IAMWEB

Dattaprasad CHOUKEKAR

Subject description

- IAMWeb is an Identity Management system accessible though web.
- It provides the capability of managing life cycle of an Identity.
- Identity represents a person in real life.
- Default credentials.
  username=max
  password=max
- By default application is configured to run with in memory Apache derby DB.
  Hence no need to set up databases for demo purpose.

Subject analysis

1. Major features

1. Ability to create, update, delete an identity.
2. Ability to search an identity by various parameters like name, surname.
3. Ability to ***dynamically define the properties of an identity*** to extend the functionality according to requirement. Ex. You can define any new attribute like city or age.
4. Simple user interface.
5. Application is secured. Only access to predefined users.
6. UI input data validation.

2. Expected results

- When the application is deployed, user should be able to manage complete life cycle of an Identity.
- Validation in the UI will prevent the wrong data to be updated. Ex. In number type of input age, strings is not accepted.
- User will be able to search any identity by entering the properties
  - search parameters are ORed.
  - Case insensitive.

3. Scope of the application

Limits:

Maximum simultaneous users : This application is not tested for max simultaneous users.

Evolution:

This application can be updated to support internationalization.
So that the UI will be updated according to the user language.

Conception

1. Technologies Used

Java : Mimimum JDK 1.5

Servlet API : 3.0

- Sevrlet mapping is directly done in servlet class. No need to add maping each time we define a servlet class.

JSTL : 1.2

Spring : 4.2.1.RELEASE

- Spring is ued for Dependency Injection. Application is highly flexible due to use of autowire.

Hibernate : 4.3.5.Final

- Hibernate is used to simplify dao development.
- No need to write jdbc code.
- Reduced code size.
- Using criteria for filtering data.
- Can be integrated with any database.

Spring Security : 3.2

- Easy to set up
- Ability to define users and roles.

Apache Tiles : 2.2.2

- Allows templating in java web application

Junit : 2.2

- Usesful for wrtting Junit tests.

2.Data structures

Application uses two data models.

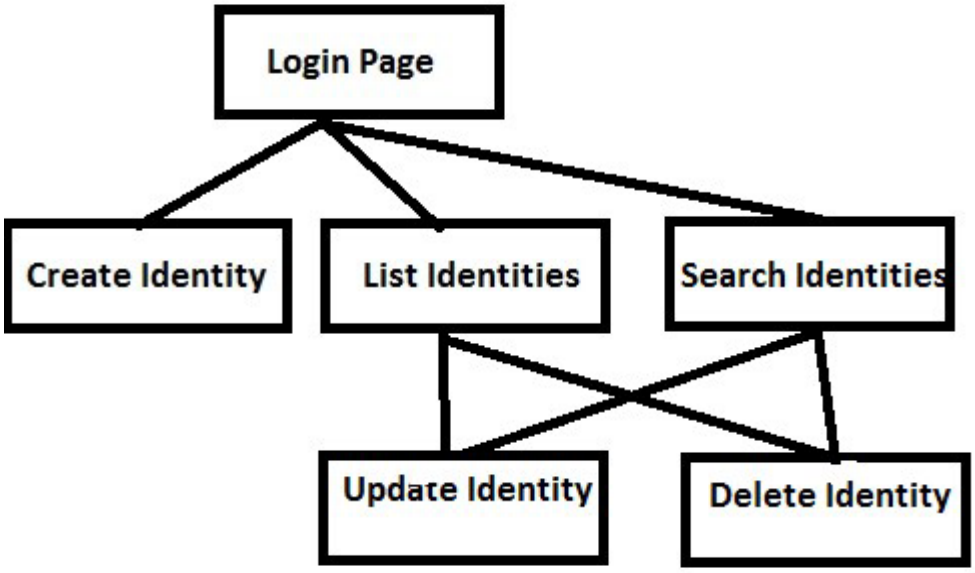
Identity model represents the identity to be stored.

```
Identity->
          ->firstName    : First Name of the user
          ->lastName     : Last name of the user
          ->email        : Email of the user
          ->birthDate    : Birth date of the user
          ->Attributes   : Defined dynamically in application
                           configuration. Key value pair. Ex. Age=12,
                          City=Paris.
```

Attribute defines the key value pairs ascociatd with entity. The properties isMandatory and type help to validate data in the form.

```
Attributes represent the key value pairs ascociated with an Identity.
Attribute->
            ->name         : Name of the attribute
            ->isMandatory  : Is this attribute is mandaotoary.
            ->type         :  Type of attribute can be one of the below
                              values TEXT, NUMBER, EMAIL, DATE, PASSWORD
```

3.Global application flow



4. Algorithms

Defining attributes dynamically.

The attributes of an Identity can be defined dynamically through "attributes-configuration.xml" configuration file.

Algorithm:

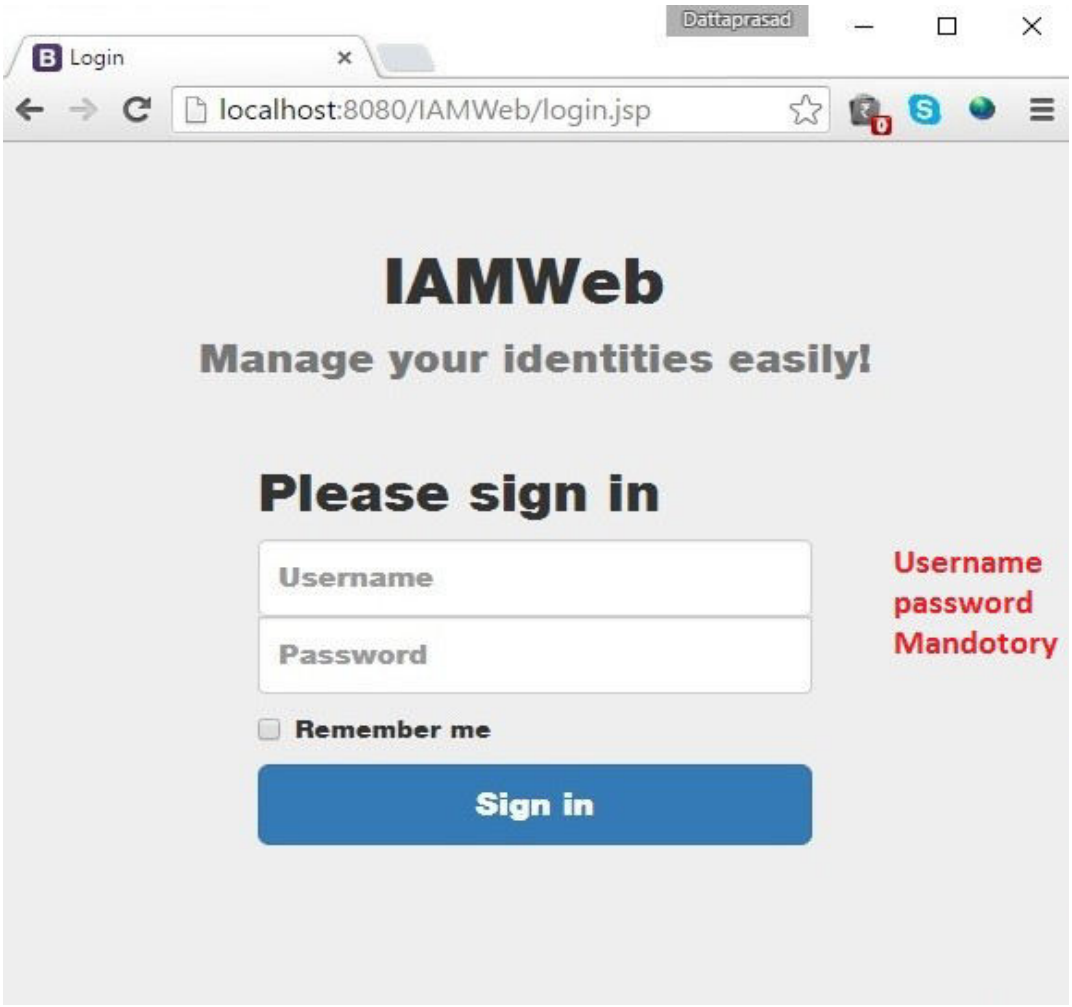
1. At the start of the application , it first reads the "attributes-configuration.xml" configuration file.
2. Reads the template file template.jsp.
3. Creates `attributetemplate.jsp` file containing each attribute defined in xml by replacing tokens.
4. `attributetemplate.jsp` is included in the all the jsp to display updated attributes.

GUI description

Comments are in red color.

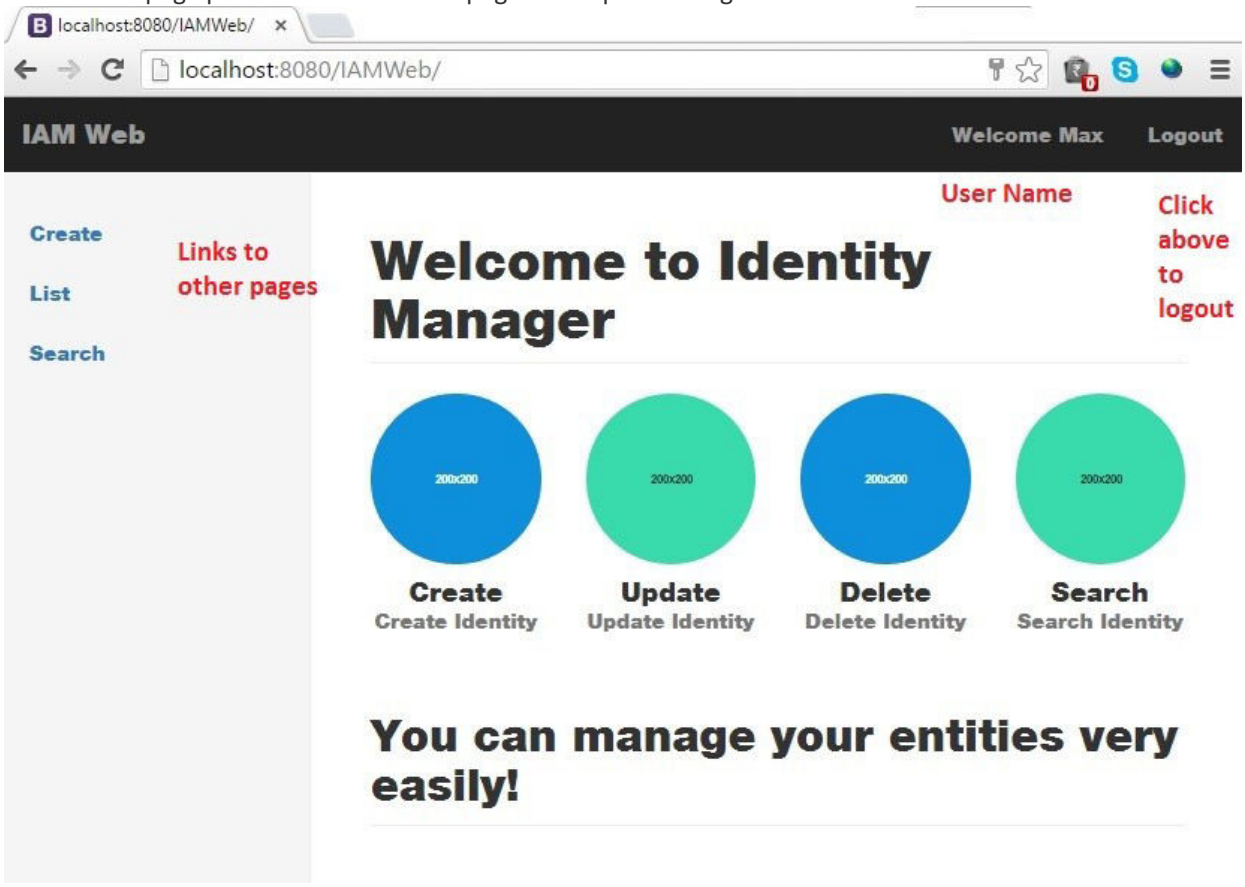
1. Login Page :

When user accesses any page it is redirected to login page.

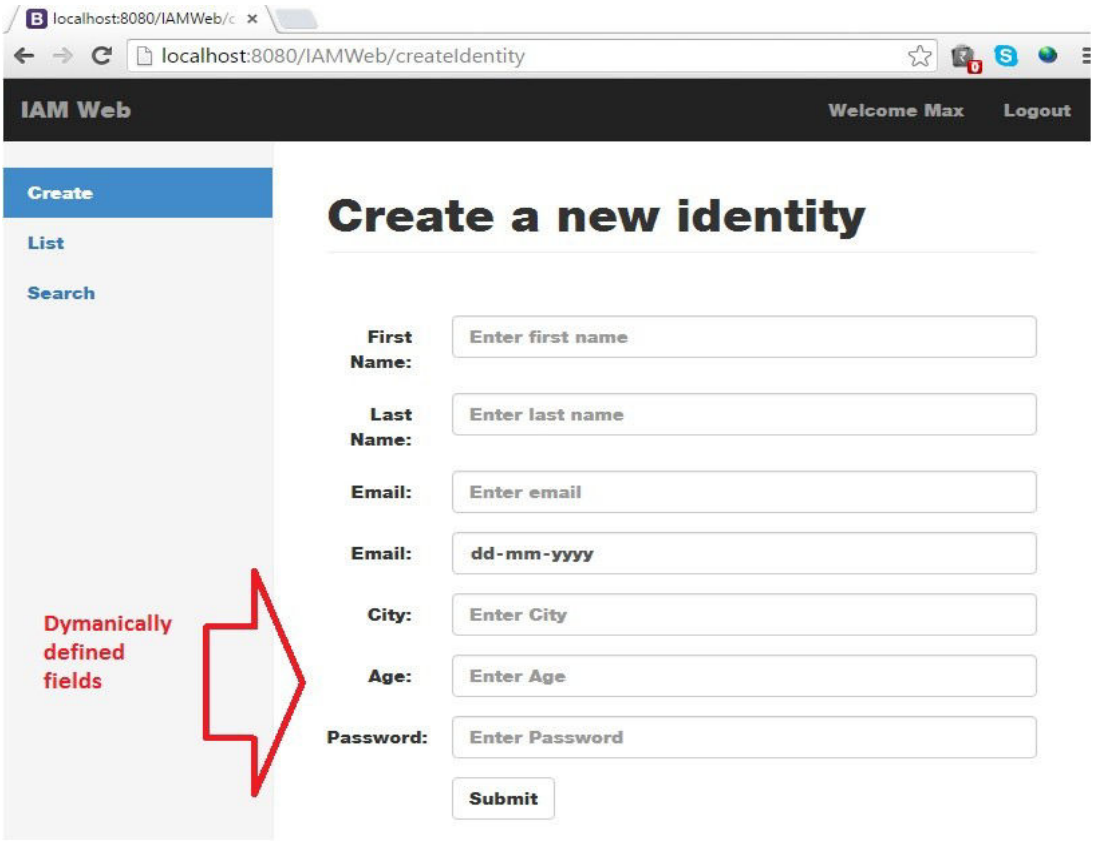


2. Homepage :

Homepage provides links to other pages and option to logout.

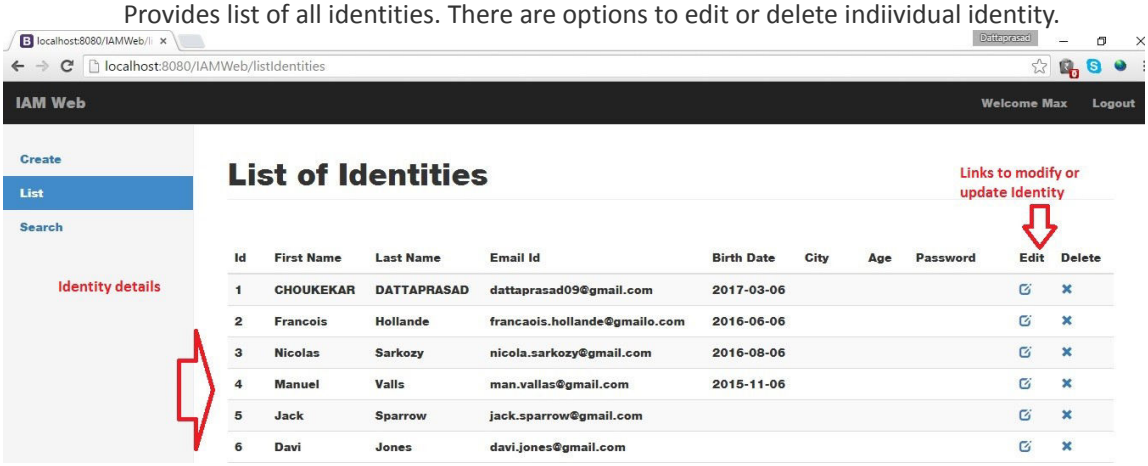


3. Create Identity :

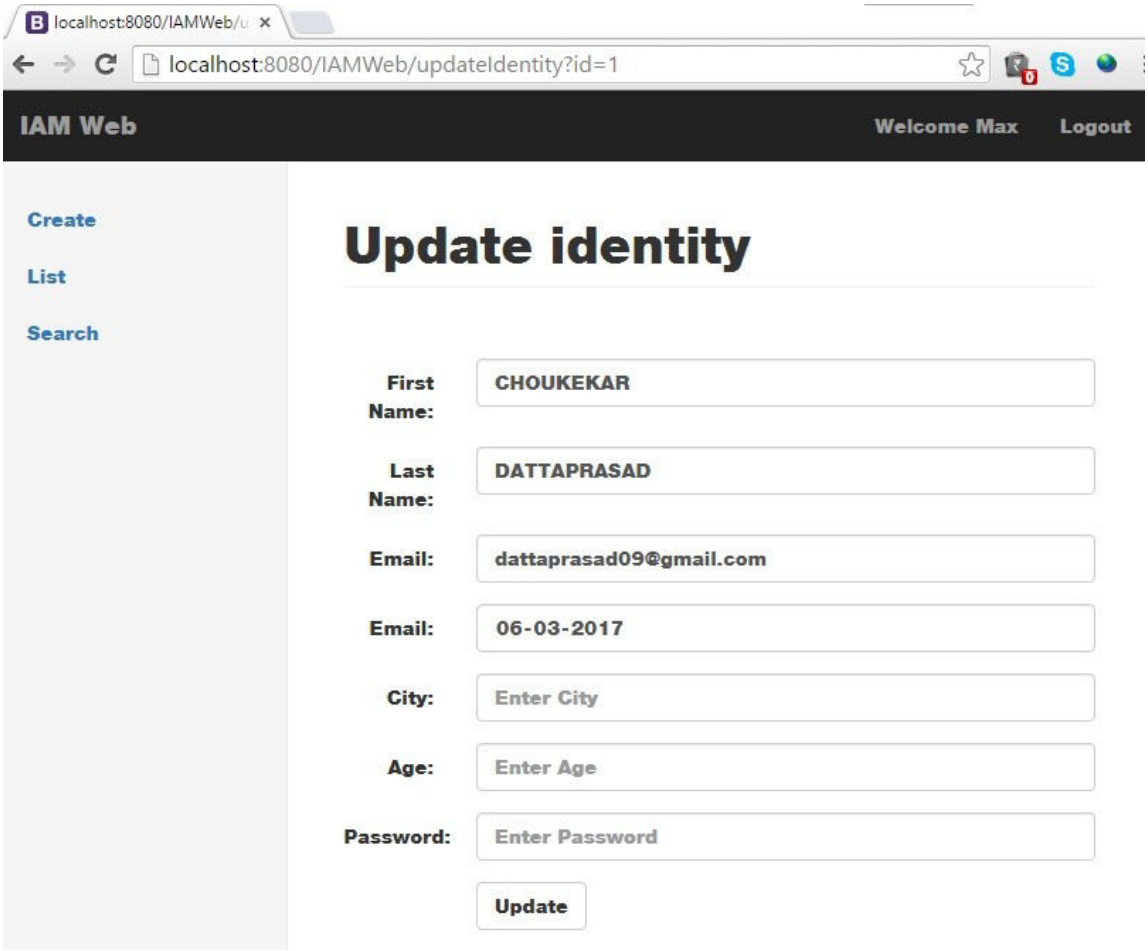


3. List of Identities :

Provides list of all identities. There are options to edit or delete indiivdual identity.

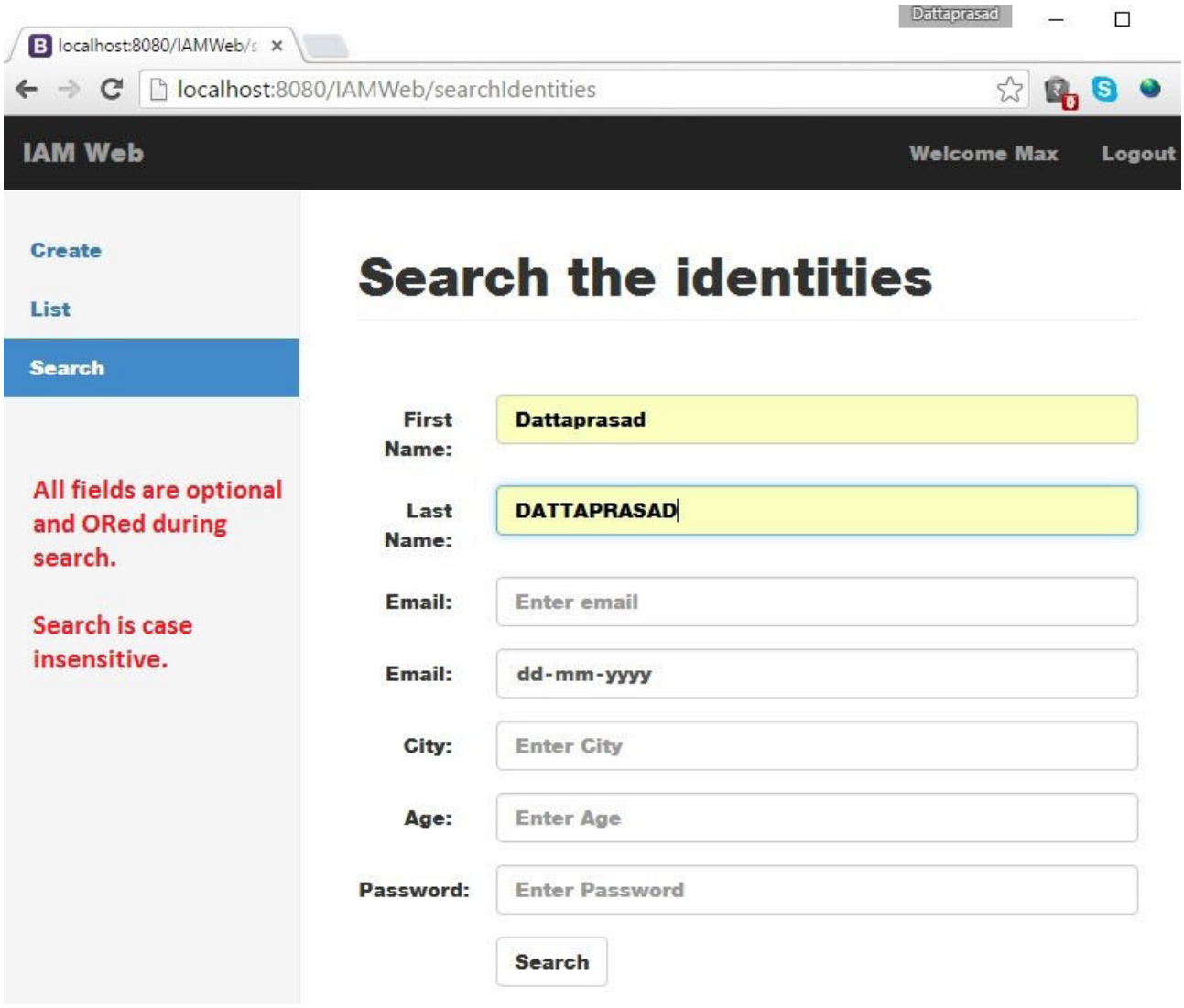


4. Update an existing identity :



5. Search Identity :

Allows to search identities by field values. All values are optional. All the attributes are searched in as OR. But all the other identity properties are searched as AND.



2.Configuration instructions

1. Defining users

Users can be defined using spring-security.xml.

By default only one user defined with username "max" and password "max"

```
<user-service>
  <user name="max" password="max" authorities="ROLE_USER" />
</user-service>
```

2.Configuring DAO implementation

Any one of the below DAO implmentation can be configured.

Configure below DAO implemenation in WEB-INF/applicationContext.xml.

- Hibernate DAO Implememtation

```
<beans:bean name="daoImpl"
class="fr.tbr.iamcore.services.impl.hibernate.HibernateDaoImpl"
scope="singleton">
</beans:bean>
```

- JDBC DAO Implementation

```
<beans:bean name="daoImpl"
class="fr.tbr.iamcore.services.impl.dbstore.IdentityDaoDbStoreImpl"
scope="singleton">
```

3. File DAO implementation

```
<beans:bean name="daoImpl"
class="fr.tbr.iamcore.services.impl.filestore.IdentityDAOFileStoreImpl"
scope="singleton">
</beans:bean>
```

4. XML DAO Implementation

```
<beans:bean name="daoImpl"
class="fr.tbr.iamcore.services.impl.xmlstore.IdentityDaoXMLStoreImpl"
scope="singleton">
</beans:bean>
```

Dynamically configuring Identity Attributes:

New attriutes can be defined by adding <attribute> element to **attributes-configuration.xml file.**
Below is the sample configuration file.

```
<attributes>
      <attribute isMandatory="true" type="TEXT">
            <name>City</name>
      </attribute>
      <attribute isMandatory="false" type="NUMBER">
            <name>Age</name>
      </attribute>
</attributes>
```

The attribute element has three values.

Name : this element defines the name of attribute in UI and DB.
IsMandatory : this attribute is true , then UI will apply validation.
Type : It defines the type of the attribute. It can have below values.

TEXT|DATE|EMAIL|PASSWORD|NUMBER

Depending on this value the UI will apply validation.

2. Load the identities at server start

Configuration for loading identites from text file at the start of web application. Set below in web.xml , identites files contains default identities.

```
<context-param>
	<param-name>loadIdentitiesFromTxtFile</param-name>
	<param-value>/WEB-INF/config/identities.txt</param-value>
</context-param>
```

3.Bibliography

1. http://getbootstrap.com/

   Template of the website is derived from boostrap.

2. http://www.mkyong.com/tutorials/spring-security-tutorials/

   Spring security is used using the above tutorial.

3. http://www.mkyong.com/tutorials/hibernate-tutorials/

   Tutorials were used for use of variaous hibernate components.

4. https://db.apache.org/derby/papers/DerbyTut/embedded_intro.html

   Details of Apache derby tutorial.

5. https://dzone.com/articles/spring-mvc-tiles-3-integration

   Using Apche tiles with Spring.