Course name : Java Real-Time Project (Fullstack Development)

(Spring Boot, Microservices & Angular - 11)

Start Date : 22-Apr-2021

Class Timings: 10:00 AM IST - 11:30 AM IST (1.5 hour) (Mon-Sat)

Duration : 3 months (10 days +/-)

Trainer : Mr. Ashok (8+ yrs experience, Working Professional)

(5+ Years of experience in Training)

YouTube Channel : www.youtube.com/c/AshokIT

Facebook Group : www.facebook.com/groups/ashokitschool

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

Pre-Requisites:

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

Core Java

Adv Java (JDBC, Servlets & JSP)

SQL

ORM Framework (Good To have)

Spring Boot & Microservices (You can do this course parallely - @7:30 AM IST)

Angular - 11 with Typescript (You can do parallely - @11:30 AM IST - FREE FOR ALL)

Course Content:

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

Part-1 : Software Industry Terminology

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

1) How many type of software companies

2) Interview Process In Software Companies

3) What is Software Project

4) How many types of software projects & work culture

5) How many types of teams & Roles of teams

6) Role Chart In Software Company

7) Bridge Calls & VC

8) Conclusion

Part-2 : Java Real-Time Tools (30+ Realtime Tools)

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

1) Maven

2) Gradle

3) SVN (Videos available in our Youtube)

4) GIT HUB

5) BitBucket

6) Source Tree

7) Log4J, LogBack & SLF4J

8) Log Monitoring Tools (Putty, WinScp & Splunk)

9) JUnit (For Unit Testing)

10) Mocking (Easy Mock, PowerMock, Mockito)

12) Code Coverage (Jacocco)

13) Sonar Qube (Code Review)

14) Agile with JIRA (Project Management)

15) JMETER (Performance Testing)

16) Swagger (Generating Documentation for REST APIs)

17) SOAP UI & POSTMAN (API Testing Tools)

18) JENKINS (CI & CD Software)

19) DOCKER (Containerization Tool)

20) Reports Using Apache POI & IText API

21) Java Mail API

22) Apache Kafka

23) Redis Cache

24) AWS Deployment

Part-3 : Three Mini Projects (Spring Boot & Microservices with Angular)

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

1) Project-1

2) Project-2

3) Project-3

Part-4 : Major Project (This keep project you can keep in Resume)

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

- Health Insurance

- 7 Modules

- Project Architecture

- Project Flow

- Batch Processing

- Multi Threading

- Exception Handling

- Reports

Part-5 : Interview Guide

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

- Resume Preparation

- Profile Creation In Job Portals

- How to apply for Jobs

- How to cover gap

- Joining Formalities

- First Day in Company

- Last Day in company

- Full and Final Settlement

- Do's and Don't in IT company

JAVA REALTIME PROJECT (FULLSTACK DEVELOPMENT) - ONLINE TRAINING

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

Start date : 22-Apr-2021

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

Session-1 (22-Apr) : Pre-requisites & Course-content

(https://www.youtube.com/watch?v=exSnWlLdT7E)

Session-2 (23-Apr) : Doubts Clarification for students

(https://www.youtube.com/watch?v=f\_kGbvI3CFg)

24-Apr & 25-Apr : No classes

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

Session-3 (26-Apr-2021) : Part-1 (Software Industry Details

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

-> Software companies can be categorized into 3 types

1) Product Based Companies

2) Service Based Companies

3) Outsourcing Companies

Product Based Companies

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

-> The companies which will develop product/project and selling to the people in the market is called as Product Based Companies.

-> For Product Based Companies clients will not be available they will have only Customers.

Ex : Microsoft, HP, IBM, Google etc...

-> In Product Based Companies interviews, they will concentrate mainly on below concepts

1) Data Structures

2) Algorithms

3) Design Patterns

4) System Design

5) Problem Solving

6) Time Complexity and Space Complexity

Note: In Product Based Companies they will conduct Coding Test using Hacker Ranker or Hacker Earth.

-> Product Based Companies provide high packages

No.of.years of exp \* 6 Lakhs

3 \* 6 = 18 Lakhs

Service Based Companies

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

-> The companies which are providing services to others are called as Service Based Companies.

-> Service Based Companies will deveop the projects based on client given requirements.

Ex : TCS, Wipro, Infy, Capgemini, Accenture, Cognizent, Deloitte etc..

-> In Service Based Companies interviews, they will ask questions related to latest technologies

1) Spring Boot

2) REST Apis

3) Microservices

4) Fullstack development

-> Below package structure will be there in Service based companies

Years of exp \* 3 lakhs

-> In service based companies onsite opportunity will be there.

Outsourcing Companies

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

-> Outsourcing companies will provide employees to other companies on contract basis.

Ex: Magna info Tech, SourceOne mgmt etc..

Course Details

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

15-JRTP Course Fee : 6000 INR (Angular-11 FREE)

Daily class notes and class recording will be shared

class notes life time access

class recording will be avaialble for only 2 days

Spring Boot & Microservices : 5000 INR

Yesterday's session : Types of companies in IT industry

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

-> We can see 3 types of companies in IT industry

a) Product Based Companies

b) Service Based Companies

c) Outsourcing Companies

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

Today's session : How many types of projects in companies

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

-> In IT companies we can see 3 types of projects

1) Scratch development projects

2) Maintenence Projects

3) Migration Projects

-> If we develop a brand new project from the beginning then it is called as Scratch Development Project.

-> The projects which are already running in the market will be under maintenence.

-> In the maintenence projects we will have below types of works

1) Change Request

2) Enhancements

3) Bug Fixing

-> If we are changing our project from one technology to another technolor then it is called as Migration Project.

Part-2 of 15-JRTP : Java Realtime Tools

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

-> In part-2 we will learn some realtime tools which we will use in realtime project.

-> We will learn how to download and install those tools

-> We will learn how to work with those tools in project

-> We will learn what are common problems we will face while working with those tools and how to rectify them.

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

-> After class is completed everybody has to practise the tools concept

-> Sometimes in class i will ask students randomely to perform operations with tools (By giving co-host)

-> Weekly assignments also will be there

Note: Now we will learn few tools then will start Mini Projects development. Once first mini project is completed we will learn remaining tools.

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

Maven : Build tool

Git Hub & Git : Source Code Repository Tools

BitBucket : Source Code Repository Tool

Source Tree : Repository client software

Log4J : Logging

Putty, WinScp, Splunk : Log monitoring tools

Jenkins : CI & CD tool

Agile JIRA : Project management tool

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

Mini Project Development

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

Junit

Mocking (Easy Mock & PowerMock)

Jacoco (Code Coverage)

SonarQube ----------Code review

Swagger

PostMan

JMETER ---> Performance Testing

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

2 Mini Project

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

Docker

Apache Kafka

Redis Cache

Apache poi api

itext api

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

3 mini project

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

Major Project

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

Interview Guide

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

Note: Please check your email before joining 15-JRTP class daily

Today we will share new zoom link with all previous videos links.

Yesterday's session : Introduction for Part-2 in 15-JRTP

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

Build Tools

-----------

-> Build tools are used to automate application build process

Ex : Ant, Maven and Gradle etc...

Note: Ant is outdated in market people are using Maven or Gradle as build tool for java projects development.

-> Apache Organization provided Maven software to simplify application build process easy.

Maven Main objectivies

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

-> Make build process easy

-> Maintain uniform build process

-> Provide Quality information about project

-> Encouraging Better development practises

-> Download maven software using below url

https://maven.apache.org/download.cgi

(apache-maven-3.8.1-bin.zip)

-> Extract maven zip file then add below 2 things in Environment variables

1) MAVEN\_HOME

2) MAVEN PATH

Yesterday's session : Maven Introduction

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

-> Maven software provided by apache org

-> Maven is a free & open-source software

-> Maven is used to simplify java projects build process

-> Maven follows uniform build process for all the applications

-> Maven provides quality information related to project

-> Maven is encouraging to follow better development practises

-> Maven setup also got completed

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

-> We can create maven project using cmd and using IDE also

-> 'ArcheType' Selection is required to create maven project. It represents type of the project.

quick-start : Java Standalone app

web-app : Java Web application

Yesterday's session : Maven Project Folder Structure

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

-> Maven project contains pom.xml file which acts as maven configuration file

-> All the dependencies which are required for our application we will add in pom.xml file only

Note: We can identify maven dependencies in mvnrepository.com website

---------------------------------Adding maven dependency----------------

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.3.6</version>

</dependency>

</dependencies>

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

-> We can add multiple dependencies under <dependencies/> tag in pom.xml file

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

-> To perform application build process maven provided several goals for us

1) clean

2) compile

3) test

4) package

5) install etc.

-> 'clean' goal is used to delete target folder

-> 'compile' goal is used to compile java classes and generate .class files in target folder

-> 'test' goal is used to execute junit classes available in application for unit testing

-> 'package' goal is used to generate 'jar' or 'war' file for our application

-> 'install' goal is used to represent our project as dependency for other projects

10-may

Last session : Maven Goals

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

-> Maven provided several goals to perform application build process

-> Every maven goal having its plugin to perform the actual operation

1) clean

2) compile

3) test

4) package

5) install

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

Today's session : Maven install goal

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

11-may

Yesterday's session : Maven install goal

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

-> To create our own dependency we will use maven install goal

-> Once we create dependency using maven install goal, then we can use that project as a dependency in other projects by configuring dependency details in pom.xml file.

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

clean, compile, test, package and install

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

Today's session : Maven Repositories

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

-> Maven supports 3 types of repositories

1) Central Repository

2) Local Repository

3) Remote Repository

-> Central Repository will be maintained by apache org (all jars will be available)

-> When we configure maven it will create local repository in our system

(Local Repo Path : C:\Users\<username>\.m2 )

-> Every Software Company will maintain their own repository for jars that repository is called Remote Repository.

-> When we add dependency in project pom.xml file then maven will search for that dependency in local repo first. If dependency available in local it will add to project build path.

-> If dependency not availbale in local repo then it will download that dependency from Central Repository.

Note: If we configure Remote Repository in our IDE, if dependency not available in local then it will download from Remote Repository.

Note: In company we will connect to Remote REpository only

-> To configure Remote Repository in IDE we will follow below steps

IDE -> Window -> Preferences -> Maven -> User Settings -> Add settings.xml file

Note: Once we join in project, TL or Manager will provide settings.xml file for us

14-may

12-May-2021 & 13-May-2021 : No classes

-------------------------------------------------------------------------Last session : Maven Repositories (Central, Local and Remote)

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

-> Central Repo will be maintained by Apache Org

-> In our machine Maven Local Repo will be available

-> Every company will maintain their own Remote Repository

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

Today's session : Pom.xml tags

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

-> pom stands for project object model

<project>

<groupId>in.ashokit</groupId>

<artifactId>01-Maven-App</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<properties>

<spring.version>5.3.7</spring.version>

</propertites>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

<scope>runtime</scope>

<dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

<scope>runtime</scope>

<dependency>

<dependencies>

<build>

<finalName>First-App</finalName>

<plugins>

<plugin>

.....

</plugin>

</plugins>

</build>

</project>

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

-> Maven dependency 'scope' decides when to include that dependency in class path.

-> There are 6 scope elements available in maven

compile --------- This is by default

provided

runtime

test

system

import

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

Q) How to exclude maven dependency in pom.xml?

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

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

<exclusions>

<exclusion>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

</exclusion>

</exclusions>

</dependency>

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

<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>03-Maven-App</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<properties>

<spring.version>5.3.6</spring.version>

</properties>

<dependencies>

<dependency>

<groupId>in.ashokit</groupId>

<artifactId>02-Pwd-Utils</artifactId>

<version>1.0</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

<exclusions>

<exclusion>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

</exclusion>

</exclusions>

</dependency>

</dependencies>

<build>

<finalName>Maven-App</finalName>

</build>

</project>

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

1) What is Maven ?

2) What is Archetype selection in Maven?

3) What is maven-archetype-quickstart ?

4) What is the difference between src/main/java and src/test/java folders?

5) What is the purpose of src/main/resources folder?

6) What is pom.xml file ?

7) What are maven goals available?

8) What is the difference between clean and compile?

9) What is the purpose of maven test goal?

10) What is the purpose of maven package goal?

11) What is the use of install goal?

12) What is transitive dependency in maven?

13) How to exclude dependency in maven?

14) What is scope in maven?

15) What repositories available in maven?

16) What is settings.xml in maven?

17) What is <finalName> tag in pom.xml ?

18) What is Maven plugin?

19) What is force update in maven?

20) What is .m2 folder?

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

Today's assignment : SDLC life cycle steps & Waterfall Methodology

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

15-may

Last session : Maven Tool

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

Today' session : Agile Methodology

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

Q) What is SDLC ?

Q) What is waterfall methodology ?

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

-> SDLC stands for Software development life cycle.

-> It includes several phases in application development process

1) Requirements Gathering

2) Requirements Analysis

3) Design

4) Development

5) Testing

6) Deployment

7) Maintenence

-> Waterfall methodology follows linear process in SDLC.

-> In Waterfall methodology everything happens sequentially.

-> In Waterfall methodology requirements & Project budget is fixed

-> Waterfall methodology is suitable for small scale applications

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

-> Now a days in industry Agile methology is using

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

-> AGILE methodology is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project.

-> In the Agile model, both development and testing activities are concurrent, unlike the Waterfall model.

-> Agile methodology is best sutiable for large scale applications.

-> In Agile methodology requirements & budget is not fixed.

-> All the given requirements will be divided into multiple releases in Agile methodology.

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

Agile Terminology

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

Product Owner

Scrum Master

Tech Lead

Agile team members

-> Product Owner is responsible for client deliverables

-> Scrum Master is responsible to manage agile team

-> Tech Lead is responsible to resolve any technical problems facing by agile team members

-> Agile team contains both developers & testers (they will work paralelly)

1) Story

2) Story Points

3) Backlog Grooming

4) Backlog

5) Sprint Planning

6) Sprint

7) Scrum

8) Mid Iteration Review

9) Release

10) Retrospective

-> Story means a task in the project

ex : logging in project, exception handling, unit testing, code review etc...

-> Every story will have story points. Story points decides duration to complete the task

3 points --- 8 hours duration (1 day)

5 points --- 16 hours duration (2 days)

8 points ----24 hours duration (3 days)

Note: In the team anybody can create story and assign story points

-> Backlog grooming is the meeting conducted by scrum master to identify pending tasks in the project. All the team members will join in the backlog grooming meeting will identify pending tasks in the project.

-> For all the pending tasks stories will be created in JIRA that is called as Backlog bucket.

Note: The stories which are pending are called as Backlog Stories

-> Scrum Master will conduct a meeting with Agile team to identify the stories to complete from backlog that meeting is called as Sprint Planning. All the team members will join in Sprint Planning session.

-> The stories which are identified in Sprint Planning are called as one Sprint. Sprint will have start date and end date. Sprint duration will be 2 weeks (10 working days).

-> Once sprint started everyday scrum master will conduct scrum call for 15-20 mins of time.

In scrum call every team member will provide work status to scrum master (like which story we are working on and what is the status and when it will complete).

-> Mid Iteration review meeting will be conducted by Scrum Master to evoluate sprint stories once half of sprint duration got completed.

-> Retrospective meeting will be conducted by Scrum Master to check sprint work.

1) Lessons learnt in previous sprint

2) Improvement areas

3) Achievements

-> For every sprint we don't release project to client. After 3 to 4 sprints we will do one release to client.

16-may

Last session : Agile methodology & Agile Terminology

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

-> Agile is an iterative approach to develop software applications.

-> In Agile methodology developers & testers will work paralelly to complete the project.

-> In Agile methodology requirements are not fixed.

-------------------------------------------------------------------------Agile Terminology

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

-> Product Owner

-> Scrum Master

-> Tech Lead

-> Agile Team

-> Story

-> Story Points

-> Backlog Grooming

-> Backlog

-> Sprint Planning

-> Sprint

-> Scrum

-> Mid-Iteration Review

-> Retrospective

-> Release

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

-> Agile project work will be managed by using JIRA software

-> JIRA is an Atlasian company product.

-> JIRA is used to manage project work and jira is used for bug reporting also.

-> Software companies will purchase JIRA product from Atlasian company to manage projects work.

Note: We can use trail version of jira to understand jira functionality

17-may

Last session : Agile methodology & JIRA

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

Today's session : Version Control Softwares

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

-> In company multiple developers will be available to develop the project.

-> All the developers will not work from same location.

-> If developers are woring from multiple locations we will face below 2 problems

1) Code Integration

2) Monitoring Code Changes

-> To avoid above 2 problems we will use Version Control Software For Project.

-> Version Control Software will provide solution for code integration and providing monitored access for code.

-> There are several version control softwares are available in market. They are

1) Clear Case

2) CVS

3) SVN

4) Git Hub

5) BitBucket etc..

Working with GIT Hub

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

-> Create account in Git Hub (Its free of cost)

-> Install Git Client software

18-may

Last session : Purpose source code repository / Version Control Software

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

-> In one project mulitple developers will be available

-> Developers will be working from different locations.

-> If developers are working from multiple location we will face below problems

1) Code Integration

2) Monitoring code changes

-> To resolve the above 2 problems we will use Version Control Softwares.

1) Clear Case

2) CVS

3) SVN

4) GIT Hub

5) Bit Bucket etc..

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

-> We have created Account in git hub and we have installed git client software.

Note: When we create account in git hub it provides some space for free of cost to host our applications.

-> Git Hub is a cloud platform which is used to store project code and for hosting applications.

-> Git is the version control software.

-> Git Hub cloud platform using Git as a version control software.

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

Note: Once we join in a project in IT company, they will provide git account credentials for us (We no need to create).

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

-> In Git account we will create repository to store project source code.

Note: For one project we will create one repository

Note: In IT company, GIT admin will create repository for the project.

-> In Git we can create 2 types of repositories.

1) Public Repository

2) Private Repository

-> Public Repository means anybody can see and we will choose who can commit

-> Private repository means we will choose who can see and commit to our repository.

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

-> Open Git bash to execute git commands

Introduce ourself to git using config command

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

git config --global user.name "Ashok"

git config --global user.email "ashokitschool@gmail.com"

19-may

Last session : Git Repo Creation & Types of Repositories

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

-> Git hub is supporting for two types of repositories

1) public repo

2) private repo

-> Once Git Repository got created it will generate one unique url for that repo

https://github.com/Ashok-IT-School/15JRTP-01-App.git

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

Git Bash commands

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

git help

git help <command-name>

git config --global user.name "Ashok"

git config --global user.email "ashokitschool@gmail.com"

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

git init : To initialize our folder as git operations folder. If the .git folder is not available change file explorer options view can change display hidden files.

git add <filename> : To add one file to git staging area

Note: The files which are added to staging area are eligible for commit.

git status: To see files added and not added to staging area

git commit -m <msg> : To commit files from staging area to git local repo

git add –a : Add all the newly and modified files can added.

Git rm- -cached : all the file go under unstaging area.

$ git restore --staged B.java is unstaging file from staging area

git push : To move changes from local repo to central repository.

git pull : To take changes from git central repo to local repo

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice

$ git init

Initialized empty Git repository in D:/GIt Practice/.git/

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (master)

$ git config --global user.name

narendra

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (master)

$ git config --global user.email

dasaranarendra99@gmail.com

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (master)

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

demo1.java

nothing added to commit but untracked files present (use "git add" to track)

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (master)

$ git add demo1.java

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: demo1.java

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (master)

$ git commit -m " my first commit"

[master (root-commit) 9c4ee0d] my first commit

1 file changed, 2 insertions(+)

create mode 100644 demo1.java

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (master)

$ git branch -M main

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (main)

$

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (main)

$ git remote add origin https://github.com/narendradasara99/15-JRTP-02-app.git

HP@DESKTOP-6E2LN7K MINGW64 /d/GIt Practice (main)

$ git push -u origin main

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Writing objects: 100% (3/3), 235 bytes | 235.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/narendradasara99/15-JRTP-02-app.git

\* [new branch] main -> main

Branch 'main' set up to track remote branch 'main' from 'origin'.

s

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

20-may

Last session : Git Bash Commands

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

git help

git help <command-name>

git config --global user.name "ashok"

git config --global user.email "ashokitschool@gmail.com"

git init

git status

git add <filename>

git commit -m 'msg'

git remote add origin <repo-url>

git push

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

git clone <repo-url> : To take existing project from git central repo to local system

Staged : The files which are added for commit (will display in green color)

Un-staged : The files which are already available in git repo but modified in local (Will display in red color)

Un-Tracked : The files which are newly created in local (Will diplay in red color)

-> In order to commit un-staged and un-tracked files we have to add those files to staging using git add command.

-> In order to display staged, un-staged and un-tracked files we will use 'git status' command.

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

22-may

Last session : Git Commands

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

git help

git help <command-name>

git config

git init

git add <file-name>

git add --a

git status

git commit -m <msg>

git remoted add

git push

git reset

git clone

git checkout

git log

git rm

git pull

git stash

What is staging?

What is un-staging?

What is un-tracked?

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

Git branches & Pull request (4th video in playlist)

Merge Conflicts (5th video in playlist)

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

Setting up existing project

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

-> When we join in the project, team member or TL will give project repo URL then we have to clone the project and setup workspace.

git clone <repo-url>

git clone -b <branch-name><repo-url>

Setting Up New repository

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

-> When we need to start new project development we need a repository for that.

-> Git Repository will be created by Git Admin.

-> As a developer we need to send request to Git admin team to create Git Repository for our project.

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

To : gitadminteam@ibm.com

cc: manager-email, scrum-master, team-members

Subject : Git Repo Creation Request | IRCTC

Body:

Hi Team,

Please create new git repo for IRCTC project.

Repository Name : IRCTC\_Payments\_Service

Thanks,

Ashok.

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

-> After sending this email, git admin team will create repository and will share repository url in email as a reply.

-> Once repository URL is recieved, we will create project with folder strcture and will push to git repo.

-> Other developers will clone the project from repository and will start development.

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

Repo URL : https://github.com/Ashok-IT-School/15-JRTP-SBApp.git

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

git clone <https://javaproject027am@bitbucket.org/javaproject027am/15-jrtp-03-app.git>­­

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

23-may

Last session : BitBucket

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

-> BitBucket is an Atlasian company product

-> BitBucket providing version control mechanism

-> BitBucket internally uses git a version control software

-> Bitbucket is having rich UI when compared with Git Hub.

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

-> We can use git bash as a client to communicate with BitBucket repositories.

-> In Bitbucket also we can create both public and private repositories.

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

-> We can use Source Tree as a client software to communicate with BitBucket repositories.

-> Source Tree is an open source GUI based application.

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

Interview Questions

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

1) Which version control software is using in your project?

2) Which client software you are using to perform operations in source code repository ?

3) Who will create repository in your project?

4) What are branches in repository and why we need them ?

5) Have you ever faced conflicts ?

6) What is pull request ?

7) What is the difference between push and pull ?

8) What is the difference between clone and pull?

9) How code integration will happen in your project?

10) Have you ever created repo in bitbucket ?

11) Have you ever created branches ?

12) What is branch locking?

13) What is code freeze ?

Maven

Agile with JIRA

Git Hub

BitBucket

Source Tree

Debugging: https://www.youtube.com/watch?v=2WxsClYhreE

31-may

Last Session (23-May-2021) - Version Control Software Completed

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

I have shared videos for log4j and Jenkins

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

Today's session : Application Environments

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

-> Environment is nothing but a platform which is used to run our application.

-> To run one application we need below setup

1) Machine

2) Operating system

3) JEE Server

4) Database Server

-> In Realtime we will have mulitple environments to run our application.

1) Local Env

2) DEV Env

3) SIT Env (QA Env)

4) UAT Env

5) Pilot Env

6) Prod Env

-> Local Env is used for the development

-> DEV env, SIT Env, UAT Env and Pilot Env are used to test our application behaviour.

-> Prod Env is used for 'Go Live'.

Note: The project which is in Prod env is accessible by end users.

1-jun

Last session : Application Environments

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

-> In Realtime every application will have multiple environments like below

1) Local Env

2) Dev Env

3) SIT Env (QA Env)

4) UAT Env

5) Pilot Env

6) Prod Env

-> In Local env we will do the development

-> Dev env will be used by developers to perform Integration testing

-> SIT env will be used by Testing Team to perform System Integration Testing

-> UAT Env will be used by client side team to perform User Acceptance Testing.

-> Pilot env is used to perform Cluster setup. Pilot is also called as Pre-Prod env.

-> Prod env is used to for Live Deployment.

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

-> All these environments setup will be done by DevOps team.

-> DevOps team will create Jobs in Jenkins Software for application deployment.

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

To : irctcdevops@ibm.com

CC : irctc@ibm.com

Subject : Jenkins Jobs Creation Request For IRCTC

Body :

Hi Team,

Please create jenins jobs for IRCTC project for below environments

1) DEV

2) SIT

3) UAT

4) Pilot

5) PROD

BitBucket URL : <repo-url>

Thanks,

Ashok.

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

2-jun

Last session : Application Environments & Deployment Process

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

-> For every project devops team will be available and they are responsible to create environments required for our application.

-> We will send an email for devops team to create jobs for our application deployment.

-> When we need to deploy the application we have to run the job which is created by devops team in jenkins software.

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

-> If our code is not working as expected in local env then we will use Debugging technique to understand the problem and we will fix it.

-> If code is not working in higher environments then we will check log files to identify the problem.

\*\*\*\*\*\* For every project logging will be implemented in realtime\*\*\*\*\*\*\*\*\*

package com.nare;

import java.io.IOException;

import org.apache.log4j.ConsoleAppender;

import org.apache.log4j.FileAppender;

import org.apache.log4j.HTMLLayout;

import org.apache.log4j.Level;

import org.apache.log4j.Logger;

import org.apache.log4j.PatternLayout;

import org.apache.log4j.SimpleLayout;

import org.apache.log4j.xml.XMLLayout;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Test {

static Logger l = Logger.getLogger(Test.class);

static {

//SimpleLayout sl=new SimpleLayout();

// HTMLLayout hl= new HTMLLayout();

//XMLLayout hl=new XMLLayout();

PatternLayout hl=new PatternLayout();

// ConsoleAppender c=new ConsoleAppender(hl);

try {

FileAppender f=new FileAppender(hl,"log.html");

l.addAppender(f);

l.setLevel(Level.DEBUG);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public static void main(String[] args) {

l.debug("debug");

l.info("ifno");

l.warn(" warn messages");

l.fatal("fatal");

l.error("error");

// TODO Auto-generated method stub

ApplicationContext ac = new ClassPathXmlApplicationContext("AppConfig.xml");

Atm a = (Atm) ac.getBean("atm");

a.print("hello factory method is called");

}

}

Log messages we can write by level

DEBUG<INFO<WARN<FATAL<ERROR

1. DEBUG : in debug we write just information about class like main method executed
2. INFO: here we can write connection established like messages we can write
3. WARN: warning messages like this method is deprecated we can write here
4. Fatal : in the try block we can write
5. Error: in catch block we can write

Note: we can configure log properties in properties file

In java class we can write like this

static Logger l = Logger.getLogger(Test.class);

PropertyConfigurator.configure("src\\main\\resources\\log.properties");

l.debug("debug");

l.info("ifno");

l.warn(" warn messages");

l.fatal("fatal");

l.error("error");

log4j.rootLogger=DEBUG,r

log4j.appender.r=org.apache.log4j.FileAppender

log4j.appender.r.file=D:/info.html

log4j.appender.r.layout=org.apache.log4j.HTMLLayout

-> We will use 'Log Monitoring Tools' to check log messages of our application.

1) putty

2) winscp

3) splunk (Trending)

-> Putty is a commandline based software (We have to execute commands to see log messages of our application)

-> WinScp is a gui based application which is used to download log files from remote system to local system.

-> Splunk is an advanced web application which is used for log monitoring.

Maven

GIT Hub

BitBucket

Source Tree

Agile with JIRA

Log4j

Log Monitoring Tool

Jenkins (Ci & Cd)

3-jun

Last session : Log Monitoring Tools

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

-> To identify the problems in application we will use log files

-> To monitor log files we will use log monitoring tools

1) Putty (CLI Based)

2) WinScp (GUI Based)

3) Splunk - ( Web app )(Highly demanding tool)

-> Devops team will provide details of remote mahcine or splunk URL to access logs of our application.

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

Maven : Build Tool (To automate application build process)

GIT Hub : Version Control Software

BitBucket : Version Control Software

Source Tree : Client software for BitBucket

Agile with JIRA : Project Management Software

Log4j : To perform logging in the application

Putty, WinScp, Splunk : For logs monitoring

Jenkins : To automate application deployment process (CI & CD)

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

Part-3 : Mini Projects Development

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

-> For next one month we will do Mini projects implementation

Note: Everybody has to implement code for mini projects in this one month.

-> We will implement these mini projects with Realtime standards.

-> In these 3 mini projects we are going to use tools also.

Note: Daily assignments will be available related to mini projects and assignment checking will be there. The zoom link will be provided only for assignment completed students.

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

What is project?

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

Set of programs we can call as a project

Why we need to develop project?

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

Software projects are used to reduce human efforts

Ex: Netbanking, Ticket Booking, Online Shopping

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

-> Service based companies will take requirements from client and will develop the project based on client requirement.

-> Product based companies will develop the product and will sell that product to public in the market.

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

-> In Every company functional team will be available.

-> Functional team is responsible to collect requirements from the client.

-> Functional team will prepare BRD (Bussiness Requirement Document).

-> Functional team will submit BRD to client for approval.

-> Client will review BRD and will approve BRD.

-> Once BRD got approved, Functional team will prepare FDD (Functional Design Document).

-> Functional team will share FDD to client for review

-> Client will review FDD and will approve the FDD if everything is correct.

-> Once FDD got approved, functional team will share FDD to Developers & Testers.

-> Developers and Testers should read the FDD and should understand requirements clearly.

-> If any developer or tester having doubts in FDD then they should note down their doubts and they should ask functional team for clarifications.

4-jun

Last session : BRD and FDD

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

-> BRD stands for Business Requirement Document

-> BRD will be prepared by functional team

-> BRD document contains highlevel requirements

-> FDD stands for Functional Design Document

-> FDD will be prepared by functional team

-> FDD contains detailed information related to requirements

-> Developers will write the code based on FDD provided information

-> Testers will test the application behaviour based on FDD provided information

-> Once FDD got approved by client, developers and testers will read the fdd and will notedown the queries on FDD.

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

Q-1) Do we need client side validations or not?

Q-2) What is minimum and maximum length for Name field?

Q-3) Phone Number should take how many digits ?

Q-4) Do we need to implement Soft-Delete or Hard-Delete? - Softdelete

Q-5) Can we save duplicate contact ? - No

Q-6) Do we need to implement Pagination or not ? -

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

-> After understanding FDD, developers and testers will get some queries.

-> Developers and testers will write their queries in query log document.

-> Functional team will read query log and will answer queries

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

Today's assignment : Analyze how many classes/interfaces required for 01-mini project.

-> Analyze methods which are required for each class with proper methodname, method parameters and method return type.

ContactEntity.java

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

ContactRepo.java

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

ContactService.java

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

ContactController.java

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

FriendsEntity.java ------Entity class

@GeneratedValue

private Integer id;

private String name;

private String email;

private int phone\_no;

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

FriendsRepo.java ------Interface

public FriendsRepo extends JpaRepository<CreateContactEntity, Serializable>

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

FriendsService.java ------Service Class

public boolean saveContact(contact){

public List<friends> viewContact(){

public void delete(Interger id){

public contact editContact(Interger id)

Steps to create first mini project:

1. Create Spring boot starter project with below dependencies

a) Starter-web

b) starter data jpa

c) H2

d) project Lombok

e) Tomcat Embed japer

f) JSTL

2. Configure below properties in the application.properties file

a) Data source properties

b) View resolver

c) Embedded Server port number

d) ORM properties

3. Create Entity class and Repository Interface

4. Create Service Interface and provide implementation

5. Create a Controller with required methods

6. Create View file(Jsp)

7. Run Application and Test the functionalities

Last session : 01-Mini Project Design & Steps for Development

-------------------------------------------------------------------------Today's session : 01-Mini Project Structure

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

1) Create Spring Boot application with below dependencies

a)spring-boot-starter-web

b)spring-boot-starter-data-jpa

c)h2

d)project-lombok

e)devtools

f)tomcat-embed-jasper

g)jstl

2) Configure below properties in application.properties file

a) datasource

b) ORM properties

c) server port

d) view resolver

-------------------------------------------------------------------------<?xml version="1.0" encoding="UTF-8"?>

<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.5.0</version>

<relativePath /> <!-- lookup parent from repository -->

</parent>

<groupId>in.ashokit</groupId>

<artifactId>01-Contact-App</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<name>01-Contact-App</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-tomcat</artifactId>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<configuration>

<excludes>

<exclude>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

</exclude>

</excludes>

</configuration>

</plugin>

</plugins>

</build>

</project>

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

spring.datasource.driver-class-name=org.h2.Driver

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.username=sa

spring.datasource.password=

server.port=9090

spring.mvc.view.prefix=/views/

spring.mvc.view.suffix=.jsp

server.servlet.context-path=/App1

-------------------------------------------------------------------------package in.ashokit.controller;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

@Controller

public class TestController {

@GetMapping("/welcome")

public String welcome(Model model) {

// setting data in model to send to UI

model.addAttribute("msg", "Welcome to Ashok IT..!!");

// returning logical view name

return "index";

}

}

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

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h1>${msg}</h1>

</body>

</html>

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

Last session : 01-Mini Project-Environment-Setup

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

-> Spring boot provided embedded server to run web applications.

-> When we add 'spring-boot-starter-web' dependeny 'Tomcat' server will come as default embedded container. It runs on 8080 port Number.

-> To change embedded server port number we can use below property in application.properties file

server.port = 9090

-> For temporary purpose we can use H2 database. It is embedded database.

-> If application is running then h2 db will be available and if application stopped then h2 db will not be available.

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

Today's session : Coding for 01-Mini Project

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

Last session : We have implemented below components in application

1) Entity class (Mapping with DB table)

2) Repository interface ( To perform DB Operations )

3) Service Interface (Business methods)

4) Service Implementation class (Business methods implementation)

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

Today's session : Controller classes Implementation

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

-> In Realtime, for every screen one controller will be created. That controller class will contain methods to handle requests coming from that screen.

-> In Our 'Contact Application' we have 2 screens so we will create 2 controllers.

-> In Spring MVC we have @Controller annotation to represent java class as Controller class.

-> Inside controller class we will write methods to handle requests.

Last session : We have implement Save Contact Functionality

Today's session : View Contacts Functionality

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

-> When user click on 'View Contacts' hyperlink we have to retrieve all contact records available in database and we have to display in jsp page in table format.

Last session : We have implement Save Contact Functionality

Today's session : View Contacts Functionality

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

-> When user click on 'View Contacts' hyperlink we have to retrieve all contact records available in database and we have to display in jsp page in table format.

Last session : Pagination Introduction

Today's session : Implementinig Pagination in 01-Mini Project

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

-> The process of dividing records into multiple pages is called as Pagination.

-> To implement pagination we have to decide page size.

-> Page Size represents how many records should be displayed in each page.

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

-> In Spring Data JPA, we have JpaRepository which supports for Pagination.

Page pageObj = PageRequest.of(pageSize, pageNo);

Syntax : repo.findAll(pageObj);

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

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (101, 'Y', 'john@gmail.com', 'John', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (102, 'Y', 'smith@gmail.com', 'Smith', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (103, 'Y', 'Charles@gmail.com', 'charles', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (104, 'Y', 'Ram@gmail.com', 'Ram', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (105, 'Y', 'lakshman@gmail.com', 'Lakshman', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (106, 'Y', 'Bharat@gmail.com', 'Bharat', 79797979);

SELECT \* FROM CONTACT\_DTLS INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (107, 'Y', 'Sita@gmail.com', 'Sita', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (108, 'Y', 'Gita@gmail.com', 'Gita', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (109, 'Y', 'Rani@gmail.com', 'Rani', 79797979);

INSERT INTO CONTACT\_DTLS(CONTACT\_ID, ACTIVE\_SW, CONTACT\_EMAIL, CONTACT\_NAME, CONTACT\_NUMBER)

VALUES (110, 'Y', 'Ching@gmail.com', 'Ching Chong', 79797979);

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

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>

<html>

<head>

<script>

function confirmDelete() {

return confirm("Are you sure, you want to delete?");

}

</script>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h3>View Contacts</h3>

<a href="loadForm">+Add New Contact</a>

<table border="1">

<thead>

<tr>

<th>S.No</th>

<th>Name</th>

<th>Email</th>

<th>Number</th>

<th>Action</th>

</tr>

</thead>

<tbody>

<c:forEach items="${contacts}" var="c" varStatus="count">

<tr>

<td>${count.index+1}</td>

<td>${c.contactName}</td>

<td>${c.contactEmail}</td>

<td>${c.contactNumber}</td>

<td><a href="editContact?cid=${c.contactId}">Edit</a> &nbsp; <a

href="deleteContact?cid=${c.contactId}"

onclick="return confirmDelete()">Delete</a></td>

</tr>

</c:forEach>

</tbody>

</table>

<c:if test="${currPno > 1 }">

<a href="viewContacts?pno=${currPno-1}">Previous</a>

</c:if>

<c:forEach begin="1" end="${tp}" var="pageNo">

<c:choose>

<c:when test="${currPno==pageNo}">

${pageNo}

</c:when>

<c:otherwise>

<a href="viewContacts?pno=${pageNo}">${pageNo}</a>

</c:otherwise>

</c:choose>

</c:forEach>

<c:if test="${currPno < tp }">

<a href="viewContacts?pno=${currPno + 1}">Next</a>

</c:if>

</body>

</html>

Last session : Implemented Pagination in 01-Mini Project

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

-> Pagination is the process of divinding the records into multiple pages.

-> To implement pagination we are using JpaRepository provided

findAll(pageReq) method.

-------------------------------------------------------------------------Today's session : Assignment checking (01-Mini Project)

Last session (16-Jun-21) : 01-Mini Project Implementation

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

17-Jun-21: No class

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

Today's session : Code Review

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

-> The process of verifying source code is called as Code Review

-> Code Review is used to identify mistakes we have done in the code

-> Code Review will be performed in 2 ways

1) By Using Tool

2) Peer Review

-> When we do the code review by using tool, it will check coding standards, duplicate code, NullPointerExceptions etc...

-> Peer means reviewing code manullay (In this process we will identify mistakes in logic).

-> The main intention to perform code review is to provide bug-free and quality code.

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

-> In Realtime we will use SonarQube software to perform Code Review

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

What is Sonar Qube?

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

-> SonarQube is an open source software

-> SonarQube is used to perform code review

-> SonarQube supports 27 programming languages

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

-> We can use Sonar Qube Community Edition for free of cost.

-> Download SonarQube 6.3.1 version from below URL

https://www.sonarqube.org/downloads/

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

-> Download sonar software zip file and extract it

-> Go to sonar folder sonar/bin/windows64/StartSonar.bat

Note: Based your PC os choose folder to start sonar

-> Once sonar server is started it will display 'Sonar is up' message in command promot.

-> Once Sonar is Up and Running then access Sonar Dashboard using below URL

http://localhost:9000/

Last session : Code Review Introduction

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

Today's session : Code Review Using Sonar Qube

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

-> Sonar Qube is an open source software which is used to perform Code Review

-> Sonar Qube supports for 27 programming languages

-> Sonar Qube provided 'Community Edition' to use it for free of cost

Note: We have commericial plans also for using Sonar Qube

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

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 install goal

mvn clean compile package install

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

mvn sonar:sonar

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

-> We can our project report under Projects

-> Report contains below details

1) Bugs(Blocker, Critical, Major & Minor)

2) Vunerabilities

3) Code Smells

4) Code Coverage

5) Duplicates etc...

-> Bugs means the problems available in source code like ArthematicException, NullPointerException, Resource Leak etc (We have to fix them)

-> Vunerabilities nothing but security issues (If we don't fix these hackers can attack). We have to fix them

-> Code Smells are minor issues available in code. Recommneded to fix

(private methods which are not called, un-used imports, un-used method parameters etc)

-> Code Coverage represents how much code we have covered in Unit Testing.

-> Duplicates represents in how many places we have written same logic in project.

Last session : Code Review Using Sonar Qube

-------------------------------------------------------------------------->

To perform code review for project we have to do below 2 things

1) Download and Run Sonar Server

2) Add Sonar-Maven-Plugin in pom.xml and run (goal -> sonar:sonar)

-> When we perform code review using sonar qube it will generate report with below details

1) Bugs

2) Vulnerabilities

3) Code Smells

4) Code Coverage

5) Duplicate Code

-> Bugs means the problems available in source code (we have to fix them)

-> Vulnerabilities are security issues (hackers can attack)

-> Code smells are the issues which are recommended to fix

-> Code Coverage related to Unit testing

-> Duplicate code means the repeated code which is available in project.

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

-> In project if we want to declare variables like password and ssn we should not declare them directley.

-> Instead of declaring password we will declare it as pazzword

-> Instead of declaring ssn we will declare it as zzn

Last session : Code Review Using Sonar Qube and Issues Fixed

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

Today's session : Stop Duplicate Contact Saving & Validations

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

---------------------------------pom.xml---------------------------------

<?xml version="1.0" encoding="UTF-8"?>

<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.5.0</version>

<relativePath /> <!-- lookup parent from repository -->

</parent>

<groupId>in.ashokit</groupId>

<artifactId>01-Contact-App-15-JRTP</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<name>01-Contact-App-15-JRTP</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-validation</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-tomcat</artifactId>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<configuration>

<excludes>

<exclude>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

</exclude>

</excludes>

</configuration>

</plugin>

<plugin>

<groupId>org.sonarsource.scanner.maven</groupId>

<artifactId>sonar-maven-plugin</artifactId>

<version>3.4.0.905</version>

</plugin>

</plugins>

</build>

</project>

------------------------application.properties---------------------------

spring.datasource.driver-class-name=org.h2.Driver

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.show-sql=true

server.port=9090

spring.mvc.view.prefix=/views/

spring.mvc.view.suffix=.jsp

server.servlet.context-path=/App1

-----------------------AppConstants.java-------------------------------

package in.ashokit.constants;

public class AppConstants {

public static final String CONTACT = "contact";

private AppConstants() {

}

}

----------------ContactInfoController.java------------------------------

package in.ashokit.controller;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.validation.Valid;

import org.springframework.data.domain.Page;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.servlet.ModelAndView;

import in.ashokit.constants.AppConstants;

import in.ashokit.entity.Contact;

import in.ashokit.service.ContactService;

@Controller

public class ContactInfoController {

private ContactService contactService;

public ContactInfoController(ContactService contactService) {

this.contactService = contactService;

}

@GetMapping(value = { "/loadForm", "/" })

public String loadForm(Model model) {

Contact cobj = new Contact();

model.addAttribute(AppConstants.CONTACT, cobj);

return AppConstants.CONTACT;

}

@PostMapping("/saveContact")

public String handleSaveBtnClick(@Valid Contact contact, BindingResult result, Model model) {

if (result.hasErrors()) {

return AppConstants.CONTACT;

}

Boolean contactExists = contactService.isContactExists(contact);

if (contactExists) {

model.addAttribute("errMsg", "Duplicate Contact Found");

} else {

Boolean isSaved = contactService.saveContact(contact);

if (isSaved) {

model.addAttribute("succMsg", "Contact Saved Successfully");

} else {

model.addAttribute("errMsg", "Failed To Save Contact");

}

}

return AppConstants.CONTACT;

}

@GetMapping("/viewContacts")

public ModelAndView handleViewAllContactsClick(HttpServletRequest req) {

Integer pageSize = 3;

Integer pageNumber = 1;

String reqParam = req.getParameter("pno");

if (reqParam != null && !"".equals(reqParam)) {

pageNumber = Integer.parseInt(reqParam);

}

Page<Contact> page = contactService.getAllContactsNew(pageNumber - 1, pageSize);

int totalPages = page.getTotalPages();

List<Contact> allContacts = page.getContent();

ModelAndView mav = new ModelAndView();

// Setting data to model in Key-Value pair format

mav.addObject("contacts", allContacts);

mav.addObject("tp", totalPages);

mav.addObject("currPno", pageNumber);

// Setting Logical view name

mav.setViewName("viewContacts");

return mav;

}

}

-------------------------ViewContactsController.java--------------------

package in.ashokit.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.servlet.ModelAndView;

import in.ashokit.entity.Contact;

import in.ashokit.service.ContactService;

@Controller

public class ViewContactsController {

@Autowired

private ContactService contactService;

@GetMapping("/editContact")

public ModelAndView handleEditLinkClick(@RequestParam("cid") Integer contactId) {

ModelAndView mav = new ModelAndView();

Contact cobj = contactService.getContactById(contactId);

mav.addObject("contact", cobj);

mav.setViewName("contact");

return mav;

}

@GetMapping("/deleteContact")

public String handleDeleteLinkClick(@RequestParam("cid") Integer contactId) {

contactService.deleteContactById(contactId);

return "redirect:viewContacts";

}

}

------------------------ContactService.java-----------------------------

package in.ashokit.service;

import java.util.List;

import org.springframework.data.domain.Page;

import in.ashokit.entity.Contact;

public interface ContactService {

public Boolean saveContact(Contact contact);

public List<Contact> getAllContacts();

public Page<Contact> getAllContactsNew(Integer pageNo, Integer pageSize);

public Contact getContactById(Integer contactId);

public Boolean deleteContactById(Integer contactId);

public Boolean isContactExists(Contact contact);

}

-------------------------ContactServiceImpl.java-------------------------

package in.ashokit.service;

import java.util.List;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Example;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.stereotype.Service;

import in.ashokit.entity.Contact;

import in.ashokit.repository.ContactRepository;

@Service

public class ContactServiceImpl implements ContactService {

private ContactRepository contactRepo;

@Autowired

public ContactServiceImpl(ContactRepository contactRepo) {

this.contactRepo = contactRepo;

}

@Override

public Boolean saveContact(Contact contact) {

contact.setActiveSw("Y");

Contact savedEntity = contactRepo.save(contact);

if (savedEntity != null && savedEntity.getContactId() != null) {

return true;

}

return false;

}

@Override

public List<Contact> getAllContacts() {

Contact contactFilter = new Contact();

contactFilter.setActiveSw("Y");

Example<Contact> example = Example.of(contactFilter);

return contactRepo.findAll(example);

}

@Override

public Page<Contact> getAllContactsNew(Integer pageNo, Integer pageSize) {

Contact contactFilter = new Contact();

contactFilter.setActiveSw("Y");

Example<Contact> example = Example.of(contactFilter);

PageRequest pageRequest = PageRequest.of(pageNo, pageSize);

return contactRepo.findAll(example, pageRequest);

}

@Override

public Contact getContactById(Integer contactId) {

Optional<Contact> findById = contactRepo.findById(contactId);

if (findById.isPresent()) {

return findById.get();

}

return null;

}

@Override

public Boolean deleteContactById(Integer contactId) {

Optional<Contact> findById = contactRepo.findById(contactId);

if (findById.isPresent()) {

Contact contact = findById.get();

contact.setActiveSw("N");

contactRepo.save(contact);

return true;

}

return false;

}

@Override

public Boolean isContactExists(Contact contact) {

Example<Contact> example = Example.of(contact);

return contactRepo.exists(example);

}

}

-----------------------Contact.java-------------------------------------

package in.ashokit.entity;

import java.time.LocalDate;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.Id;

import javax.persistence.Table;

import javax.validation.constraints.Email;

import javax.validation.constraints.NotEmpty;

import javax.validation.constraints.NotNull;

import org.hibernate.annotations.CreationTimestamp;

import org.hibernate.annotations.UpdateTimestamp;

@Entity

@Table(name = "CONTACT\_DTLS")

public class Contact {

@Id

@GeneratedValue

@Column(name = "CONTACT\_ID")

private Integer contactId;

@Column(name = "CONTACT\_NAME")

@NotEmpty(message = "Name Is Mandatory")

private String contactName;

@Column(name = "CONTACT\_NUMBER")

@NotNull(message = "Number Is Mandatory")

private Long contactNumber;

@Column(name = "CONTACT\_EMAIL")

@NotEmpty(message = "Email is mandatory")

@Email(message = "Please Enter Valid Email Id")

private String contactEmail;

@Column(name = "ACTIVE\_SW")

private String activeSw;

@CreationTimestamp

@Column(name = "CREATED\_DATE", updatable = false)

private LocalDate createdDate;

@UpdateTimestamp

@Column(name = "UPDATED\_DATE", insertable = false)

private LocalDate updatedDate;

private String password;

private Long ssn;

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public Long getSsn() {

return ssn;

}

public void setSsn(Long ssn) {

this.ssn = ssn;

}

public Integer getContactId() {

return contactId;

}

public void setContactId(Integer contactId) {

this.contactId = contactId;

}

public String getContactName() {

return contactName;

}

public void setContactName(String contactName) {

this.contactName = contactName;

}

public Long getContactNumber() {

return contactNumber;

}

public void setContactNumber(Long contactNumber) {

this.contactNumber = contactNumber;

}

public String getContactEmail() {

return contactEmail;

}

public void setContactEmail(String contactEmail) {

this.contactEmail = contactEmail;

}

public String getActiveSw() {

return activeSw;

}

public void setActiveSw(String activeSw) {

this.activeSw = activeSw;

}

public LocalDate getCreatedDate() {

return createdDate;

}

public void setCreatedDate(LocalDate createdDate) {

this.createdDate = createdDate;

}

public LocalDate getUpdatedDate() {

return updatedDate;

}

public void setUpdatedDate(LocalDate updatedDate) {

this.updatedDate = updatedDate;

}

@Override

public String toString() {

return "Contact [contactId=" + contactId + ", contactName=" + contactName + ", contactNumber=" + contactNumber

+ ", contactEmail=" + contactEmail + ", activeSw=" + activeSw + ", createdDate=" + createdDate

+ ", updatedDate=" + updatedDate + "]";

}

}

-------------------------ContactRepository.java-------------------------

package in.ashokit.repository;

import java.io.Serializable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import in.ashokit.entity.Contact;

@Repository

public interface ContactRepository extends JpaRepository<Contact, Serializable> {

}

----------------------------contact.jsp--------------------------------

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h3>Save Contact Here</h3>

<p>

<font color='green'>${succMsg}</font>

</p>

<p>

<font color='red'>${errMsg}</font>

</p>

<form:form action="saveContact?contactId=${contact.contactId}" modelAttribute="contact" method="POST">

<table>

<tr>

<td>Contact Name</td>

<td><form:input path="contactName" /></td>

<td><form:errors path="contactName"/></td>

</tr>

<tr>

<td>Contact Email</td>

<td><form:input path="contactEmail" /></td>

<td><form:errors path="contactEmail"/></td>

</tr>

<tr>

<td>Contact Number</td>

<td><form:input path="contactNumber" /></td>

<td><form:errors path="contactNumber"/></td>

</tr>

<tr>

<td></td>

<td><input type="submit" value="Save" /></td>

</tr>

</table>

</form:form>

<a href="viewContacts">View All Contacts</a>

</body>

</html>

-------------------------------viewContacts.jsp--------------------------

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>

<html>

<head>

<script>

function confirmDelete() {

return confirm("Are you sure, you want to delete?");

}

</script>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h3>View Contacts</h3>

<a href="loadForm">+Add New Contact</a>

<table border="1">

<thead>

<tr>

<th>S.No</th>

<th>Name</th>

<th>Email</th>

<th>Number</th>

<th>Action</th>

</tr>

</thead>

<tbody>

<c:forEach items="${contacts}" var="c" varStatus="count">

<tr>

<td>${count.index+1}</td>

<td>${c.contactName}</td>

<td>${c.contactEmail}</td>

<td>${c.contactNumber}</td>

<td><a href="editContact?cid=${c.contactId}">Edit</a> &nbsp; <a

href="deleteContact?cid=${c.contactId}"

onclick="return confirmDelete()">Delete</a></td>

</tr>

</c:forEach>

</tbody>

</table>

<c:if test="${currPno > 1 }">

<a href="viewContacts?pno=${currPno-1}">Previous</a>

</c:if>

<c:forEach begin="1" end="${tp}" var="pageNo">

<c:choose>

<c:when test="${currPno==pageNo}">

${pageNo}

</c:when>

<c:otherwise>

<a href="viewContacts?pno=${pageNo}">${pageNo}</a>

</c:otherwise>

</c:choose>

</c:forEach>

<c:if test="${currPno < tp }">

<a href="viewContacts?pno=${currPno + 1}">Next</a>

</c:if>

</body>

</html>

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

Last session : Duplicate Record and Server Side Validations

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

Today's session: 02-Mini-Project

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

-> 02-Mini-Project is all about User Management.

-> We will have below functionalities in 02-Mini-Project

1) User Registration

2) User Registration Email

3) Lock & Unlock Account

4) User Login

5) Forgot Password

Last session : 02-Mini Project Requirements & DB Design

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

Today's session: Project Setup

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

1) Create Spring Boot application with below dependencies

a) web starter

b) data jpa starter

c) oracle / mysql / h2 dependency (any one)

d) mail starter

e) project lombok

f) swagger & swagger-ui

g) actuator starter

h) devtools

2) Configure below properties in application.yml file

a) server port

b) data source properties

c) ORM properties

d) SMTP properties

e) actuator properties

3) Create packages required in application

in.ashokit (basepackage)

in.ashokit.constants

in.ashokit.controller

in.ashokit.service

in.ashokit.entity

in.ashokit.repository

in.ashokit.util

in.ashokit.properties

in.ashokit.config

4) Create Entity Classes For DB tables (4 entity classes required)

Country.java

State.java

City.java

User.java

5) Create Repository interfaces (4 repositories required)

CountryRepo.java

StateRepo.java

CityRepo.java

UserRepo.java

6) Create Form Binding Classes

LoginForm.java

RegForm.java

UnlockAccForm.java

7) Create Service Interface & Impl class

UserService.java (I)

UserServiceImpl.java (C)

8) Create RestControllers

LoginRestController.java

RegRestController.java

UnlockAccRestController.java

ForgotPwdRestController.java

9) Create Constant class

AppConstants.java

10) Create Properties class

AppProps.java

11) Create Utility classes

EmailUtils.java

PwdUtils.java

12) Create Swagger Config class

SwaggerConfig.java

13) Insert data into static tables (country\_master, state\_master & city\_master)

-------------------------------------------------------------------------Today's assignment : Project setup based on above steps (13 steps should be completed)

Tomorrow's assignment: Identify methods, method parameters and return types in Service & Controllers.

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

Last session : 02-Mini-Project Setup and Assignment

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

Today's session : Assignment checking

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

-> I will call names randomley. Share your screen and show me your project setup and your analysis for service layer methods.

Last session : Assignment checking

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

Service layer methods

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

public String loginCheck(LoginForm loginForm)

public Map<Integer, String> getCountries();

public Map<Integer, String> getStates(Integer countryId);

public Map<Integer, String> getCities(Integer stateId);

public String emailCheck (String emailId);

public boolean saveUser(UserForm userForm)

public boolean unlockAccount(UnlockAccForm unloackAccForm);

public boolean forgotPwd(String emailId);

Yesterday (30-Jun-2021) : Class Not Happend

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

In previous session (29-Jun-2021) : Service Interface Methods

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

----------------------------pom.xml------------------------------------

<?xml version="1.0" encoding="UTF-8"?>

<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.3.4.RELEASE</version>

<relativePath /> <!-- lookup parent from repository -->

</parent>

<groupId>com.ashokit</groupId>

<artifactId>05-user-management-web-app</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<name>05-user-management-web-app</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-mail</artifactId>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger2</artifactId>

<version>2.6.1</version>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger-ui</artifactId>

<version>2.6.1</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

<exclusions>

<exclusion>

<groupId>org.junit.vintage</groupId>

<artifactId>junit-vintage-engine</artifactId>

</exclusion>

</exclusions>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

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

server:

port: 9090

spring:

datasource:

username: sa

password: null

url: jdbc:h2:mem:testdb

driver-class-name: org.h2.Driver

jpa:

properties:

hibernate:

show\_sql: true

format\_sql: true

hibernate:

ddl-auto: update

mail:

host: smtp.gmail.com

port: 587

username: javaproject.02.7am@gmail.com

password: 452

properties:

mail:

smtp:

starttls:

enable: true

auth: true

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

Insert Into COUNTRY\_MASTER Values(1,'+91','India');

Insert Into COUNTRY\_MASTER Values(2,'+1','USA');

Insert Into STATES\_MASTER Values(1,1,'Andhra Pradesh');

Insert Into STATES\_MASTER Values(2,1,'Karnataka');

Insert Into STATES\_MASTER Values(3,2,'New Jersy');

insert into STATES\_MASTER values(4,2,'Ohio');

Insert Into CITIES\_MASTER Values(1,'Vizag',1);

Insert Into CITIES\_MASTER Values(2,'Guntur',1);

Insert Into CITIES\_MASTER Values(3,'Banglore',2);

Insert Into CITIES\_MASTER Values(4,'Mysore',2);

Insert Into CITIES\_MASTER Values(6,'Maywood',3);

Insert Into CITIES\_MASTER Values(7,'Westwood',3);

Insert Into CITIES\_MASTER Values(8,'Oakwood',4);

Insert Into CITIES\_MASTER Values(9,'Cuyahoga County',4);

Last session : 2nd Mini Project Setup

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

Today's session : Reading application message from yml file

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

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

server:

port: 9090

spring:

datasource:

username: sa

password: null

url: jdbc:h2:mem:testdb

driver-class-name: org.h2.Driver

jpa:

properties:

hibernate:

show\_sql: true

format\_sql: true

hibernate:

ddl-auto: update

mail:

host: smtp.gmail.com

port: 587

username: javaproject.02.7am@gmail.com

password: 452

properties:

mail:

smtp:

starttls:

enable: true

auth: true

app:

messages:

welcomeMsg: Welcome To Ashok IT..!!

greetMsg: Good Morning...!!

-------------------------AppProperties.java-----------------------------

package com.ashokit.props;

import java.util.HashMap;

import java.util.Map;

import org.springframework.boot.context.properties.ConfigurationProperties;

import org.springframework.boot.context.properties.EnableConfigurationProperties;

import org.springframework.context.annotation.Configuration;

@Configuration

@ConfigurationProperties(prefix = "app")

@EnableConfigurationProperties

public class AppProperties {

private Map<String, String> messages = new HashMap<>();

public Map<String, String> getMessages() {

return messages;

}

public void setMessages(Map<String, String> messages) {

this.messages = messages;

}

}

------------------------AppConstants.java-------------------------------

package com.ashokit.constants;

public class AppConstants {

public static final String WELCOME\_MSG = "welcomeMsg";

public static final String GREET\_MSG = "greetMsg";

}

------------------------------WelcomeRestController.java----------------

package com.ashokit.rest;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

import com.ashokit.constants.AppConstants;

import com.ashokit.props.AppProperties;

@RestController

public class WelcomeRestController {

@Autowired

private AppProperties appProps;

@GetMapping("/welcome")

public String getWelcomeMsg() {

Map<String, String> msgs = appProps.getMessages();

String msg = msgs.get(AppConstants.WELCOME\_MSG);

return msg;

}

@GetMapping("/greet")

public String getGreetMsg() {

Map<String, String> messages = appProps.getMessages();

String greetMsg = messages.get(AppConstants.GREET\_MSG);

return greetMsg;

}

}

Last session : How to read data from YML file into Java class

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

-> It is not at all recommended to hard code application messages in java program.

-> If we want to change any message text then we have to modify our java program.

-> If we modify source code then we have build and deploy the entire application.

-> Building and Deploying entire application is time consuming process.

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

-> To avoid this problem we can store application messages in yml file and we can read from there.

-> If we change any message text in yml, build and deployment is not required just server re-start is sufficient to load latest msgs from yml.

-> Server restart will take 1-2 mins only.

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

Last session : We have started service implementation class

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

Today's session : service implementation coding

Last session : RestController development

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

Today's session : swagger documentation for Rest api

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

-> Swagger is used to generate documentation for REST api

-> Swagger UI is used for rest api testing. It provides user interface for testing.

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

1) Add below swagger dependencies in pom.xml file

2) Create Swagger Config class

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

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger2</artifactId>

<version>2.6.1</version>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger-ui</artifactId>

<version>2.6.1</version>

</dependency>

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

package com.ashokit.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import springfox.documentation.RequestHandler;

import springfox.documentation.builders.ApiInfoBuilder;

import springfox.documentation.builders.PathSelectors;

import springfox.documentation.builders.RequestHandlerSelectors;

import springfox.documentation.service.ApiInfo;

import springfox.documentation.spi.DocumentationType;

import springfox.documentation.spring.web.plugins.Docket;

import springfox.documentation.swagger2.annotations.EnableSwagger2;

@Configuration

@EnableSwagger2

public class SwaggerConfig {

@Bean

public Docket postsApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("public-api")

.apiInfo(apiInfo())

.select()

.apis(RequestHandlerSelectors.basePackage("com.ashokit.rest"))

.paths(PathSelectors.any()).build();

}

private ApiInfo apiInfo() {

return new ApiInfoBuilder().title("Ashok IT API")

.description("Ashok IT API reference for developers")

.termsOfServiceUrl("http://www.ashokit.in")

.contact("ashokitschool@gmail.com")

.license("Ashok IT License")

.licenseUrl("ashokitschool@gmail.com")

.version("1.0")

.build();

}

}

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

Last session : Swagger & Swagger UI Configuration

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

-> Swagger is used to generate documentation for Rest api

-> Swagger UI is used for testing rest api.

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

Today's session : Email Functionality In Spring Boot

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

-> To work with Email functionality we need to configure SMTP properties

Note: Here we will use gmail smtp properties but in company they will provide company SMTP properties.

-> To send emails, spring boot provided 'spring-boot-starter-mail'

-> By adding above dependency we will get 'JavaMailSender'

-> 'JavaMailSender' is used to send emails in boot application.

-> SimpleMessage object is used to send mails with simple text.

-> MimeMessage object is used to send mails with rich text and attachments.

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

package com.ashokit.util;

import javax.mail.internet.MimeMessage;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.mail.javamail.JavaMailSender;

import org.springframework.mail.javamail.MimeMessageHelper;

import org.springframework.stereotype.Component;

@Component

public class EmailUtils {

@Autowired

private JavaMailSender mailSender;

public boolean sendEmail(String to, String subject, String body) {

boolean isSent = false;

MimeMessage mimeMessage = mailSender.createMimeMessage();

try {

MimeMessageHelper mimeMessageHelper = new MimeMessageHelper(mimeMessage, true);

mimeMessageHelper.setSubject(subject);

mimeMessageHelper.setTo(to);

mimeMessageHelper.setText(body);

mailSender.send(mimeMessageHelper.getMimeMessage());

isSent = true;

} catch (Exception e) {

e.printStackTrace();

}

return isSent;

}

}

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

Hi {FNAME}, {LNAME} :

<br/>

Welcome to Ashok IT family, Your registration is almost completed.

Plese use below details to unlock your account.<br/>

<br/>

Temporary Password : {TEMP-PWD}<br/>

<br/>

<a href='http://localhost:9090/unlockAcc?email={EMAIL}'>

Click Here To Unlock Your Account</a> <br/>

<br/>

Thanks & Regards,<br/>

Ashok IT.<br/>

Last session : Email functionality in 02-Project

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

-> Once user registered in our application then our application is responsible to send email to the registered user to unlock his/her account

-> In email body we need to display temporary password and hyperlink to unlock account.

-> Email body content having some static text and some dynamic text.

-> For email body we have created a text file (email body template)

-> In our program we will read that mail body from text file and will replace with dynamic values and will use that for body.

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

-> To read the data from text file in java we have below classes

1) FileReader

2) BufferedReader

-> FileReader class will read the data from file character by character

-> BufferedReader will read data from the file line by line

Note: BufferedReader can't read the data from file directley. We will connect FileReader object with BufferedReader object to read data line by line.

private String readUnlockAccEmailBody(UserAccountEntity entity) {

StringBuffer sb = new StringBuffer(AppConstants.EMPTY\_STR);

String mailBody = AppConstants.EMPTY\_STR;

try {

String fileName = appProps.getMessages().get(AppConstants.UNLOCK\_ACC\_EMAIL\_BODY\_FILE);

FileReader fr = new FileReader(fileName);

BufferedReader br = new BufferedReader(fr);

String line = br.readLine();

while (line != null) {

sb.append(line);

line = br.readLine();

}

br.close();

mailBody = sb.toString();

mailBody = mailBody.replaceAll(AppConstants.FNAME, entity.getFname());

mailBody = mailBody.replaceAll(AppConstants.LNAME, entity.getLname());

mailBody = mailBody.replaceAll(AppConstants.TEMP\_PWD, entity.getPazzword());

mailBody = mailBody.replaceAll(AppConstants.EMAIL, entity.getEmail());

} catch (Exception e) {

e.printStackTrace();

}

return mailBody;

}

Last session : Doubts clarifications

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

-> Most of the people are doing mistakes in application.yml configuration

-> Some people done mistakes in readingMailBody( ) method.

-> Some People done mistakes in Entity classes & Repositories interfaces

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

Login Functionality

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

---------------------------Forgot pwd body template file--------------

Hi {FNAME}, {LNAME} :

<br/>

<br/>

Welcome to Ashok IT, Please find your password below<br/>

<br/>

Password : {PWD}<br/>

<br/>

Thanks & Regards,<br/>

Ashok IT.<br/>

Last session : 02-Project-Completion

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

Today's session : En-coding & De-Coding | En-cryption & De-Cryption

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

-> In our application we are storing password given by user as it is

-> If we store password like that then it will be visible for the people who have access for our Database.

-> In Security standpoint, we should not store passwords directley like that.

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

-> If we are dealing with Passwords in our application then we have to encrypt and decrypt the passwords.

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

Encoding & Decoding

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

-> Encoding and Decoding techniques will be used to transmit the data from one place to another place or from one application to another application.

-> To work with Encoding and Decoding we have java.util.Base64 class

-

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

package in.ashokit.beans;

import java.util.Base64;

import java.util.Base64.Decoder;

import java.util.Base64.Encoder;

public class EncodeDecodeUtils {

public static String encode(String text) {

Encoder encoder = Base64.getEncoder();

String encodedString = encoder.encodeToString(text.getBytes());

return encodedString;

}

public static String decode(String encodedStr) {

Decoder decoder = Base64.getDecoder();

byte[] decode = decoder.decode(encodedStr);

return new String(decode); // converting byte[] to string

}

public static void main(String[] args) {

String encodedStr = encode("ashokit@123");

System.out.println("Encoded String :: " + encodedStr);

String decodedStr = decode(encodedStr);

System.out.println("Decoded String :: " + decodedStr);

}

}

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

Encryption & De-Cryption

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

-> Encryption is the process of converting data from readable format to un-readable format.

-> To covert encrypted data into orginal format we will use Decryption.

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

package in.ashokit.beans;

import java.security.MessageDigest;

import java.util.Base64;

import java.util.Base64.Encoder;

public class Test {

public static void main(String[] args) throws Exception {

String pwd = "abc@123";

MessageDigest msgDigest = MessageDigest.getInstance("SHA-1");

msgDigest.reset(); //clearing data

msgDigest.update(pwd.getBytes()); //setting the data to encrypt

byte[] digest = msgDigest.digest();

String digestedPwd = new String(digest);

System.out.println("Digested Pwd :: " + digestedPwd);

Encoder encoder = Base64.getEncoder();

String encodeToString = encoder.encodeToString(digest);

System.out.println("Digested + Encoded Val :: " + encodeToString);

}

}

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

Note: MessageDigest will support only for Encryption it is not supporting for Decryption.

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

package in.ashokit.beans;

import java.util.Base64;

import javax.crypto.Cipher;

import javax.crypto.KeyGenerator;

import javax.crypto.SecretKey;

public class EncryptionDecryptionAES {

static Cipher cipher;

public static void main(String[] args) throws Exception {

/\*

\* create key If we need to generate a new key use a KeyGenerator If we have

\* existing plaintext key use a SecretKeyFactory

\*/

KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");

keyGenerator.init(128); // block size is 128bits

SecretKey secretKey = keyGenerator.generateKey();

/\*

\* Cipher Info Algorithm : for the encryption of electronic data mode of

\* operation : to avoid repeated blocks encrypt to the same values. padding:

\* ensuring messages are the proper length necessary for certain ciphers

\* mode/padding are not used with stream cyphers.

\*/

cipher = Cipher.getInstance("AES"); // SunJCE provider AES algorithm, mode(optional) and padding

// schema(optional)

String plainText = "ashokit@123";

System.out.println("Plain Text Before Encryption: " + plainText);

String encryptedText = encrypt(plainText, secretKey);

System.out.println("Encrypted Text After Encryption: " + encryptedText);

String decryptedText = decrypt(encryptedText, secretKey);

System.out.println("Decrypted Text After Decryption: " + decryptedText);

}

public static String encrypt(String plainText, SecretKey secretKey) throws Exception {

byte[] plainTextByte = plainText.getBytes();

cipher.init(Cipher.ENCRYPT\_MODE, secretKey);

byte[] encryptedByte = cipher.doFinal(plainTextByte);

Base64.Encoder encoder = Base64.getEncoder();

String encryptedText = encoder.encodeToString(encryptedByte);

return encryptedText;

}

public static String decrypt(String encryptedText, SecretKey secretKey) throws Exception {

Base64.Decoder decoder = Base64.getDecoder();

byte[] encryptedTextByte = decoder.decode(encryptedText);

cipher.init(Cipher.DECRYPT\_MODE, secretKey);

byte[] decryptedByte = cipher.doFinal(encryptedTextByte);

String decryptedText = new String(decryptedByte);

return decryptedText;

}

}

Last session : Encoding, Decoding, Encryption & Decryption

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

-> The process of converting data from readable format to un-readable format is called as Encryption.

-> The process of converting data from unreadable format to readable format is called as Decryption.

-> If we are dealing with sensitive data (pin , pwd , ssn, aadhar ) in applications then we should use Encryption and Decryption techniques.

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

-> Encoding is used to convert data into compatiable characters.

-> To convert encoded data back to original format we will use Decoding.

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

package com.ashokit.util;

import java.util.Base64;

import javax.crypto.Cipher;

import javax.crypto.KeyGenerator;

import javax.crypto.SecretKey;

public class PwdUtils {

private static KeyGenerator keyGenerator = null;

static SecretKey secretKey = null;

static Cipher cipher;

public static String encrypt(String plainText) {

byte[] plainTextByte = plainText.getBytes();

String encryptedText = null;

try {

keyGenerator = KeyGenerator.getInstance("AES");

keyGenerator.init(128); // block size is 128bits

secretKey = keyGenerator.generateKey();

cipher = Cipher.getInstance("AES");

cipher.init(Cipher.ENCRYPT\_MODE, secretKey);

byte[] encryptedByte = cipher.doFinal(plainTextByte);

Base64.Encoder encoder = Base64.getEncoder();

encryptedText = encoder.encodeToString(encryptedByte);

} catch (Exception e) {

e.printStackTrace();

}

return encryptedText;

}

public static String decrypt(String encryptedText) {

Base64.Decoder decoder = Base64.getDecoder();

String decryptedText = null;

try {

keyGenerator = KeyGenerator.getInstance("AES");

keyGenerator.init(128); // block size is 128bits

secretKey = keyGenerator.generateKey();

cipher = Cipher.getInstance("AES");

byte[] encryptedTextByte = decoder.decode(encryptedText);

cipher.init(Cipher.DECRYPT\_MODE, secretKey);

byte[] decryptedByte = cipher.doFinal(encryptedTextByte);

decryptedText = new String(decryptedByte);

} catch (Exception e) {

e.printStackTrace();

}

return decryptedText;

}

}

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

02-Mini Project Coding Completed and we have done very basic testing using Swagger. We have below tasks pending in 02-Mini Project

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

1) Exception Handling

2) Server Side Validation

3) Logging

4) Unit Testing Using (Junit with Mockito)

5) Code Coverage (Minimum 85 %)

6) Code Review

7) Performance Testing

8) Deploy application using Jenkins

Last session : Password En-cryption & De-Cryption

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

-> When we are dealing with sensitive data in our application then we will use Encryption and decryption techniques.

-> The process of converting data from readable format to unreadable format is called as Encryption

-> The process of converting data from un-readable format to readable format is called as De-Cryption.

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

In our application we are using pwd encryption & decryption at below places

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

-> When new user registered, our application will generate one temp-pwd. We are encrypting that temp-pwd and storing into db.

Note: User will recieve email with temp-pwd to unlock his/her account.

-> When user trying to unlock his/her account we will recieve temp-pwd user but in db temp-pwd available in encrypted mode.

-> To unlock account, our code should encrypt the temp-pwd given by user then verify in db with email and encrypted temp-pwd a record is present or not.

-> At the time of unlocking account we will encrypt new pwd given by user then we will update that in db.

-> At the time of login, user will provide original pwd but in db user pwd stored with encryption.

-> To check login functionality we need to encrypt the pwd given by user then verify with given email and encrypted pwd a record present in db or not.

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

package com.ashokit.util;

import java.util.Base64;

import javax.crypto.Cipher;

import javax.crypto.spec.IvParameterSpec;

import javax.crypto.spec.SecretKeySpec;

public class PwdUtils {

private static final String SECRET\_KEY = "abc123xyz123abcd";

private static final String INIT\_VECTOR = "abc123xyz123abcd";

public static String encryptMsg(String msg) throws Exception {

IvParameterSpec ivParamSpec = new IvParameterSpec(INIT\_VECTOR.getBytes());

SecretKeySpec secretKeySpec = new SecretKeySpec(SECRET\_KEY.getBytes("UTF-8"), "AES");

Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");

cipher.init(Cipher.ENCRYPT\_MODE, secretKeySpec, ivParamSpec);

byte[] encrypted = cipher.doFinal(msg.getBytes());

return Base64.getEncoder().encodeToString(encrypted);

}

public static String decryptMsg(String msg) throws Exception {

IvParameterSpec ivParamSpec = new IvParameterSpec(INIT\_VECTOR.getBytes());

SecretKeySpec secretKeySpec = new SecretKeySpec(SECRET\_KEY.getBytes("UTF-8"), "AES");

Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");

cipher.init(Cipher.DECRYPT\_MODE, secretKeySpec, ivParamSpec);

byte[] decodedMsg = Base64.getDecoder().decode(msg);

byte[] decrypted = cipher.doFinal(decodedMsg);

return new String(decrypted);

}

}

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

Last session : Password Encryption & Decryption Implementation In Project

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

Today's session : Exception Handling In REST API

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

-> We can handle exceptions in 2 ways

1) Global Exception Handling Mechanism

2) Controller Based Exception Handling Mechanism

-> Global Exception Handling means it will be applicable for entire application.

-> Controller Based Exception Handling means it will handle the exceptions occured in that particular controller.

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

-> When we are doing any experiment or when we are implementing any functionality for first time don't directley implement in project.

-> First develop a POC (Proof Of Concept) and show that POC to higher management (Lead / Manager).

-> Once your POC got approved by Lead or Manager then you can implement that in project.

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

Last session : How HTTP Post request working internally

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

1) DispatcherServlet process

2) HandlerMapper

3) HttpMsgBodyReader

4) HttpMsgBodyWriter

5) MessageConverters

6) consumes

7) produces

8) Content-Type

9) Accept

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

-> We understood What is HTTP GET request ?

-> We understood what is HTTP POST request ?

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

-> If rest api method is responsible to provide data to client then we will bind our method to HTTP GET request using @GetMapping annotation.

-> HTTP GET request will not have any request body

-> If client wants to send some data to rest api GET request method then client will use Query Parameters & Path Parameters

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

-> If REST api method is responsible to create a new record/resource at server then we will bind our method to HTTP POST request using @PostMapping annotation.

-> HTTP Post request will have request body

-> Client can send data to HTTP POST Request method using Request Body.

-> HTTP Request Methods will return status code as "201 Created"

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

-> By default DispatcherServlet will decide HTTP Status code and will send to client.

-> If we want to customize Http status code for our method we can use ResponseEntity.

Syntax : ResponseEntity(T body, HttpStatus status)

Ex: new ResponseEnity(ticketResObj, HttpStatus.CREATED);

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

HTTP PUT Request

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

-> If our REST API method is responsible to update an existing record then we will bind our method to HTTP PUT Request Method.

-> HTTP PUT Request method contains Request body.

@PutMapping

public ResponseEntity<String> updateTicket(@RequestBody Ticket request){

//logic to update the ticket

return new ResponseEntity<>(body, HttpStatus.OK);

}

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

HTTP DELETE Request

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

-> If our REST API method is responsible for deleting a record then we will bind that method to HTTP DELETE Request using @DeleteMapping annotation.

-> HTTP DELETE request contains Request Body

@DeleteMapping("/delete/{empId}")

public String deleteEmp(@PathVariable Integer empId){

//logic to delete record

return "Record Deleted";

}

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

Last session : Handling User-Defined Exceptions

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

-> As per our requirement we can create our own Exceptions in the application.

public class NoDataFoundException extends Exception{

public NoDataFoundException(){ }

public NoDataFoundException(String msg){

super(msg);

}

}

-> To handle exception in our application, we will create Global Exception Handler Class using @RestControllerAdvice annotation.

-> To map exception to particular method we will use @ExceptionHandler annotation.

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

Implement Logging In Application

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

1) What is Logging?

2) Why we need logging?

3) Logger, Layout and Appender

4) Log Levels

5) Logging methods

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

-> Logging is the process of storing application execution details.

-> When our application running in local system we can use debugging to understand execution flow and we can identify bugs available in source code.

-> When application is executing in higher environment like DEV, QA, UAT, PILOT and PROD we can't debug the code to understand the flow. To understand application execution flow we will use log messages.

-> Logger is a class which provided methods to perform logging in our application.

-> Layout will decide format of log msg

1) SimpleLayout

2) PatternLayout

-> Appender will decide destination for storing log message

1) ConsoleAppender

2) FileAppender

4) JdbcAppender

5) MailAppender etc...

-> Logging is having some levels those levels will decide when to print that log msg.

1)trace

2)debug

3)info

4)warn

5)error

6)fatal

7)off

-> For every log level Logger class provided method.

20-Jul-2021 Session : What is Logging & Why Logging

21-Jul-2021 Session : No Session

22-Jul-2021 Session : How to implement Logging in application

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

Logger

Layout

Appender

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

Log Levels

trace

debug

info

warn

error

fatal

Note: When we set log level, it will print from that level to all higher level msgs.

TRACE > DEBUG > INFO > WARN > ERROR > FATAL

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

-> In realtime projects we will use logback to implement logging in our application.

-> If logback.xml file available in clas path then Spring Boot will give first priority to logback

-> If logback not available then it will execute default spring boot logging.

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

-> If we store log msgs to single log file then file size will become very heavy.

-> It would be difficult for the developers also if one log file having all messages.

-> To overcome this problem we will use below types of appenders

1) Time Based Rolling File Appender

2) Size Based Rolling File Appender

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

logging.file.name=MyApp.log

logging.pattern.rolling-file-name=MyApp-%d{yyyy-MM-dd}.%i.log

logging.file.max-size=1MB

logging.file.total-size-cap=10MB

logging.file.max-history=30

logging.file.clean-history-on-start=true

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

Last session : Logging Implementation using Logback

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

-> Logback is the default option for logging in Spring Boot

-> In realtime project we will configure FileAppender in 2 ways

1) TimeBasedRolling

2) SizeBasedRolling

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

Log Monitoring

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

-> The process of checking log messages is called as Log Monitoring

-> We will use below softwares to perform log Monitoring

1) Putty

2) WinScp

3) Splunk

Last Session : What is Log Monitoring & How to do it

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

Today's session : Code Review In 2nd Project

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

1) Run Sonar Qube Server

2) Add Sonar Qube plugin in project pom.xml file

3) Build Our Project using Maven Goals

4) Run application with sonar goal

sonar:sonar

5) Open Sonar dashboard and check the issues

1) Don't use Generic Exception. Create a dedicated exception and use it.

2) Instread of using printStackTrace() log or re-throw the exception

3) When class doesn't have instance method declare private constructor

4) Application should not contain duplicate code

5) Use try with resources for closing files

6) String literals should be declared in constants file

7) Remove un-necessary curley braces when we have single stmt in that

8) Constructor should be declared after variables

9) Unused import stmts should be removed

10) Commented code should be removed

11) Instead of StringBuffer use StringBuilder

12) Return variable directley from the method when it is not used

13) Don't use deprecated methods

Last session : We have noted down lessons learnt in coding

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

Today's session : 03-Mini Project

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

-> In this 3rd minit project we will learn below things

1) How to implement dynamic search operations

2) How to implement Server Side Pagination

3) How to generate reports in Excel and Pdf formats

//category selected

select \* from products where category='Mobiles & Accessories';

//category & brand selected

select \* from products where category = 'Mobiles & Accessories' and brand='SAMSUNG';

//category , brand, budget selected

select \* from products where category = 'Mobiles & Accessories' and brand='SAMSUNG' and price <=30000;

//category, brand, budget, RAM selected

select \* from products where category = 'Mobiles & Accessories' and brand='SAMSUNG' and price <=30000 and RAM = 4;

Last session : 03-Mini Project Requirement

-------------------------------------------------------------------------->

-> Dynamic Search Implementation

-> Server Side Pagination

-> Reports Generation In Excel & Pdf formats

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

MOBILES

--------

MOBILE\_ID NUMBER

BRAND\_NAME VARCHAR2(100)

MOBILE\_PRICE NUMBER

MOBILE\_RATING NUMBER

MOBILE\_RAM NUMBER

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (101, 'SAMSUNG', 20000, 4, 4)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (102, 'SAMSUNG', 25000, 5, 6)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (103, 'SAMSUNG', 40000, 4, 8)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (104, 'APPLE', 50000, 4, 4)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (105, 'APPLE', 55000, 4, 6)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (106, 'APPLE', 70000, 4, 8)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (107, 'REDMI', 10000, 4, 4)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (108, 'REDMI', 15000, 4, 6)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (109, 'REDMI', 20000, 4, 8)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (110, 'OPPO', 11000, 4, 4)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (111, 'OPPO', 14000, 4, 6)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (112, 'OPPO', 18000, 4, 8)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (113, 'REALME', 12000, 4, 4)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (114, 'REALME', 15000, 4, 6)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (115, 'REALME', 19000, 4, 8)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (116, 'MOTOROLA', 12000, 4, 4)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (117, 'MOTOROLA', 15000, 4, 6)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (118, 'MOTOROLA', 19000, 4, 8)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (119, 'NOKIA', 12000, 4, 4)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (120, 'NOKIA', 15000, 4, 6)

INSERT INTO MOBILES(MOBILE\_ID, BRAND\_NAME, MOBILE\_PRICE, MOBILE\_RATING, MOBILE\_RAM) VALUES (121, 'NOKIA', 19000, 4, 8)

Last session: How to send headers in request & exchange method

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

-> To set headers to the request we will use HttpHeaders class

-> To add Headers to the request we will uses HttpEntity class

HttpHeaders headers = new HttpHeaders();

headers.set(key1,value1);

headers.set(key2,value2);

HttpEntity entity = new HttpEntity(body,headers);

rt.exchange(Url, Method, entity, type);

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

-> Now a days people are not using RestTemplate bcz Spring itself saying to use WebClient

-> RestTemplate introduced in Spring 3.0 and WebClient introduced in Spring 5.0

-> RestTemplate supports only synchronus communication where as WebClient supports both Sync and Async communication.

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

Q) What is Synchronus Communication?

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

-> After making the request if our thread is waiting for the response to continue its execution then it is called as Sync communication.

Q) What is Asynchronus Communication?

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

-> After making the request if our thread will continue its execution without waiting for the response then it is called as Aync communication.

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

-> RestTemplate class is part of spring-boot-starter-web dependency where as WebClient is part of spring-boot-starter-webflux dependency

Last Session : Server Side Pagination & Client Side Pagination

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

Today's session : Reports Generation Using Java

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

-> In applications we can see below 2 types of reports

1) Excel Reports

2) Pdf Reports

-> To generate Excel report using java, we have third party api called Apache POI.

-> Using Apache POI api we can write the data to excel and we can read the data from Excel.

-> To generate PDF report using java, we have third party api called IText api.

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

Working with Apache POI api

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

Workbook

Document

Sheet

Row

Cell

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

<dependencies>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi</artifactId>

<version>3.15</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-ooxml</artifactId>

<version>3.15</version>

</dependency>

</dependencies>

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

package in.ashokit.beans;

import java.io.File;

import java.io.FileOutputStream;

import java.util.ArrayList;

import java.util.List;

import org.apache.poi.ss.usermodel.Cell;

import org.apache.poi.ss.usermodel.Row;

import org.apache.poi.ss.usermodel.Sheet;

import org.apache.poi.ss.usermodel.Workbook;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class ExcelWriter {

public static void main(String[] args) throws Exception {

Emp e1 = new Emp("101", "Nithin", "15000.00");

Emp e2 = new Emp("102", "Steve", "25000.00");

Emp e3 = new Emp("103", "John", "35000.00");

Emp e4 = new Emp("104", "Somu", "45000.00");

Emp e5 = new Emp("105", "Ramesh", "55000.00");

Emp e6 = new Emp("106", "Butler", "65000.00");

List<Emp> empList = new ArrayList<>();

empList.add(e1);

empList.add(e2);

empList.add(e3);

empList.add(e4);

empList.add(e5);

empList.add(e6);

ExcelWriter writer = new ExcelWriter();

writer.write(empList);

}

public void write(List<Emp> empList) throws Exception {

Workbook workbook = new XSSFWorkbook();

Sheet sheet = workbook.createSheet("Emps");

Row headerRow = sheet.createRow(0);

Cell headerCell0 = headerRow.createCell(0);

Cell headerCell1 = headerRow.createCell(1);

Cell headerCell2 = headerRow.createCell(2);

headerCell0.setCellValue("Emp Id");

headerCell1.setCellValue("Emp Name");

headerCell2.setCellValue("Emp Salary");

for (int i = 0; i < empList.size(); i++) {

Row dataRow = sheet.createRow(i + 1);

Cell dataCell0 = dataRow.createCell(0);

Cell dataCell1 = dataRow.createCell(1);

Cell dataCell2 = dataRow.createCell(2);

Emp emp = empList.get(i);

dataCell0.setCellValue(emp.getEmpId());

dataCell1.setCellValue(emp.getEmpName());

dataCell2.setCellValue(emp.getEmpSalary());

}

FileOutputStream fos = new FileOutputStream(new File("Emps.xlsx"));

workbook.write(fos);

workbook.close();

}

}

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

25 horses

5 tracks are available

how to find fastest 5 horses with minimum races (5 races)

how to find which is fastest in all 25 ---> (5 + 1 races)

9 balls are there all are same color same shape but one ball is less in wait. i will give weighing machine... with in how many tries u can find less weight ball

Last Session : Apache POI API

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

Today's session : ITEXT API

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

package in.ashokit.beans;

public class Course {

private String courseName;

private String courseTiming;

public Course() {

// TODO Auto-generated constructor stub

}

public Course(String courseName, String courseTiming) {

super();

this.courseName = courseName;

this.courseTiming = courseTiming;

}

@Override

public String toString() {

return "Course [courseName=" + courseName + ", courseTiming=" + courseTiming + "]";

}

public String getCourseName() {

return courseName;

}

public void setCourseName(String courseName) {

this.courseName = courseName;

}

public String getCourseTiming() {

return courseTiming;

}

public void setCourseTiming(String courseTiming) {

this.courseTiming = courseTiming;

}

}

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

package in.ashokit.beans;

import java.util.ArrayList;

import java.util.List;

public class CourseService {

public List<Course> getCourses() {

Course c1 = new Course("10-SBMS", "6:30 AM IST");

Course c2 = new Course("09-SBMS", "7:45 AM IST");

Course c3 = new Course("15-JRTP", "10:15 AM IST");

Course c4 = new Course("16-JRTP", "11:45 AM IST");

List<Course> courses = new ArrayList<>();

courses.add(c1);

courses.add(c2);

courses.add(c3);

courses.add(c4);

return courses;

}

}

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

package in.ashokit.beans;

import java.io.File;

import java.io.FileOutputStream;

import java.io.OutputStream;

import java.util.Date;

import java.util.List;

import com.itextpdf.text.Document;

import com.itextpdf.text.Paragraph;

import com.itextpdf.text.pdf.PdfPTable;

import com.itextpdf.text.pdf.PdfWriter;

public class PdfReportGenerator {

public static void main(String args[]) {

try (OutputStream file = new FileOutputStream(new File("Contacts.pdf"))) {

// Create a new Document object

Document document = new Document();

// You need PdfWriter to generate PDF document

PdfWriter.getInstance(document, file);

// Opening document for writing PDF

document.open();

// Writing content

document.add(new Paragraph("Hello World, Creating PDF documents in Java is very easy"));

document.add(new Paragraph("You are customer # 2345433"));

document.add(new Paragraph(new Date(new java.util.Date().getTime()).toString()));

document.add(new Paragraph(" "));

CourseService service = new CourseService();

List<Course> courses = service.getCourses();

PdfPTable table = new PdfPTable(2);

table.addCell("Course Name");

table.addCell("Timings");

courses.forEach(course -> {

table.addCell(course.getCourseName());

table.addCell(course.getCourseTiming());

});

document.add(table);

// Add meta data information to PDF file

document.addCreationDate();

document.addAuthor("Ashok IT");

document.addTitle("How to create PDF document in Java");

document.addCreator("Thanks to iText, writing into PDF is easy");

// close the document

document.close();

System.out.println("Your PDF File is succesfully created");

} catch (Exception e) {

e.printStackTrace();

}

}

}

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

Last Session (03-Aug-2021) : Excel & PDF files creation

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

04-Aug-2021: No class

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

Today's session (05-Aug-2021): Major Project

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

Part-1: Industry Details

Part-2: Realtime Tools

Part-3: 03-Mini Project

-> 08-Aug-2021 we have Junit-5 continution workshop for REST api unit testing + code coverage using Jacoco

-> REST API Performance Testing using JMETER

-> Front and Backend Applications Integration

-> Application Deployment Using Jenkins

-> Docker Image creation

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

Major Project

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

Name : IES (Integrated Eligibility System)

Client : US State Govt (Rhode Island)

Domain : Health & Insurance

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

-> INDIA

- Central Govt (Prime Minister --> Modi)

- State Govts (Chief Minister, Governer)

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

-> USA

- Federal Govt

- State Govts

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

1) If you are having low or no income, US govt will provide Food Benefits

2) If you are having low & limited resource, US govt will take care of your childs

3) US govt will provide free Medical insurance (Free Hospitality)

4) For Unemployees, US govt will provide free trainings, monthly expenses during training, job opportunities

Last Session : Major Project Introduction

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

Today's session: Major Project Details

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

-> Our Major project name is IES

-> IES stands for Integrated Eligibility System

-> This project is developed for USA Rhode Island State Govt.

-> Rhode Island State Govt is using this project to provide health and insurance plans for Rhode Island State Citizens.

-> RI Govt is providing health & insurance plans for Rhode Island State Citizens based on some conditions (income, age, properties etc)

-> Below are the health and insurance plans offering by IES for RI state citizens

1) SNAP

2) CCAP

3) Medicaid

4) Medicare

5) RIW

6) QHP

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

-> SNAP (Supplemental Nutrition Assistence Program) is related food plan

-> CCAP (Child Care Assistance Program) is realted to Kids

-> Medicaid and Medicare plans are related Health Insurance

-> RIW (Rhode Islands Works) is realted to Un-employeed people skills improvement plan

-> QHP (Quality Health Plan) is commercial health plan with some discounts

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

-> IES application is an Intranet Application.

-> IES application is accessible only with in DHS offices

-> DHS ofc employees only can access IES application.

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

-> RI citizens should visit DHS ofc to apply for a Plan in IES.

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

Last session : IES project details & Plans

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

Today's session : IES architecture & IES Modules

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

Admin

AR (Application Registration)

DC (Data Collection)

ED (Eligibility Determination)

CO (Correspondence)

BI (Benefit Issuance)

Batches

Reports

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

-> Admin module contains below functionalities

1) Users (Case Workers & Admins) Management

2) Plan Management

-> Application Registration (AR) is entry point for citizen to apply for a plan. AR module will interact with SSA-WEB (Federal Govt) project to verify citizen identity

-> If citizen is not belongs to RI state, he can't apply for a plan in IES.

-> If citizen is belongs to RI state then Case worker will collect citizen data to apply for plan.

-> DC (data collection) module is used to collect data from the citizen to apply for a plan.

-> ED (Eligibility Determination) module is used to verify Citizen eligibility for the plan. ED module will communicate with ED-RULES-API.

-> CO (Correspondence) module is used generate & send notices to citizens regarding their eligibility. It will send notices to citizens thourgh post (pdf file).

-> BI (Benefit Issuance) module is used to send benefit amount for approved citizens bank accounts.