Hibernate

Day28 Revisit –

1. Devops CI/CD [Continuous Integration / Continuous Deployment & Delivery]
2. Jenkins [Installing & Configuring, accessing]
3. SonarQube, Sonar cloud & Sonar Lint – Code Quality Analysing Tool

Jenkins is a CI/CD Tool (DevOps Tool) – Build Automation Tool

Jenkins is a Open Source Tool.

Jenkins can be integrated with lot of plugins.

Jenkins is suitable for Projects of various size.

SonarQube – Stand-alone Code Quality Analysing Tool

Sonar Cloud – Cloud version of SonarQube

Sonar Lint – Is a Eclipse Plugin of SonarQube

SonarQube Requirements

1. JDK11

SonarQube is Code Quality Analysing Tool for all the programming Languages

1. JAVA
2. .Net
3. JavaScript
4. Python
5. Php …..

Editions of SonarQube

1. Personal/Community Edition
2. Developer Edition
3. Enterprise
4. Data Center -- Cloud Service

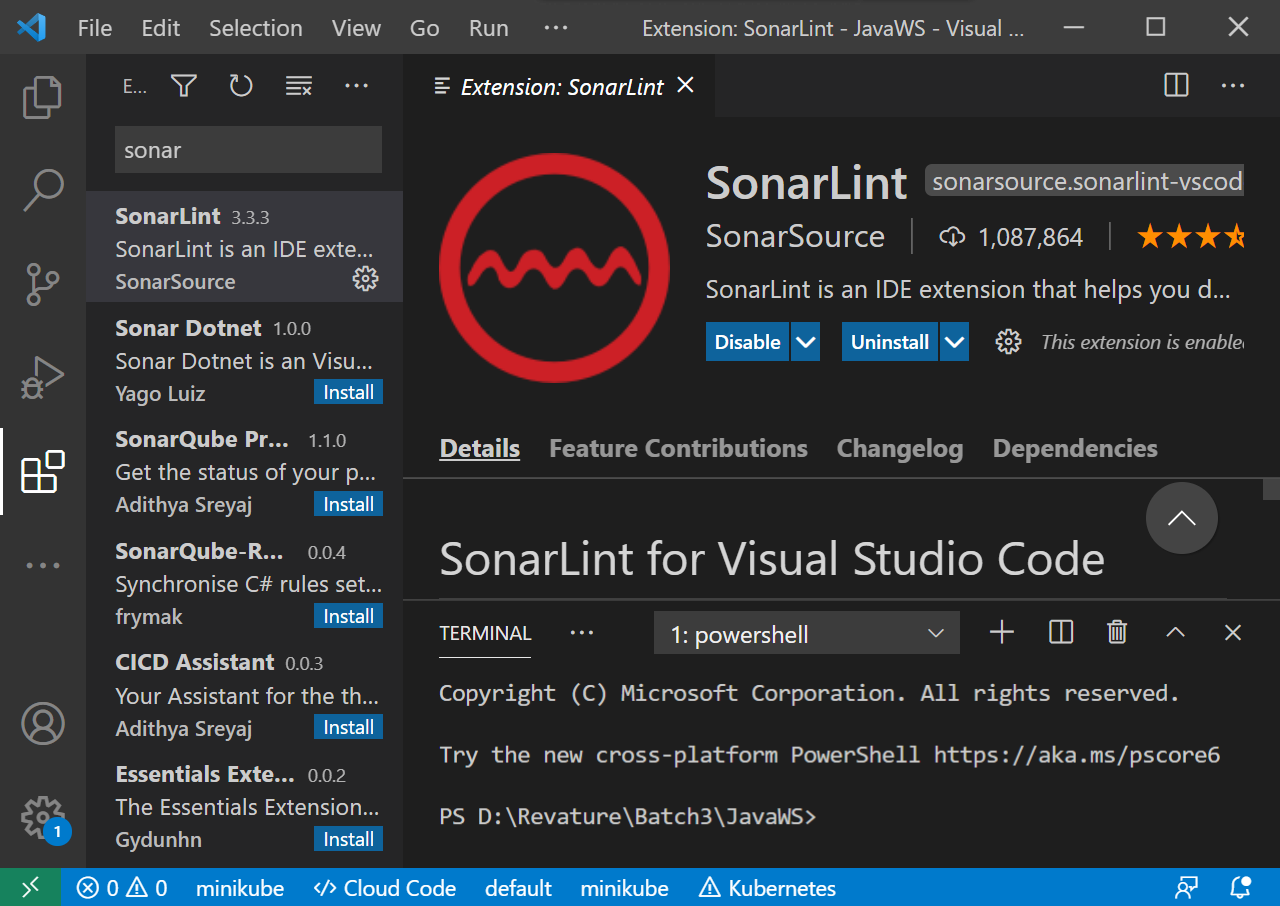
Default port number for sonarqube server is localhost:9000

Default username/password : admin/admin

<https://docs.sonarqube.org/latest/setup/get-started-2-minutes/>

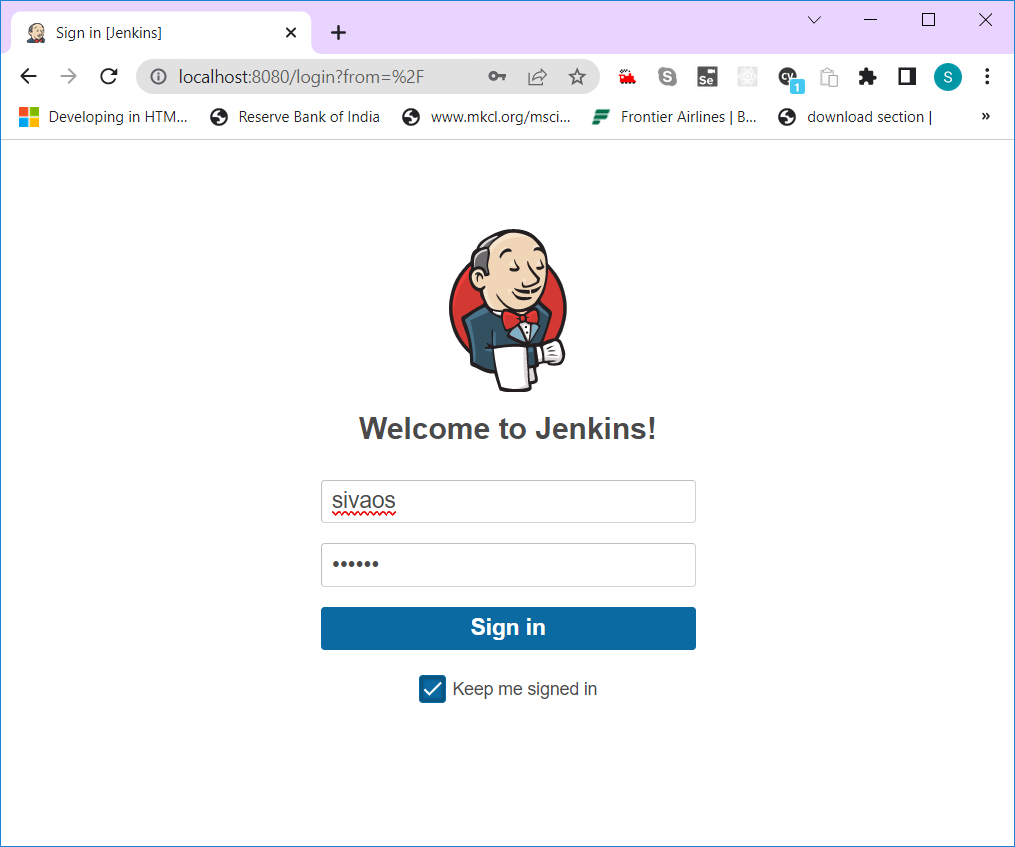
Sonar Lint – Is a Eclipse Plugin that can be added to Eclipse from the “Market Place” (Help menu)

Sonar Lint – can also be added to VS Code.

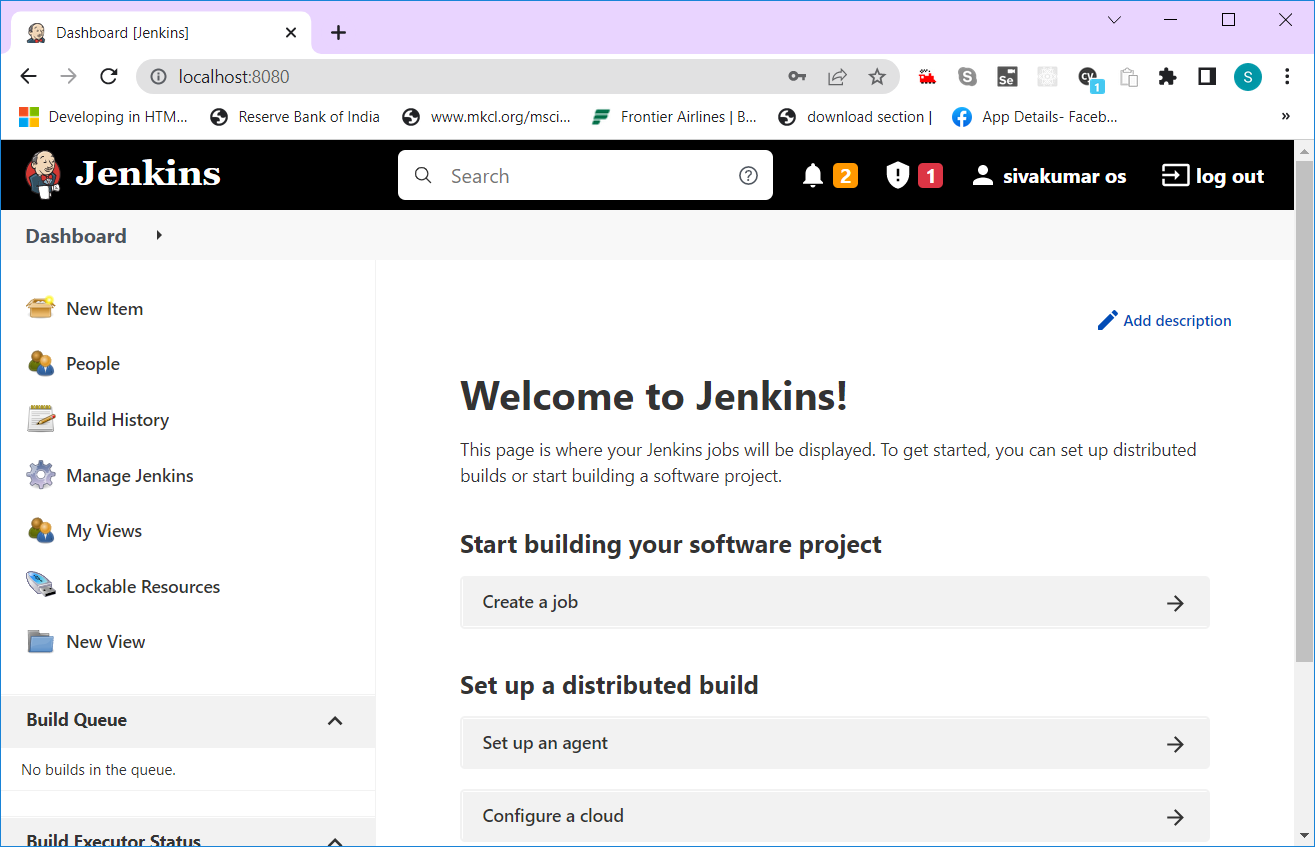


Integrating Github with Jenkins

* Open Jenkins (open localhost:8080 in browser)

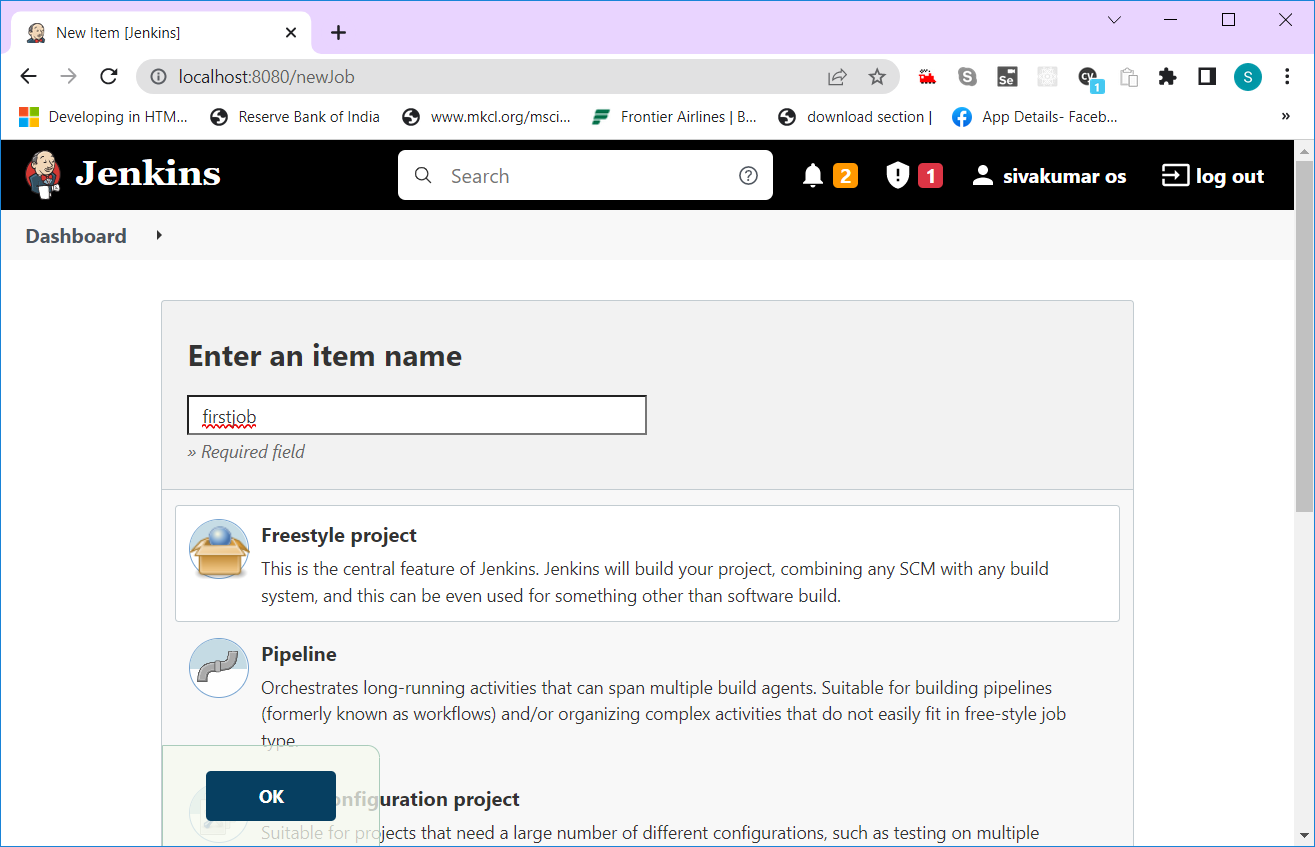


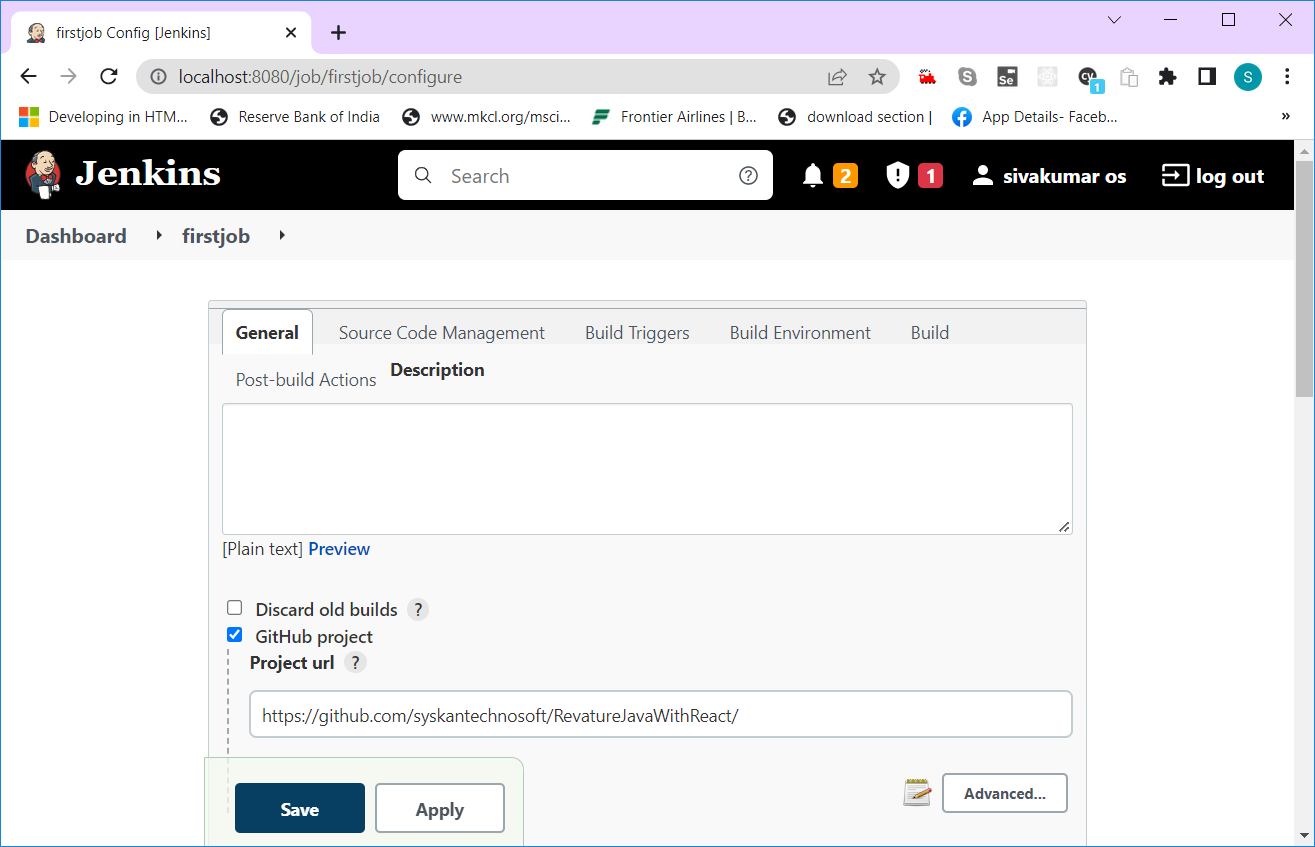
* Enter username & password (sivaos/sivaos)

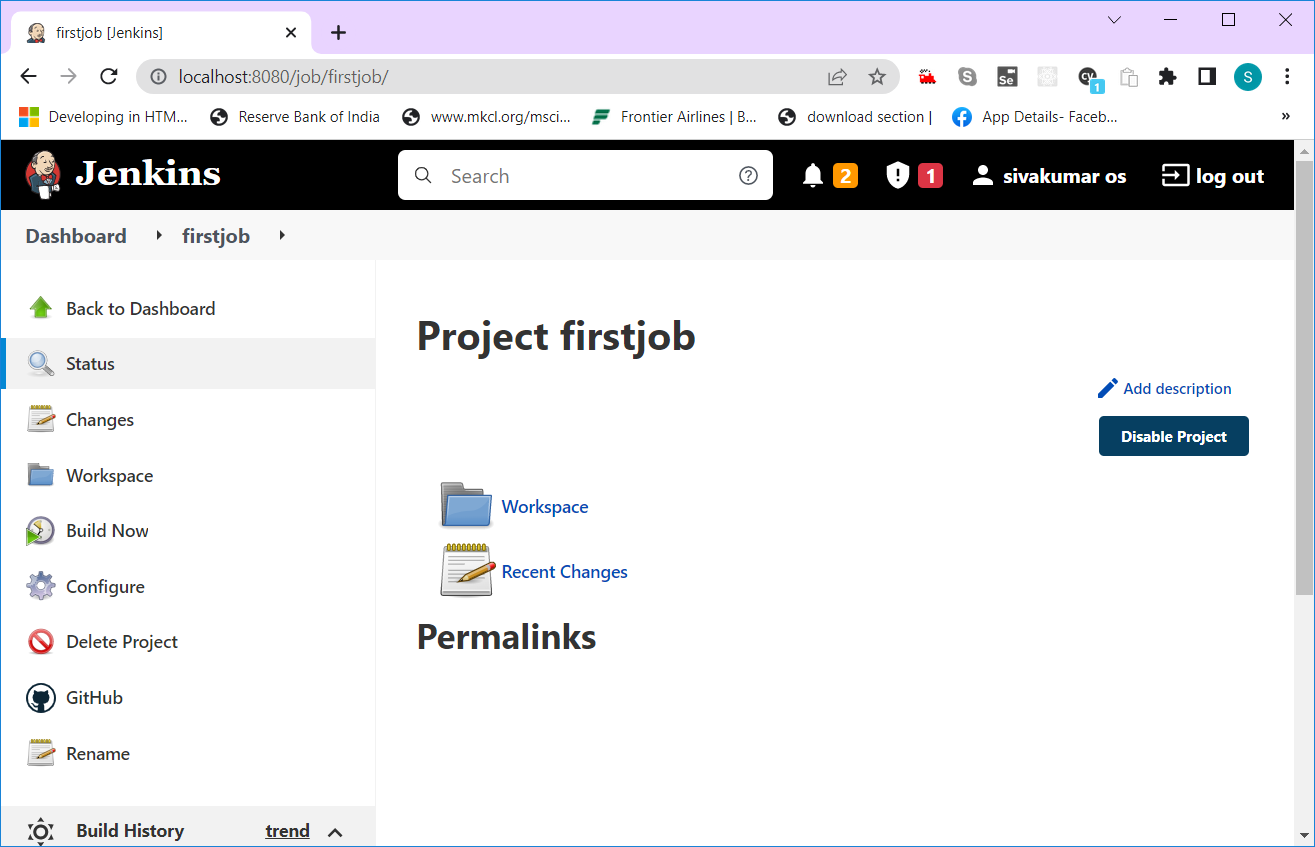


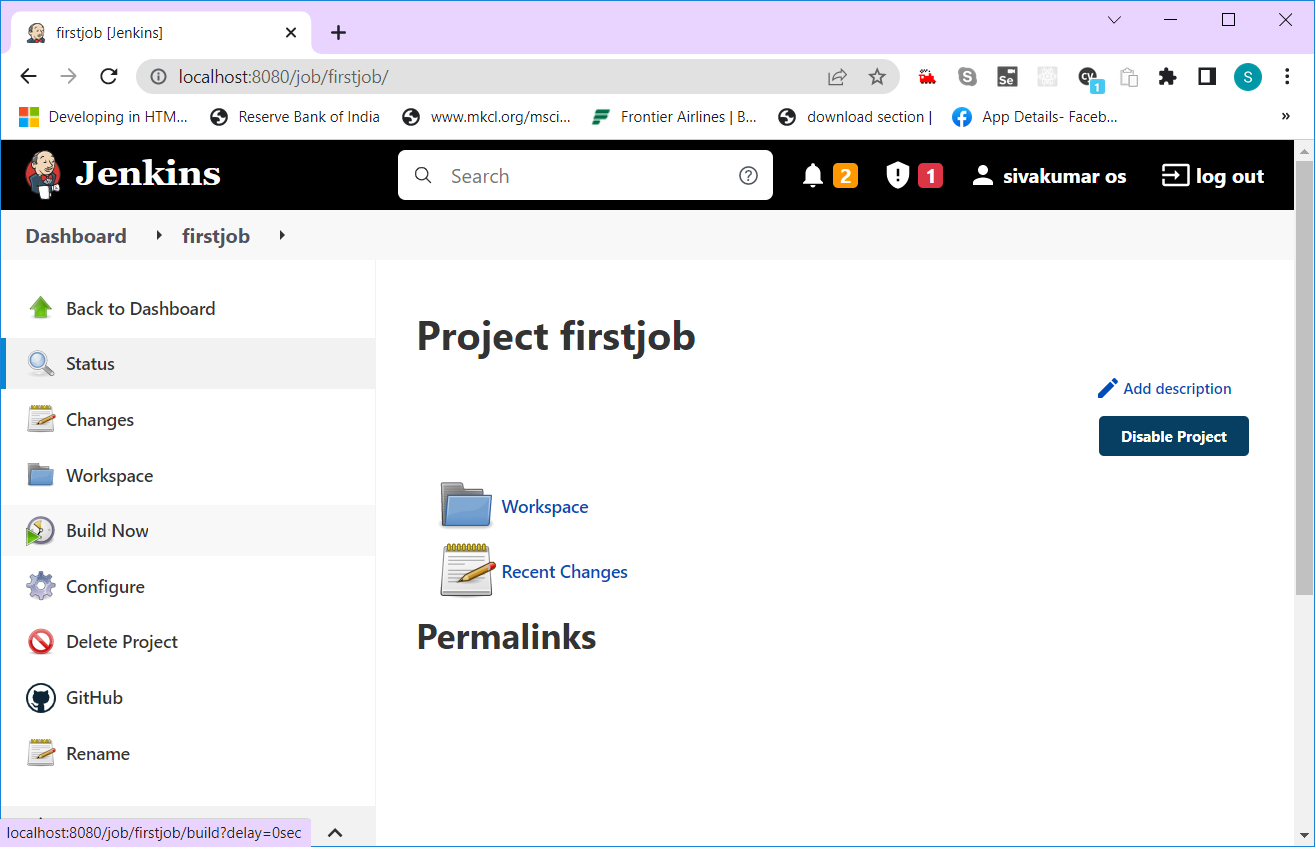
In Jenkins, all the process are called as Job (Job represents a particular sequence of process)

Click on “Create Job” link

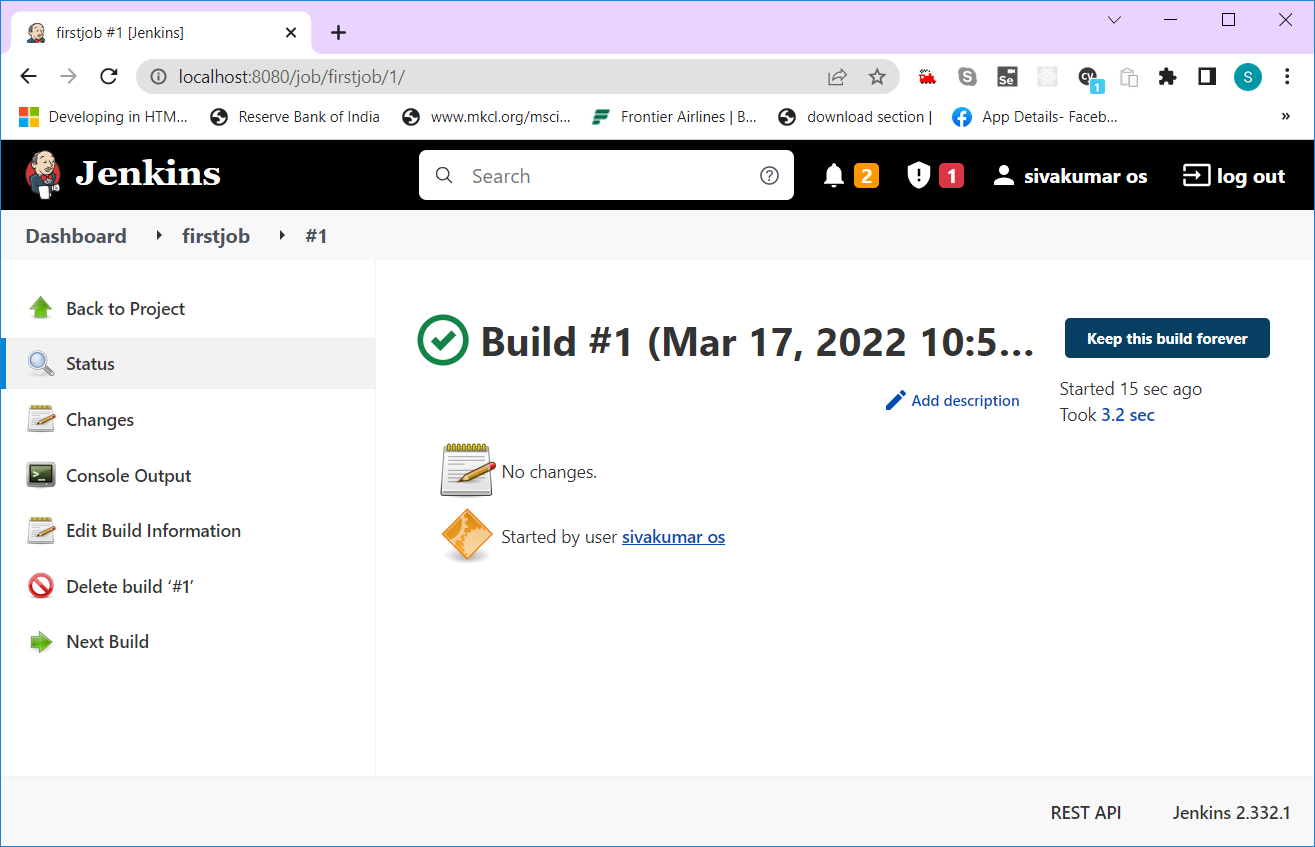


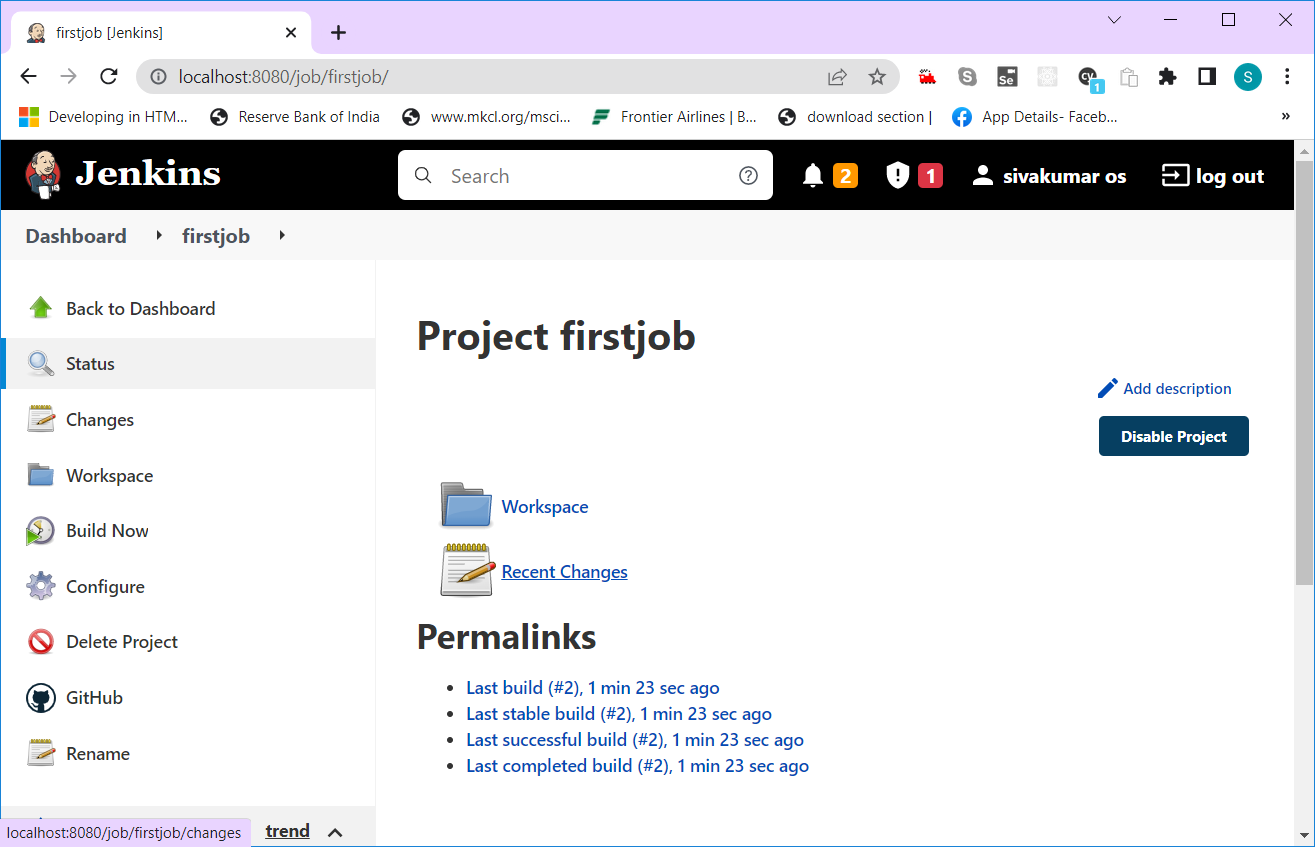


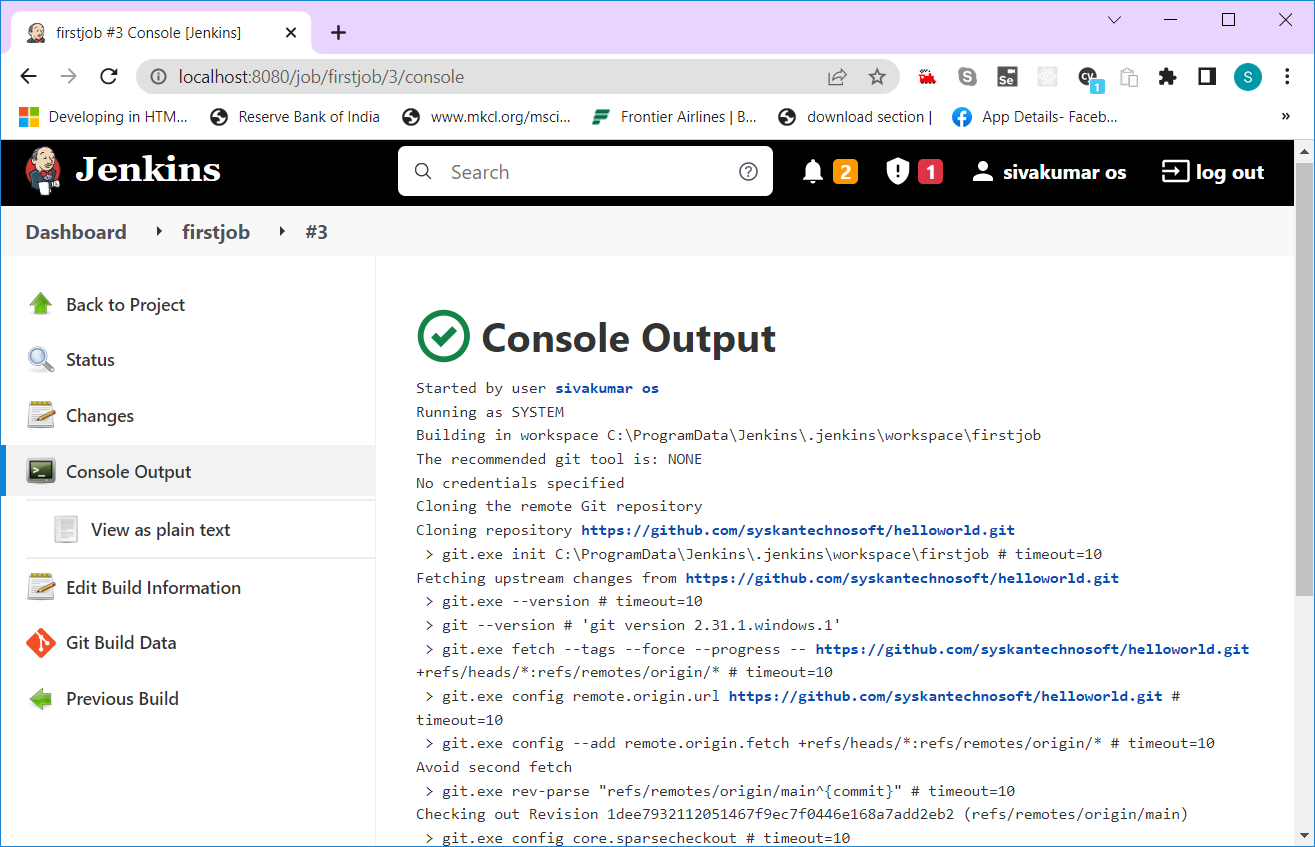












Hibernate – ORM Framework

ORM – Object Relational Mapping

Object represents Java Beans

Relational represents Database Entities (Table/Trigger/Views/Stored Procedures/Functions/Sequences etc.,)

ORM – Linking/Connecting JAVA with DATABASE

Hibernate is ORM Framework – We can interact with any RDBMS (Relational Database Management System) easily.

CRUD Operations – Create/Insert, Read, Update, Delete Operation do in a Database using Programming Language.

JDBC – Java Data Base Connectivity (API – provided by java to interact with any rdbms)

Challenges in JDBC

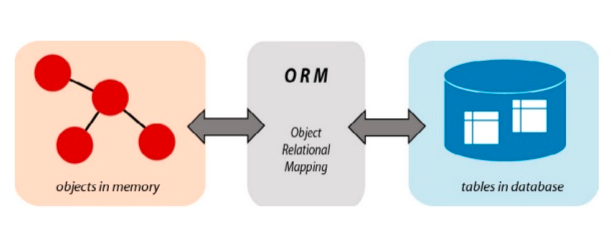
1. DB dependent Query (Insert query, update query, delete query, select query)
2. Needs to manage connection related properties (Driver, Username, Password)
3. No Caching, Scalable & High Availability

JPA – Java Persistence API (Serializing the Entity bean class in a RDBMS table – Persistence)

JPA is specification (Only Interfaces & Abstract Classes are defined)

So many popular JPA Implementations

1. Hibernate
2. EclipseLink
3. iBatis ….



RDBMS uses SQL (Structured Query Language)

JPA uses JPQL (Java Persistence Query Language – Database Independent Query) – It uses Bean classes instead of database tables in JPQL.

Hibernate uses HQL (Hibernate Query Language – Database Independent Query)

Hibernate is an open source ORM framework developed in 2001 by Gavin King.

It simplifies the process of interacting to the RDBMS.

If you have tables already, the entity bean class can be created automatically.

If you have entity bean class, then database tables can be created automatically.

**Hibernate is JPA Implementation.**

The official site : --- <https://hibernate.org>

OXM – Object XML Mapping

The latest version of Hibernate ORM is v6.0.0.CR2 as of 17Mar2022.

Hibernate 5.6 download url --- <https://hibernate.org/orm/releases/5.6/> (download the zip file)

For Maven project --- hibernate 5.6.5.final

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

<project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd" xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<modelVersion>4.0.0</modelVersion>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-java8</artifactId>

<version>5.6.5.Final</version>

<name>Hibernate ORM - hibernate-java8</name>

<description>(deprecated - use hibernate-core instead) Support for Java8-specific features - mainly Java8 Date/Time (JSR 310)</description>

<url>https://hibernate.org/orm</url>

<organization>

<name>Hibernate.org</name>

<url>https://hibernate.org</url>

</organization>

<licenses>

<license>

<name>GNU Library General Public License v2.1 or later</name>

<url>https://www.opensource.org/licenses/LGPL-2.1</url>

<distribution>repo</distribution>

<comments>See discussion at https://hibernate.org/community/license/ for more details.</comments>

</license>

</licenses>

<developers>

<developer>

<id>hibernate-team</id>

<name>The Hibernate Development Team</name>

<organization>Hibernate.org</organization>

<organizationUrl>https://hibernate.org</organizationUrl>

</developer>

</developers>

<scm>

<connection>scm:git:https://github.com/hibernate/hibernate-orm.git</connection>

<developerConnection>scm:git:git@github.com:hibernate/hibernate-orm.git</developerConnection>

<url>https://github.com/hibernate/hibernate-orm</url>

</scm>

<issueManagement>

<system>jira</system>

<url>https://hibernate.atlassian.net/browse/HHH</url>

</issueManagement>

<dependencies>

<dependency>

<groupId>org.jboss.logging</groupId>

<artifactId>jboss-logging</artifactId>

<version>3.4.3.Final</version>

<scope>compile</scope>

<exclusions>

<exclusion>

<artifactId>xml-apis</artifactId>

<groupId>xml-apis</groupId>

</exclusion>

</exclusions>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.6.5.Final</version>

<scope>compile</scope>

<exclusions>

<exclusion>

<artifactId>xml-apis</artifactId>

<groupId>xml-apis</groupId>

</exclusion>

</exclusions>

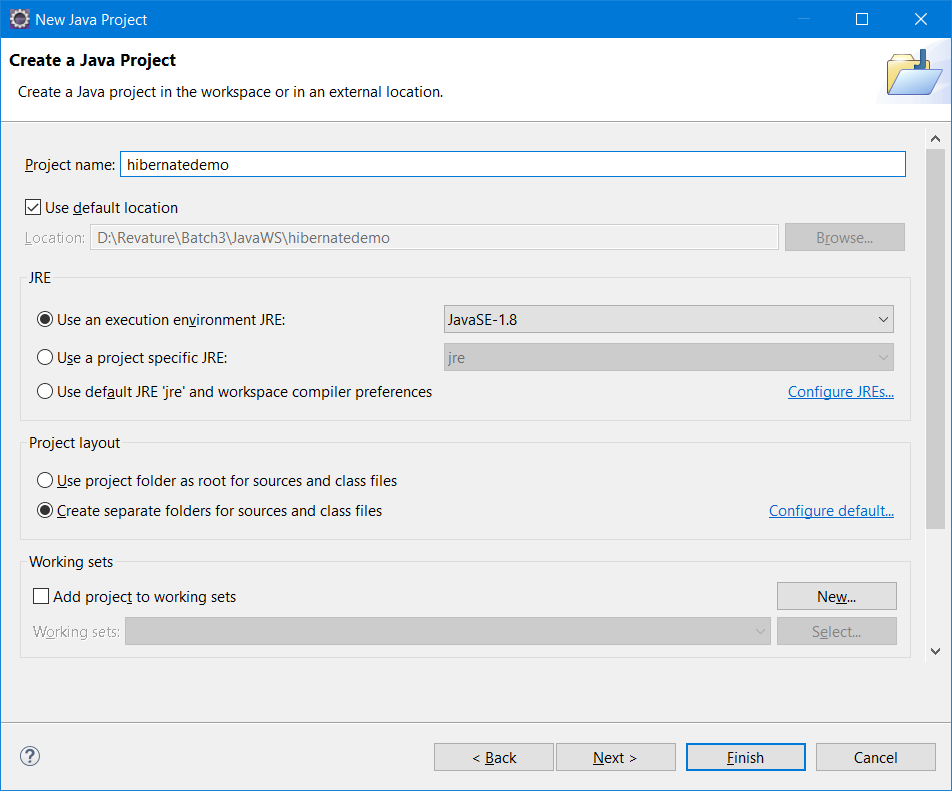
</dependency>

</dependencies>

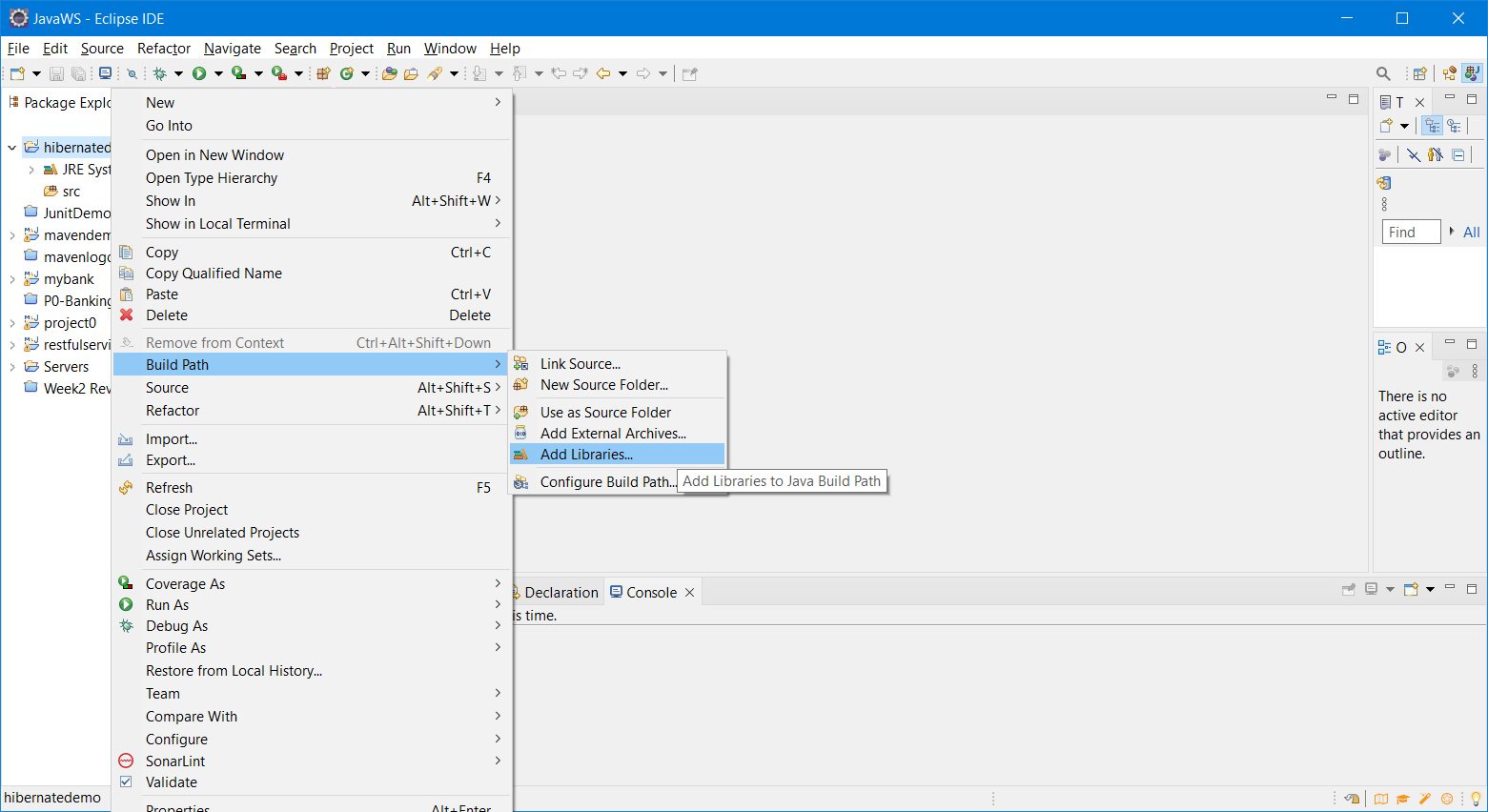
</project>

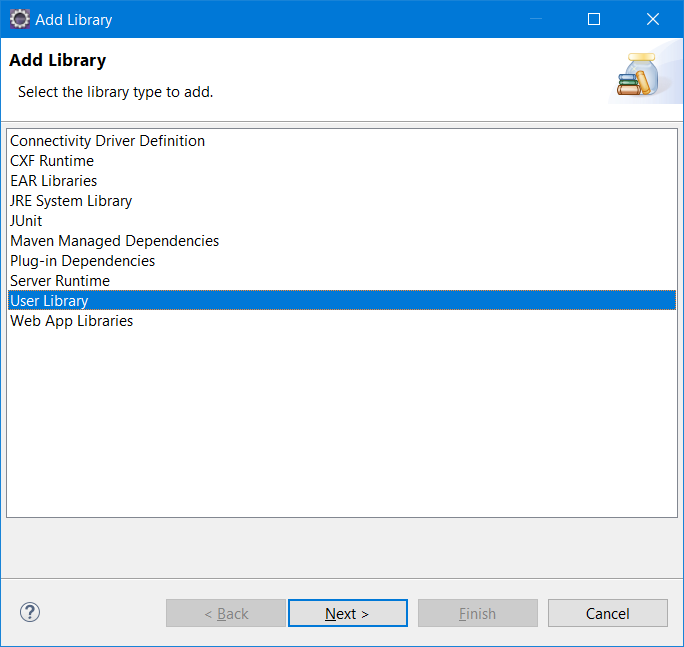
Method 1) Creating hibernate stand-alone project

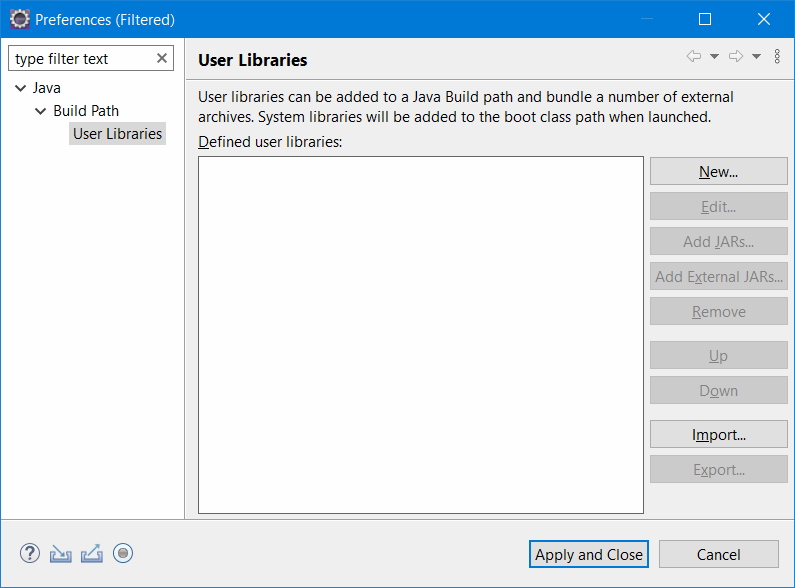
* Open Eclipse
* Create a new Java Project (File🡪New🡪Java Project)
* Name the project as “hibernatedemo”



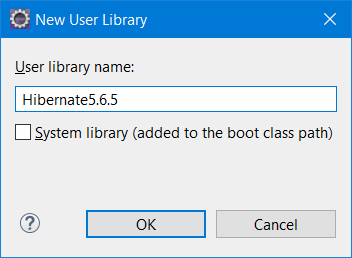
* Click the “finish” button
* Add hibernate jars to the build path (as a user library)

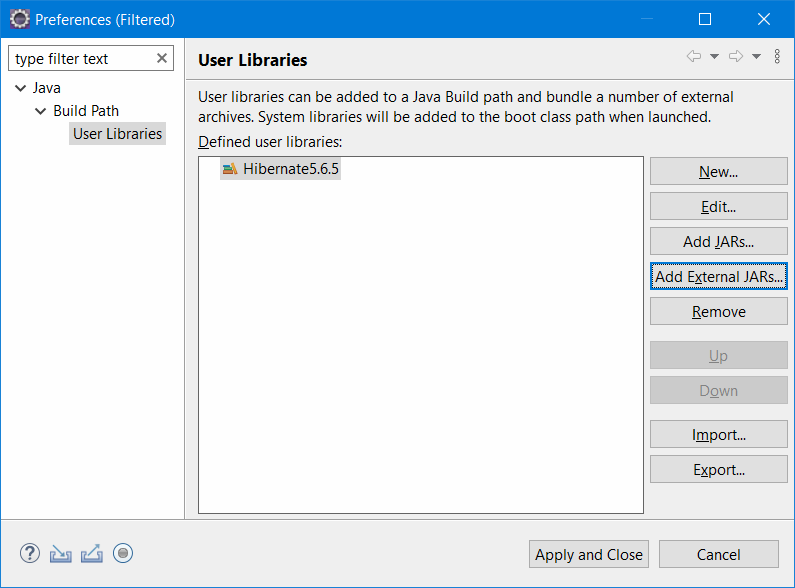




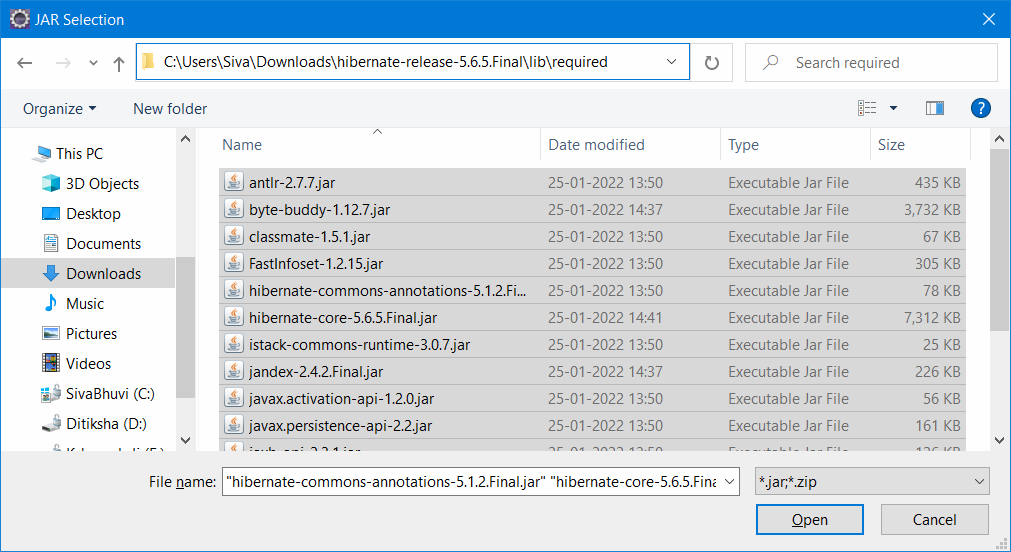


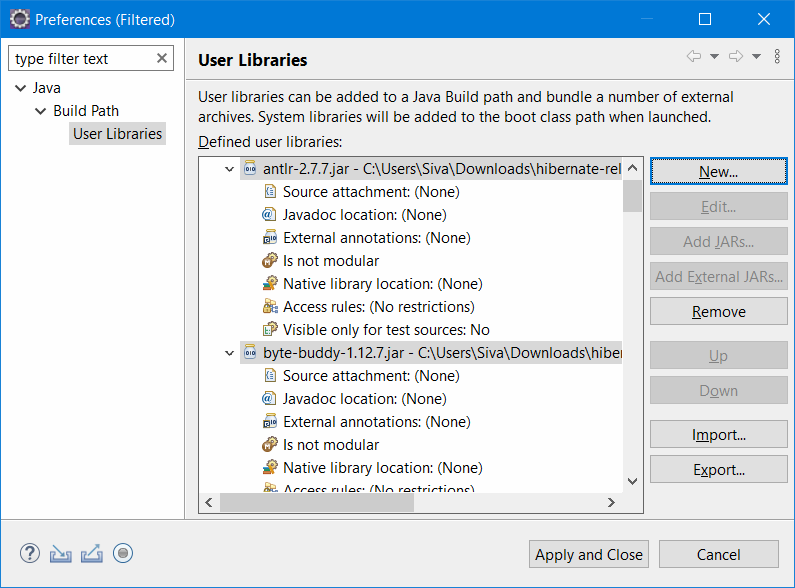
Click New



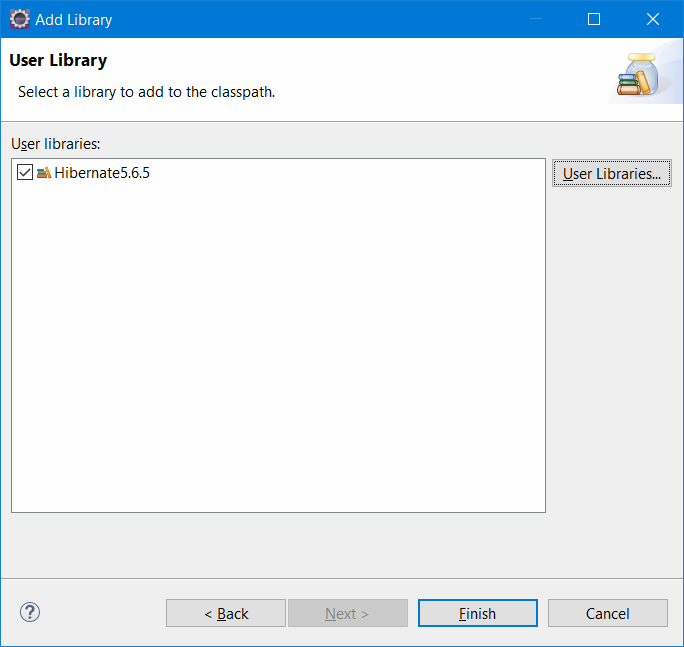


Click “Add External Jars”





Click on “Apply and Close”

Click on “Finish”

<https://www.javatpoint.com/steps-to-create-first-hibernate-application>

Name of hibernate configuration file is hibernate.cfg.xml

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

<!DOCTYPE hibernate-configuration PUBLIC

          "-//Hibernate/Hibernate Configuration DTD 5.3//EN"

          "http://hibernate.sourceforge.net/hibernate-configuration-5.3.dtd">

<hibernate-configuration>

    <session-factory>

        <property name="hbm2ddl.auto">update</property>

        <property name="dialect">org.hibernate.dialect.MySQLDialect</property>

        <property name="connection.url">jdbc:mysql://localhost:3306/revature</property>

        <property name="connection.username">root</property>

        <property name="connection.password">root</property>

        <property name="connection.driver\_class">com.mysql.cj.jdbc.Driver<property>

    <mapping resource="trainer.hbm.xml"/>

    </session-factory>

</hibernate-configuration>

Name of hibernate mapping file is trainer.hbm.xml (hbm- hibernate mapping)

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

<!DOCTYPE hibernate-mapping PUBLIC

 "-//Hibernate/Hibernate Mapping DTD 5.3//EN"

 "http://hibernate.sourceforge.net/hibernate-mapping-5.3.dtd">

 <hibernate-mapping>

  <**class** name="com.revature.bean.Trainer" table="trainer">

    <id name="id">

     <generator **class**="assigned"></generator>

    </id>

    <property name="name"></property>

    <property name="email"></property>

<property name="mobile"></property>

  </**class**>

 </hibernate-mapping>

Server Side Coding (Servlets & JSP /JSF)

JSP – Java Server Pages [HTML+JAVA code -- Java is embedded in HTML]

JSF – Java Server Faces

Servlet – Java Class to generate dynamic Web page. [JAVA+HTML – HTML is embedded in java code]