PROJECT REPORT ON

ONLINE ASSESSMENT RESTful WEB APPLICATION USING JWT BASED AUTHENTICATION

Submitted in partial fulfilment of the requirements for the award
of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
OF
SASTRA UNIVERSITY

Submitted by
PENUKONDA SAI NAVYA SREE 117003142

Under the Guidance of
ANUJA RANI,SASIDHARAN
iNautixTechnologies,Chennai

SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAMTHANJAVUR – 613 401
April 2017

SCHOOL OF COMPUTING

SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM, THANJAVUR – 613401

BONAFIDE CERTIFICATE

Certified that this project work entitled "**ONLINE ASSESSMENT RESTful WEB APPLICATION USING JWT BASED AUTHENTICATION**" submitted to the Shanmugha Arts, Science, Technology & Research Academy (SASTRA University), Tirumalaisamudram-613401 by **PENUKONDA SAI NAVYA SREE - 117003142**in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** is the original and independent work carried out under my guidance, during the period December 2016 - April 2017.

[illegible]

EXTERNAL GUIDE
ANUJA RANI,SASIDHARAN,
iNautix Technologies,Chennai.
asasidharan@inautix.co.in

ASSOCIATE DEAN
Dr.A. UMAMAKESWARI
SCHOOL OFCOMPUTING

Submitted for University Examination held on______________________

EXAMINER - I

EXAMINER - II

SCHOOL OF COMPUTING

SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM, THANJAVUR – 613401

I submit this project work entitled "**ONLINE ASSESSMENT RESTful WEB APPLICATION USING JWT BASED AUTHENTICATION**" to the Shanmugha Arts, Science, Technology & Research Academy (SASTRA) University, Tirumalaisamudram–613401,in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** and declare that it is our original and independent work carried out under the guidance of **ANUJA RANI,SASIDHARAN**, Learning & Development Team, iNautix Technologies, Chennai.

Place : Chennai, **Name:**P. Sai Navya Sree Signature: P. Navya

Date:06-04-2017. **Reg.no:**117003142

First and Foremost, we take pride in thanking the almighty to gave us strength for the successful completion of this project.

We have immense pleasure in expressing my heartfelt thanks to our Vice-Chancellor **prof.R.Sethuraman**for the benevolent advice and guidance during my tenure in the college.

We would like to express our gratitude to **Dr.Balachandran, Registrar,**SASTRA UNIVERSITY, for his benevolent guidance through-out my college life.

We wish to express our thanks to **Dr.S.Vaidhyasubramanian, Dean of Planning and Development** for his encouragement and providing us with all the needed amenities.

We would like to express our gratitude to **Dr.A.Umamakeshwari,** Associate Dean **,** Computer Science and Engineering,School of Computing, SASTRA UNIVERSITY, for her benevolent guidance through-out my college life.

We wish to express our thanks to **Anuja Rani, Sasidharan,** Learning & Development Team**, iNautix Technologies, Chennai** for her encouragement and providing us with all the needed amenities.

We deeply thank our family and friends for supporting me in all the tasks that I have carried for the successful completion of this project.

TABLE OF FIGURES USED IN THIS PROJECT:

TABLE OF CONTENTS

1)ABSTRACT

This project is aimed to develop a web based assessment application using Rest API web Services using JWT based authentication. Various Assessment applications are currently being used in the process of recruiting the employees for an organization and they are paying a huge amount for that to the Third party vendor. Assessment Application is a web service based one, which is having further more features than the third party vendor applications, and also in which it supports customization based on the organization's present needs which will be beneficial to the organization in long run.

The interface is designed in such a way that it enables candidates and organization to have efficient and user-friendly interaction by using latest technologies like AngularJS which makes UI high responsive.

The application is designed using Core Java and latest technologies like AngularJS which modifies DOM directly and makes it highly responsive instead of modifying inner HTML code.Project has been updated to MAVEN which helps in obtaining package dependencies easily by building WAR files which speeds testing and helps in maintaining standard directory structure.

JSON Web Tokens(JWT) based Authentication is a standard which helps in secure transmission of information between parties as a JSON Object.The basic objective of JSON web token based authentication is to enhance the transmission of credentials between the client and the server by overcoming the disadvantages of traditional session and cookie creation procedures.This mechanism is stateless authentication which uses HMAC algorithm which involves cryptographic hash function.

Spring MVC Framework has been used which achieves loose coupling through Dependency Injection and it also provides extensible JDBC abstraction framework.

RESTful API allows manipulation of objects through URL which acts as the identifier of every resource which makes easy access of application. A WAR file is created and deployed in Tomcat Server. The challenging part after developing the application is to move into other platform which helps the end user to use other similar applications signed in smoothly.

2)INTRODUCTION

OVERVIEW:

JWTokens have a established claims as an object of Json which has the structure of encoded JWE or JWS. This Json object is the JWT Claims has this object. The names represented by JW Token Claims Set are claim names and values corresponding to these names can be called as claim values. Cryptographic functions smeared to the JWToken claims are described in the header of Jose by contents in it.

Nested signing and nested encryption can be envoked by a single JWToken which is inside another JWS (or) JWE layout to build nested JWToken. JWToken is symbolised as a URL with sequence in which parts are detached by a single period '.' typescripts. Every part of this URL stores the encoded value of base64url. The representation of JWE or JWS (compact serialization) signifies the number of parts that the URL of JWToken must contain.

Assessment Application is used for the purpose of conducting assessments which helps in understanding the knowledge gained by the users using it. It will be much more useful in the recruitment process of any organization and the specialty of using it is customization based on the present needs of the organization and it leads to the productive development of the organization. This application uses Json token when a user wants to access protected route or resource instead of creating a session id in server and returning cookies.

OBJECTIVES:

To develop an Assessment Application which provides

- Organisation based customization.
- Upgradability with upcoming database technologies.
- Lightweight processing.
- Helps in easy recruitment and organization development process.
- Concurrency & Backup Option.
- Loose Coupling by implementing Dependency Injection using Spring MVC Architecture.
- Secure transformation of information between parties through JWT based authentication.

- Both external and internal attacks are prevented by establishing a secure connection which helps to create a database which is strong and secure.
- High availability or disaster recovery.
- Storage and compression.
- Ease of integration with external libraries.
- Integration with reporting and archive tools.

3)PROBLEM STATEMENT

When we want to build our own Application Programming Interfaces, questions over security concerns of API's used arises which are solved using Json web tokens. JWTokens can be implemented across different platforms (.net , Python, Java, Node.js, Ruby, PHP, Go, Haskell and JavaScript). This is being used in different scenarios now-a-days. Mainly, JWT will be beneficiary for SSO (Single Sign On) Property. Different applications uses different authentication mechanisms, so SSO must store the login credentials during initial authentication and translating them whenever required to other domains. JWT's are selfcontained, which carries all the information within it.

4)LITERATURE SURVEY:

EXISTING SYSTEM:

The existing thirdparty applications are designed in a fixed way so that the organizations needs to pay alone for their customized designing and anyspecialization that need to be wished by the organization should be done by the vendor providing it,any changes in the application must be carried out only with the help and support of the vendor providing the application.

JWT:

This includes both cryptographic algorithms and an identifier to be castoff with Json web encryption (JWE) [JWE], Json web signature (JWS) [JWS], Json web key (JWK) [JWK] stipulations. The identifiers c are represented by the definitions of IANA archives. JSON-based [RFC7159] data structures are utilized by these specifications. The algorithms and identifiers remain constant when they are registered here rather than in any other specifications (JWE/JWK/JWS). Changes can be done in the specifications(JWE/JWK/JWS) without actually changing the corresponding document.

JSON Web Encryption-M.Jones and J.Hilderbrand

Json web encryption (JWE) uses Json-based data structures (RFC7159) to represent the encrypted data. Sequence of octets are secured and protected by cryptographic tools of Jwe.

The two serializations which are meticulously associated to JWE's are well-defined below. Jwe Serialization is a compressed and URL-secure depiction which is meant for Uri query constraints and space inhibited situations like Http authorization header. The Serialization techniques signifies Jwe's as Json objects and multiple parties are encrypted to the same content. The common thing which is shared by both is that underpinnings of cryptography are same. Separate Json web algorithms specifications is used to explain the use of identifiers and cryptographic algorithms. Associated MAC capabilities and digital signature are depicted in distinct Json Web Signature specification. Names defined in this specification are short because representations should be compact.

DRAWBACKS OF EXISTING SYSTEM

- **PRICE:**

The costs of the third party vendor applications are cost wise higher and even though the organization gets the service, these organizations cannot be able to work independently on its own pace. Any modifications in those applications can be done only with the vendor support.

- **INABILITY TO EXPAND:**

There are several applications which uses SQL database and it has the inability to withstand when the number of records exceeds certain limits, the ability to retrieve gets slow and as it is web service based application the user may have a slower response as the database gets loaded.

- **MIGRATION TO OTHER DATABASE:**

There are so many third party applications which doesn't supports No SQL databases, it cannot be further scaled up to a higher data contented database and it will not be useful for the organization in a long term use.

PROPOSED SYSTEM:

The proposed system is based on completely supportable to the organization in all cases. The proposed system supports No SQL database as it uses java based spring REST web services.

The customization of organization based assessment can be done with the proposed system.

The proposed system is lightweight and it doesn't get slow over loading of data.

FEATURES OF PROPOSED SYSTEM:

- **Light in weight:**

Transparency and size of spring are frivolous. The elementary sort of spring framework is about 2MB.

- **Inversionof control:**

Inversionof control helps in achieving the loose coupling. It minimizes the very amount of code in your application.

- **Dependency Injection:**

You just describe the process to create the objects but not actually create them. A configuration file acts as a medium to describe the services needs of components and there is no necessity to directly connect them.

- **Container:**

Application object's conformations and life cycles are controlled and coped in the Spring.

5)SOFTWARE/HARWARE REQUIREMENT SPECIFICATION

HARDWARE REQUIREMENT SPECIFICATION:

Processor	Intel core 2 duo and advance
Speed	2.0 GHz
Hard Disk Drive	250 GB and above.
Operating System	Windows, linux
Memory	2 GB RAM and above
System Type	32,64 bit Operating System

HARDWARE-INTERFACES:

Server-side hardware:

- Hardware good and prescribed by all the product utilized
- Communication equipment which is utilized to perform customer demands.

Client-side hardware:

- Hardware utilized by individual customer's side Operating framework and web program.
- Communication equipment to recover information from server.

SOFTWARE-INTERFACES:

Server-side software:

- Apache Tomcat Server v7.0
- Server side scripting tools : JSP
- Database tools : Derby 10.1
- Compatible Operating System :Windows **Client-side software:**
- Web browser supporting JavaScript

COMMUNICATION PROTOCOL:

Server-side :

HTTP incoming request and HTTPS incoming request for secure gateway.

Client-side :

HTTP outgoing request and HTTPS outgoing request for secure gateway.

SOFTWARE REQUIREMENTS SPECIFICATION:

1)Apache Server Installation:

Programming some portion of the venture for the most part includes in network between web application and Tomcat Server. Apache tomcat is an open---source web server and servlet container.It requires a Java Standard Edition RuntimeEnvironment (JRE) adaptation 6 or later.

STEPS TO INSTALL:

1. Download and introduce JRE from http://www.oracle.com/technetwork/java/javase/downloads/index.html.

2. Download and introduce Apache Tomcat, a double appropriation of tomcat from http://tomcat.apache.org/

3. Unpack the double appropriation with the goal that it lives in its own index (expectedly named "apache-tomcat-[version]").

4. Configure Environment Variables

4.1.1. Set CATALINA_HOME and CATALINA_BASE(optional).The CATALINA_HOME condition variable ought to be set to the area of theroot index of the "parallel" appropriation of Tomcat.

4.1.2. The CATALINA_BASE condition variable indicates area of the rootdirectory of the "dynamic setup" of Tomcat. It is discretionary. It defaults to be equivalent to
CATALINA_HOME.

5. Set JRE_HOME or JAVA_HOME.

5.1.1. The JRE_HOME variable is utilized to indicate area of a JRE. The JAVA_HOME variable is utilized to determine area of a JDK.

5.1.2 . Using JAVA_HOME gives access to certain extra startup choices that are not permitted when JRE_HOME is used.If both JRE_HOME and JAVA_HOME are determined, JRE_HOME is used.The best place to incorporate these factors is a "setenv" script.

6. Other Variables like CATALINA_OPTS are discretionary to set with. It permits particular of extra alternatives for the java charge to begin tomcat.

7. Start Tomcat

7.1.1. On Windows

%CATALINA_HOME%\bin\startup.bat

or, on the other hand

%CATALINA_HOME%\bin\catalina.bat begin

or, on the other hand

$CATALINA_HOME/container/catalina.sh begin

7.1.2. After startup, the default web applications included Tomcat will be accessible by going to:

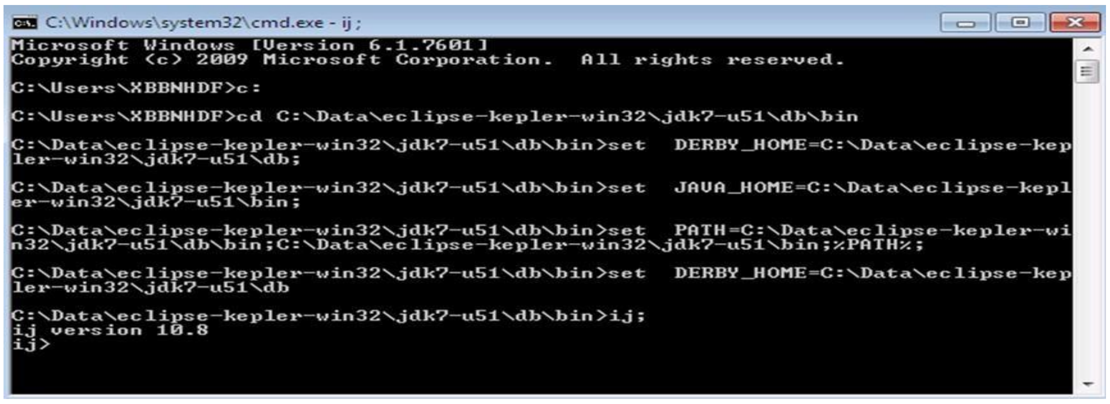
http://localhost:8080/

8. Shut Down Tomcat

<u>2)DERBY DATABASE INSTALLATION:</u>

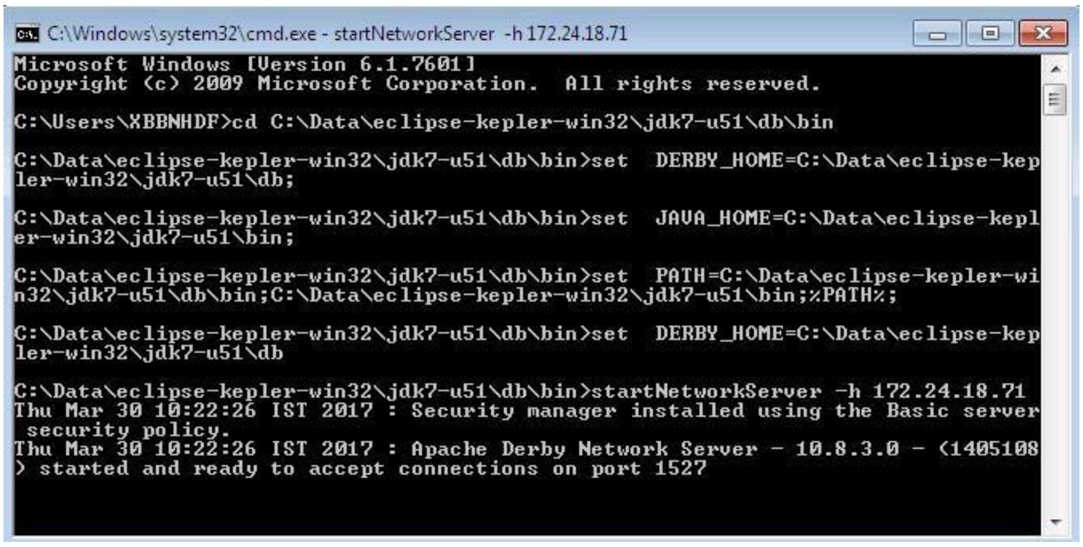
Apache Derby is a social database administration framework (RDBMS) created by Apache Software Foundation which is implanted in Java programs and utilized for online exchange handling. It has a 2.6 MB plate space impression.

The Derby arrange server expands the scope of database engineby giving essential customer server usefulness. The server permits customers to associate over TCP/IP utilizing the standard DRDA convention. Derby joins Java 7 and has been named as "JavaDB" however it is exactlythe same piece for-bit as Derby may be. For engineers with Java 6, they can even now download Derby as some time recently, however for designers utilizing JRE 7 or later, Derby is incorporated into the Java API.



C:\Windows\system32\cmd.exe - ij ;

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\XBBNHDF>c:

C:\Users\XBBNHDF>cd C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  DERBY_HOME=C:\Data\eclipse-kep
ler-win32\jdk7-u51\db;

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  JAVA_HOME=C:\Data\eclipse-kepl
er-win32\jdk7-u51\bin;

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  PATH=C:\Data\eclipse-kepler-wi
n32\jdk7-u51\db\bin;C:\Data\eclipse-kepler-win32\jdk7-u51\bin;%PATH%;

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  DERBY_HOME=C:\Data\eclipse-kep
ler-win32\jdk7-u51\db

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>ij;
ij version 10.8
ij>
```



C:\Windows\system32\cmd.exe - startNetworkServer -h 172.24.18.71

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\XBBNHDF>cd C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  DERBY_HOME=C:\Data\eclipse-kep
ler-win32\jdk7-u51\db;

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  JAVA_HOME=C:\Data\eclipse-kepl
er-win32\jdk7-u51\bin;

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  PATH=C:\Data\eclipse-kepler-wi
n32\jdk7-u51\db\bin;C:\Data\eclipse-kepler-win32\jdk7-u51\bin;%PATH%;

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>set  DERBY_HOME=C:\Data\eclipse-kep
ler-win32\jdk7-u51\db

C:\Data\eclipse-kepler-win32\jdk7-u51\db\bin>startNetworkServer -h 172.24.18.71
Thu Mar 30 10:22:26 IST 2017 : Security manager installed using the Basic server
 security policy.
Thu Mar 30 10:22:26 IST 2017 : Apache Derby Network Server - 10.8.3.0 - (1405108
) started and ready to accept connections on port 1527
```

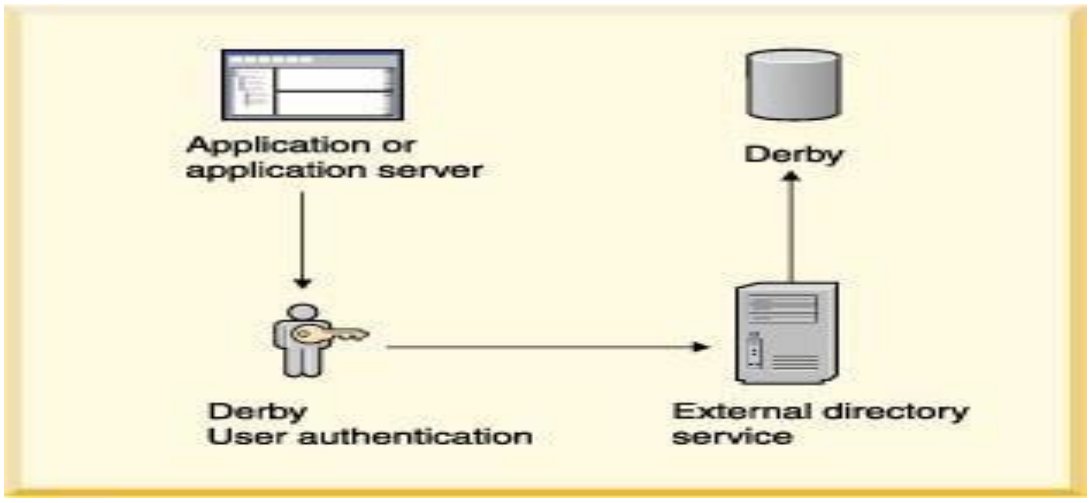


Fig 5.1 Derby Data Base Design

3)npm and Node.js Installation:

Node.js and npm are essential to modern web development with Angular and other platforms. Node powers client development and build tools. The npmpackage manager, itself a node application, installs JavaScript libraries.

The major advantage using AngularJs 2 is to have configuration files in Typescript which will convert to javascript file when loaded. This particular feature enables us to write the code as we write in Java classes. TypeScript is superset of Javascript which enhances the language with type annotations. These Type annotations are used to detect errors easily before they happen at runtime.

<u>Version:</u> node --
4.x.x / Higher npm

--3.x.x / higher

The versions of node and npm can be checked using following commands

: **node -v npm**

–v

Older versions produce errors.

After installing node and npm, the following steps are performed

1. Create a project folder named quickstart.
2. Clone the QuickStart seed into project folder.
3. Install npm packages.
4. Run npm start to launch the sample application.

git clone https://github.com/angular/quickstart.git quickstart

The above command is given in GIT bash console to clone the quickstart folder from the GitHub link. We can also download it separately and save it in a folder.

cdquickstart

Switch to the quickstart folder **npm**

install

Install all the packages necessary to run Angular JS 2 application **npm start**

Once the command is executed, the npm server is started and https://localhost:8080 path is set by default.

The general code structure looks like:

-src

-app

-app.component.ts

-app.module.ts

-main.ts app.component.ts

:

It is the root component of what will become a tree of nested components as the application evolves. **app.module.ts:**

Defines AppModule, the root module that tells Angular how to assemble the application.

Right now it declares only the AppComponent. **main.ts:**

Compiles the application with the JIT compiler and bootstraps the application's main module (AppModule) to run in the browser.

The JIT compiler is a reasonable choice during the development of most projects and it's the only viable choice for a sample running in a *live-coding* environment.

Bootstrap framework:

Angular Js 1 uses ng-app directives to point that Angular content has started whereas Angular 2 uses Bootstrapper framework.

The main module contains the bootstrapping code for your Angular 2 application.

Bootstrap is an open-source gathering of devices which is utilized as a part of making sites and web applications. It contains HTML and CSS based plan layouts for structures, catches, and route et cetera, and also discretionary Java Script augmentations. This expects to facilitate the advancement of element sites and furthermore web applications.

Bootstrap is a front end web system which is an interface for the client, not at all like the server-side code which lies on the "back end" or server. Bootstrap accompanies a few
JS(JavaScript) segments as jQuery modules. They give extra UI(user interface) components, for example, discourse boxes, tooltips, et cetera. They additionally amplify the usefulness of existing interface components, including auto-finish work for info fields. The accompanying JavaScript modules are upheld by bootstrap: Dropdown, Scroll spy, Modal, Tab, Tooltip, Popover, Collapse, Alert, Carouse, lButton and Type ahead.

ADVANTAGES OF BOOTSTRAP

- Ease of Use
- Highly Flexible
- Responsive Grid
- Comprehensive List of Components
- Leveraging JavaScript Libraries
- Frequent Updates
- Detailed Documentation and Vast Community

- Consistency

The following configuration files are necessary for Spring Framework:

Pom.xml

Spring-Servlet.xml

Web.xml

Pom.xml:

This file contains all the dependencies that are required to get the application work. The code dependencies and the properties tag are given below.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"

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

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/maven- v4_0_0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>pledge</groupId>

<artifactId>pledge.test</artifactId>

<packaging>war</packaging>

<version>0.0.1-SNAPSHOT</version>

<name>Test Pledge MavenWebapp</name>

<url>http://maven.apache.org</url>

<properties>

<jdk.version>1.7</jdk.version>

<spring.version>4.1.1.RELEASE</spring.version>
```

```
<jstl.version>1.2</jstl.version>

<junit.version>4.11</junit.version>

<logback.version>1.0.13</logback.version> <jcl-over-
slf4j.version>1.7.5</jcloverslf4j.version> </properties>

<dependencies>

<!-- Unit Test -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

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

</dependency>

<!-- Spring Core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

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

<exclusions>

<exclusion>

<groupId>commons-logging</groupId>

<artifactId>commons-logging</artifactId>

</exclusion>

</exclusions>

</dependency>
```

```
<dependency>

<groupId>org.slf4j</groupId>

<artifactId>jcl-over-slf4j</artifactId>

<version>${jcl-over-slf4j.version}</version>

</dependency>

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>${logback.version}</version> </dependency>

<dependency>  <groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

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

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

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

</dependency>

<!-- jstl -->

<dependency>

<groupId>jstl</groupId>

<artifactId>jstl</artifactId>

<version>${jstl.version}</version>
```

```
</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

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

</dependency>

<!-- Jackson JSON Processor -->

<dependency> <groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>  <version>2.4.1</version>

</dependency>

</dependencies>

<build>

<finalName>Test</finalName>

</build>

</project>
```

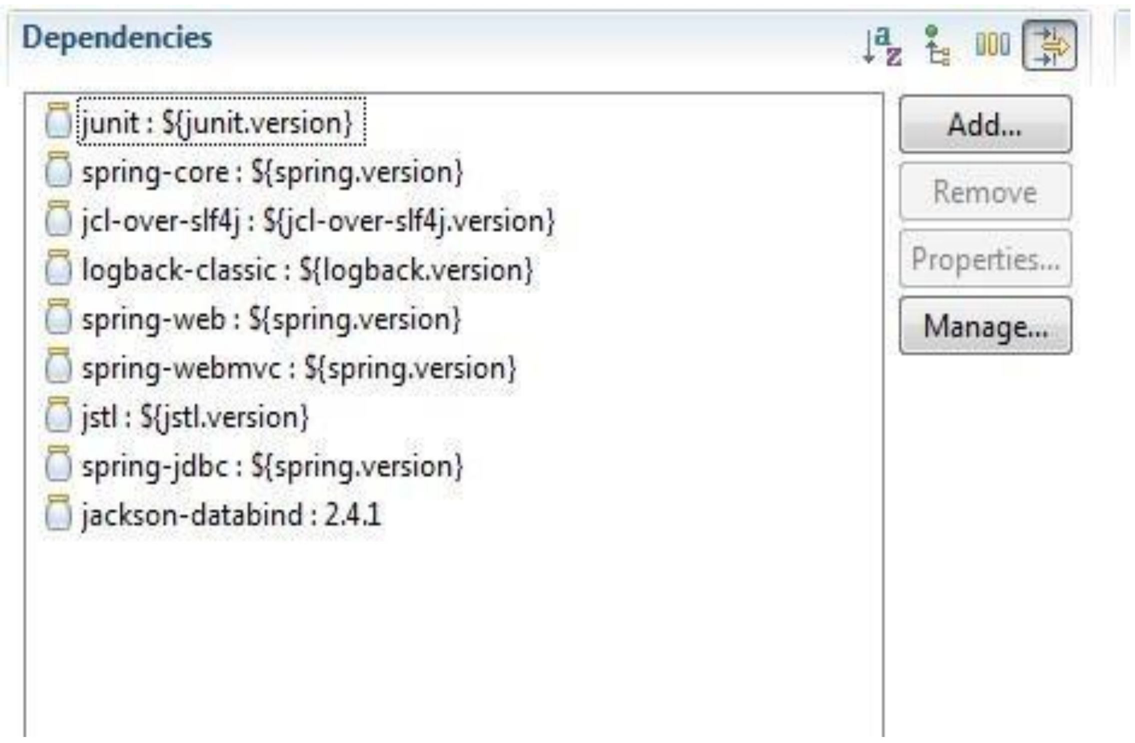
The below code is responsible for invoking rest services from

Angular JS 2

As shown in the above snippet, the AutoAuthenticator class which calls the rest service using the url and gets the list of records in JSON format and return it to the called method. This result is printed using console.log() method.

Maven Dependency :

Maven is a powerful tool that allows users to import dependencies into their software projects and also automatically manage transitive dependencies. In order to use Maven, it is necessary to explicitly add dependencies to the Maven pom.xml file. Once added to the Maven pom.xml file, dependencies will be automatically downloaded, updated, and have their sub-dependencies managed by Maven.



```
junit : ${junit.version}
spring-core : ${spring.version}
jcl-over-slf4j : ${jcl-over-slf4j.version}
logback-classic : ${logback.version}
spring-web : ${spring.version}
spring-webmvc : ${spring.version}
jstl : ${jstl.version}
spring-jdbc : ${spring.version}
jackson-databind : 2.4.1
```

The above screenshot contains some of the core dependency injected and each one is responsible for importing some specific packages. For example, Jackson-databind will import the packages at run time which will convert list into JSON objects and vice versa.

Spring-servlet.xml:

```
<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"

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

xmlns:tx="http://www.springframework.org/schema/tx"
```

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

http://www.springframework.org/schema/mvc/spring-mvc.xsd

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

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

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

http://www.springframework.org/schema/tx/spring-tx-4.1.xsd

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

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

</beans>
```

The above shown configuration file contains a beans tag which is used to download the required packages from the url provided.

6) Conceptual Model / Proposed Architecture

SPRING MVCARCHITECTURE:

It offers model---view---controller structural design and in-built components which could be helpful to develop supple and loosely coupled web based applications. The pattern of MVC assists in unravelling the different parts of this application [User Interactive ,input and business logics], by establishing a loose coupling amongst these units.

Model(M) – Stores the Pojo and aslo encapsulates the whole data that the allpication contains.

View(v) –Generates the client side HTML data that any browser can interpret responsible for rendering the model data and in general it generates HTML output that the client's browser can interpret.

Controller - responsible for processing user requests and building appropriate model and passes it to the view for rendering.

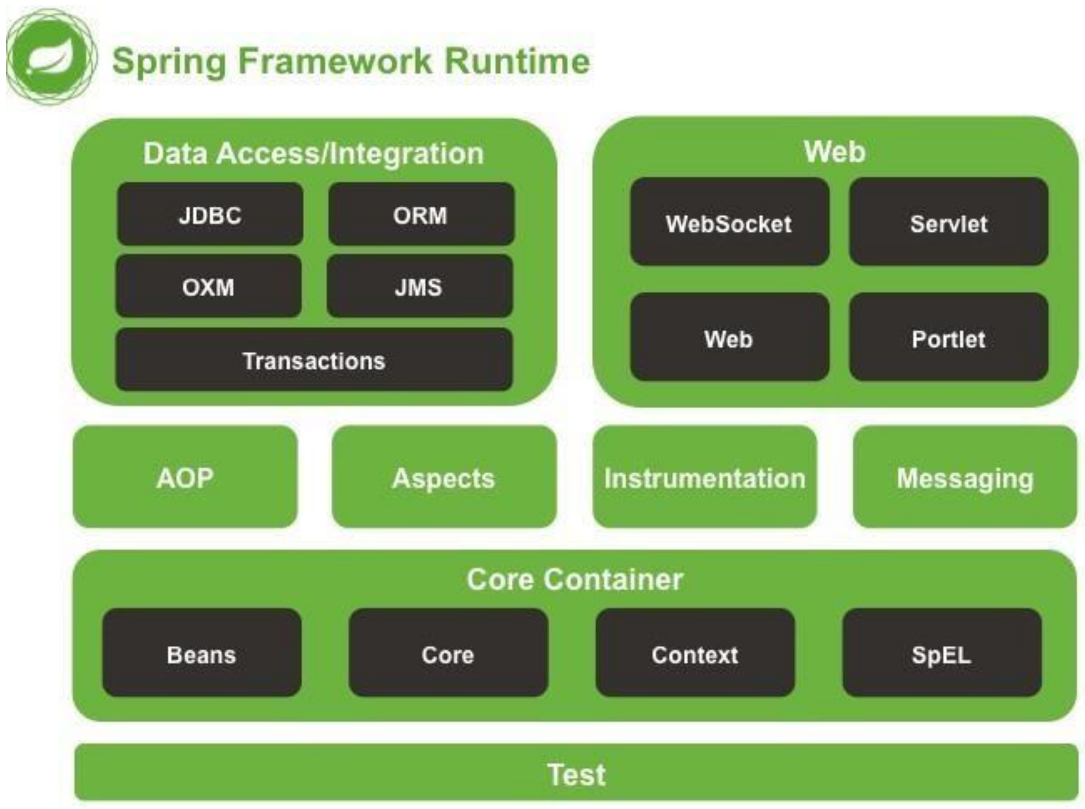


Fig 6.1 Spring Framework Runtime Diagram

<u>Advantages OF SPRING FRAMEWORK:</u>

- Spring helps designers to create endeavor class applications utilizing POJOs. The benefit of utilizing just POJOs is that you require have an EJB compartment item, for example, an application server rather utilize just a vigorous servlet holder, for example, Tomcat Server. Spring is composed in a measured structure.
- Spring genuinely makes utilization of a portion of the current advancements like ORM structures, logging systems, JEE, Quartz and JDK clocks.
- Spring applications are anything but difficult to test and basic since condition subordinate code is moved into system. Likewise, by utilizing Java Bean-style POJOs, it winds up noticeably simpler to utilize reliance infusion for infusing test information.

- Spring gives an appropriate and perfect API to decipher innovation particular special cases (tossed by JDBC, Hibernate, or JDO) into reliable, unchecked exemptions.
- Lightweight IoC compartments are helpful for creating and sending applications on frameworks with restricted memory and CPU assets.
- Spring gives steady exchange administration interface which downsizes to a nearby exchange (utilizing a solitary database, for instance) and scales up to worldwide exchanges (utilizing JTA, for instance).
- Spring Session gives an API and executions to deal with client's session data. It additionally furnishes straightforward incorporation with:
    1. Http Session - replaces the Http Session in an application holder (i.e.Tomcat)
    2. Clustered Sessions - Spring Session bolster bunched sessions without being associated with application holder particular arrangement.
    3. Multiple Browser Sessions - Spring Session bolsters dealing with numerous clients' sessions in a solitary program occurrence.
    4. RESTful APIs - Spring Session gives session ID in headers to work with RESTful APIs.
    5. Web Socket - keeps the Http Session alive when receiving Web Socket messages.

SPRING IOC CONTAINERS:

The spring compartment is the center component of Spring Framework. The compartment makes objects, wire them together, design them, and deal with their total lifecycle from creation till obliteration.

The spring holder utilizes reliance infusion (DI) to deal with the segments which constitute an application. These articles are called Spring Beans .The holder gets data on what items to make, design, and amass from setup metadata gave which can be spoken to either by XML, Java explanations, or Java code. The accompanying graph is an abnormal state perspective of how spring functions. The Spring IoC compartment makes utilization of POJO classes and arrangement metadata to create a totally designed and executable application.

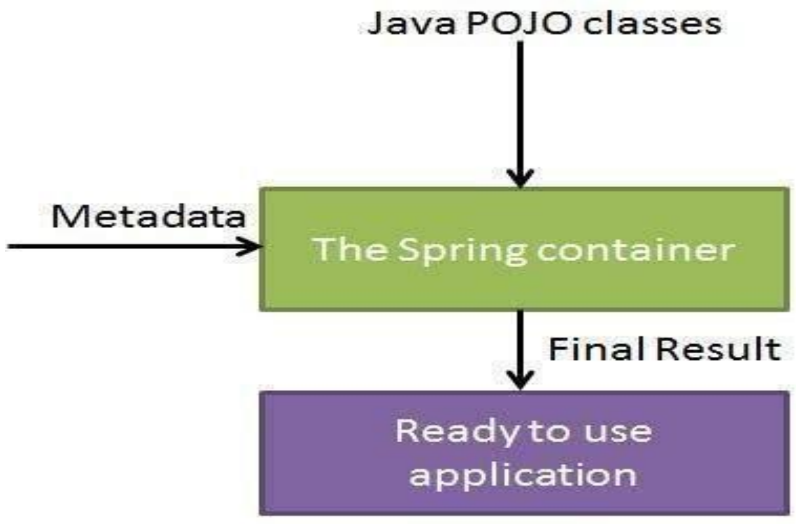


Fig 6.2 Spring IoC Container Layout Diagram

DEPENDENCY INJECTION(DI):

The spring system is perceived and utilized broadly to have Dependency Injection (DI) as kind of Inversion of Control. Reliance Injection is a solid case of Inversion of Control.

When we assemble complex Java application, DAO classes ought to be autonomous as conceivable of other Java classes to expand the reusability and to test freely Dependency Injection helps in interfacing these classes together and same time keeping them autonomous.

Reliance is something which converts into a relationship between two classes. For instance, class An is subject to class B. Presently, Injection is that class B will get infused into class A by the IoC. Reliance infusion can be starting at passing parameters to the constructor or by post-development utilizing setter techniques.

Perspective - ORIENTED PROGRAMMING:

Perspective arranged programming (AOP) is one of the key segments of spring structure. The capacities which traverse numerous purposes of an application are called crosscutting concerns and they are hypothetically separate from the application's business rationale. There are various cases of perspectives including logging, definitive exchanges, security, and reserving.

The key part of measured quality in OOP is the class, though in AOP the unit of seclusion is the perspective. The AOP module gives viewpoint situated programming execution which empowers you to characterize technique interceptors and direct cuts toward unmistakably decouple code which actualizes usefulness that ought to be isolated.

- Aspect: modularization of a worry that plays hooky. Eg :Transaction administration
- Join point: a point which happens amid the execution of a program or the treatment of a special case. In Spring AOP, it generally speaks to a technique execution.
- Advice: Action taken at a specific joinpoint happened. Distinctive sorts of exhortation are "after", "before" and "around". Spring model a guidance as an interceptor.
- Pointcut: a predicate which is coordinated with join focuses. Guidance is related with a pointcut expression and keeps running at any join point coordinated by the pointcut.

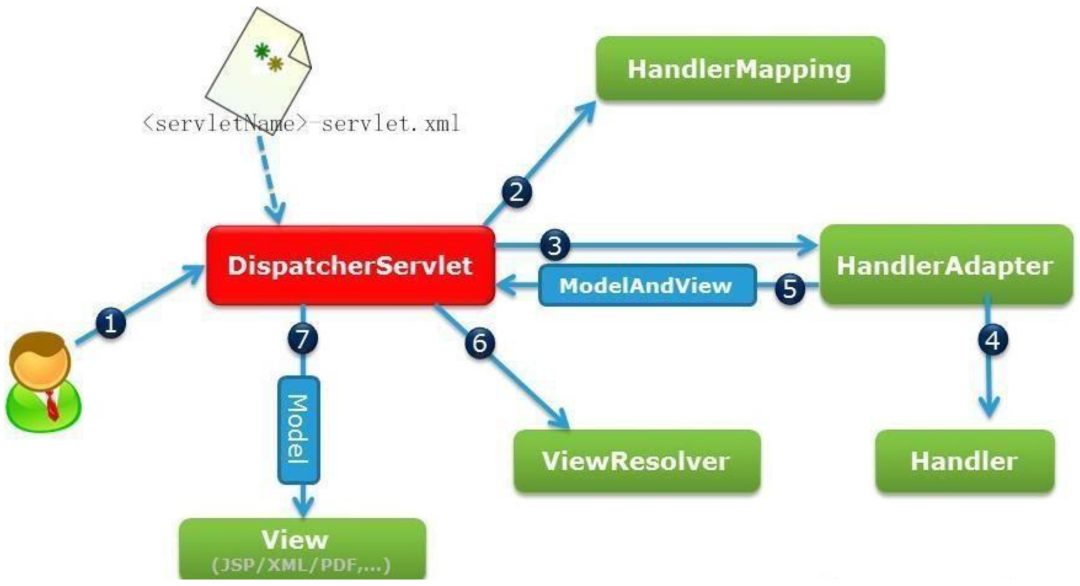


Fig 6.3Aspect Oriented Programming Layout

DISPATCHER-SERVLET:

The Spring Web(MVC) structure is composed with a DispatcherServlet that handles all the HTTP solicitations and reactions. Therequest handling work process of DispatcherServlet is portrayed in the accompanying outline

Succession of occasions for an approaching HTTP ask for to DispatcherServlet:

- After getting a HTTP ask for, DispatcherServlet goes to handler mapping to call the proper Controller.
- The Controller forms demand and calls the suitable administration strategies in view of GET or POST techniques. The administration strategy will set model information in view of business rationale characterized and see name is come back to the Dispatcher-Servlet.

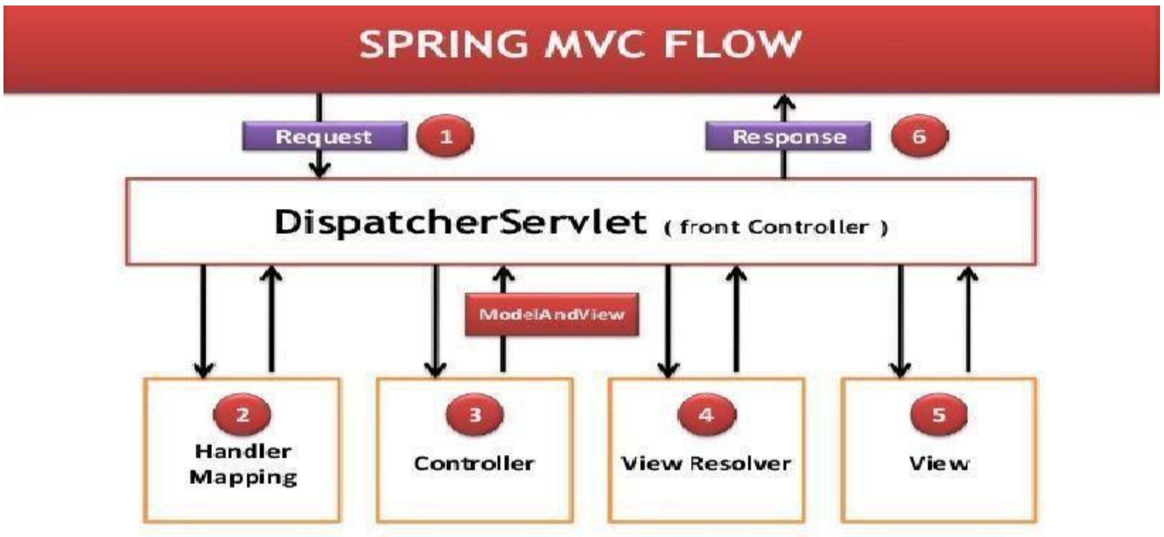


Fig 6.4 Spring MVC Flow Diagram

- With the help of ViewResolver, DispatcherServlet picks defined view for the request.
- DispatcherServlet passes show information to view when concluded which is rendered on the program.

JSON OBJECT:

JSON (JavaScript Object Notation) is a lightweight information interchangeformat. which is anything but difficult to peruse and compose and to parse andgenerate for machines. It depends on a subset of the JavaScript ProgrammingLanguage.

JSON is a textformat that is dialect free however utilizes traditions that aresimilar to C-group of dialects which incorporates C, C#, C++, Java,JavaScript,Python,Perl and numerous others which makes JSON an idealdata-exchange dialect.

JSON uses JavaScript syntax, but the JSON format istext only, just like XML.

JSON is built on two structures:

- A collection of name/value pairs. In other languages, this is considered asan record,object, dictionary,struct, hash table, keyed list, or associativearray.
- A list of values ordered. In other languages, which is realized as an array,vector, list, or sequence.

JSON WEB TOKENS (JWT):

JWT helps in ensuring trust and security in many applications. JWT allow claims also called as attributes, such as user data, which are to be represented in a secure manner.

A Json web token is a JSON object which is contained in RFC 7519 standard as a secure way to signify a set of information between two parties. A single JSON Web Token has a header, a payload, and a signature. Format of a token can be represented as:

header.payload.signature

To illustrate how and why JWTS are actually used, we will use a 3 entity example. The three entities are:

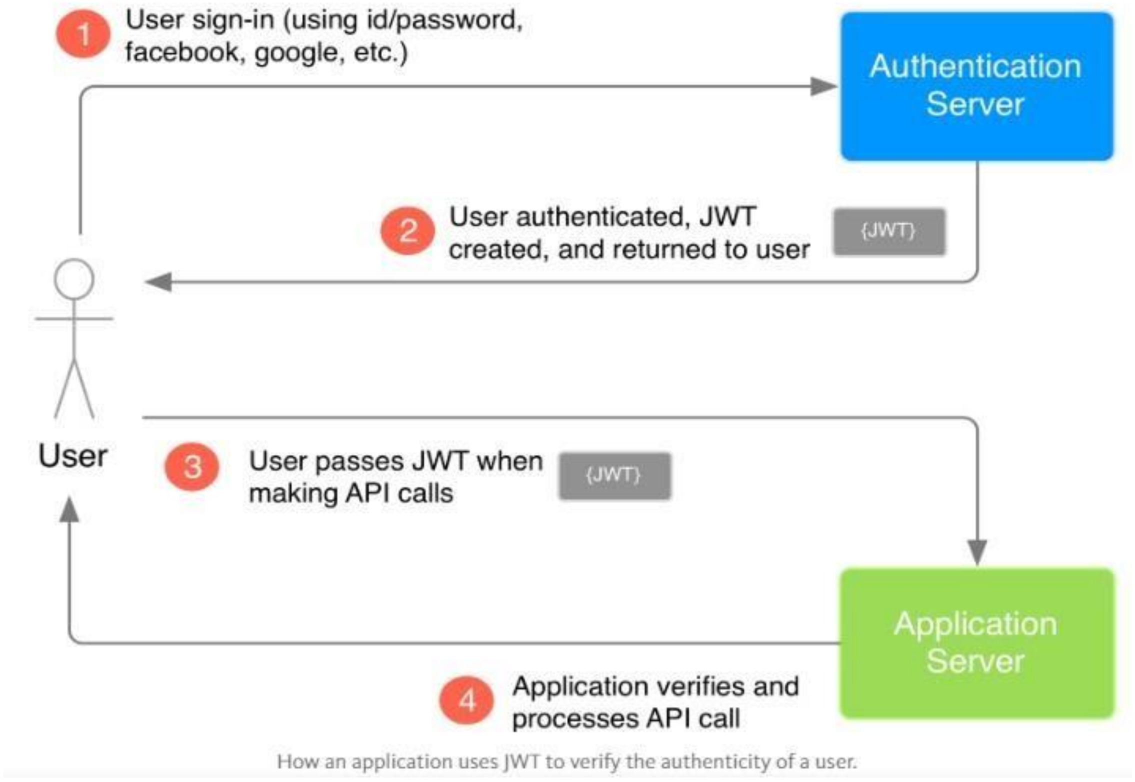
- Authentication server-It provides the JWT to user
- User-Receives the JWT
- Application server-Communicates between user and application safely

Features of Framework:

- Compact
- Self Contained
- URL Safe
- Highly Secure

ADVANTAGES of JWT:

- Usability
- Easy access
- Version management
- Enhanced Security



1
User sign-in (using id/password, facebook, google, etc.)
Authentication Server
2
User authenticated, JWT created, and returned to user
{JWT}
User
3
User passes JWT when making API calls
{JWT}
Application Server
4
Application verifies and processes API call
How an application uses JWT to verify the authenticity of a user.

Fig 6.5 Internal Working of JWT Design

Anatomy of JWT:

When the user signs into the respective authentication server using the application'slogin system. Unique JWT is created to that user by the authentication server .API calls with JWT attached to them are sent to application upon logged in. The configuredapplication server verifies the creator of JWT. The creator should be the authentication server and that particular authenticated user should be the one who making the API calls.

How to use JWT?

1)Create HEADER

The header component of the JWT has thedata about how the JWT signature isfigured.

Format:

```
{
   "typ": "JWT",
   "alg": "HS256"
}
```

- “typ” key - states the object is JWT
- “alg” key - states the type of algorithm being used to create JWT signature
- HMAC-SHA256 - algorithm, a hashing algorithm that computes the signature using

secret key **2)Create PAYLOAD**

It has the data about user information (“user id”) in the form of claims(attributes). Three types of claims can be used:

- Registered/Reserved claims - Standard claims contained in JWT (“iss”,”sub”,”exp”)

 Userdefined claims – Defined by users, are the two most used claims (“userId”).

Format:

```
{
        "userId": "b08f86af-35da-48f2-8fab-cef3904660bd"
      }
```

3)Create SIGNATURE

The signature is figured using the following pseudo code:

Fomat: data = base64urlEncode(header) + “.” + base64urlEncode(

payload) signature

= Hash(data, secret);

- base64url-Encodes the header and the payload
- Period(.)-Joins the encoded strings
- Secret key –Creates the JWT signature using hashing algorithm

Encoding the HEADER and PAYLOAD:

// header

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9

// payload

eyJ1c2VySWQiOiJiMDhmODZhZi0zNWRhLTQ4ZjItOGZhYi1jZWYzOTA0NjYwY
mQif
Q

Signature obtained looks like:

// signature

-xN_h82PHVTCMA9vdoHrcZxH-x5mb11y1537t3rGzcM

Final JTW looks like:

// JWT Token

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJiMDhmODZhZi0zN
WRhL
TQ4ZjItOGZhYi1jZWYzOTA0NjYwYmQifQ.xN_h82PHVTCMA9vdoHrcZxHx5mb
11y1537t3rGzcM

<u>How the data is protected using JWT?</u>

JWT do not hide or obscure the user data. It is just a verification of API calls from authenticated user.The user data in a JWT is not ENCRYPTED but just ENCODED and SIGNED.

- ‘Encoding’-Transforms the data’s structure
- ‘Signing’-verifies the authenticity

<u>JWT is ENCODING not ENCRYPTION</u>

ENCODING- transform data so that it can be safely consumed by a different type of system.

ENCRYPTION- transforms the data in order to keep it secret from others.

TYPES OF CLAIMS:

1.RESERVED CLAIMS:

- "iss": Issuer of the JWT Token
- "sub": Subject of the JW Token
- "aud": Audience of the JW Token
- "exp": Most utilized JWT which characterizes the termination in Date(Numerical). The close should be after the present date/time.
- "nbf": Defines the breaking point of time, just past which the JWT must be acknowledged for preparing
- "iat": The time at which the JWT was issued. Used to figure the present period of JWT
- "jti": A Unique and Disposable JWT which is utilized to abstain from replaying of JWT.
  Alluded as a solitary time utilize token.

2.PUBLIC CLAIMS:

Created by user like credentials and other user related information.

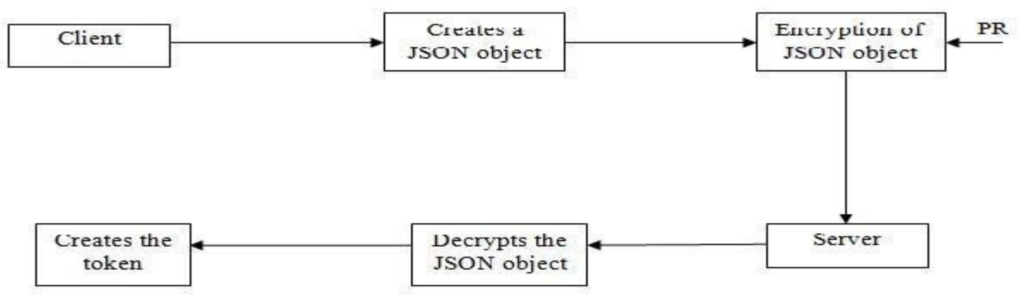
3. PRIVATE CLAIMS:

Two parties can use them privately upon agreement. As they are subject to crash, we should use them with carefulness.

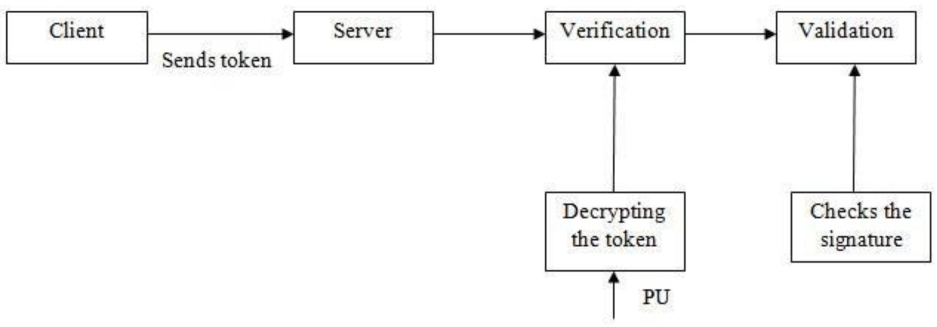
ARCHITECTURE DESIGN:

Fig 6.6 Creation of Token

TOKEN CREATION



USAGE OF TOKEN



When user provides the login credentials, the information provided will be converted to a json object which is encrypted with HMAC algorithm and sent to the server. Server Generates the token by using the decrypted information provided by the client and returns back to the client.

7) INTERACTION SCENARIO:

USECASE DIAGRAM:

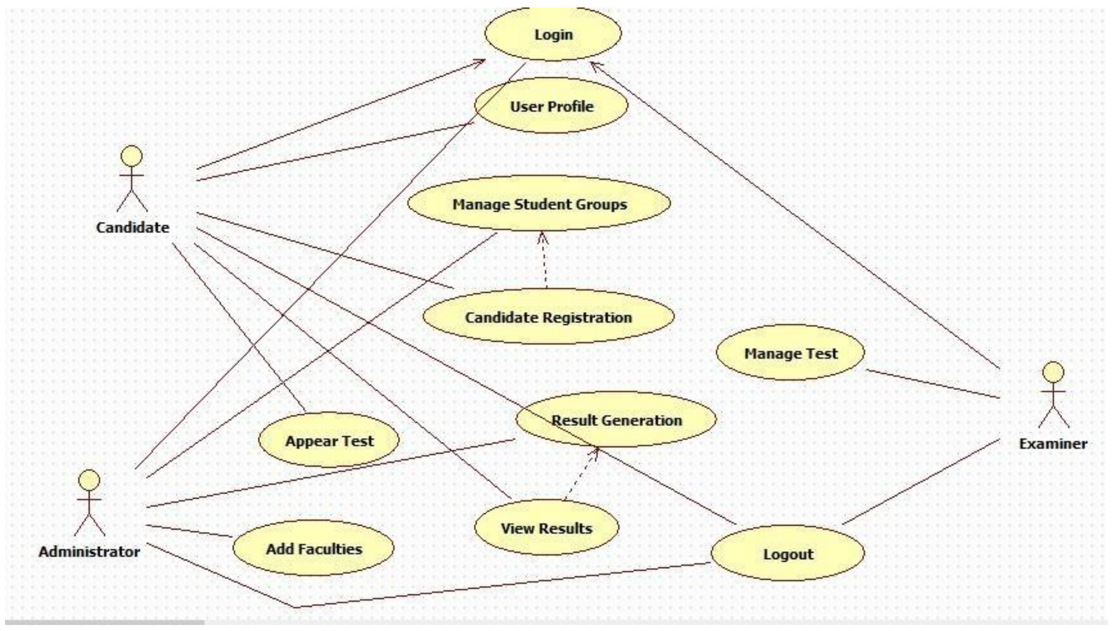


Fig 7.1 Usecase Diagram

Use case Diagrams usually talk about the set of actions performed by the actors in collaboration with the other actors of the system.

Here Candidate could be able to login with their credentials and go through the functionalities like attending test which has been registered, viewing his profile details and to edit. Examiner will prepare Test and upload the questions. Admin can add the Examiners to the application and sends e-mail notification to the students

ACTIVITY DIAGRAM:

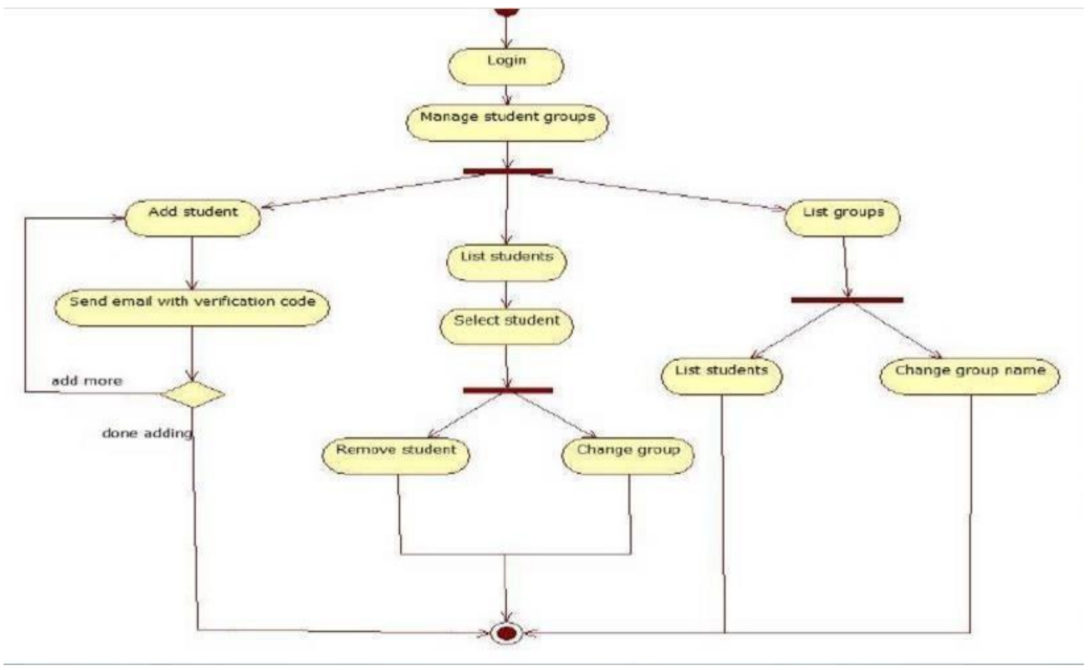


Fig 7.2 ACTIVITY DIAGRAM FOR MANAGING STUDENT GROUPS

Managing Student groups is being administered by admin. He will have control to remove the student in a group or changing a particular group name under examiner request.

COLLABORATION DIAGRAM FOR JWT:

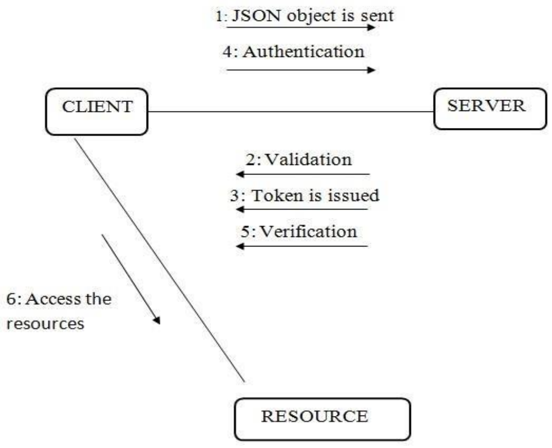


Fig 7.3

Sequence Diagram for JWT:

A Sequence outline is a communication graph that shows how forms work with each other and in what arrange.

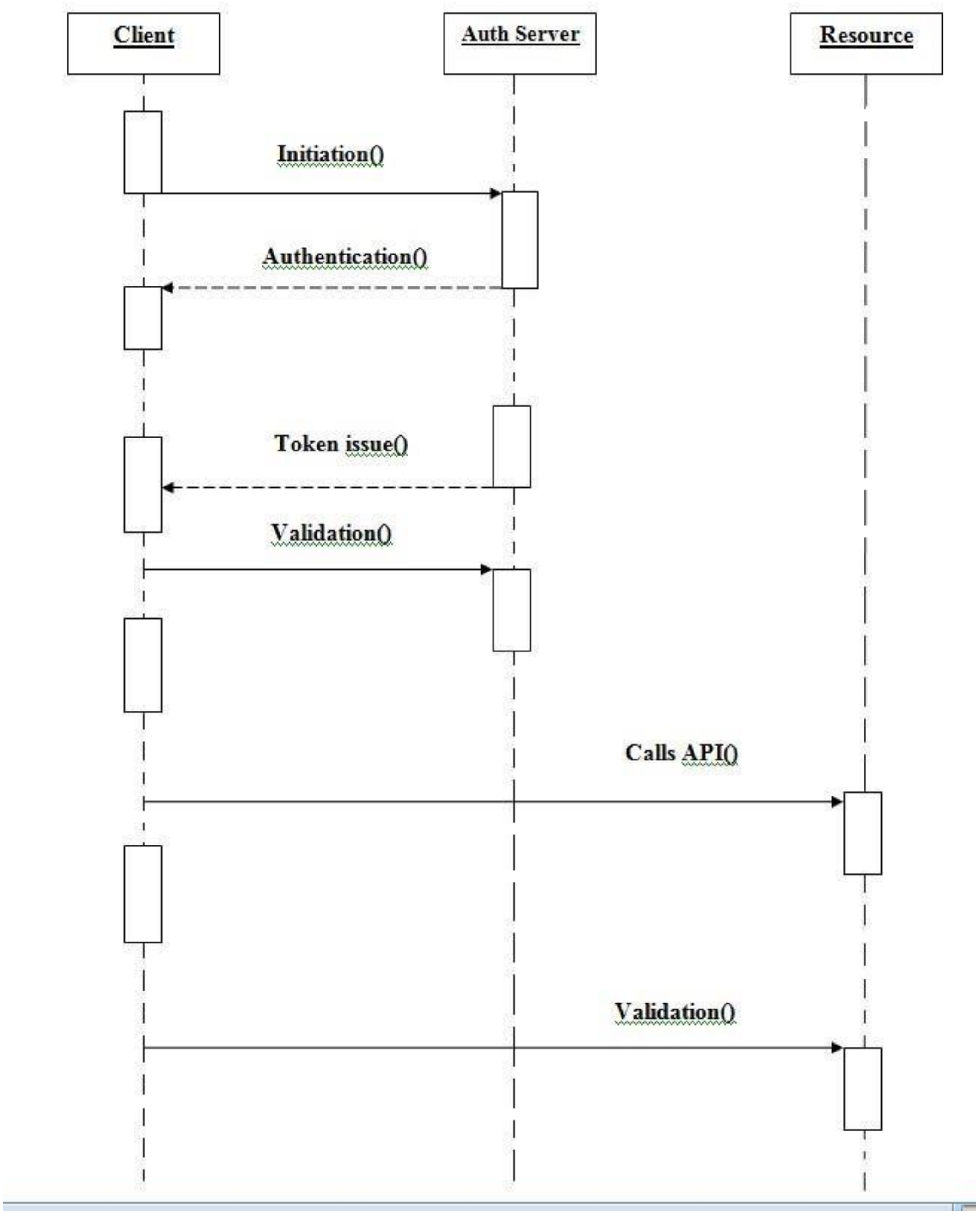
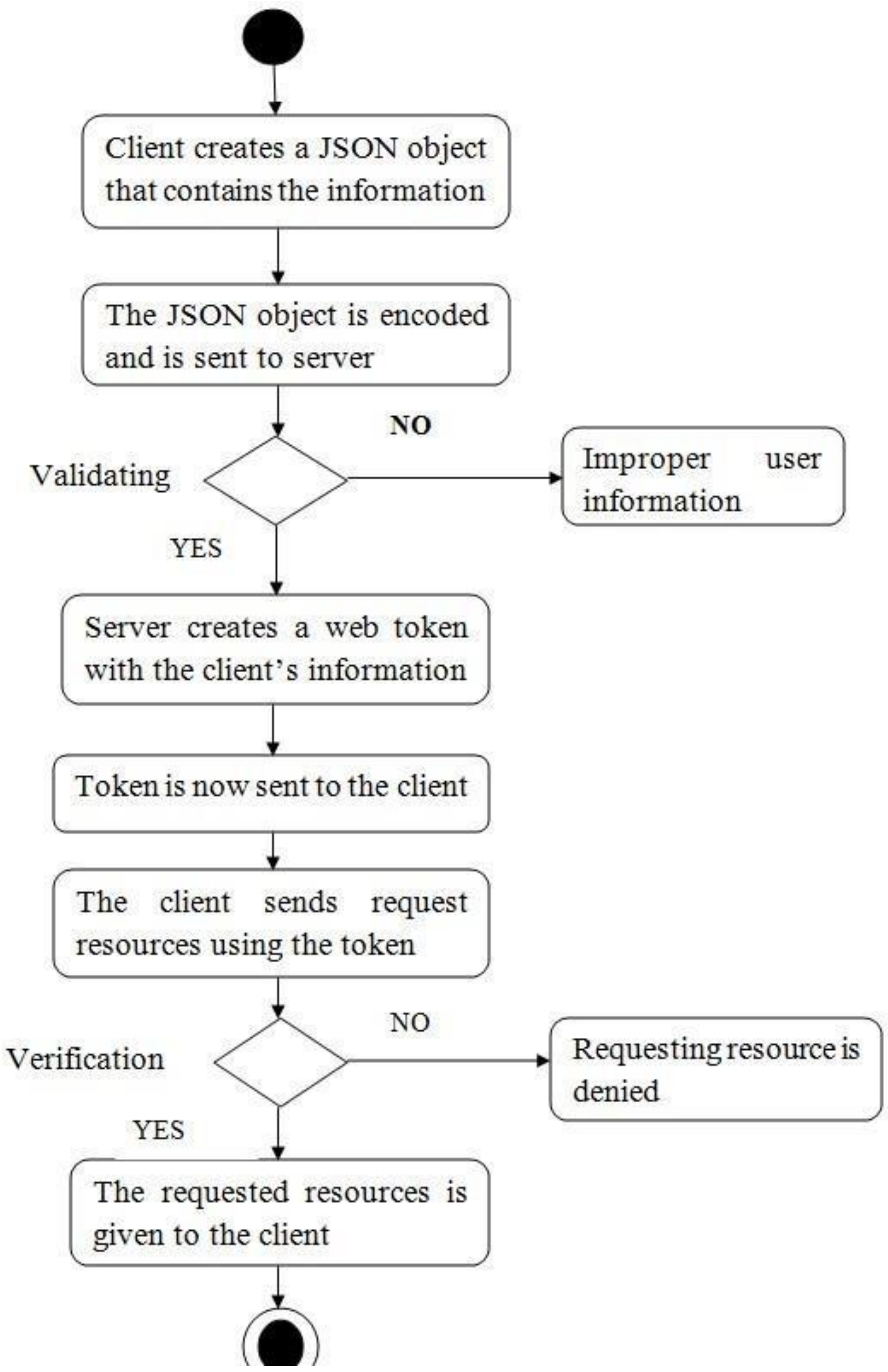


Fig 7.4 Sequence Diagram

ACTIVITY DIAGRAM JWT:



Client creates a JSON object that contains the information
The JSON object is encoded and is sent to server
NO
Validating
Improper user information
YES
Server creates a web token with the client's information
Token is now sent to the client
The client sends request resources using the token
NO
Verification
Requesting resource is denied
YES
The requested resources is given to the client

Fig 7.5 Activity Diagram

SEQUENCE DIAGRAM:

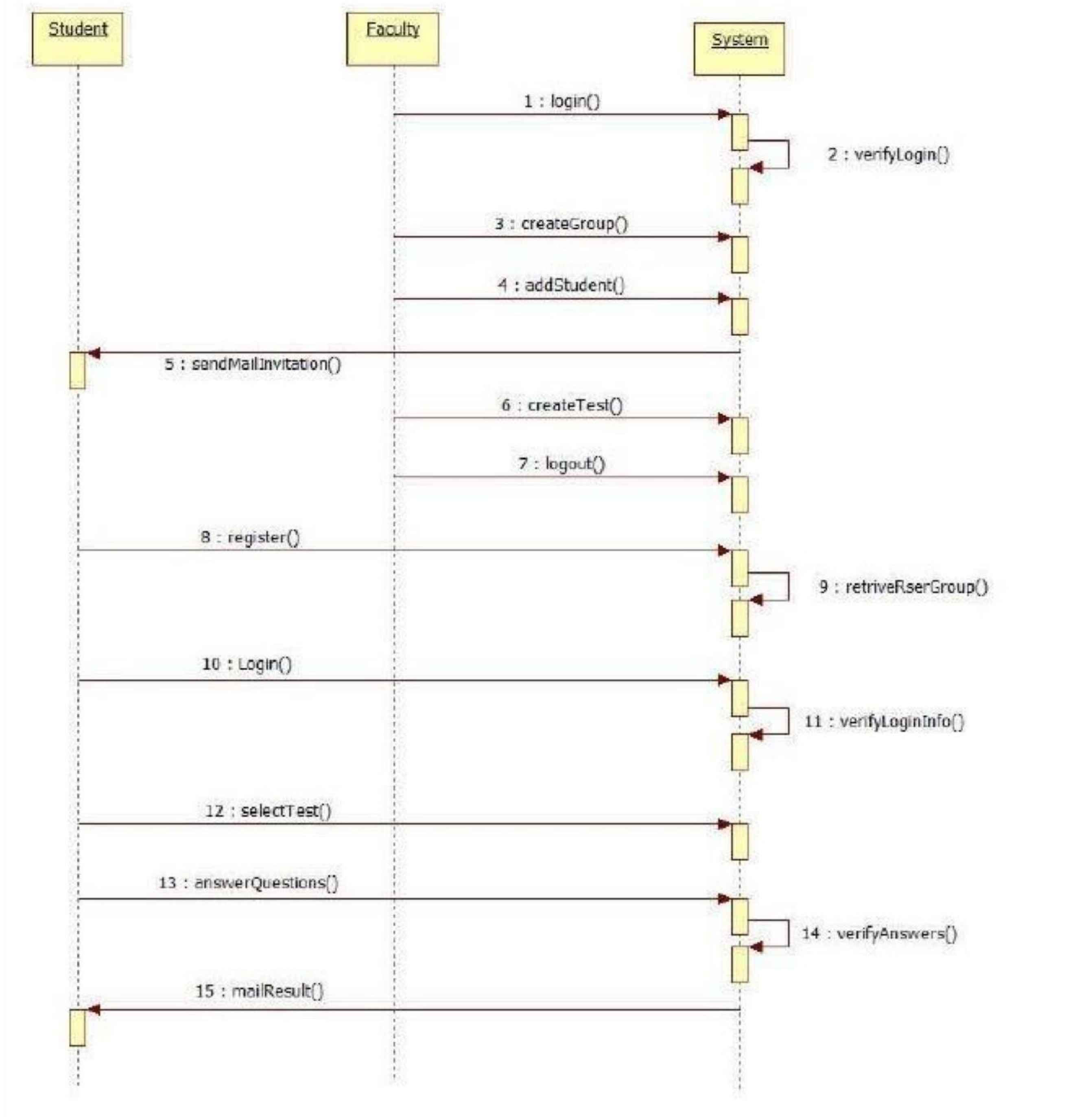


Fig 7.6 Sequence Diagram

CLASS DIAGRAM:

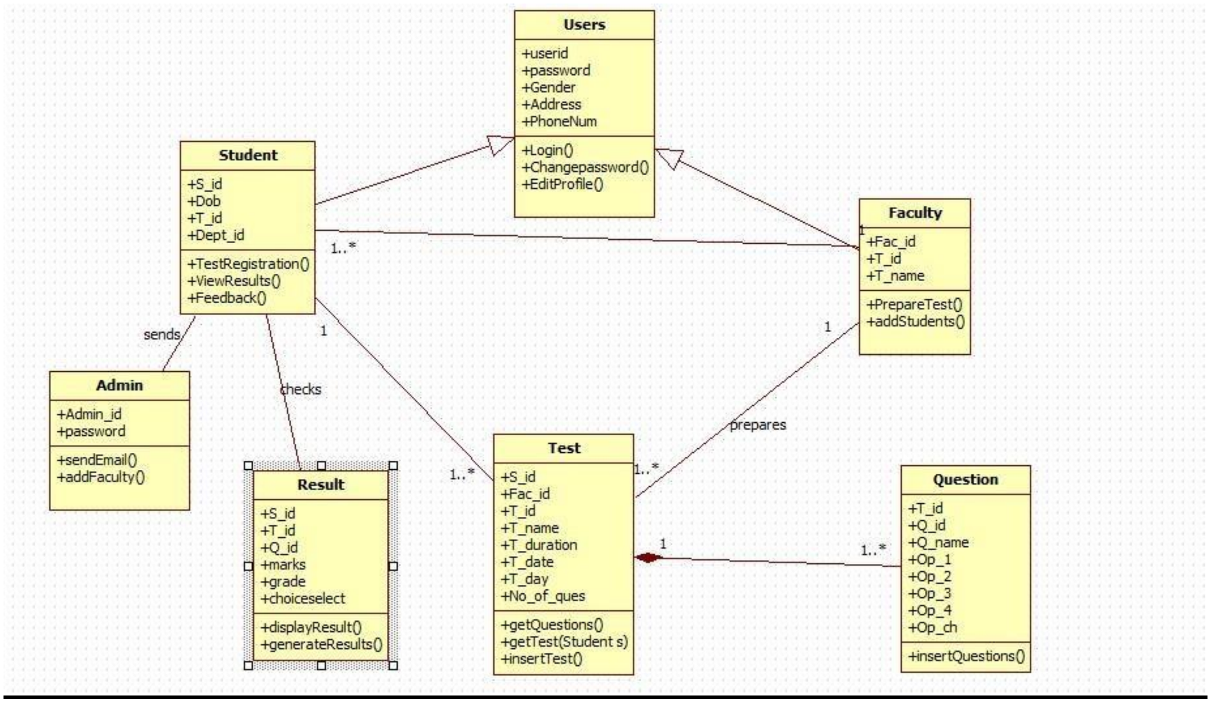


Fig 7.7 Class Diagram

8)METHODOLOGY AND APPROACH

The following are the modules:

- Creation of token
- Verification and Validation

CREATION OF TOKEN:

To make a JWT, the underneath steps got the chance to be taken after. The request of the means isn't imperative therefore of that there are no conditions between the data sources and yields.

1. Produce a JWT Claims Set containing the specified claims.
2. JWT ‘s alerts and messages should be represented in UTF-8’s octets.
3. Manufacture a Header that holds the wanted set of Parameters.
4. If either nested language or cryptography operation is performed, and give the Message a chance to be the JWS or JWE, then come back to Step 3, utilizing a substance kind cost of "JWT" among the new JOSE Header made in this progression.
5. Otherwise, let the ensuing JWT be the JWS or JWE .

JWT would be rejected when one of the mentioned steps fail and treated as invalid input.

VERIFICATION AND VALIDATION

1. JWT must contain no less than 1 ('.') character.
2. Encoded JOSE Header ought to be put before the character('.') in a JWT.
3. Base64url decode the Encoded JOSE Header taking after the limitation that no whitespace, line breaks, or entirely unexpected extra characters are utilized.
4. Octet portrayal of UTF-8 ought to affirm to the standard RFC 7159 [RFC7159],JOSE Header ought to be this JSON question.

5. JOSE Header must contain both parameters and their qualities bolstered by their phonetics and punctuation.
6. Check whether a JWT can be with an example of JWS or JWE.
7. If the JOSE Header contains a "cty" (content sort) estimation of "JWT", then the Message is a JWT that was the subject of settled marking or encryption operations. In such case, come back to Step1, utilizing the Message as the JWT.
8. Otherwise, base64url unravel the Message taking after the limitation that no whitespace, line breaks, or other extra characters have been utilized.

9) OUTPUT/RESULTS:

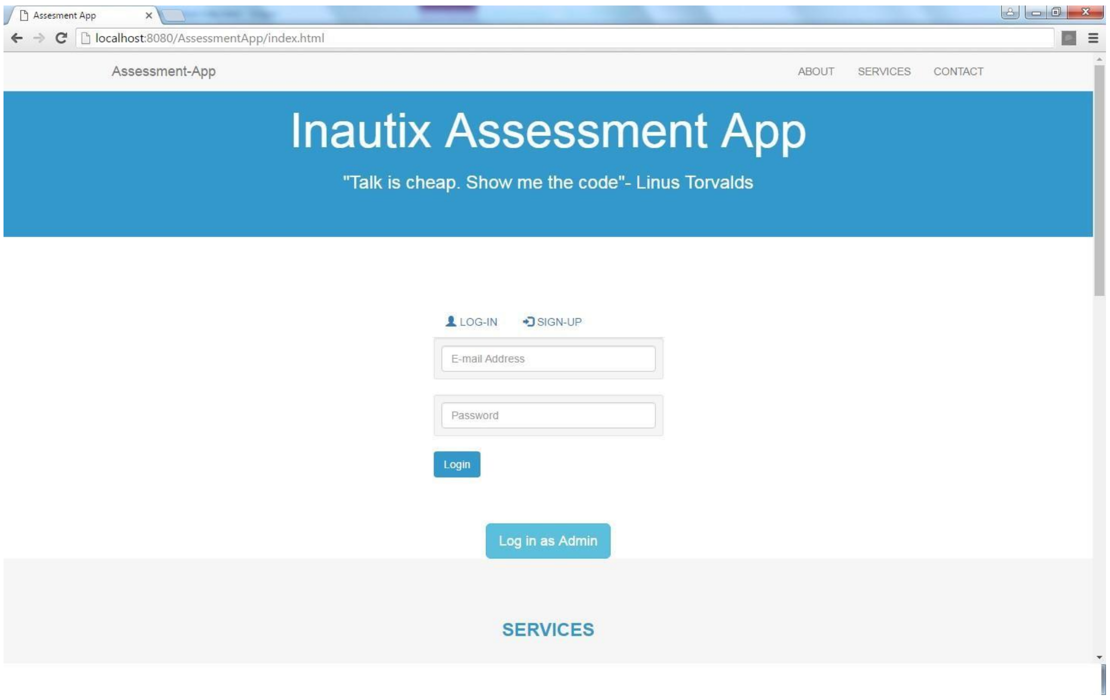
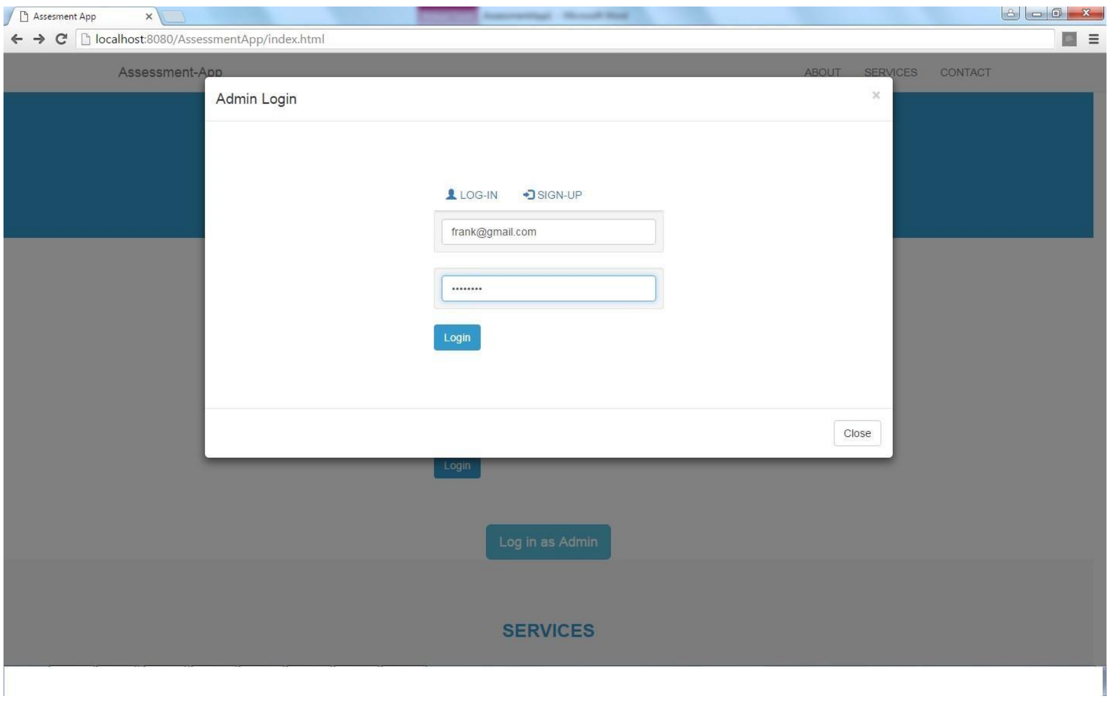


Fig 9.1 Output Screen of Application

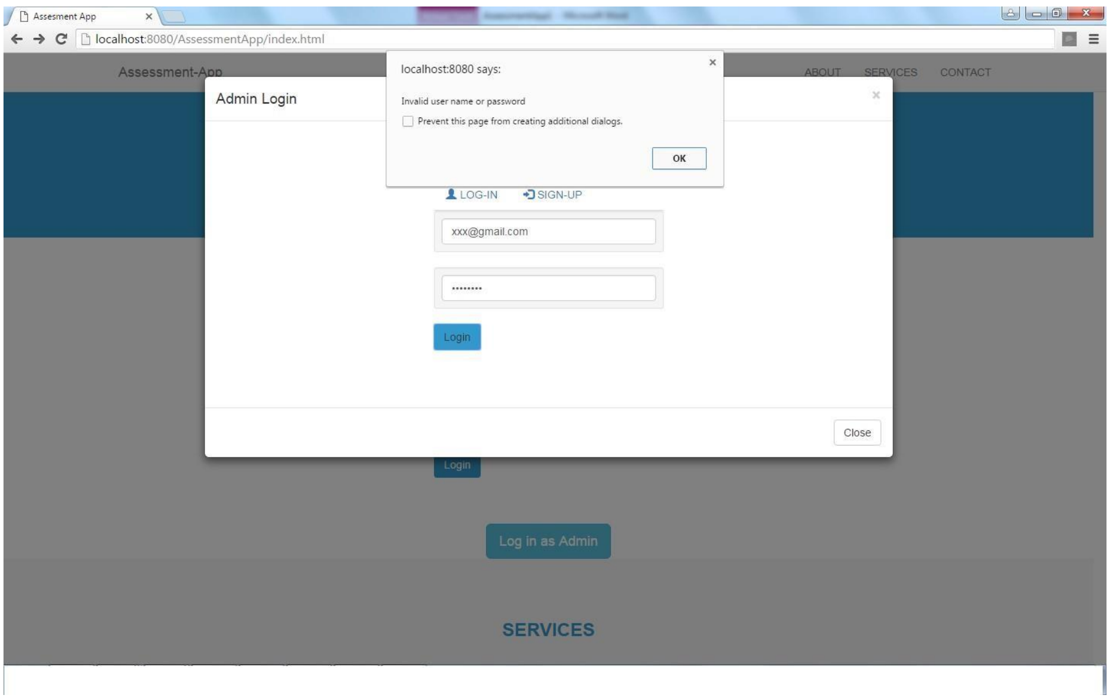
Admin Signup Screen for Authentication:

- The admin signup with their name,emailid,employeeid and password.
- The application identifies the admin using their employeeid,other than admin non can't signup.
- The admin details are stored in **Apache Derby** database.

FIGURE: ADMIN SIGNUP



- Using **AngularJS** by sending **AJAX** request and response to valid the admin emailid and password.
- The application valid the admin emailid and password , if it is invalid the application will show an **Alert message.**

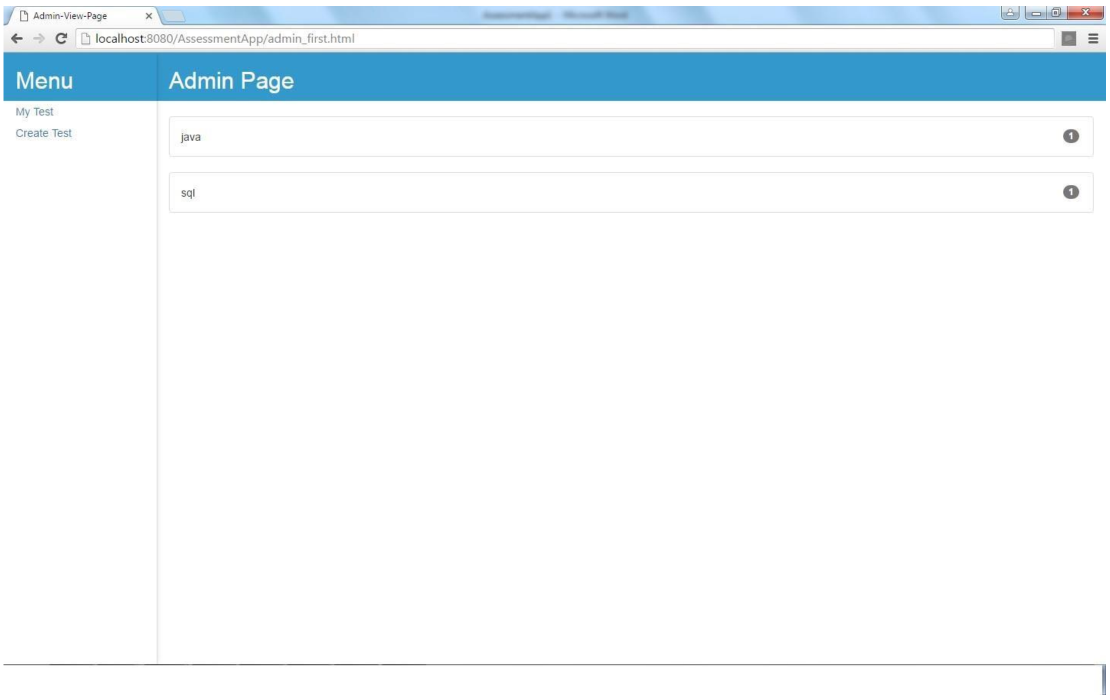

localhost:8080 says:

Invalid user name or password

Prevent this page from creating additional dialogs.

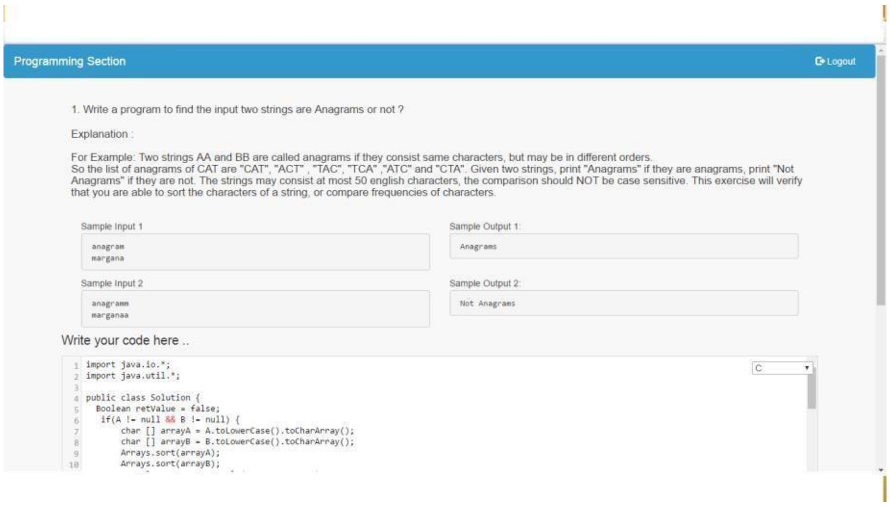
Admin Page:

- This is a page used to create a test and view already existing test details. On clicking **MY TEST,** it will show the already existing test created by the respective admin.
- On clicking **CREATE TEST** to create new test like on multiple choice Questions (MCQs) and Programming task.



USER PROGRAM TEST:

- In this programming section, the user get question based on the domain names they selected.
- The question will contain explanation along with two sample inputs and outputs .
- This makes the user solve the program efficiently and quickly.



1. Write a program to find the input two strings are Anagrams or not ?

Explanation :

For Example: Two strings AA and BB are called anagrams if they consist same characters, but may be in different orders.
So the list of anagrams of CAT are "CAT", "ACT" , "TAC", "TCA" ,"ATC" and "CTA". Given two strings, print "Anagrams" if they are anagrams, print "Not Anagrams" if they are not. The strings may consist at most 50 english characters, the comparison should NOT be case sensitive. This exercise will verify that you are able to sort the characters of a string, or compare frequencies of characters.

Sample Input 1

```
anagram
margana
```

Sample Output 1:

```
Anagrams
```

Sample Input 2

```
anagramm
marganaa
```

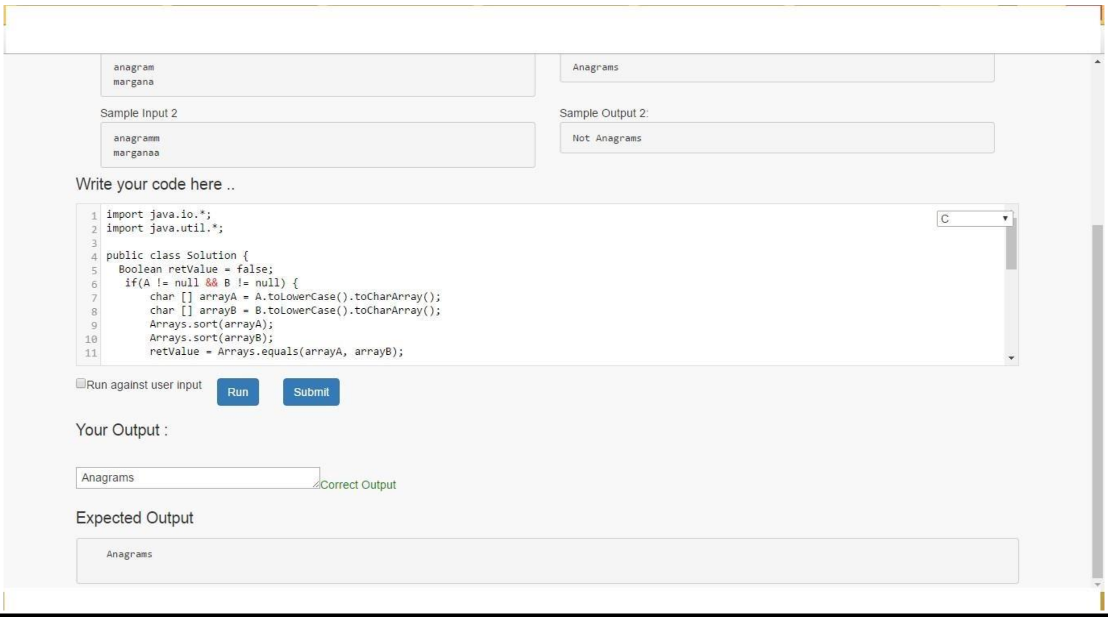
Sample Output 2:

```
Not Anagrams
```

Write your code here ..

```
import java.io.*;
import java.util.*;

public class Solution {
  Boolean retValue = false;
   if(A != null && B != null) {
       char [] arrayA = A.toLowerCase().toCharArray();
       char [] arrayB = B.toLowerCase().toCharArray();
       Arrays.sort(arrayA);
       Arrays.sort(arrayB);
```

TEST RESULT:



```
anagram
margana
```

```
Anagrams
```

Sample Input 2

```
anagramm
marganaa
```

Sample Output 2:

```
Not Anagrams
```

Write your code here ..

```
import java.io.*;
import java.util.*;

public class Solution {
  Boolean retValue = false;
   if(A != null && B != null) {
       char [] arrayA = A.toLowerCase().toCharArray();
       char [] arrayB = B.toLowerCase().toCharArray();
       Arrays.sort(arrayA);
       Arrays.sort(arrayB);
       retValue = Arrays.equals(arrayA, arrayB);
```

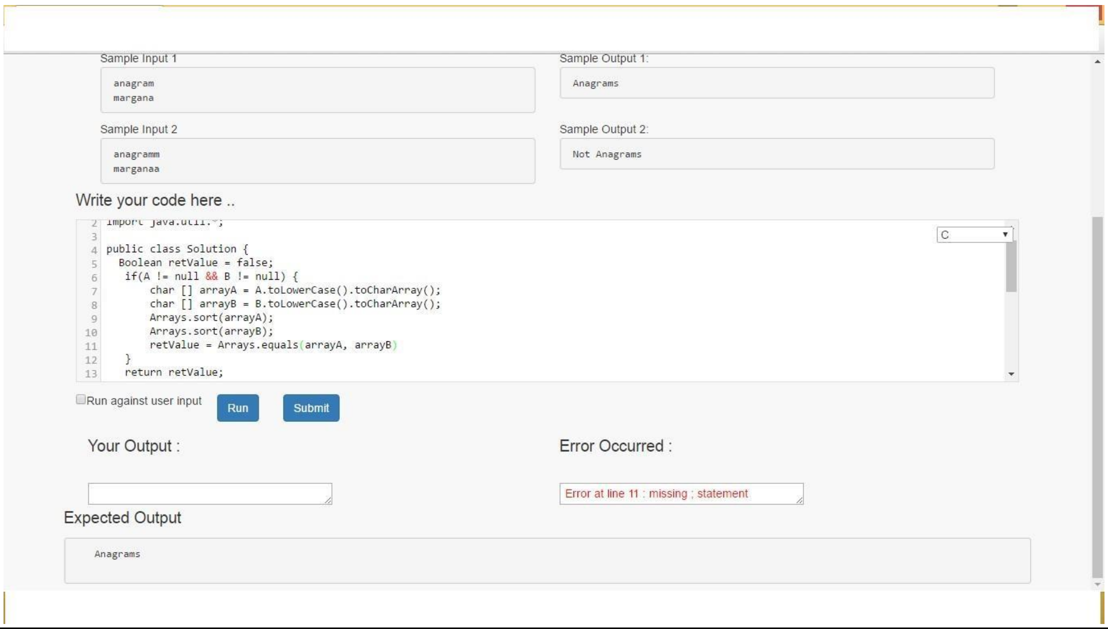
Your Output :

```
Anagrams
```

Correct Output

Expected Output

```
Anagrams
```


Sample Input 1

```
anagram
margana
```

Sample Output 1:

```
Anagrams
```

Sample Input 2

```
anagramm
marganaa
```

Sample Output 2:

```
Not Anagrams
```

Write your code here ..

```
 2  import java.util.*;
 3
 4  public class Solution {
 5    Boolean retValue = false;
 6     if(A != null && B != null) {
 7         char [] arrayA = A.toLowerCase().toCharArray();
 8         char [] arrayB = B.toLowerCase().toCharArray();
 9         Arrays.sort(arrayA);
10         Arrays.sort(arrayB);
11         retValue = Arrays.equals(arrayA, arrayB)
12     }
13     return retValue;
```

Your Output :

Error Occurred :

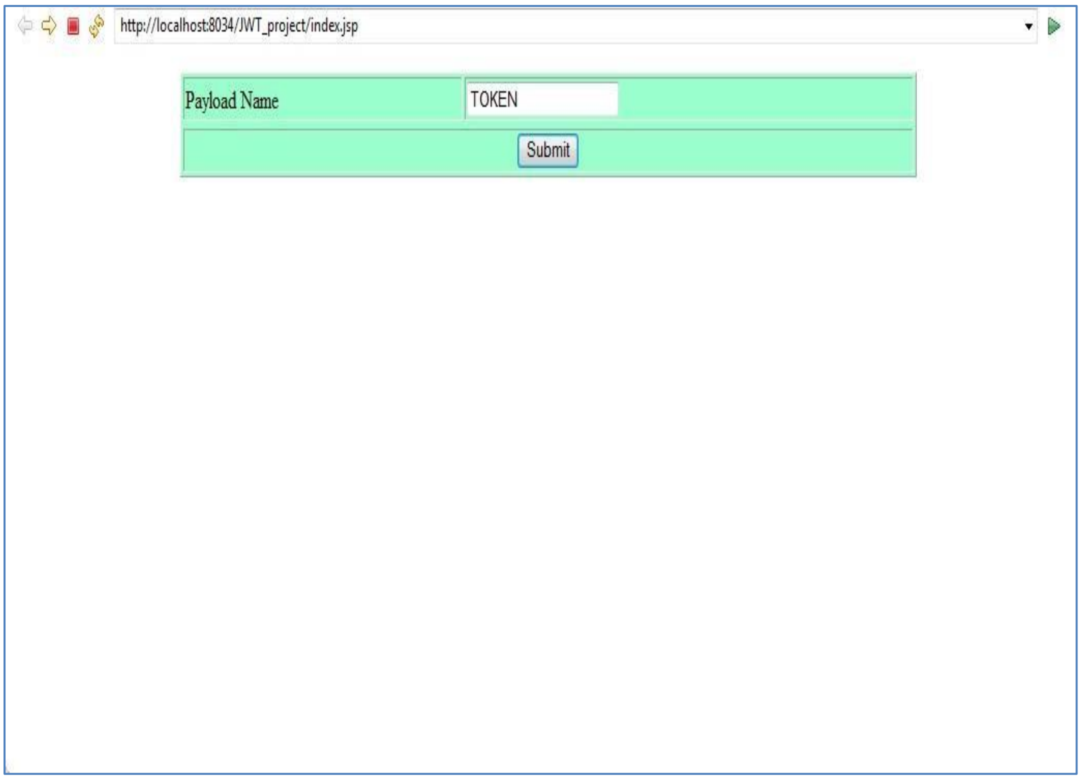
```
Error at line 11 : missing ; statement
```

Expected Output

```
Anagrams
```

INPUT PAGE

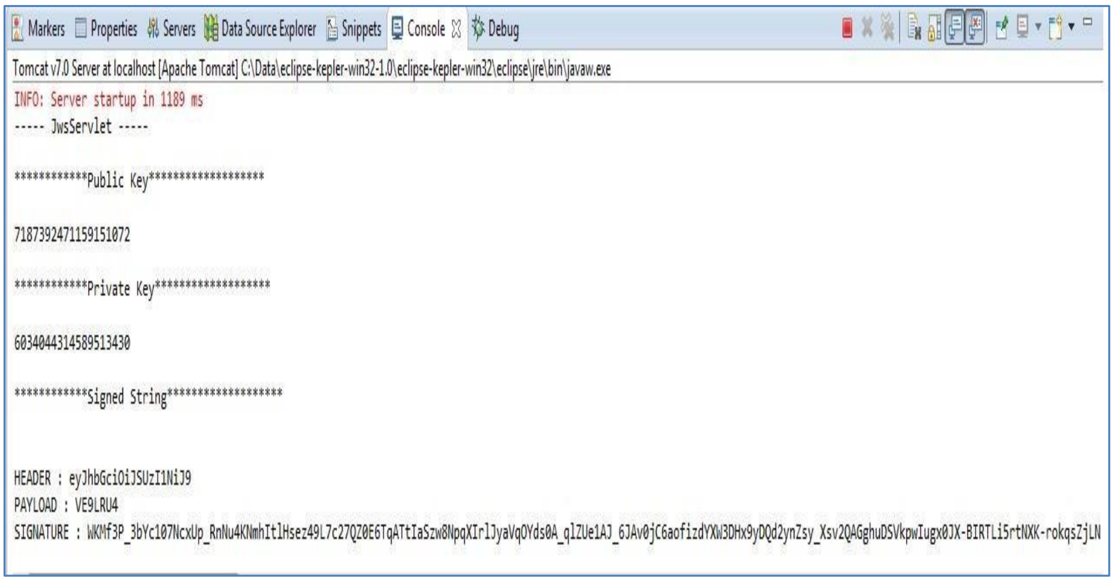
The input page is about entering the payload information of the token that has to be transmitted over the two parties.


http://localhost:8034/JWT_project/index.jsp

Payload Name	TOKEN

TOKEN GENERATION

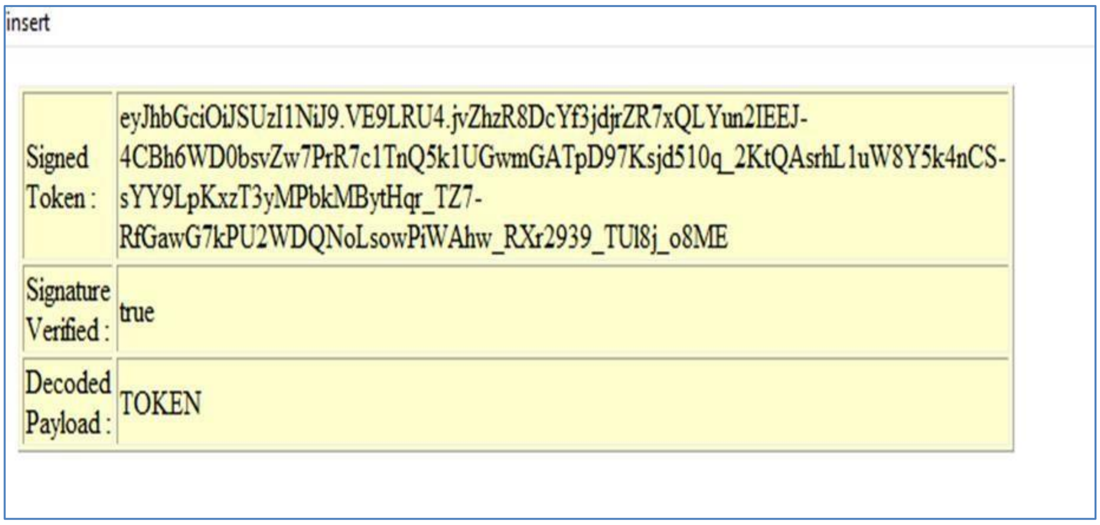
This is the process in the server side where the entire token is generated along with its required contents that include private and primary key.



```
Tomcat v7.0 Server at localhost [Apache Tomcat] C:\Data\eclipse-kepler-win32-1.0\eclipse-kepler-win32\eclipse\jre\bin\javaw.exe
INFO: Server startup in 1189 ms
----- JwsServlet -----

************Public Key******************

7187392471159151072

************Private Key******************

6034044314589513430

************Signed String******************


HEADER : eyJhbGciOiJSUzI1NiJ9
PAYLOAD : VE9LRU4
SIGNATURE : WKMf3P_3bYc107NcxUp_RnNu4KNmhItlHsez49L7c27QZ0E6TqATtIaSzw8NpqXIrlJyaVqOYds0A_qlZUe1AJ_6JAv0jC6aofizdYXW3DHx9yDQd2ynZsy_Xsv2QAGghuDSVkpwIugx0JX-BIRTLi5rtNXK-rokqsZjLN
```

OUTPUT PAGE

The output page therefore displays the token, the decoded payload information that has been sent and also verifies the signature.


insert

Signed Token :	eyJhbGciOiJSUzI1NiJ9.VE9LRU4.jvZhzR8DcYf3jdjrZR7xQLYun2IEEJ-4CBh6WD0bsvZw7PrR7c1TnQ5k1UGwmGATpD97Ksjd510q_2KtQAsrhL1uW8Y5k4nCS-sYY9LpKxzT3yMPbkMBytHqr_TZ7-RfGawG7kPU2WDQNoLsowPiWAhw_RXr2939_TUl8j_o8ME
Signature Verified :	true
Decoded Payload :	TOKEN

Fig 9.2 Token Generation Reference

10)CONCLUSION

In this project we have proposed a technique to enhance the security of the information traversing in a network between a client and server using a web token and using HMAC algorithm. It ensures safe transfer of the credentials by generating a secret key, that signs the token and checks for the same every time the details are sent to the server.

FUTURE ENHANCEMENTS:

Future work would be focused on a change to the standard that could help in preventing the future vulnerabilities, as many libraries are prone to attacks. Libraries treat tokens that holds 'none' value in the algorithm field as valid token that results in arbitrary account access on any systems, that can be enhanced by providing a secret key, that fails for the token using 'none'. Attackers can modify the algorithm field where the server does not have standard algorithm. In order to standardize the algorithm, the server specifies the algorithm that is to be incorporated statically into the verification function.

11)REFERENCES

1. Bradley, J. and N.Sakimura (editor), "JSON Simple Sign", September 2010

2. Panzer, J., Ed., Laurie, B., and D. Balfanz, "Magic Signatures", January 2011.

3. Jones, M., "JSON Web Algorithms (JWA)", RFC 7518, May 2015.

4. Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)",RFC 7516, May 2015

MSDN – the Microsoft Developer Network.

5. https://jwt.io/introduction/

p2

by P SAI NAVYA SREE

%9	%8	%2	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.ijitee.org Internet Source	%4
2	www.vytcdc.com Internet Source	%1
3	www.scribd.com Internet Source	%1
4	r4r.in Internet Source	%1
5	www.tutorialspoint.com Internet Source	%1
6	www.slideshare.net Internet Source	<%1

7	Advanced API Security, 2014. Publication	<%1
8	blogs.sap.com Internet Source	<%1
9	Mittal. "Installing Tomcat", Pro Apache Tomcat 6, 2007 Publication	<%1
10	stackoverflow.com Internet Source	<%1
11	angular.io Internet Source	<%1

EXCLUDE QUOTES ON EXCLUDE MATCHES OFF EXCLUDE OFF BIBLIOGRAPHY