



Name: Subhash Ravindra SRIVATSA

Professor: Thomas Broussard

Session: 2017 - Fall

Project: Identity and Access Management

The IAM Web Application

Subject: JAVA

## Table of Contents

1. Project Overview.....	3-8
1.1 Java .....	3
1.2 Maven .....	3
1.3 Spring.....	4
1.4 Hibernate.....	5
1.5 Log4j.....	8
1.6 JUnit.....	8
2. Project Analysis.....	9-10
2.1 Modules.....	9
2.2 Data Description.....	9
2.3 Expected Results.....	10
2.4 Scopes and Limitations.....	10
3. Conception.....	11
3.1 Chosen Algorithm.....	11
3.1 Data Structures.....	11
3.1 Application Flow Diagram.....	11
4. Configuration .....	12-13
4.1 Maven.....	12
4.2 Server.....	13
5. Screenshots.....	14-17
Bibliography.....	18

# CHAPTER – 1

## PROJECT OVERVIEW

### 1.1 JAVA

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

### 1.2 MAVEN

Maven, a Yiddish word meaning *accumulator of knowledge*, was originally started as an attempt to simplify the build processes in the Jakarta Turbine project. There were several projects each with their own Ant build files that were all slightly different and JARs were checked into CVS. We wanted a standard way to build the projects, a clear definition of what the project consisted of, an easy way to publish project information and a way to share JARs across several projects.

The result is a tool that can now be used for building and managing any Java-based project. We hope that we have created something that will make the day-to-day work of Java developers easier and generally help with the comprehension of any Java-based project.

Maven's primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this goal there are several areas of concern that Maven attempts to deal with:

- Making the build process easy
- Providing a uniform build system
- Providing quality project information
- Providing guidelines for best practices development
- Allowing transparent migration to new features

## 1.3 SPRING

Spring is the most popular application development framework for enterprise Java. Millions of developers around the world use Spring Framework to create high performing, easily testable, and reusable code.

Spring framework is an open source Java platform. It was initially written by Rod Johnson and was first released under the Apache 2.0 license in June 2003. Spring is lightweight when it comes to size and transparency. The basic version of Spring framework is around 2MB.

The core features of the Spring Framework can be used in developing any Java application, but there are extensions for building web applications on top of the Java EE platform. Spring framework targets to make J2EE development easier to use and promotes good programming practices by enabling a POJO-based programming model.

Following is the list of few of the great benefits of using Spring Framework –

- Spring enables developers to develop enterprise-class applications using POJOs. The benefit of using only POJOs is that you do not need an EJB container product such as an application server but you have the option of using only a robust servlet container such as Tomcat or some commercial product.

- Spring is organized in a modular fashion. Even though the number of packages and classes are substantial, you have to worry only about the ones you need and ignore the rest.
- Spring does not reinvent the wheel, instead it truly makes use of some of the existing technologies like several ORM frameworks, logging frameworks, JEE, Quartz and JDK timers, and other view technologies.
- Testing an application written with Spring is simple because environment-dependent code is moved into this framework. Furthermore, by using JavaBeanstyle POJOs, it becomes easier to use dependency injection for injecting test data.
- Spring's web framework is a well-designed web MVC framework, which provides a great alternative to web frameworks such as Struts or other over-engineered or less popular web frameworks.
- Spring provides a convenient API to translate technology-specific exceptions (thrown by JDBC, Hibernate, or JDO, for example) into consistent, unchecked exceptions.
- Lightweight IoC containers tend to be lightweight, especially when compared to EJB containers, for example. This is beneficial for developing and deploying applications on computers with limited memory and CPU resources.
- Spring provides a consistent transaction management interface that can scale down to a local transaction (using a single database, for example) and scale up to global transactions (using JTA, for example).

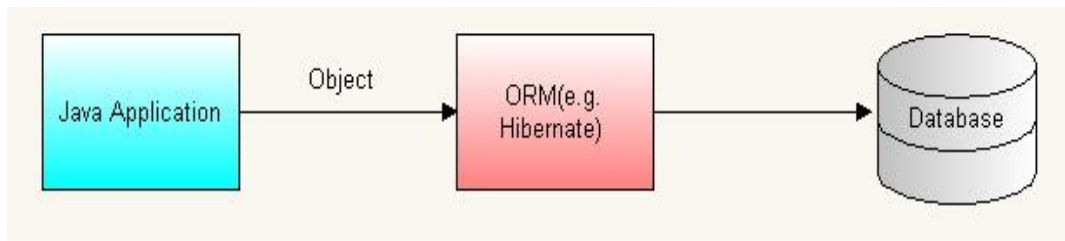
## 1.4 HIBERNATE

Hibernate is a high-performance Object/Relational persistence and query service which is licensed under the open source GNU Lesser General Public License (LGPL) and is free to download. Hibernate not only takes care of the mapping from Java classes to database tables (and from Java data types to SQL data types), but also provides data query and retrieval facilities.

Before understanding hibernate framework, we need to understand Object Relational Mapping(ORM).

### ***What is ORM?***

ORM is a programming method to map the objects in java with the relational entities in the database. In this, entities/classes refers to table in database, instance of classes refers to rows and attributes of instances of classes refers to column of table in database. This provides solutions to the problems arise while developing persistence applications using traditional JDBC method. This also reduces the code that needs to be written.



### ***Need for tools like hibernate:***

The main advantage of ORM like hibernate is that it shields developers from messy SQL. Apart from this, ORM provides following benefits:

#### ***Improved productivity:***

- High-level object-oriented API
- Less Java code to write
- No SQL to write

#### ***Improved performance:***

- Sophisticated caching
- Lazy loading
- Eager loading

***Improved maintainability:***

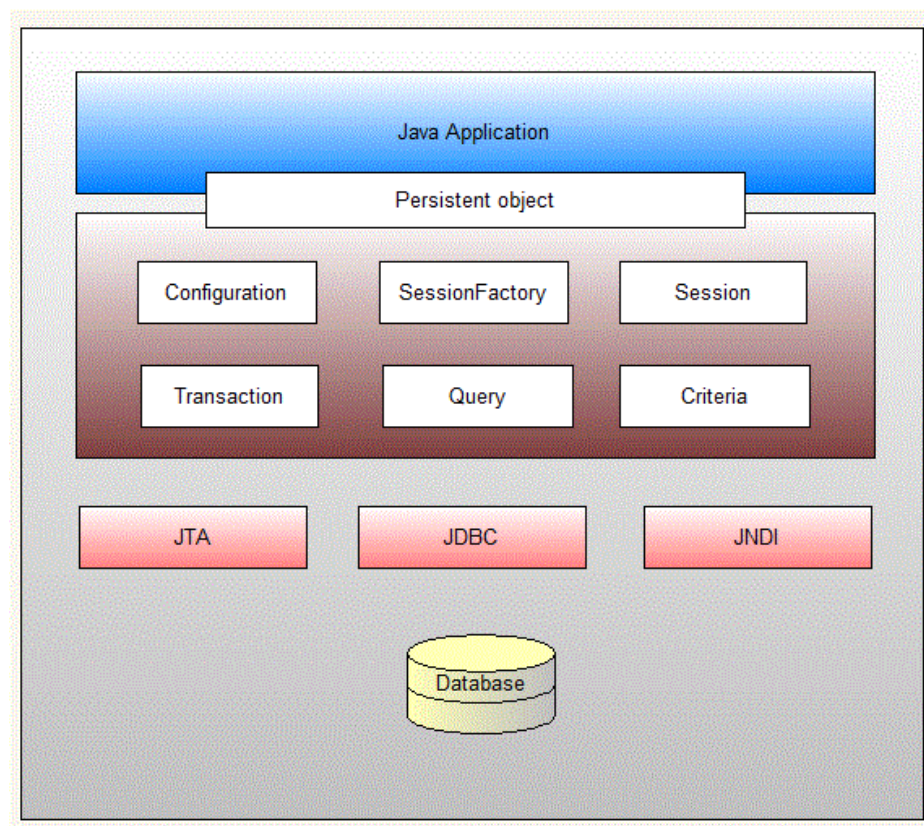
- A lot less code to write

***Improved portability:***

- ORM framework generates database-specific SQL for you

***Architecture of hibernate :***

Following is a detailed view of the Hibernate Application Architecture with few important core classes.



The Hibernate architecture is layered to keep you isolated from having to know the underlying APIs. Hibernate is like a bridge between java application and relational database.

## 1.5 LOG4J

Apache **Log4j** is a Java-based logging utility. It was originally written by Ceki Gülcü and is now a project of the Apache Software Foundation. Log4j is one of several Java logging frameworks. Gülcü has since started the SLF4J and Logback projects, with the intention of offering a successor to Log4j.

The Apache Log4j team has created a successor to Log4j 1 with version number 2. Log4j 2 was developed with a focus on the problems of Log4j 1.2, 1.3, java.util.logging and Logback, and addresses issues which appeared in those frameworks. In addition, Log4j 2 offers a plugin architecture which makes it more extensible than its predecessor. Log4j 2 is not backwards compatible with 1.x versions,<sup>[5]</sup> although an "adapter" is available.

## 1.6 JUNIT

JUnit is a unit testing framework for the Java programming language. JUnit has been important in the development of test-driven development, and is one of a family of unit testing frameworks which is collectively known as xUnit that originated with SUnit. JUnit is linked as a JAR at compile-time; the framework resides under package junit.framework for JUnit 3.8 and earlier, and under package org.junit for JUnit 4 and later.



## CHAPTER – 2

### PROJECT ANALYSIS

#### 2.1 MODULES

The following are the modules of this project

- **Authentication:** First user enters username and password to access the application. User can register itself and after to access the other features user need to authenticate himself which can be achieve by using login.
- **Create Identity:** User can create identity by providing display name, e-mail address and Date of birth and these attributes will be added in the database.
- **Search an Identity:** User can search for an entity by giving name or email address. User enter display name and email address to fetch the related record in the database if it's already present in the db.
- **Update an Identity:** User can update an entity by selecting the searched entity. User can modify entity by utilizing an already present information form which is result of user's search selection. User enter display name and email address to fetch the related record in the database if it's already present in the db.
- **Delete an Identity:** User can delete an entity by selecting the searched entity. User can delete the entity by selecting it using radio button which is generated by search.

This application is operating in reduced cost, independent.

#### 2.2 DATA DESCRIPTION

Modular approach is used for the development of this project.

There are three basic modules of this project:

## **1. IAMCORE:**

- a. Users:
  - i. This module is used to handle user related activities like user registration.
- b. DAO service:
  - i. This module is handling all database related queries using HQL.
- c. Identity data module:
  - i. This module is used to handle all identity related operations
- d. Test module:
  - i. This module is handling Junit Tests.

## **2. IAM-WEB:**

- a. Servlets:
  - i. Servlets are used to handle all control all DAO and User related activities it's acting like a controller.
- b. JSP:
  - i. JSP is acting as a view. Jsp are used to build custom tags and perform java operations and it can directly call java beans.

## **2.3 EXPECTED RESULTS**

This application can authenticate the user and can create, delete, update and search for the identity without any issue. The communication between application and database is smooth. Highly sophisticated, user friendly and secure tool created for Identity and access management.

Junit Tests are utilized in this project for the testing purpose and to make sure that there are no loopholes in the application.

## **2.4 SCOPE AND LIMITATIONS**

The password is storing in database without encryption.

## CHAPTER – 3

### CONCEPTION

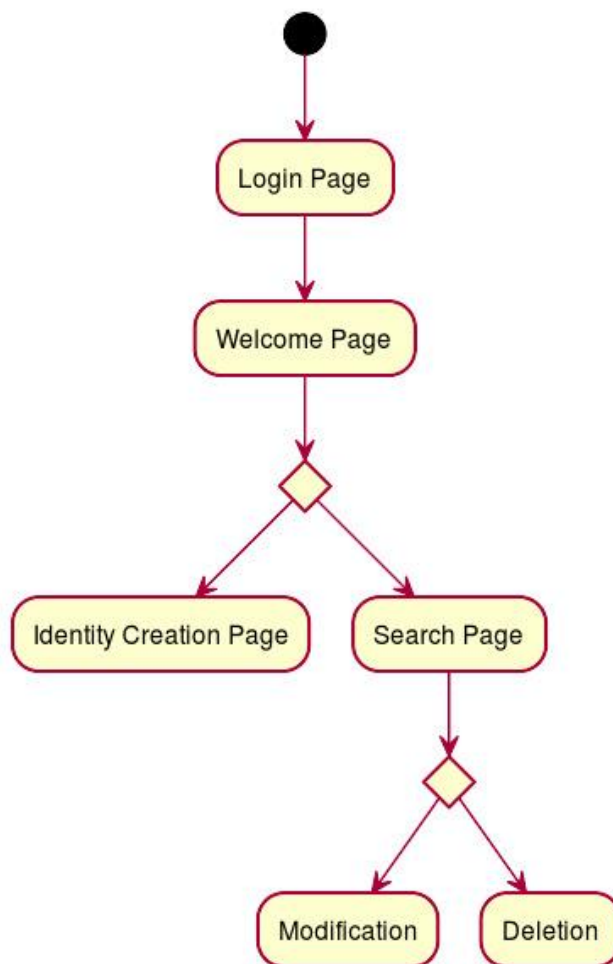
#### 3.1 ALGORITHM

The algorithms which is used for search of the identity is based on user name and user email which user can enter to search the specific identity if it already exists.

#### 3.2 DATA STRUCTURE

Data structure which is used based on presentation, business and data access layer. Which is a good technique. We can modify our application without affecting its layers.

#### 3.3 APPLICATION FLOW DIAGRAM



## CHAPTER – 4

### CONFIGURATION

#### 4.1 MAVEN

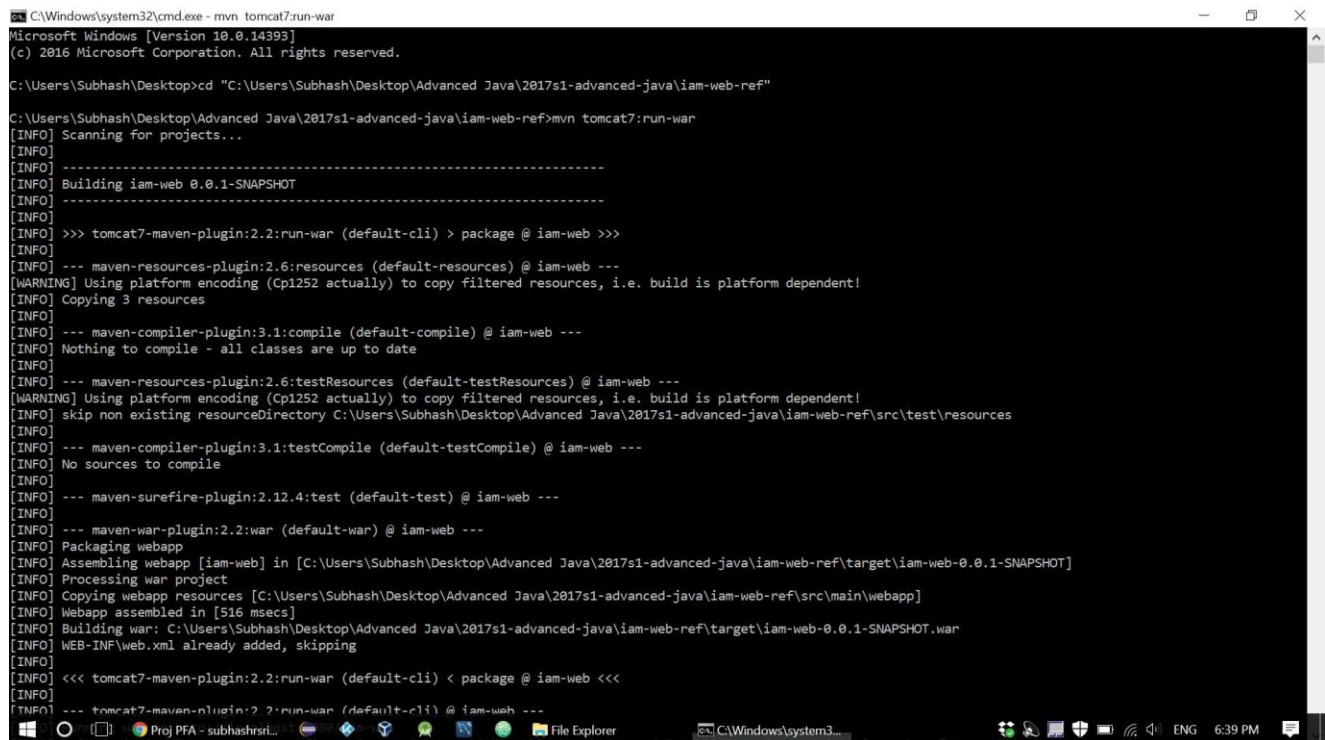
- Open Eclipse and go to: Help / Eclipse Marketplace.
- Insert on search: "Maven" and look for "Maven Integration for Eclipse WTP" and install it.
- If the installation was successfully made in Window/Preferences, the Maven option will be visible.
- Download the last version from here:
  - <http://maven.apache.org/download.html> and put it on a folder in C partition (or other partitions) - just **remember** that the path to the downloaded maven shouldn't contain spaces (you will have errors) - for example, path is: "c:\Apache\_Maven\apache-maven-3.0.4...".
- After downloading it, go in Eclipse -> Window -> Preferences -> Maven (from left menu and expand it) -> Installations -> and add path to the downloaded maven.
- To create the project: New -> Other -> Maven -> Maven Project -> and search on filter the archetype you would like to use (for example: use maven-archetype-webapp for web application etc.). Click Next and put project details: **Group Id, Artifact Id, Version** etc.
- After creating the project, the most important file is **pom.xml**. There add dependencies. Another important thing, maven local repository will be created here: "c:\Users\YOUR\_USERNAME\m2."
- To perform maven, install/build/clean etc, right click on project and click "Run as" -> Select option.

## 4.2 SERVER

### Add Server

- Windows -> Show View -> Servers.
- Then in the servers view, right click and add new.
- It will show a pop up containing many server vendors.
- Under Apache select Tomcat v7.0 (Depending upon your downloaded server version).
- And in the run time configuration point it to the Tomcat folder downloaded.

The starting of the TOMCAT7 Server to the localhost/8080



```
C:\Windows\system32\cmd.exe - mvn tomcat7:run-war
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Subhash\Desktop>cd "C:\Users\Subhash\Desktop\Advanced Java\2017s1-advanced-java\iam-web-ref"

C:\Users\Subhash\Desktop\Advanced Java\2017s1-advanced-java\iam-web-ref>mvn tomcat7:run-war
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building iam-web 0.0.1-SNAPSHOT
[INFO] -----
[INFO]
[INFO] >>> tomcat7-maven-plugin:2.2:run-war (default-cli) > package @ iam-web >>>
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ iam-web ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 3 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ iam-web ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ iam-web ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory C:\Users\Subhash\Desktop\Advanced Java\2017s1-advanced-java\iam-web-ref\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ iam-web ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ iam-web ---
[INFO]
[INFO] --- maven-war-plugin:2.2:war (default-war) @ iam-web ---
[INFO] Packaging webapp
[INFO] Assembling webapp [iam-web] in [C:\Users\Subhash\Desktop\Advanced Java\2017s1-advanced-java\iam-web-ref\target\iam-web-0.0.1-SNAPSHOT]
[INFO] Processing war project
[INFO] Copying webapp resources [C:\Users\Subhash\Desktop\Advanced Java\2017s1-advanced-java\iam-web-ref\src\main\webapp]
[INFO] Webapp assembled in [516 msecs]
[INFO] Building war: C:\Users\Subhash\Desktop\Advanced Java\2017s1-advanced-java\iam-web-ref\target\iam-web-0.0.1-SNAPSHOT.war
[INFO] WEB-INF\web.xml already added, skipping
[INFO]
[INFO] <<< tomcat7-maven-plugin:2.2:run-war (default-cli) < package @ iam-web <<<
[INFO]
[INFO] --- tomcat7-maven-plugin:2.2:run-war (default-cli) @ iam-web ---
```

## CHAPTER – 5

### SCREENSHOTS

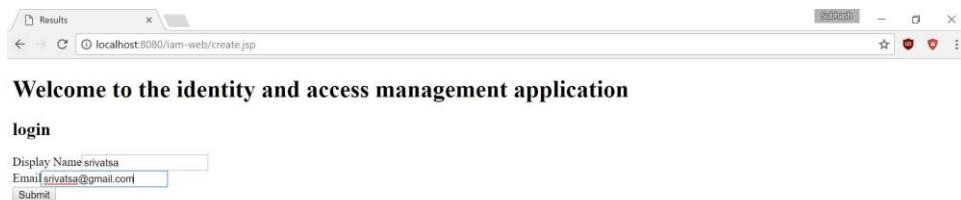
1. The Index page, where we need to enter the Login and Password for Authentication



2. The HOME page



3. Entering the first value for Create Identity page



4. After the insertion of 3 elements



**Number of objects:**

3

5. The Search page for entering user name



**Welcome to the identity and access management application**

**Search of user here**

Display Name

6. Element found after successful search



**User details**

**Display Name: subhash**

**Email: subhashrsrivatsa.paris@gmail.com**

## 7. Searching an invalid name



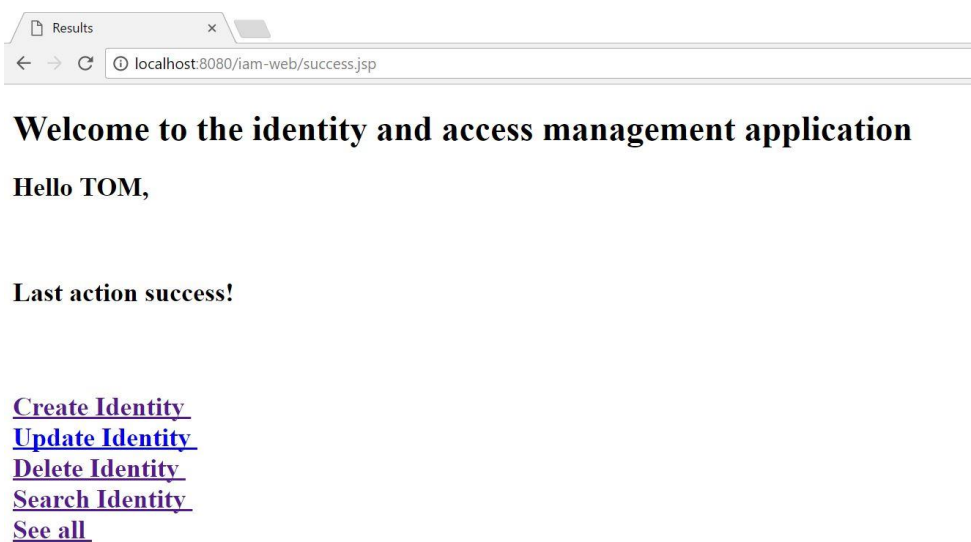
The screenshot shows a web browser window with a single tab titled 'Results'. The address bar displays 'localhost:8080/iam-web/search.jsp'. The page content includes a heading 'Welcome to the identity and access management application', a subheading 'Search of user here', and a form with the label 'Display Name' and the input value 'invalidname'. A 'Submit' button is located below the input field.

## 8. Entering the user name to delete



The screenshot shows a web browser window with a single tab titled 'Results'. The address bar displays 'localhost:8080/iam-web/delete.jsp'. The page content includes a heading 'Welcome to the identity and access management application', a subheading 'Give display name to delete user.', and a form with the label 'Display Name' and the input value 'subhash'. A 'Submit' button is located below the input field.

## 9. A success message after the last action



The screenshot shows a web browser window with a single tab titled 'Results'. The address bar displays 'localhost:8080/iam-web/success.jsp'. The page content includes a heading 'Welcome to the identity and access management application', a greeting 'Hello TOM,', a message 'Last action success!', and a list of links: 'Create Identity', 'Update Identity', 'Delete Identity', 'Search Identity', and 'See all'.



10. Updated list after the deletion of an Identity



**Number of objects:**

**2**

11. After deletion of all the User details



**Number of objects:**

**0**

## BIBLIOGRAPHY

- [\*https://en.wikipedia.org\*](https://en.wikipedia.org)
- [\*https://www.tutorialspoint.com\*](https://www.tutorialspoint.com)
- [\*https://www.youtube.com\*](https://www.youtube.com)
- [\*https://www.javatpoint.com/jsp-tutorial\*](https://www.javatpoint.com/jsp-tutorial)
- [\*https://www.w3schools.com\*](https://www.w3schools.com)