug free code

-> To perform code review we will use Sonar Qube software

Sonar Qube

-> Sonar Qube is an automatic code review tool

-> Sonar Qube supports for 29 programming languages

-> Sonar Qube will identify

1) Bugs
2) Vulnerabilities
3) Code Smells
4) Duplicate Code Blocks

Note: SonarQube will not check our logic is correct not

```
public void findNonRepeatedCharInString(String str){

    //logic to find
}
```

Note: Juniour developers code checking will done by seniour developers in the team. This is called as Peer Review. Peer Review is a manual process. In peer review, logic checking will be done.

Installing Sonar Software in Local

-> Download Sonar Software from below url

URL : https://www.sonarqube.org/downloads/
Version : 6.3.1 (historical version download)

-> Start Sonar Server by executing StartSonar.bat

Location : Sonar-Folder/bin/windows64/StartSonar.bat

-> Once Sonar Server is started it will display Sonar up and running message in console.

-> By Default Sonar Server will run on 9000 port number (We can customize port number)

Location : sonar-folder/conf/sonar.properties file

-> Open Sonar Server Dashboard using below URL

URL : http://localhost:9000/

Running Project with SonarQube Server

-> Add below 1 plugin in project pom.xml file (in <build> tag)

```
<plugin>
   <groupId>org.sonarsource.scanner.maven</groupId>
   <artifactId>sonar-maven-plugin</artifactId>
   <version>3.4.0.905</version>
</plugin>
```

-> Do Maven build of project with package goal

```
mvn clean package
```

-> For project do maven build with below goal To Do Code Review

```
mvn sonar:sonar
```

-> After maven build completed, check sonar server dashboard.

Lessons Learnt in Code Review
---

1) String which we want to compare should present at left side

```
if (userAcc.getAccStatus().equals("LOCKED"))  // bad-practise

if ("LOCKED".equals(userAcc.getAccStatus()))  // good practise
```

2) Replace StringBuffer with StringBuilder to improvate performance

StringBuffer -> is synchronized (only one thread can access at a time)

StringBuilder -> is not-syncronized (multiple threads can access at a time)

3) Either log or re-throw the exception

```
//bad practise
catch (Exception e) {
        e.printStackTrace( );
}

//good practise
catch (Exception e) {
        logger.error("Exception :: "+e.getMessage(), e);
}
```

4) Instead of Math.random( ) use java.util.Random.nextInt ( ) method to generate random number

6) Don't use "password" or "pwd" in our code directley. It will be considered as vulnerable. Use "pazzword" or "pzzwd" for variable names.

7) Declare variables before constructor

8) Remove un-necessary curly braces from lambda when we have single line

9) follow camel case for variables names declaration

10) Declare private constructor for class if it is not getting instantiated anywhere.

Software Industry
++++++++++++++++++
What is Project
Types of projects
Types of companies
Interview Process in companies
Role Chart
Bridge Calls

Realtime Tools
+++++++++++++++
Maven
Git Hub
BitBucket
Source Tree
JIRA
Lombok
Sonar Qube
JMETER
Swagger
Redis
Heroku Cloud

Angular
+++++++
What is Angular
Angular Advantages
Angular JS vs Angular
Angulr Architecture
Angular Env Setup
Type Script
Module in TS
Variables
Data Types
Loops
Functions
Components
Templates
Directives
Pipes
Services
DI
Forms & validations
Routes
Ajax
Boot Integration
Mini Project Development

Spring Boot - Mini Projects
++++++++++++++++++++++++++
Curd Operations App
User Management (Register, Login, Unlock-Account, Forgot Pwd)

Dynamic Search
Reports Generation

Microservices
++++++++++++++
What is Monolith Architecture
Pros and Cons of Monolith
Microservices Introduction
Pros and Cons with Microservices
Microservices Architecture
Service Registry (Eureka Server)
Inter Service Communication (FeignClient)
Api Gateway (Cloud Gateway)
Filters in Gateway
Routes in Gateway
Load Balancing
Sleuth & Zipkin
Admin Server & Client
Mini Project Implementatioin using Microservices Architecture

Reference Videos
+++++++++++++++++
â™¨ï¸ Spring Boot Actuators : https://youtu.be/lMDZ6dple9Q

ðŸ‘‰ Spring Boot Profiles : https://youtu.be/I8r858beW0o

â™¨ï¸Spring Boot with Security - By Mr. Ashokâ™¨ï¸
Part-1 : https://youtu.be/p-nxpoHXhp8
Part-2 : https://youtu.be/vNZ5H7zm3us
Part-3 (JWT) : https://youtu.be/J-g0FYPHY-k

â™¨ï¸Spring Boot with Mongo DB - By Mr. Ashokâ™¨ï¸
Part-01 : https://youtu.be/3qz8dRABscs
Part-02 : https://youtu.be/JzAPqmnaKco
Part-03 : https://youtu.be/kZOpf7kDiU8

â™¨ï¸JUnit-5 with Mocking - By Mr. Ashokâ™¨ï¸
https://youtu.be/MEFoGR07qgw

ðŸ‘‰ Docker : https://www.youtube.com/watch?v=VO818de8sdk

ðŸ‘‰ Apache Kafka : https://youtu.be/VInk1_9vvCY

ðŸ‘‰ Spring Boot + AWS : https://youtu.be/PkZzHtSFMa0

ðŸ‘‰ Spring Cloud Config Server : https://youtu.be/NJh70xz_CXY

Git Hub Repository
++++++++++++++++++
ðŸ‘‰ https://github.com/ashokitschool/ashokit_weekend_workshops
---

PLAN_CATEGORY_MASTER (STATIC)

CATEGORY_ID   INTEGER PRIMARY KEY
CATEGORY_NAME   VARCHAR
CREATED_DATE   DATE
UPDATED_DATE   DATE
CREATED_BY   VARCHAR
UPDATED_BY   VARCHAR

APP_PLANS (DYNAMIC)

PLAN_ID	INTEGER	PRIMARY KEY
PLAN_NAME	VARCHAR	
PLAN_START_DATE	DATE	
PLAN_END_DATE	DATE	
CATEGORY_ID	NUMBER	
ACTIVE_SW	CHAR	
CREATED_DATE	DATE	
UPDATED_DATE	DATE	
CREATED_BY	VARCHAR	
UPDATED_BY	VARCHAR	

CW_ACCOUNTS

ACC_ID	INTEGER	PRIMARY KEY
FULLNAME	VARCHAR	
EMAIL	VARCHAR	
PWD	VARCHAR	
MOBILE_NUM	INTEGER	
GENDER	CHAR	
DOB	DATE	
SSN	INTEGER	
ACTIVIE_SW	CHAR	
CREATED_DATE	DATE	
UPDATED_DATE	DATE	
CREATED_BY	VARCHAR	
UPDATED_BY	VARCHAR	

CITIZEN_APPS

CASE_NUM	INTEGER	PRIMARY KEY
FULLNAME	VARCHAR	
EMAIL	VARCHAR	
MOBILE_NUM	INTEGER	
GENDER	CHAR	
DOB	DATE	
SSN	INTEGER	UNIQUE
STATE_NAME	VARCHAR	
ACTIVIE_SW	CHAR	
CREATED_DATE	DATE	
UPDATED_DATE	DATE	
CREATED_BY	VARCHAR	
UPDATED_BY	VARCHAR	

CITIZEN_PLANS

CITIZEN_ID	INTEGER	PRIMARY KEY
CASE_NUM	INTEGER	
PLAN_ID	INTEGER	
CREATED_DATE	DATE	
UPDATED_DATE	DATE	
CREATED_BY	VARCHAR	
UPDATED_BY	VARCHAR	

CITIZEN_INCOME_DTLS

INCOME_ID	INTEGER	PRIMARY KEY
CASE_NUM	INTEGER	
SALARY_INCOME	INTEGER	
RENT_INCOME	INTEGER	

```
PROPERTY_INCOME INTEGER
CREATED_DATE            DATE
UPDATED_DATE            DATE
CREATED_BY              VARCHAR
UPDATED_BY              VARCHAR


GRADUATION_YEARS
----------------
YEAR_ID         INTEGER PRIMARY KEY
YEAR                    INTEGER
CREATED_DATE    DATE
UPDATED_DATE    DATE
CREATED_BY      VARCHAR
UPDATED_BY      VARCHAR


CITIZEN_GRADUATION_DTLS
-----------------------
GRADUATION_ID           INTEGER PRIMARY KEY
CASE_NUM                                INTEGER
HIGHEST_DEGREE          VARCHAR
GRADUATION_YEAR_ID      INTEGER
UNIVERSITY                      VARCHAR
CREATED_DATE                    DATE
UPDATED_DATE                    DATE
CREATED_BY                      VARCHAR
UPDATED_BY                      VARCHAR


CITIZEN_CHILD_DTLS
------------------
CHILD_ID                INTEGER PRIMARY KEY
CASE_NUM                INTEGER
CHILD_NAME      VARCHAR
CHILD_DOB       DATE
CHILD_SSN       INTEGER
CREATED_DATE    DATE
UPDATED_DATE    DATE
CREATED_BY      VARCHAR
UPDATED_BY      VARCHAR


ELIG_DTLS
----------
ELIG_ID                 INTEGER PRIMARY KEY
CASE_NUM                        INTEGER
PLAN_NAME               VARCHAR
PLAN_STATUS             VARCHAR
START_DATE              DATE
END_DATE                        DATE
BENEFIT_AMT             VARCHAR
DENIAL_REASON           VARCHAR
CREATED_DATE            DATE
UPDATED_DATE            DATE
CREATED_BY              VARCHAR
UPDATED_BY              VARCHAR


CO_TRIGGERS
------------
TRG_ID          INTEGER  PRIMARY KEY
CASE_NUM                INTEGER
TRG_STATUS      CHAR (P - pending, C - Completed) - Default value : P
NOTICE          BLOB
```

MYSQL DB Setup In Local
++++++++++++++++++++++++

1) Install MySQL Server in your PC

2) Install MySQL Workbench in your PC

MySQL Server : DB Server

MySQL Work Bench : DB Client

MySQL Workbench ------------> MySQL DB Server
SQL

3) Open MySQL Workbench and Create DB Connection

4) Create Database in MySQL DB

Query : create database ihis

Note: ihis is the name of database

5) Use Database

Query : use ihis

6) Check tables available in the Database

Query : show tables

++
Create DB Manager API
++
1) Create Spring Starter Project with below dependencies

a) data-jpa
b) mysql-connector
c) lombok

2) Configure Datasource properties in application.properties file

3) Create Entity classes & Repository interfaces for all the tables

4) Execute Maven Goals (clean & package)

5) Install DB Manager as dependency in maven local repo using 'install goal'

Note: Very db-manager dependency in maven local repo

(c://users/uname/.m2/repository)

6) Copy DB-Manager dependency details

Example

```
<groupId>com.ashokit</groupId>
<artifactId>IHIS-DB-Manager</artifactId>
```

```
        <version>1.0.RELEASE</version>
```

Note: If we make any changes in DB-Manager then we have to execute 'clean install' goals again.

Admin API Development

---

1) Create Spring starter project with below dependencies

   a) web-starter
   b) devtools
   c) lombok
   d) mail-starter
   e) actuator
   f) db-manager
   g) swagger & swagger-ui

2) Develop functionality for Case Worker Account Management

3) Develop Functionality for Plans Management

Correspondence (CO)

++++++++++++++++++

-> Correspondence is used to generate notices to citizens

-> When citizen eligibility determined, ED module inserting one record into CO_TRIGGERS table with TRG_STATUS as pending

-> CO api should retrieve all pending triggers and should generate notice based on eligiblity details and send that notice to citizen through email

-> After sending Notice to citizen, store that notice in DB table and update TRG_STATUS as Completed

CO_API BUSINESS LOGIC

// Read all pending triggers from CO_TRIGGERS TABLE

// Get Eligibility data for each trigger using Case Number from ELIGIBILITY_DTLS table

// Generate PDF for trigger with Eligibility Data

// Send Generated PDF to citizen email as an attachment

// Store Generated PDF into CO_TRIGGERS table

// Once PDF sent to email and stored in DB then update trigger status as Complted

// Return api execution summary (total triggers, success count and failure count)

1) Create Spring Boot application with below dependencies

        -> web-starter
        -> starter-data-jpa
        -> h2 db
        -> starter-mail
        -> open-pdf
    -> swagger & swagger-ui
        -> devtools

2) Configure properties in application.yml file

         -> server-port
         -> datasource properties
         -> ORM properties
         -> SMTP properties

3) Create Entities & Repositories for DB communication

Note: CO-API will communicate with below 3 tables

CO_TRIGGERS ==> To get all pending triggers

ELIGIBILITY_DTLS ==> To get citizen eligiblity based on case_number

APP_REG ===> To get citizen email id based on case_number

4) Create Service Interface with implementation

```
public interface CoService {

        public Map<String, Integer> generateNotices();

}
```

5) Create Rest Controller to handle request and response

6) Create Swagger Configuration for documention

7) Run the application and test it

Eligibility Determination

-> After Data Collection completed, then Eligibility Determination module execution will start

-> ED module is responsible to determine citizen is eligble for the applied plan or not based on plan conditions and citizen data

-> After Eligiblity Determination Completed, ED module will insert eligibility data into ED_ELIG_DTLS table and it will insert a record into CO_TRIGGERS table to send notice to citizen

ED-REST-API
++++++++++
Input : case_number

Output: Plan_Info
(case_num, holder_name, holder_ssn, plan_name, plan_status, start_date, end_date, benefit_amt, denial_reason)

Note: If citizen approved for the plan then start_date, end_date and benefit_amt will be available (denial_reason will not be available)

Note: If citizen denied for the plan then denial_reason will be available (start_date, end_date and benefit_amt will not be available)

SNAP Condition : If employment_income <=300$ then citizen is eligible for SNAP

CCAP Condition : If emplyment_income <=300$ and kids count > 0 and each kid age <=16 then eligible for CCAP

Medicaid : If employment_income <=300$ and Property Income is 0 then eligible for Medicaid

Medicare : If citizen age is >=65 then eligible for Medicare

NJW : If citizen is un-employed and graduated then eligible for NJW

-> After Eligiblity Determination Completed then insert data into below two tables

ED_ELIG_DTLS

ED_TRACE_ID PK
CASE_NUM
HOLDER_NAME
HOLDER_SSN
PLAN_NAME
PLAN_STATUS
PLAN_START_DATE
PLAN_END_DATE
BENEFIT_AMT
DENIAL_REASON

CO_TRIGGERS

CO_TRG_ID PK
CASE_NUM
CO_PDF (blob) (deafult : null)
TRG_STATUS (Default : Pending)

```
DROP PROCEDURE IF EXISTS insertRowsToCoTriggers;
DELIMITER //
CREATE PROCEDURE insertRowsToCoTriggers()
BEGIN
DECLARE i INT DEFAULT 1;
WHILE (i <= 5000) DO
    INSERT INTO `CO_TRIGGERS` (TRG_ID, CASE_NUM, TRG_STATUS) values (i, i,
'Pending');
    SET i = i+1;
END WHILE;
END;
//
DELIMITER ;

call insertRowsToCoTriggers();
```

```
++++++++++++++++++++++++++++++++++++++++++++++++++

DROP PROCEDURE IF EXISTS insertRowsToCitizenApps;
DELIMITER //
CREATE PROCEDURE insertRowsToCitizenApps()
BEGIN
DECLARE i INT DEFAULT 1;
DECLARE j INT DEFAULT 1;
WHILE (i <= 5000) DO
    INSERT INTO `CITIZEN_APPS` (EMAIL,FULLNAME,SSN) values
('ashokitschool@gmail.com','John',123456+j);
    SET i = i+1;
        SET j = j+1;
END WHILE;
END;
//
DELIMITER ;

call insertRowsToCitizenApps();

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

DROP PROCEDURE IF EXISTS insertRowsToEligDtls;
DELIMITER //
CREATE PROCEDURE insertRowsToEligDtls()
BEGIN
DECLARE i INT DEFAULT 1;
WHILE (i <= 5000) DO
    INSERT INTO `ELIG_DTLS` (CASE_NUM, PLAN_NAME,PLAN_STATUS,DENIAL_REASON) values
(i, 'SNAP','Denied','High Income');
    SET i = i+1;
END WHILE;
END;
//
DELIMITER ;

call insertRowsToEligDtls();

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
------------------------------------------------------------------------------
-------------------------------------------------------------

=================
AWS-ACCOUNT
=================

What is Cloud Computing?
++++++++++++++++++++

-> The process of delivering IT resources on demand basis is called as Cloud
Computing

                        a) Machines
                        b) Database
                        c) Storage
                        d) Network
                        e) Computing etc....

-> Cloud providers are providing IT infrastructure over the web

-> Cloud Providers giving services based on "Pay As You Go" model. We need to pay
the money for using Cloud Services

                Ex: Post paid bill, Credit Card Bill etc.
```

-> There are several cloud providers available in the market

a) Amazon Webservices (AWS)
b) Microsoft Azure
c) Google Cloud (GCP)
d) Salesforce Cloud
e) Alibaba Cloud
f) IBM cloud etc....

-> Cloud Services are divided into 3 types

1) IAAS : Infrastructure as a service

2) PAAS : Platform as a service

3) SAAS : Software as a service

-> IAAS means cloud provider will give infrastructure then we have to setup that infrastructure to run our application

-> PAAS means cloud provider will give platform run our application

-> SAAS means cloud provider will give their application to use

Ex: google drive, dropbox, zoom, Salesforce etc...

What is AWS?
++++++++++++

-> AWS stands for Amazon Webservices

-> AWS is one of the leading cloud platform available in the market

-> Amazon company managing this AWS cloud

-> AWS cloud providing both IaaS & PaaS

-> AWS cloud services are using in 190+ countries

-> AWS providing 200+ services

AWS Services Overview
+++++++++++++++++++++

1) EC2 : Elastic Compute Cloud (It is used to create virtual machines)

2) EBS : Elastic Block Store (It is used to add additional memory to our virtual machines)

3) ELB : Elastic Load Balancer (It is used to distribute load to multiple servers)

4) S3 : Simple Storage Service (It is used to store unlimited data)

5) RDS : Relational Database System (It is used to create SQL databases)

6) Route 53 : DNS mapping (It is used to map application url to domain name)

7) VPC : Virtual Private Cloud (It is used to create isolated network for our application in AWS)

8) Elastic Beanstack : It is a PAAS in AWS. It will provide platform to run our application

9) ECS : Elastic Contianer Service (It is used to run our docker containers)

10) ECR : Elastic Container Registry (It is used to store our docker images)

11) EKS : Elastic Kubernetes Service (It is used to run K8S cluster)

12) IAM : Identity and Access Management (To manage permissions for our team members to use AWS services)

13) AWS Lambdas : Lambds are used for serverless computing (Run the code without thinking about servers)

14) EFS : Elastic File System (To share files among multiple virtual machines)

Environment Setup
+++++++++++++++

1) Create AWS Free Tier Account (AWS providing 1 year free access)

2) Download and Install MobaXterm software

++++++++++++++++++++++++++++++
What is Operating System ?
++++++++++++++++++++++++++++++

-> Operating System is a software which is used to interact with computer

-> Operating System is acting as mediator between users and computer hardware components

-> Operating System is mandatory to use any computer

-> OS provides environment to run other applications

(browser, notepad, paint, calc)

-> The OS came into market in 1950

-> Microsoft released it's first OS in 1981 (MS DOS)

+++++++++++++++
Windows OS
+++++++++++++++

-> Developed by Microsoft Company

-> It is having GUI (Graphical user interface)

-> It is single user based Operating System (only one person can use this at a time)

-> It is commercial (paid)

-> Less Security

-> It is recommended for personal use

- Watch Movies
- Using Internet
- Playing Games
- Attending Online classes
- Store data

++++++++++++++++
Linux OS
++++++++++++++++

-> Linux is Community Based OS

-> Linux is Free & Open Source OS

-> Linux is Multi User Based OS

-> High Security

-> Recommended to use for Applications, Servers, Databases etc..

Note: In realtime, we will use Linux OS only to setup infrastructure required to run our application

-> Linux OS is not only for Administrators, even developers and testers also will use Linux OS in realtime to monitor our application and application servers and application logs.

++++++++++++++++
History Of Linux
++++++++++++++++

-> In 1991, a student 'Linus Torvalds' developed this Linux OS

-> Linux Torvalds identified some challenges in UNIX OS and he suggested some changes
for Unix OS but UNIX OS Team rejected 'Linux Torvalds 'suggestions

-> Linus Torvalds used Minux OS to develop Linux

Linus + Minux

-> First Two letters from his name and last 3 letters from Minux OS

LI + NUX => LINUX

-> Linus Torvalds released LINUX OS with source code into market so that anybody can modify LINUX OS thatys why it is called as Open Source Operating System.

-> As Linux OS is open source, so many people and companies taken that Linux OS and modified according to their requirements and released into market with different names those are called as Linux Distributions.

RHEL -- RED HAT
Ubuntu OS
Cent OS
Fedora
Open SUSE
Kali Linux
Debian

Amazon Linux

Note : 200+ Linux Distributions are available in the market.

+++++++++++++++
Environment Setup
+++++++++++++++

1) Create account in AWS (it will ask debit/credit card) - selected cards are accepted

2) Create Linux VM using AWS EC2 service (download key pair)

3) Download Mobaxterm software to connect with Linux VM

4) Connect with Linux VM using Public IP and pem file

Note: Once work is completed then stop your Linux VM in EC2 to avoid billing

+++++++++++++++
Linux Commands
+++++++++++++++

$ whoami : It will display currently logged in username

$ pwd : present working directory

$ date : To display current date

$ cal : To display calendar

-> In Linux everything will be represented as a file

-> We have 3 types of files in linux

1) Ordinary File / Normal File (it will start with -)
2) Directory File (it will start with d)
3) Link File (it will start with l)

-> The file which contains data is called as ordinary file

-> Directory file is equal to folder (it can contain files and folders)

-> The file which is having linking is called as Link File

=> touch : it is used to create empty file

```
$ touch f1.txt
$ touch f2.txt
$ touch f3.txt f4.txt
```

=> To display files we will use 'ls' command

```
$ ls
```

=> To create a file with data we will use 'cat' command

```
$ cat > hello.txt
//write data
press CTRL + d (to save and exit)
```

```
$ cat hello.txt   (To display file data)

$ cat >> hello.txt (To append data in the file)
//write data
press CTRL + d (to save and exit)
```

-> To create directory we will use 'mkdir' command

```
$ mkdir dirname
```

-> To remove the file we will use 'rm' command

```
$ rm filename
```

-> To remove empty directory we will use 'rmdir' command

```
$ rmdir dirname
```

=> 'ls' is used to list out all files & directories available in the given directory

Note: we can pass several options for 'ls' commands

-> ls : It will display all files in alphabetical order (a to z)

-> ls -r : It will display all files in reverse of alphabetical order (z to a)

-> ls -l : It will display long listing of files

-> ls -t : It will display all files based on last modified date and time. Most recent file will be display at top and old files will display at bottom.

-> ls -rt : It will display all files based on reverse of last modified date and time. Old files will display at top and recent files will display bottom.

-> ls -a : It will display all files including hidden files (hidden files will start with .)

-> ls -li : It will display files with inode numbers.

-> ls -lR : It will display all files and directories along with sub directoris content

Note: -R represents recursive

Note: We can use several options for ls command at a time. When we are using multiple options order of the options is not important

```
$ ls -ltr
$ ls -tlr
$ ls -l -t -r
$ ls -trl
```

Note: All the above commands will give same output

-> To display content of given directory we can execute like below

```
$ ls -l <dirname>
```

-> To delete a file we will use 'rm' command

```
$ rm <filename>
```

-> To delete empty directory we will use 'rmdir' command

```
$ rmdir <dirname>
```

-> To delete non-empty directory we will use 'rm' command like below

```
$ rm -r dirname
```

-> To display file content we will use 'cat' command

```
$ cat filename
```

-> To display file content with line numbers we will use '-n' option

```
$ cat -n filename
```

-> To display multiple files content at a time execute command like below

```
$ cat file1 file2 file3
```

-> Copy one file data into another file using 'cat' command

```
$ cat f1.txt > f8.txt
```

-> Copy more than one file data into another file

```
$ cat f1.txt f2.txt > f9.txt
```

-> 'tac' command is used to reverse file content

```
$ tac filename
```

-> 'rev' command is used to reverse each line content of the file

```
$ rev filename
```

-> head command is used to display file data from top (default 10 lines)

```
$ head filename
$ head -n 5 data.log  (first 5 lines data)
$ head -n 20 data.log (first 20 lines data)
```

-> tail command is used to display file data from bottom (default 10 lines)

```
$ tail filename (last 10 lines data)
$ tail -n 100 filename (last 100 lines data)

$ tail +25 filename (it will display data from 25th line to bottom)
```

Note: To see on-growing logs we can use '-f' option

```
$ tail -f data.log   (Live log message we can see)
```

-> It is used to count no.of lines, no.of words and no.of characters in the file

```
$ wc f1.txt
```

-> To copy the data from one file to another file

```
$ cp one.txt two.txt  ( or ) $ cat one.txt > two.txt

$ cp f1.txt f2.txt f3.txt  (invalid syntax)
```

-> We can't copy morethan one file data using 'cp' command. To copy multiple files data we should go for 'cat' command

```
$ cat f1.txt f2.txt > f3.txt
```

-> To rename files we will use 'mv' command

```
$ mv f1.txt f1111.txt

$ mv  dirname dirnewname
```

Note: We can use 'mv' command for renaming and moving files

-> To compare files we can use below commands

```
$ cmp f1.txt f2.txt
$ diff f1.txt f2.txt
```

-> cmp command will display only first difference in given 2 two files
-> diff command will display all the differences in the content

-> 'grep' stands for global regular expression print.

-> 'grep' command will process the text line by line and prints any lines which matches given pattern.

```
Ex: I want to print all lines which contains 'NullPointerException'

$ grep -i 'NullPointerException'  *
```

Note: We can install grep using below command

```
$ sudo yum install grep
```

```
//search for the lines which contains given word in the given filename
$ grep 'word' filename

//search for the lines which are having exception keyword in server.log file
$ grep -i 'exception' server.log

//search for the given text in present directory and in sub-directories also
$ grep -R 'exception'
```

=> We can pass several options for 'grep' command

-c : This prints only the count of files that matches give pattern

-i : ignore case-sentitivty

-n : Display the matched lines and their line numbers

-l : Displays only file names that matches the pattern

-h : Displays matched lines without file names

-R : Displays matched lines with file names

Working with Text Editor In Linux
++++++++++++++++++++++++++

=> To edit a file we can use 'vi' eiditor (visual editor)

```
        $ vi filename

-> To edit data we should press 'i' (insert mode)

-> After editing the file To save the changes press 'Esc' then :wq  (To save and
close the file)

-> After editing the file If we don't want to save the changes press 'Esc' then :q!
 (To close the file)


+++++++++++++++
File Permissions
+++++++++++++++

-> In Linux OS, file permissions are divided into 3 types

                1) User Permissions
                2) Group Permissions
                3) Others Permissions

-> File Permissions will be represented with below characters

                r   - read
                w  - write
                x   - execute

-> For Every file, permissions are divided into 3 sections

                rwxrwxrwx

-> First 3 characters will represent user permissions
-> Next 3 characters will represent group permissions
-> Last 3 characters will represent others permissions

Example-1 :   rw-rw-r--

        -> user having read & write permissions
        -> group having read & write permissions
        -> others having only read permission

Example-2 :     rwxrwxr-x

        -> user having read , write and execute
        -> group having read, write and execute
        -> others having read and execute


Example-3 :   r---w---x

        -> User having only read permission
        -> Group having only write permission
        -> Others having only execute permission


-> Addring write permission fo user

        $ chmod u+w filename

-> Removing read permission for others

        $ chmod o-r filename


++++++++++++++++++
```

Creating User in Linux
++++++++++++++++++

```
$ sudo useradd ashok
$ sudo passwd ashok
```

Note: Enter pwd & confirm password

-> Switch to newly created account

```
$ sudo su ashok
```

Note: For every user one directory will be created in home directory with our username

For Ec2 User: /home/ec2-user

For ashok : /home/ashok

-> To install any software in linux we will use root user permissions

-> To switch to root user account we will use below command

```
        $ sudo su
```

++++++++++++++++++
Install Java in Linux
++++++++++++++++++

-> java -version (Check java version)

```
$ sudo su
$ sudo yum install java-devel
```

Hosting static website using EC2 VM
++++++++++++++++++++++++++++++

-> Website means collection of web pages (HTML pages)

-> To run the web pages we need a web server

        Ex: Apache, Httpd, Nginix etc....

-> We can install Webserver in our EC2 VM

-> Webservers will run on the port number : 80 with http protocol

Note: To allow the traffic for Web server, we need enable HTTP protocol with 80 port number in Security Group of our EC2
(Inbound Rules)

```
$ sudo su
$ yum install httpd -y
$ cd /var/www/html
$ echo "<html><h1>Hotel Server - 1</h1></html>" > index.html
$ service httpd start
```

-> Once httpd service is started we can access our website using EC2 VM public ip address

URL : http://13.126.19.99/

=> When we deploy our application then multiple users will access our application at a time then burden will increase on the server

=> If burden increased on server then below are the side effects

a) Request processing may take more time
b) Response will be delayed
c) Customers will get irrirated with our application response
d) We will loose customers
e) We will loose revenue
f) We will loose trust
g) We will loose our brand value

=> To overcome these problems we will use Load Balacing concept

++++++++++++++
Load Balancing
++++++++++++++

-> The process of distributing load to multiple servers is called as Load Balancing

-> Instead of running our application in one server, we will run on multiple servers

-> We will setup Load Balancer to distribute load to multiple servers

-> Create 2 Ec2-Instances
-> Install Webserver (Httpd) in both servers and setup static website
-> Start httpd server in both instances
-> Enable Http port : 80 in security groups for both ec2-instances
-> Create Load Balancer
-> Scheme: Internet facing
-> Target Group (collection of servers)
-> Add both Ec2 instances for target group and select target group in LBR
-> Add HTTP-80 port for LBR security group
-> Access app using LBR DNS name

++
How to implement LBR for Microservices based application
++

1) Create EC2 Instance with below user-data (Name it as HotelServer-1)

```
#! /bin/bash
sudo su
yum install httpd -y
cd /var/www/html
echo "<html><h1>Hotel Server - 1</h1></html>" > index.html
service httpd start
```

2) Create EC2 instance with below user-data (Name it as HotelServer-2)

```
#! /bin/bash
sudo su
yum install httpd -y
cd /var/www/html
echo "<html><h1>Hotels Server - 2</h1></html>" > index.html
service httpd start
```

3) Create HotelServers Target group with above 2 instances

4) Create EC2 instance with below user-data (Name it as FlightServer-1)

```
#! /bin/bash
sudo su
yum install httpd -y
cd /var/www/html
echo "<html><h1>Flights Server - 1</h1></html>" > index.html
service httpd start
```

4) Create EC2 instance with below user-data (Name it as FlightServer-2)

```
#! /bin/bash
sudo su
yum install httpd -y
cd /var/www/html
echo "<html><h1>Flights Server - 2</h1></html>" > index.html
service httpd start
```

5) Create FlightsServers Target group with above 2 instances

6) Create Load Balancer by select HotelServers Target Group

7) Goto LBR Listeners and configure Route Based Routing for Flights Target Group

8) Test it with DNS name

Note: Once practise completed, delete all instances and LBR also.

Maven Environment Setup
++++++++++++++++++++++

1) Download maven software from official website (zip file)

URL : https://maven.apache.org/download.cgi
File Name: apache-maven-3.8.5-bin.zip

2) Extract zip file and set MAVEN_HOME in environment variable

3) Set Path for Maven in Environment variables

4) Set JAVA_HOME also in environment variables

JAVA_HOME = C:\Program Files\Java\jdk1.8.0_201
MAVEN_HOME = C:\apache-maven-3.6.0

Note: Maven Path we have to set upto maven bin directory

C:\apache-maven-3.6.0\bin

=> To verify maven setup, open command prompt and execute below command

```
$ mvn -v
```

Note: It should display version of the maven. If it is displaying version that means maven setup is successfull.

Creating standalone application using maven
++++++++++++++++++++++++++++++++++++

1) Create one folder for maven practise

2) Open Command prompt from that folder

3) Execute below command to create maven project

```
>> mvn archetype:generate -DgroupId=in.ashokit -DartifactId=01-Maven-App
-DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

++++++++++++++++++++++++++++++++++++
creating web application using maven
++++++++++++++++++++++++++++++++++++

```
>> mvn archetype:generate -DarchetypeArtifactId=maven-archetype-webapp
-DgroupId=in.ashokit -DartifactId=01-maven-web-app -DinteractiveMode=false
```

++++++++++++++++++++
Apache Tomact Server
++++++++++++++++++++

=> Apache Tomcat is a web server

=> Apache Tomcat is used to run Java Web Applications

=> Apache Tomcat is free & open source

=> Apache Tomcat runs on 8080 port bydefault (we can change that port in server.xml file)

++++++++++++++++++++++++++++
Apache Tomcat Folder Structure
++++++++++++++++++++++++++++

bin : It contains commands to start and stop tomcat server

conf : It contains configuration files

lib : It contains libraries (jar files)

logs : It contains server log files

temp : Temp files will be created here (we can delete them)

webapps : This is called as deployment folder (war file should be kept here to deploy)

Note: We will keep war file in webapps folder for deployment

++++++++++++++++++++++++++++++++++
Working with Apache Tomact in Linux
++++++++++++++++++++++++++++++++++

-> Login into AWS Management Console

-> Create EC2 Instance (Amazon Linux AMI)

-> Connect to EC2 instance using MobaXterm / Putty

-> install java software using below command

```
$ sudo yum install java-1.8.0-openjdk
```

-> Verify the version of java installed in our machine

```
$ java -version
```

Note: If we have multiple java versions installed then we can switch to particular version using below command

```
$ alternatives --config java
```

-> We can download apache tomcat from official website

URL : https://tomcat.apache.org/download-90.cgi

-> We can find apache tomcat urls to download in official website downloads page

-> Copy the URL of tar file and execute below command in linux machine

```
$ wget
https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.63/bin/apache-tomcat-9.0.63.tar.gz
```

-> After tomact tar file got downloaded then extract Tomcat Tar file using below command

```
$ tar -xvf <tomact-tar-file-name>
```

-> Go inside tomcat folder and see folder structure

```
$ cd tomact-folder
$ ls -ltr
```

-> Go to tomact bin directory and run tomact server

```
$ cd bin
$ ./startup.sh
```

Note: Tomcat Server runs on 8080 port by default. Enable this port in security group as custom tcp

Type : Custom TCP
Protoal : TCP
Port Range: 8080
Source : Custom (0.0.0.0/0)

-> Access Tomact server from your browser

URL : http://EC2-VM-Public-IP:8080/

Note: It should open tomact server home page.

++
Providing remote access for Tomcat Admin Console
++

=> By default the Host Manager is only accessible from a browser running on the same machine as Tomcat. If you wish to modify this restriction, you'll need to edit the Host Manager's context.xml file.

=> File Location : <tomcat-folder>/webapps/manager/META-INF/context.xml

=> In Manager context.xml file, change <Valve> section like below (allow attribute value changed)

```
<Context antiResourceLocking="false" privileged="true" >
     <Valve className="org.apache.catalina.valves.RemoteAddrValve" allow=".*" />
</Context>
```

Add tomact users in "tomact-folder/conf/tomact-users.xml" file like below

++

```
<role rolename="manager-gui" />
<role rolename="manager-script" />
<user username="tomcat" password="tomcat" roles="manager-gui" />
<role rolename="admin-gui" />
<user username="admin" password="admin"
roles="manager-gui,admin-gui,manager-script"/>
```

-> Stop the tomact server and start it

++

We can change tomcat server default port in tomact/conf/server.xml file

++

-> When we change tomact port number in server.xml file then we have to enable that port in Security Group which is associated with our EC2 instance.

++

Steps to display Maven Web Application in Tomcat Server

++

1) Create Maven Web application

2) Edit "index.jsp" file like below (File Location : project-folder\src\main\webapp)

```
<html>
<body>
<h1><font color='red'>Welcome to Ashok IT..!!<font></h1>
<h2>Learn Here.. Lead Anywhere..!! </h2>

<a href="https://ashokitech.com/online-training-schedules">Click Here To See
Training Schedules</a>
</body>
</html>
```

3) Package maven web application as war file using maven goals

```
        $ mvn clean package
```

4) Go to Tomcat Server Admin Dashboard and click on "Manager App"

5) Select War file to upload and click on 'deploy' button

6) War file will be deployed and it will display in applications

7) Click on Application Path (It will open the application in browser)

Conclusion

++++++++++

-> Stop Apache Tomact Server

-> Stop Ec2 instance

++

++++++++++

Jenkins installation on AWS EC2 using YUM
##

Create EC2 VM in AWS (Amazon Linux)

Connect to EC2 VM using MobaXterm software

Add Jenkins repo to your yum repository
$ sudo wget -O /etc/yum.repos.d/jenkins.repo
https://pkg.jenkins.io/redhat/jenkins.repo

Import a key file from Jenkins-CI to enable installation from the package
$ sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key

For Amazon Linux 2
$ sudo amazon-linux-extras install epel

Install Jenkins
$ sudo yum install jenkins -y

install java software using below command

$ sudo yum install java-1.8.0-openjdk

Start and enable Jenkins service
$ sudo systemctl start jenkins
$ sudo systemctl enable jenkins

Check Jenkins Server Status (it should be active)
$ sudo systemctl status jenkins

Get the initial administrative password
$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword

pwd : 5fe6ddcc9db244cab6aca5ccf2d6a83a

Step3:
Open your EC2 instance public DNS or public IP
(http://<PUBLIC_DNS/PUBLIC_IP>:8080/) along with port 8080 in your favorite
browser. And provide the administrative password obtained during the installation.

Note: Make sure you enabled 8080 port in Security Group Inbound Rules which is
attached to EC2 instance.

                        URL : http://43.204.214.21:8080/

-> Provide pwd which we have copied to unlock jenkins

-> Select "Install Suggested Plugins" card (it will install those plugins)

-> Create Admin account

Install Git & Maven In Jenkins Server Machine

$ sudo yum install git -y
$ sudo yum install maven

++
++++++++++++

Jenkins Job with with GIT Hub Repo + Maven + Tomcat Server - Integration
++

1 Go to Jenkins Dashboard -> Manage Jenkins --> Manage Plugins -> Goto Available -> Search For "Deploy Container" Plugin -> Install it.

2) Create Jenkins Job

-> New Item
-> Enter Item Name (Job Name)
-> Select Free Style Project & Click OK
-> Enter some description
-> Go to "Source Code Management" Tab and Select "Git"
-> Enter Project "Git Repo URL"
-> Add Your Github account credentials
-> Go to "Build tab"
-> Click on Add Build Step and Select 'Inovke Top Level Maven Targets'
-> Select Maven and enter goals 'clean package'
-> Click on 'Post Build Action' and Select 'Deploy war/ear to container' option
-> Give path of war file (You can give like this also : **/*.war)
-> Enter Context Path (give project name)
-> Click on 'Add Container' and select Tomcat version 9.x
-> Add Tomcat server credentials (give the username & pwd which is having manager-script role)
-> Enter Tomact Server URL (http://ec2-vm-ip:tomcat-server-port)
-> Click on Apply and Save

3) Run the job now using 'Build Now' option and see see 'Console Output' of job

4) Once Job Executed successfully, go to tomcat server dashboard and see application should be displayed.

5) Click on the applicaton name (it should display our application)

++++++++++++++++++++++++++++++++++++
Running Spring Boot Application in AWS Cloud
++++++++++++++++++++++++++++++++++++

-> We can run Spring Boot Application in below ways

1) Take Spring Boot Jar file and run it (It provides embedded server)

2) Run Boot application using Elastic Beanstack (PaaS)

Approach-1
+++++++++

1) Upload spring-boot application jar file into ec2-user home directory using MobaXterm upload option

2) Install Java Software in Ec2 VM using below commands

```
$ sudo yum update -y
$ sudo yum install java
```

3) Run the jar file in EC2 instance

$ java -jar <jar-file-name>

Note: Enable 8080 port number in Security Group which is added to our EC2 Instance

4) Access Boot application in browser with below URL

URL : http://EC2-VM-Public-IP:8080/

Approach-2
++++++++++

1) Go to S3 in AWS Web Console
2) Create Bucket in S3 (Bucket name should be unique)
3) Upload Jar file into S3 bucket with Public Access (It will generate endpoint url for our jar file)
4) Copy Jar file Endpoint URL from S3
5) Download Jar file from S3 bucket into EC2 instance using below command

$ wget <jar-file-endpoint-url>

3) Run the jar file in EC2 instance

$ java -jar <jar-file-name>

Note: Enable 8080 port number in Security Group which is added to our EC2 Instance

4) Access Boot application in browser with below URL

URL : http://EC2-VM-Public-IP:8080/

Approach-3
++++++++++

-> AWS provided Elastic Beanstack service to run applications

-> Elastic Beanstack providing Platform as a service (PaaS)

-> Goto Elastic Beanstack service

-> Create Application

- Choose platform as java
- Upload code as jar file
- Finish the process

-> Once application got created, open that application -> Configuration -> Edit Software Configuration -> Add below property

SERVER_PORT = 5000

-> After adding this port, beanstack env will be updated and we can access our application using beanstack generated url

---

Code Coverage
+++++++++++++

-> Code Coverage is the process of measuring how many lines of code is executed as

part of Unit testing

-> As per industry standard every project should have minimum 80% of coverage

-> We can generate code coverage report using "Jacoco" plugin

-> By seeing code coverage report we can identify which lines are executed in unit testing and which lines are not executed in unit testing

-> By seeing code coverage report we can improve our unit test scenarios

-> In project for few classes unit testing is not required then such classes or packages we can exclude from Code Coverage report using <exclude/> option

-> To generate code coverage report add 'jacoco' plugin in pom.xml file

-> Jacoco plugin is associated with Maven 'test' goal.

-> When Maven 'test' goal is executed thean 'Jacoco' plugin will execute and it will generate the report.

-> Jacoco Report Location : <project-folder> / target / site / jacoco / index.html

-> Open index.html in browser then we can see Jacoco report

++++++++++++++++++++++++++++++++ add below plugin in pom.xml file++++++++++++++++++++++++++++++++++

```
                        <plugin>
                                <groupId>org.jacoco</groupId>
                                <artifactId>jacoco-maven-plugin</artifactId>
                                <version>0.8.2</version>
                                <configuration>
                                        <excludes>

<exclude>in/ashokit/entities/**</exclude>

<exclude>in/ashokit/bindings/**</exclude>

<exclude>in/ashokit/constants/**</exclude>

<exclude>in/ashokit/props/**</exclude>

<exclude>in/ashokit/exception/**</exclude>

<exclude>in/ashokit/config/**</exclude>
                                        </excludes>
                                </configuration>
                                <executions>
                                        <execution>
                                                <goals>
                                                        <goal>prepare-agent</goal>
                                                </goals>
                                        </execution>
                                        <execution>
                                                <id>report</id>
                                                <phase>test</phase>
                                                <goals>
                                                        <goal>report</goal>
                                                </goals>
                                        </execution>
```

```
                        </executions>
                    </plugin>

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++


-> For New Joinees below tasks will be assigned in the project for 2 to 3 months of
time

1) Unit Test implementation task

2) Code Coverage Improvement tasks

3) Sonar Fixes

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++
--------------------------------------------------------------------------------
------------------------------------------------------------

++++++++++++++++++++++++++
Do's and Don'ts In IT Company
++++++++++++++++++++++++++

Don't s
++++++++
-> We should not send any emails from company mail to personal gmail with project
information.

        Ex: BRD, FDD, Test Cases, Bug Reports etc..

-> If you send project/client information to your personal mail then you will loose
your job.

-> Don't open social media sites in office system because everything will be
tracked in office system by networking team.

        Ex : Facebook, Gmail etc..

-> You should not discuss your package (salary) with anybody in the company and you
shouldn't ask anybody about their salary.

-> If you get the job with fake experience don't tell to anybody about your fake
experience in the company.


Do's
------
-> You should be good team player ( You should have good rapo with all team members
)

-> For every one month setup one-to-one meeting with your lead & manager to discuss
about your work performance and discuss about improvements to get into next level.

-> Declare your investments in HR portal for Tax deductions

-> Declare your's and your family memebers details in Insurance Portal.

-> Every month payslip you can send from office mail to personal gmail

-> Every Year Form-16 will be generated that you can send to your personal gmail.


++++++++++++++++
```

Joining Formalities
++++++++++++++++++

-> On the day of joining in the company 'On Boarding HR' will assit you to complete all joining formalities

-> Employment documents we have to fill up

-> Collect Laptop

-> Collect Salary Account

-> After joining formalities are completed interact with 'Resource Manager' to get the project in the company.

-> Once Project Interview got completed you will be allocated to project

-> Once you got allocated to project, they will provide KT (Knowledge Transfer)

-> After KT got completed, they will assign work to you.

Exit Formalities

-> Once you got offer letter from another company setup one-to-one meeting with your manager and tell him/her that you are planning to resign.

-> Manager will ask the reason for the resign (Tell that you got good offer with better package for carreer growth)

-> Manager may ask you to stay back in the same company (He will tell some benefits)

-> After discussion completed with manager, then resign in HR portal.

-> Once you resign you will get mail from HR team with last working date.

-> While serving Notice period we should not take any leave.

-> While serving notice period we have to hand over all our work to existing team members

-> Provide KT to team members

Note: While serving notice period we can cancel our Resignition.

-> On last working date send email to all the team members with Good Bye message

-> Submit your ID card, Laptop and all company belongings to respective team.

-> Once your last working date is completed with in 30 days company will do 'Full and Final Settlement' and will send salary, experience certificate and reliving letter.

-

Initial Docs

Offer letter
3 months payslips
Hike letters
Company Email ID

Once you got selected in the company, you have to share above docs to company to get offer letter.

-> At the time of joining / after joining they will ask you to submit
   "Reliving Letter & Experience Letter".

Note: Consultancy person will provide all required documents.

+++
+++