**GC EAD-Details**

Date of Birth: 07/06/89

Btech:2006 to 2010

**Us Entry Date: 7 th sep-2014**

**USA First Project**------- 12 th -sep 2014.

**H1 :2014-16**

**16-18**

**18-20**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CLIENT | SECTOR | PLACE | STATE | DURATION | Adress |
| **T-Mobile,** | telecom | **Glendale,** | **CA** | nov-2017 to Present (15m) |  |
| **Change Healthcare, Lombard, IL** | Health care | **lombard** | illinois | Aug 16 to oct 17  14 m |  |
| Flagstar Bank | bank | Troy, | MI | Sept14-oct 15 (13m) |  |
| **IT Soft, India** | Health care | india |  | Dec2012-aug 2014(18m) |  |
| **Serco, India** | Health care | **india** |  | July 2010-nov2012(28) |  |
|  |  |  |  |  |  |

Sure!!!

I have over 8 years of strong software experience in design, development and deployment of web-based and Client-Server business applications using Java/J2EE technologies.

Over the period of these 8 years I worked on 5 projects which gave me an opportunity to work on different kind of sectors like banking, insurance, Healthcare, telecom etc.

Working in these sectors helped me to develop some strong skills in using Struts, springs, Hibernate and Web Services. I also have a good hold on core java and advanced java as well.

In my recent project, I worked as a full stack java developer where I had a role in both front enddesigning and backend development. Primarily I focused on working on 70% on backend and 30 % in frontend, In this project, I worked with an off-shore team which

Consist of 4 developers and the onsite team which had 2 developers apart from me. I coordinated with both the teams daily and used to set up standup meetings, here we followed agile methodology with 2-week sprint, here we followed TDD(Test driven development)

As a full stack developer, on the server(IBM websphere) side I worked with Servelets, hibernate for writimh hibernate queries, several modules on Springs, JPA, Spring JMS and web services both rest and SOAP.

T-Mobile is a leading Telecom Domain company based in USA. T-Mobile Equipment Instalment Program (EIP) team works with clients to help their customer’s finance their Mobile devices from various available instalment plans introduce new plans and calculate down payment and monthly Instalment using Credit Information of an end user. Also, helps them Return or Exchange their Mobile. Device and upgrade to a new mobile device using jump Upgrade Option

On the front end I worked with JavaScript, HTML, JSP, JSTL and Java Script libraries like JQuery and Angular JS. I extensively used Angular JS 2.0 components to develop Single Page Applications.

From the database side, I worked with several DB's like Oracle, SQL Server, MySQL, and with some NoSQL DB's like MongoDB, Casandra and Dynamo DB while using Amazon Web Services (AWS)

I also worked as a backup resource to my DBA where I took up some of his activities like monitoring the DB. Releasing DB locks, managing backups and restoring DB's or any other DB activities.

Apart from all of this I used various tools like Spring tool suite (STS), eclipse, IBM RAD(rational application development),

xml spy for XML creations and conversion of xml to Json using XSLT (Extensible Stylesheet Language

Transformations),

SOAP UI for testing soap based web services

PoSTMAN for tresting rest based web services.

Server wise I have used tomcat, WebLogic, WebSphere, and JBoss in my past projects. I have experience in managing the server load, checking the server health, doing performance testing and load

testing to see if any memory leaks are there in the system.

Lastly I have experience with Amazon web services as we were working with it in my recent project. I worked creating EC2 instances and deploying code on the instances, management, Cloud Front, Creating VPC, cloud templates, AMI copies

of server instances, S3 bucket for storing data API's etc.

So, this is a brief description of me and my experience as a full stack developer over a period of 8 years.

**Project Description:**

**Express Scripts Inc :-**

**Northern Trust Bank is a global leader in delivering innovative investment management, asset and fund administration, fiduciary and banking solutions to corporations, institutions and affluent individuals. The private and wealth passport are gateway applications for clients to engage with the bank, manage their portfolio, aggregate assets &amp; liabilities and leverage investment options. I worked on the Enrollment Application for these passports that is used to enroll clients and provide entitlements to leverage the various features within. Some of the key functionalities / features that I led are –Rapid Enrollment Application: A standalone Spring MVC based application used as the enrollment system for the operations team to register/enroll users and accounts to northern trust bank, led the redesign of the entire application with enhanced features, UI and made it web-based from Power Builder (legacy system).**

**Principal Responsibilities:**  
We deals with inpatient and outpatient data from the Data Repository

Application Design   
  
· Online Payments System  
  
· Application Deals with Claims, Payments, Checks Recovery

. Generally, Every Application Will Have 2 Phases

1. Costumer Facing

2. Team Member Facing

Coming to my role in the project is Team Member Facing

If costumer Applied for claim then facing team will (APPROVE, DECLINE, REVIEW) the claim

· Application Development   
. we used backend technologies like hibernate data mapping, spring

Used **Spring dependency injection**, **annotations and Spring MVC components** to implement business layer and navigation part of application

As of now am working in this project from last one year

So am working on 2 applications 1st application is online payment system

And the 2nd application is protype in this we use all hifi technologies like angular html CSS and then we deploy application in to the AWS cloud

-----------------------------------------------------------------------------------------------·

Designs and codes complex programs   
  
· Evaluates complex client area requirements and processes

· Conducts analysis of organizational needs and goals for the development and implementation of application systems   
  
· Provides application and user support and performs troubleshooting   
  
· Provides production support as necessary    
  
· Oversees the technical implementation of projects as needed   
  
Testing   
  
· Reviews and refines test cases, scenarios and scripts to ensure applications quality   
  
· Reviews and refines usability testing scenarios, and administers portions of the testing process   
  
· Creates test transactions and runs component, assembly, and system tests for conformance to standards and adherence to design specifications   
  
· Prepares the test environment   
  
· Analyzes results of testing sessions and presents results   
  
· Works with users to review test results to ensure they meet expected results   
  
· Conforms to quality assurance test standards   
  
Documentation   
  
· Reviews documentation for the user describing the installation, use, and customization of products and services as necessary   
  
· Uses product design specifications to direct development of useful documentation, help functions, and features   
  
Knowledge and Skills Required  
  
· Demonstrates strong analytical and communication skills   
  
· Experienced in working with a geographically-separated team.   
  
· Design and development of Web based applications using modern JavaScript frameworks like react.js or angular.js   
  
· Strong java programming knowledge. Familiarity with spring framework and Restful services is a must.   
  
· Design and development of RDBMS systems using Oracle 12c, strong SQL, PL/SQL skills   
  
· Working knowledge of “Git” is required.   
  
· Familiarity with reporting tools like Perl, Unix Scripting, Jenkins, Autosys etc. will be a plus

* Presently working on enhancements code and supporting the user’s applications to process it.
* Created custom Angular JS directives to implement some of the specific functionalities.
* Implemented **REST** based web services using **JAX-RS** annotations, Jersey provider and consumed using **HTTP** services from **angular JS** modules.
* Used **JQuery/JQLite** to create drop downs, status, progress and menu bar and other drag and drop components.
* Developed Application to asses JSON and XMl from Restful web service from consumer side using Angular JS.
* Effective use of **data structures** and **algorithms** to solve problems, reducing time complexity and memory usage.
* Experience in developing views and templates with **Python** and Django’s view controller and templating language to create a user-friendly website interface.
* Worked on business logic and calculations, performs analysis, handles data transfers, and generates XSL for presentation layer and **SQL** commands for **database** layer.
* Experience in developing views and templates with **Python** and Django’s view controller and templating language to create a user-friendly website interface.
* Extensively implemented **Multithreading** concept in order to handle Transaction Management with isolation and propagation levels. Thread handling was used to maintain continuity of execution.
* Created and injected **spring** services, controllers and DAOs to achieve **dependency injection**.
* Heavily used **Spring Inheritance, Auto-wiring, Core Container, Security, AOP, ORM modules as part of migration from EJB to spring, Spring Quartz** for scheduling tasks to generate reports and emails to clients.
* Implemented **SOA** to develop **Spring Restful/Micro Services and implemented Spring Eureka, Netflix, Ribbon as part of Services Discovery** usin**g Apache Axis**.
* Used Micro **service** architecture with **Spring Boot** based services interacting through a combination of **REST** and **Apache Kafka** message brokers.
* Used **SOAP UI, Postman, Rest Client, Spring Mock MVC, Spring 4Junit** for testing the web services.
* Experienced in implementing **Micro services**, **Service Oriented Architecture (SOA)** with **XML** based Web Services (**SOAP/UDDI/WSDL**) using Top Down and Bottom Up approach.
* Experience in creating **RESTful**web services using **Jersey, spring** framework and **JAX-RS**.
* **Java Beans** are used as container of business logic and supporting the services or applications.
* Used wide implementation procedures in order to disintegrate whole monolithic application into loosely coupled reusable **Micro services**. This laid foundation for the simplicity in updating versions of existing technologies without hassle of deploying entire modules.
* Employed **fail safe** and **circuit breaker patterns** for the first time in Client’s email applications using **Hystrix** and **Hystrix Dashboard** in **spring boot** Micro service Applications.
* Good knowledge with cloud services like **Spring Eureka, Cloud Foundry and AWS** etc.
* Extensively used **Hibernate Relational mappings (One-To-Many, Many-To-Many)** in the secondary project.
* Extensively worked on **Hibernate Cache, Query, Criteria and Transactions** in secondary project.
* Developed search functionality by **Named Queries, Projections, Restrictions** using Hibernate 4.3 API
* Worked on creation of custom **Docker** container images, tagging, pushing images, integration of

**Spring boot**.

* Implemented **Maven Script to create JAR, WAR, EAR & dependency JARS** and deploy the entire project onto the **WebLogic** Application Server.
* Created the **Docker** containers and **Docker** consoles for managing the application life cycle.
* Used **DROOLS** engine as java runtime rule engine to validate business rule before executing in runtime.
* Developed the Web Services Client using **REST, RSDL** description to verify the history of the new customer.
* Worked on **XML** parsers (**JAXB**) for data retrieval and **JAXB** parser for marshalling and un marshaling.
* Writing SQL Queries to get the information from **Oracle DB**, Saving, modifying, and removing the data from the DB. Used joins to get the data from more DB Tables. Views and Unions were also part of our implementation.
* Used **CURL** to move data from or to a server using available http and https protocols.
* Used DB components like **Data Access Objects**, **Collections**, and **Value Objects.**
* Used **Maven** as build and dependency management tool for creating **EAR, WAR** and **JAR** file to be deployed in application servers and integrated with **SVN, Jenkins, and Jenkins Jobs.**
* Worked on **Pivotal Cloud Foundry** to host **Cloud based Applications**.
* Experience in re-architecting monolithic architecture service into **micro services** based architecture using spring boot and **pivotal cloud foundry**.
* **GIT** is used for the version control **Maven** is used for build script and configured **log4j, .log** files.
* Used **AWS** cloud services and deployed couple of applications on **Elastic Bean Stalk** and **EC2** Instances as part of service transition phase, but later on chose to deploy them on Eureka.
* Aware of troubleshooting issues on **AWS**, **Netflix Eureka** and other developer IDE and Cloud environments.
* Using **JUnit** test in Eclipse, developed Unit test cases and test suite**. Curl** was also implemented to test.
* Used **Jenkins** as the integration tool and improved **scalability** of applications on **cross-platforms**.
* Involved in **System Testing, Integration Testing, and Production Testing** on **WebLogic** Server.
* Resolved bugs/defects in application with team members of the project to assure a positive outcome using **JIRA.**
* Used GIT as version management, **JIRA** for defect management system. Performed unit testing using **JUnit**.

**Ref 1**:

Client Name: Express Scripts

Person Name: Balaji Katari

Title: Team-Lead

Phone: [248-971-5719](tel:(248)%20971-5719)

Email : [Bkatari@express-scripts.com](mailto:Bkatari@express-scripts.com)

**Ref 2**

Client Name: Express Scripts

Person Name: Teja Chunduri

Title: Team- Member

Phone: [207-303-8474](tel:(207)%20303-8474)

Email : : [Tchunduri@express-scripts.com](mailto:Tchunduri@express-scripts.com)

2nd Project

Client Name: Conduent

Person Name: Bharat Atkuri

Title: Team- Lead

Phone: 669 262 0072

Email : : [Bharath.atkuri@infinite.com](mailto:Bharath.atkuri@infinite.com)

|  |  |
| --- | --- |
| Full Name: | Jagan |
| Mobile/Home Phone No : | 913-912-3916 |
| Last 4 Digits of SSN: | 4012 |
| Skype ID: | jaganjj008@gmail.com |
| Email: | jaganjj008@gmail.com |
| [DOB](https://www.linkedin.com/pub/uday-singh/1b/6ab/642) (MM/DD) | 05/07/1988 |
| US work authorization & Expiring date of visa : | GC-EAD(FEB-2018) |
| Current Location: | Franklin Lakes, New Jerssy |
| Willingness to relocate across US: | **Yes** |
| Education (Bachelors with Year of Passing) | B.Tech-2009 |
| Expertise & Skills Sets: | Java |
| Experience in Years: | 8+ |
| Interview Availability: | Any day After 2pm EST WITH Prior information |
| Available to join the project: | 10-days |
| LinkedIn: |  |
| Rates: |  |

**Core Java- Recent 2 projects**

**Angular JS** - **(2+ Years exp)**

* Angular is a JavaScript-based open-source front-end web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications.

**Servlets ---- 1 st project in India**

* (3 years): Java Servlets are programs that run on a Web or Application server and act as a middle layer between a request coming from a Web browser or other HTTP client and databases or applications on the HTTP server.
* Using Servlets, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.
* Java Servlets often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But Servlets offer several advantages in comparison with the CGI.

Performance is significantly better.

**JSP --- 8 Years of exp**.-----

* Java Server Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types
* JSP is similar to PHP, ASP and React JS JSX, but it uses the Java programming language.
* To deploy and run Java server Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required.

**JavaBeans** ---

* Having Knowledge on this concept They are used to encapsulate many objects into a single object (the bean), so that they can be passed around as a single bean object instead of as multiple individual object.
* A JavaBean is a Java Object that is serializable, has a nullary constructor, and allows access to properties using getter and setter methods.

**Java Servlet** ---

* Having Knowledge on this concept----A servlet is a Java programming language class that is used to extend the capabilities of servers that host applications accessed by means of a request-response programming model.
* Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by web servers

**Spring---- 7 Years ---------**

* The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application,
* but there are extensions for building web applications on top of the Java EE platform. The Spring Framework is open source.

**Hibernate - 7 years**—

* Hibernate ORM (Hibernate in short) is an object-relational mapping tool for the Java programming language.
* It provides a framework for mapping an object-oriented domain model to a relational databas
* Hibernates primary feature is mapping from Java classes to database tables, and mapping from Java data types to SQL data types. Hibernate also provides data query and retrieval facilities
* It generates SQL calls and relieves the developer from the manual handling and object conversion of the result set.

**Apache Tomcat - 1 Year exp in India** ---

* Apache Tomcat, often referred to as Tomcat Server, is an open-source Java Servlet Container developed by the Apache Software Foundation (ASF)
* Tomcat implements several Java EE specifications including Java Servlet, Java Server Pages (JSP), Java EL, and Web Socket, and provides a "pure Java" HTTP web server environment in which Java code can run.

**HTML5- (2+ years)**

* HTML5 is a markup language used for structuring and presenting content on the World Wide Web.
* It is the fifth and current version of the HTML standard

**Java script - In all projects**.

* Java script is a programming language& integrated with HTML. It is open and cross-platform.

**JBOSS - 1+ Year—**

* JBoss Application Server (or JBoss AS) is a free software/open-source Java EE-based application server.
* An important distinction for this class of software is that it not only implements a server that runs on Java, but it actually implements the Java EE part of Java

**Web Sockets** –

* having knowledge, it’s just comes under web servers/Application

**CSS 3: 1+ exp.—**

* Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML,
* the language can be applied to any XML document

**STS** –

* Present project we are using STS The Spring Tool Suite is an Eclipse-based development environment that is customized for developing Spring applications.
* It provides a ready-to-use environment to implement, debug, run, and deploy your Spring applications, including integrations for Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven

**Junit** ---

* Recent 3 projects i have an exp --- JUnit is a Regression Testing Framework used by developers to implement unit testing in Java, and accelerate programming speed and increase the quality of code.
* JUnit Framework can be easily integrated with either of the following -

**Eclipse----**

* Eclipse is an integrated development environment (IDE) used in computer programming, and is the most widely used Java
* It contains a base workspace and an extensible plug-in system for customizing the environment.

**Ant --- Apache**

* Ant is a Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other.
* The main known usage of Ant is the build of Java applications. Ant supplies a number of built-in tasks allowing to compile, assemble, test and run Java applications
* Maven -- Recent 2 projects (2+ years exp) Maven is a build automation tool used primarily for Java projects.

**SOAP --- 7 years**

* Simple Object Access Protocol (SOAP) is a standard protocol specification for message exchange based on XML.
* Communication between the web service and client happens using XML messages

**Cloud environments: Azure**:

* Having Knowledge on this concept.

**Eureka** –

* Eureka is a REST (Representational State Transfer) based service that is primarily used in the AWS cloud for locating services for the purpose of load balancing and failover of middle-tier servers
* At Netflix, Eureka is used for the following purposes apart from playing a critical part in mid-tier load balancing.

**Kubernetes**

* is an open source container cluster manager originally designed by Google and donated to the Cloud Native Computing foundation
* It aims to provide a "platform for automating deployment, scaling, and operations of application containers across clusters of hosts

**Amazon Dynamo DB:**

* Amazon DynamoDB is a fully managed [NoSQL database](https://aws.amazon.com/nosql/) service that provides fast and predictable performance with seamless scalability.
* You can use Amazon DynamoDB to create a database table that can store and retrieve any amount of data, and serve any level of request traffic.
* Amazon DynamoDB automatically spreads the data and traffic for the table over a sufficient number of servers to handle the request capacity specified by the customer and the amount of data stored, while maintaining consistent and fast performance.

**NoSQL**

* NoSQL is a term used to describe high-performance, non-relational databases.
* NoSQL databases utilize a variety of data models, including document, graph, key-value, and columnar.
* NoSQL databases are widely recognized for ease of development, scalable performance, high availability, and resilience. Below are several resources to help you get started using NoSQL databases.

**Kafka**

* Kafka is run as a cluster on one or more servers.
* The Kafka cluster stores streams of *records* in categories called *topics*.
* Each record consists of a key, a value, and a timestamp.
* In Kafka the communication between the clients and the servers is done with a simple, high-performance, language agnostic [TCP protocol](https://kafka.apache.org/protocol.html).
* This protocol is versioned and maintains backwards compatibility with older version. We provide a Java client for Kafka, but clients are available in [many languages](https://cwiki.apache.org/confluence/display/KAFKA/Clients).

**Kafka has four core APIs:**

* The [Producer API](https://kafka.apache.org/documentation.html#producerapi) allows an application to publish a stream of records to one or more Kafka topics.
* The [Consumer API](https://kafka.apache.org/documentation.html#consumerapi) allows an application to subscribe to one or more topics and process the stream of records produced to them.
* The [Streams API](https://kafka.apache.org/documentation/streams) allows an application to act as a *stream processor*, consuming an input stream from one or more topics and producing an output stream to one or more output topics, effectively transforming the input streams to output streams.
* The [Connector API](https://kafka.apache.org/documentation.html#connect) allows building and running reusable producers or consumers that connect Kafka topics to existing applications or data systems. For example, a connector to a relational database might capture every change to a table.



**NOTES**

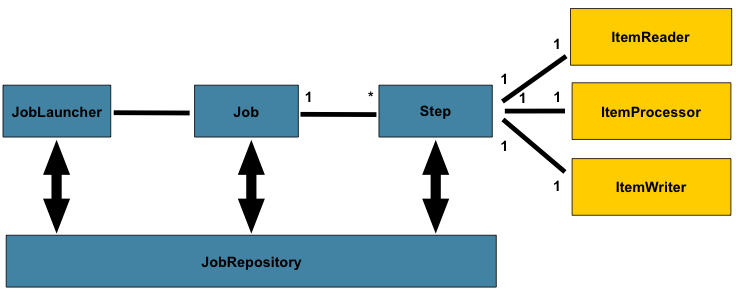


Photo credit: [*Spring Source*](http://static.springsource.org/spring-batch/reference/html/domain.html)

[Spring Batch](http://static.springsource.org/spring-batch/), is an open source framework for batch processing – execution of a series of jobs. Spring Batch provides classes and APIs to read/write resources, transaction management, job processing statistics, job restart and partitioning techniques to process high-volume of data.

The following Spring Batch tutorials and examples are tested with :

1. Spring Core 3.2.2.RELEASE
2. Spring Batch 2.2.0.RELEASE

**Singleton design pattern**

This pattern involves a single class which is responsible to create an object while making sure that only single object gets created. This class provides a way to access its only object which can be accessed directly without need to instantiate the object of the class.

Real time scenario

A few days ago, I had to implement holding a list of loaded assemblies and its references during program-execution. As this is a bigger project, we're talking about 150 dll-files to look at. In that case, I just didn't want to load the same data again and again whenever I need to get one of these assemblies. Therefore, I used a singleton-class, which creates this certain list on its first call and holds that values until the application gets closed.

**Core Java- Recent 2 projects**

**Angular JS** - **(2+ Years exp)**

* Angular is a JavaScript-based open-source front-end         web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications.

**Servlets ---- 1 st project in India**

* (3 years): Java Servlets are programs that run on a Web or Application server and act as a middle layer between a request coming from a Web browser or other HTTP client and databases or applications on the HTTP server.
* Using Servlets, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.
* Java Servlets often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But Servlets offer several advantages in comparison with the CGI.

             Performance is significantly better.

**JSP --- 8 Years of exp**.-----

* Java Server Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types
* JSP is similar to PHP, ASP and React JS JSX, but it uses the Java programming language.
* To deploy and run Java server Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required.

**JavaBeans** ---

* Having Knowledge on this concept They are used to encapsulate many objects into a single object (the bean), so that they can be passed around as a single bean object instead of as multiple individual object.
* A JavaBean is a Java Object that is serializable, has a nullary constructor, and allows access to properties using getter and setter methods.

**Java Servlet** ---

* Having Knowledge on this concept----A servlet is a Java programming language class that is used to extend the capabilities of servers that host applications accessed by means of a request-response programming model.
* Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by web servers

**Spring---- 7 Years ---------**

* The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application,
* but there are extensions for building web applications on top of the Java EE platform. The Spring Framework is open source.

**Hibernate - 7 years**—

* Hibernate ORM (Hibernate in short) is an object-relational mapping tool for the Java programming language.
* It provides a framework for mapping an object-oriented domain model to a relational databas
* Hibernates primary feature is mapping from Java classes to database tables, and mapping from Java data types to SQL data types. Hibernate also provides data query and retrieval facilities
* It generates SQL calls and relieves the developer from the manual handling and object conversion of the result set.

**Apache Tomcat - 1 Year exp in India** ---

* Apache Tomcat, often referred to as Tomcat Server, is an open-source Java Servlet Container developed by the Apache Software Foundation (ASF)
* Tomcat implements several Java EE specifications including Java Servlet, Java Server Pages (JSP), Java EL, and Web Socket, and provides a "pure Java" HTTP web server environment in which Java code can run.

**HTML5- (2+ years)**

* HTML5 is a markup language used for structuring and presenting content on the World Wide Web.
* It is the fifth and current version of the HTML standard

**Java script - In all projects**.

* Java script is a programming language& integrated with HTML. It is open and cross-platform.

**JBOSS - 1+ Year—**

* JBoss Application Server (or JBoss AS) is a free software/open-source Java EE-based application server.
* An important distinction for this class of software is that it not only implements a server that runs on Java, but it actually implements the Java EE part of Java

**Web Sockets** –

* having knowledge, it’s just comes under web servers/Application

**CSS 3: 1+ exp.—**

* Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML,
* the language can be applied to any XML document

**STS** –

* Present project we are using STS The Spring Tool Suite is an Eclipse-based development environment that is customized for developing Spring applications.
* It provides a ready-to-use environment to implement, debug, run, and deploy your Spring applications, including integrations for Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven

**Junit** ---

* Recent 3 projects i have an exp --- JUnit is a Regression Testing Framework used by developers to implement unit testing in Java, and accelerate programming speed and increase the quality of code.
* JUnit Framework can be easily integrated with either of the following -

**Eclipse----**

* Eclipse is an integrated development environment (IDE) used in computer programming, and is the most widely used Java
* It contains a base workspace and an extensible plug-in system for customizing the environment.

**Ant --- Apache**

* Ant is a Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other.
* The main known usage of Ant is the build of Java applications. Ant supplies a number of built-in tasks allowing to compile, assemble, test and run Java applications
* Maven -- Recent 2 projects (2+ years exp) Maven is a build automation tool used primarily for Java projects.

**SOAP --- 7 years**

* Simple Object Access Protocol (SOAP) is a standard protocol specification for message exchange based on XML.
* Communication between the web service and client happens using XML messages

**Cloud environments: Azure**:

* Having Knowledge on this concept.

**Eureka** –

* Eureka is a REST (Representational State Transfer) based service that is primarily used in the AWS cloud for locating services for the purpose of load balancing and failover of middle-tier servers
* At Netflix, Eureka is used for the following purposes apart from playing a critical part in mid-tier load balancing.

**Kubernetes**

* is an open source container cluster manager originally designed by Google and donated to the Cloud Native Computing foundation
* It aims to provide a "platform for automating deployment, scaling, and operations of application containers across clusters of hosts

**Amazon Dynamo DB:**

* Amazon DynamoDB is a fully managed [NoSQL database](https://aws.amazon.com/nosql/) service that provides fast and predictable performance with seamless scalability.
* You can use Amazon DynamoDB to create a database table that can store and retrieve any amount of data, and serve any level of request traffic.
* Amazon DynamoDB automatically spreads the data and traffic for the table over a sufficient number of servers to handle the request capacity specified by the customer and the amount of data stored, while maintaining consistent and fast performance.

**NoSQL**

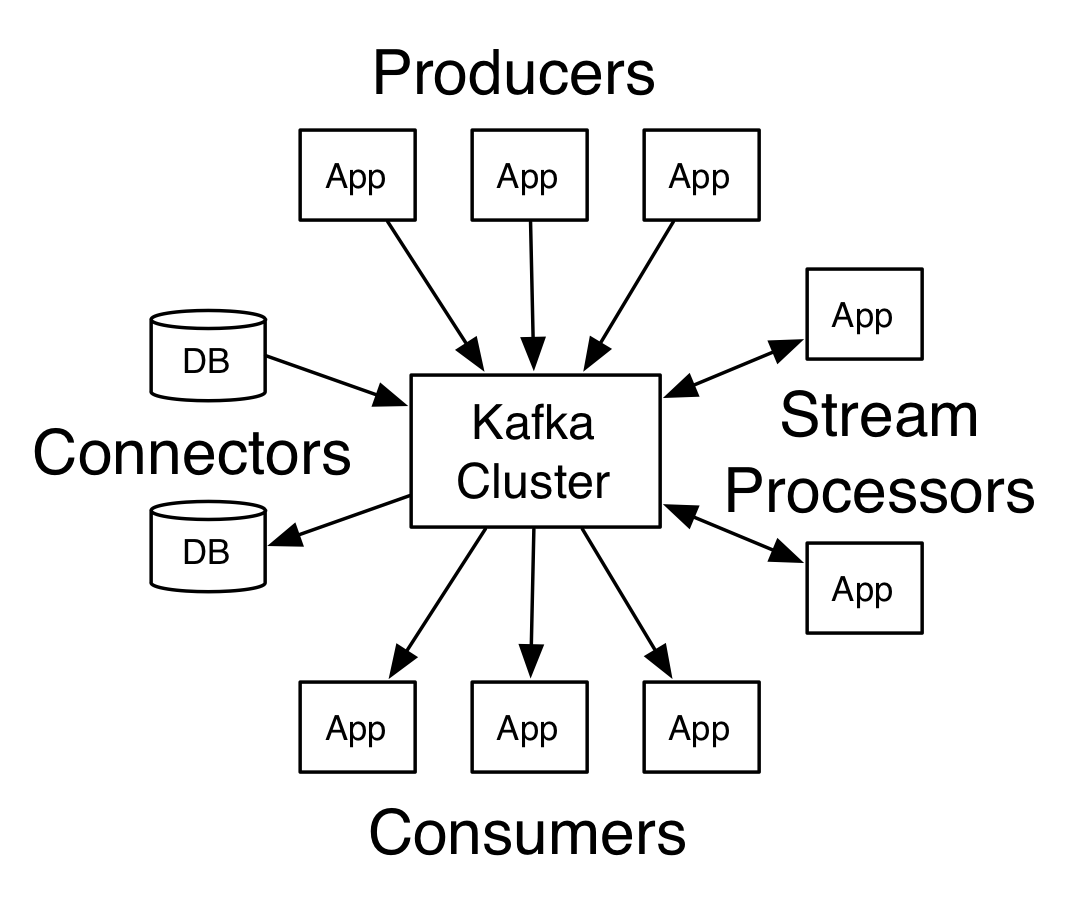
* NoSQL is a term used to describe high-performance, non-relational databases.
* NoSQL databases utilize a variety of data models, including document, graph, key-value, and columnar.
* NoSQL databases are widely recognized for ease of development, scalable performance, high availability, and resilience. Below are several resources to help you get started using NoSQL databases.

**Kafka**

* Kafka is run as a cluster on one or more servers.
* The Kafka cluster stores streams of *records* in categories called *topics*.
* Each record consists of a key, a value, and a timestamp.
* In Kafka the communication between the clients and the servers is done with a simple, high-performance, language agnostic [TCP protocol](https://kafka.apache.org/protocol.html).
* This protocol is versioned and maintains backwards compatibility with older version. We provide a Java client for Kafka, but clients are available in [many languages](https://cwiki.apache.org/confluence/display/KAFKA/Clients).

**Kafka has four core APIs:**

* The [Producer API](https://kafka.apache.org/documentation.html#producerapi) allows an application to publish a stream of records to one or more Kafka topics.
* The [Consumer API](https://kafka.apache.org/documentation.html#consumerapi) allows an application to subscribe to one or more topics and process the stream of records produced to them.
* The [Streams API](https://kafka.apache.org/documentation/streams) allows an application to act as a *stream processor*, consuming an input stream from one or more topics and producing an output stream to one or more output topics, effectively transforming the input streams to output streams.
* The [Connector API](https://kafka.apache.org/documentation.html#connect) allows building and running reusable producers or consumers that connect Kafka topics to existing applications or data systems. For example, a connector to a relational database might capture every change to a table.



**Ajax**

Ajax (Asynchronous Javascript and XML) allows developers to build highly interactive web applications. Ajax allows a web client to communicate with a server asynchronously. Among other things, Ajax can be used to provide an update to part of a web page without requiring the whole page to be refreshed. Popular Ajax libraries include jQuery and Dojo.

**How do I display file upload progress using the Dojo ProgressBar?**

we created a TestProgressListener class to allow us to monitor the progress of a file upload to a servlet[W](http://en.wikipedia.org/wiki/Java_Servlet). We utilized the Apache[S](http://www.apache.org/)[W](http://en.wikipedia.org/wiki/Apache_Software_Foundation) Commons FileUpload[S](http://commons.apache.org/fileupload/) library to handle the file upload. The TestProgressListener class implements the ProgressListener interface from the FileUpload library. The file gets uploaded to the TestServlet class, which sticks a reference to the TestProgressListener in a session. The ProgressServlet reads the TestProgressListener object from the session and displays the status of the file upload. The upload form to upload files to the TestServlet is located on upload.jsp.

I downloaded the Dojo Ajax[W](http://en.wikipedia.org/wiki/Ajax_%28programming%29) library from <http://dojotoolkit.org/>. I unpacked the library and placed it in the web directory of my project. In a production system, I would place the library somewhere on a web server rather than actually packaging it into a project, but this is fine for demonstration purposes.

Clicking the Upload button submits the form and calls the doProgress() function which disables the Upload button and initializes the max, prog, and counter variables. The max variable is the maximum progress, which is 100 (100%). The prog variable is the upload progress, which starts at 0 (0%). The counter variable is a simple counter variable. The getProgress() function is called and then the doProgressLoop() function is called.

The getProgress() function contacts the ProgressServlet (via the 'progress' URL) and puts the results in the 'progress-content' span element.

The doProgressLoop() is a recursive function that calls itself until the file upload is complete. It reads the 'progress-content' span element, which gets its innerHTML value from the ProgressServlet via the getProgress() call. If the 'progress-content' value is an integer, the prog (progress) variable is updated with this value. The ProgressBar (jsProgress) is updated with the current progress via the jsProgress.update call. The counter is incremented and its value is placed in the 'counter' span element. If the prog (progress) variable is less than 100, meaning that the file upload hasn't completed yet, then getProgress() is called in 500 milliseconds, and doProgressLoop() is called in 1000 milliseconds (1 second) with the current prog, max, and counter values.

The doProgressLoop() will continue to call itself until the progress is 100%, meaning that the file upload has completed. At this point, the TestServlet will display another page, since the upload will be complete.

**Struts Tutorials**

Apache Struts is a Java-based framework for developing Java web applications.

JSON or JavaScript Object Notation is a lightweight text-based open standard designed for human-readable data interchange. Conventions used by JSON are known to programmers, which include C, C++, Java, Python, Perl, etc.

* JSON stands for JavaScript Object Notation.
* The format was specified by Douglas Crockford.
* It was designed for human-readable data interchange.
* It has been extended from the JavaScript scripting language.
* The filename extension is **.json**.
* JSON Internet Media type is **application/json**.
* The Uniform Type Identifier is public.json.

## **Uses of JSON**

* It is used while writing JavaScript based applications that includes browser extensions and websites.
* JSON format is used for serializing and transmitting structured data over network connection.
* It is primarily used to transmit data between a server and web applications.
* Web services and APIs use JSON format to provide public data.
* It can be used with modern programming languages.

## **Characteristics of JSON**

* JSON is easy to read and write.
* It is a lightweight text-based interchange format.
* JSON is language independent.

# Hibernate and Spring Integration

We can simply integrate **hibernate application with spring application**.

In hibernate framework, we provide all the database information hibernate.cfg.xml file.

But if we are going to integrate the hibernate application with spring, we don't need to create the hibernate.cfg.xml file. We can provide all the information in the applicationContext.xml file.

The Spring framework provides **HibernateTemplate** class, so you don't need to follow so many steps like create Configuration, BuildSessionFactory, Session, beginning and committing transaction etc.

Spring annotations:

Starting from Spring 2.5 it became possible to configure the dependency injection using **annotations**. So instead of using XML to describe a bean wiring, you can move the bean configuration into the component class itself by using annotations on the relevant class, method, or field declaration.

Annotation injection is performed before XML injection. Thus, the latter configuration will override the former for properties wired through both approaches.

Annotation wiring is not turned on in the Spring container by default. So, before we can use annotation-based wiring, we will need to enable it in our Spring configuration file. So consider the following configuration file in case you want to use any annotation in your Spring application.

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

<beans xmlns = "http://www.springframework.org/schema/beans"

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

xmlns:context = "http://www.springframework.org/schema/context"

xsi:schemaLocation = "http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-3.0.xsd

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context-3.0.xsd">

<context:annotation-config/>

<!-- bean definitions go here -->

</beans>

Once <context:annotation-config/> is configured, you can start annotating your code to indicate that Spring should automatically wire values into properties, methods, and constructors. Let us look at a few important annotations to understand how they work −

|  |  |
| --- | --- |
| **Sr.No.** | **Annotation & Description** |
| 1 | [**@Required**](https://www.tutorialspoint.com/spring/spring_required_annotation.htm)  The @Required annotation applies to bean property setter methods. |
| 2 | [**@Autowired**](https://www.tutorialspoint.com/spring/spring_autowired_annotation.htm)  The @Autowired annotation can apply to bean property setter methods, non-setter methods, constructor and properties. |
| 3 | [**@Qualifier**](https://www.tutorialspoint.com/spring/spring_qualifier_annotation.htm)  The @Qualifier annotation along with @Autowired can be used to remove the confusion by specifiying which exact bean will be wired. |
| 4 | [**JSR-250 Annotations**](https://www.tutorialspoint.com/spring/spring_jsr250_annotations.htm)  Spring supports JSR-250 based annotations which include @Resource, @PostConstruct and @PreDestroy annotations. |

[Spring MVC](https://howtodoinjava.com/interview-questions/spring-mvc-interview-questions-with-answers/)

https://howtodoinjava.com/interview-questions/spring-mvc-interview-questions-with-answers/

## **JAX-RS Annotations**

The **javax.ws.rs** package contains JAX-RS annotations.

|  |  |
| --- | --- |
| **Annotation** | **Description** |
| Path | It identifies the URI path. It can be specified on class or method. |
| PathParam | represents the parameter of the URI path. |
| GET | specifies method responds to GET request. |
| POST | specifies method responds to POST request. |
| PUT | specifies method responds to PUT request. |
| HEAD | specifies method responds to HEAD request. |
| DELETE | specifies method responds to DELETE request. |
| OPTIONS | specifies method responds to OPTIONS request. |
| FormParam | represents the parameter of the form. |
| QueryParam | represents the parameter of the query string of an URL. |
| HeaderParam | represents the parameter of the header. |
| CookieParam | represents the parameter of the cookie. |
| Produces | defines media type for the response such as XML, PLAIN, JSON etc. It defines the media type that the methods of a resource class or MessageBodyWriter can produce. |
| Consumes | It defines the media type that the methods of a resource class or MessageBodyReader can produce. |
|  |  |