RMIT VIETNAM UNIVERSITY
INFORMATION TECHNOLOGY DEPARTMENT

COSC2440 - SOFTWARE ARCHITECTURE - DESIGN AND IMPLEMENTATION
DHIS 2 MODULE PROJECT - EDUCATION PROFILE MANAGEMENT FOR DOCTORS

PROJECT REPORT

s3275145 - Hoang Ngoc Thanh
s3275058 - Kieu Hoang Anh
s3343706 - Luong Duc Tu

EXECUTIVE SUMMARY

Firstly, we would like to thank to our lecturer - Mr. Nguyen Ngoc Thanh who help us focusing the aims of the project, supporting the general acknowledgement about concept and operations of the DHIS-2 system, and encouraging the team a lot.

Today, the educational systems are developed strongly in many countries. Every year, many students are certified by many colleges, and university. However, the management of these profiles still limit and not be developed widely. DHIS-2 is the web-based open-source information system with many useful built in services and high ability of information visualization, so EPM4D project is developed to help people easier when searching information about the certifications, the candidates, and the colleges. Moreover, the registered users can handle the information by the specific provided roles.

Table of Contents

1. Introduction

1.1 DHIS-2 Overview

DHIS-2 stands for District Health Information System, is the flexible, web-based open-source information system, provides many features for collect, validate, analysis, and present information. DHIS-2 is developed by the Health Information Systems Program (HISP) and globally distributed process with developers in many countries, and released under the BSD license and can be used at no cost. It runs on any platform with a Java Runtime Environment installed.

DHIS 2 is the preferred health management information system used in over 30 countries and even more organizations across four continents. The strong points of DHIS-2 system are supporting awesome visualization features such as GIS, charts, and pivot tables. Moreover, with DHIS-2, users can capture data on any type of devices and supported many solutions based on HTML5, SMS, and Java.

1.2 Project Objectives

EPM4D stands for Educational Profile Management for Doctors. The project is the new service which records the student certifications, candidates, and colleges. Then displaying data in browser, and supporting functionalities to the data.

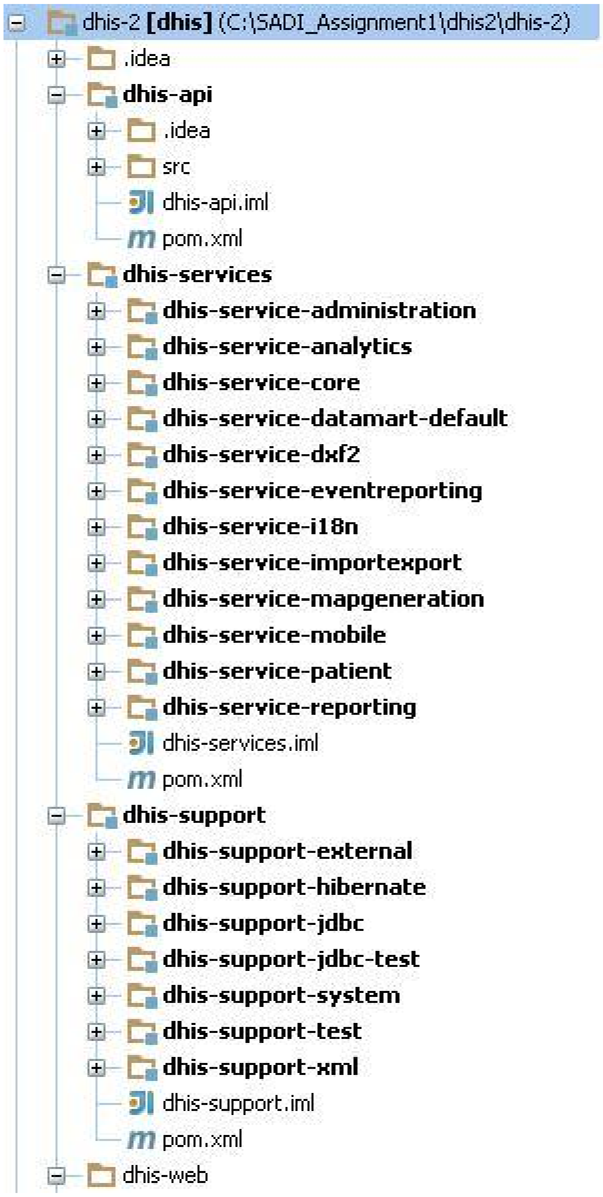
Objectives of the project

- Build new module in DHIS-2 System
- Integrated this module to DHIS-2 System and make this module visual in DHIS-2 system
- Using built-in services of DHIS-2 to implement the module functionalities
- Integrating built-in services of DHIS-2 to improve the project more convenient

2. DHIS-2

2.1 DHIS-2 Structure

DHIS-2 System contains 4 main modules which are dhis-api, dhis-services, dhis-support, and dhis-web



```
dhis-2 [dhis] (C:\SADI_Assignment1\dhis2\dhis-2)
    .idea
    dhis-api
        .idea
        src
        dhis-api.iml
        pom.xml
    dhis-services
        dhis-service-administration
        dhis-service-analytics
        dhis-service-core
        dhis-service-datamart-default
        dhis-service-dxf2
        dhis-service-eventreporting
        dhis-service-i18n
        dhis-service-importexport
        dhis-service-mapgeneration
        dhis-service-mobile
        dhis-service-patient
        dhis-service-reporting
        dhis-services.iml
        pom.xml
    dhis-support
        dhis-support-external
        dhis-support-hibernate
        dhis-support-jdbc
        dhis-support-jdbc-test
        dhis-support-system
        dhis-support-test
        dhis-support-xml
        dhis-support.iml
        pom.xml
    dhis-web
```

In DHIS2 folder tree, there are some main modules which needs to be considered.

- **\dhis-2\dhis-api:** This is the place where we store model and define service interfaces.
- **\dhis-2\dhis-services\dhis-core:** This is the place for implementing the services, and mapping the model defined in dhis-api.
- **\dhis-2\dhis-support\dhis-support-hibernate:** This is the folder we can find the hibernate-default.properties files that used to represent the project database. The default database that is H2Dialect. If developer wants to use their own database, edit that file.
- **\dhis2\dhis-2\dhis-web:** In here, developers can find the web modules.
- \dhis2\dhis-2\dhis-web-commons: contain layout of web services
- \dhis2\dhis-22\dhis-web-commons-resources: contains the web layout of the DHIS-2 system
- \dhis2\dhis-2\dhis-web\dhis-web-maintenance: maintain collection data of DHIS-2 system.
- **\dhis2\dhis-2\dhis-web\dhis-portal:** this place is the place that collects and runs all other components as a whole

2.2 How to install DHIS2 for development

1. Download and install these below components:
   Java SDK 7 (*http://www.oracle.com/technetwork/java/javase/downloads/index.html*)
   Bazaar (*http://wiki.bazaar.canonical.com/Download*)
   Maven (*http://maven.apache.org/download.cgi*)
2. Register an account at LaunchPad (*https://login.launchpad.net/BGN599r7tRgKnJ5z/+decide*) then upload a SSH key pair (*https://help.launchpad.net/YourAccount/CreatingAnSSHKeyPair*) and finally log in by invoking **bzr lp-login <username>**
3. Get a copy of the source code from **Launchpad** by invoking **bzr branch lp:dhis2**
4. To build the source code with Maven navigate to the **/dhis-2** directory and invoke *mvn install* then navigate to the **/dhis-2/dhis-web** directory and invoke *mvn install again*.
5. Each project in the **/dhis-2/dhis-web** directory is an individual web module. The **dhis-web-portal** project is an assembly of all the individual web modules. All of these modules can be started by invoking *mvn jetty:run-war* The web application can then be accessed at **http://localhost:8080**
6. Install MySQL (H2 database or PostgreSQL)

3. EPM4D - First Time of Building Project

3.1 Creating EPM4D module in DHIS-2 system

At the first time of building project, we create new module inside the DHIS-2 system in command line.

- Go to DHIS-2 web folder:

```
cd ../dhis2/dhis-2/dhis-web
```

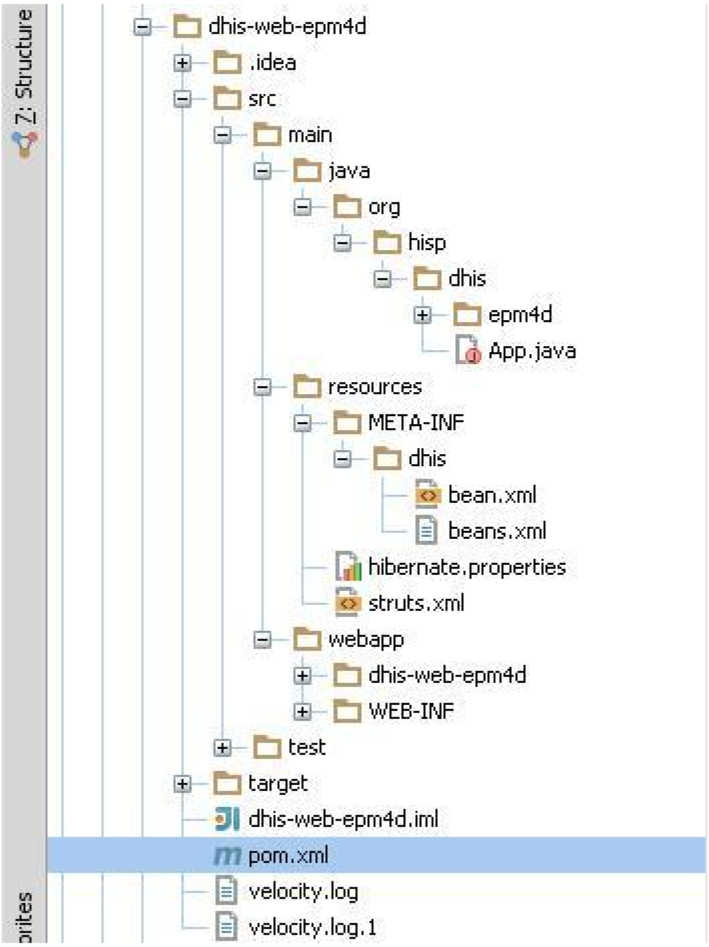
- Create new module by Maven command:

```
mvn archetype:generate -DgroupId=org.dhis.hisp. -DartifactId=dhis-web-epm4d -
DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

This command will automatically create new module with group id is org.dhis.hisp, and artifactId is dhis-web-epm4d.
Because the command is executed under the dhis-web module so in the pom.xml file which generated by run mvn command, will receive dhis-web as the parent module, and in dhis-web pom.xml file will automatically generate new module named dhis-web-epm4d which is the project module.

Build structure directory for the project

- Open project in Intellij IDE and create directory structure similar to the directory structure of each module in DHIS-2 system.



```
dhis-web-epm4d
    .idea
    src
        main
            java
                org
                    hisp
                        dhis
                            epm4d
                            App.java
            resources
                META-INF
                    dhis
                        bean.xml
                        beans.xml
                hibernate.properties
                struts.xml
            webapp
                dhis-web-epm4d
                WEB-INF
        test
    target
    dhis-web-epm4d.iml
    pom.xml
    velocity.log
    velocity.log.1
```

3.2 Hibernate and MySQL

After successfully installing DHIS2 folders for development, you now need to configure hibernate file and install MySQL (in this project, we use MySQL to develop, if you are already committed in H2 or PostgreSQL, feel free to use your own. The configuration can be found at *www.dhis2.org/development*).

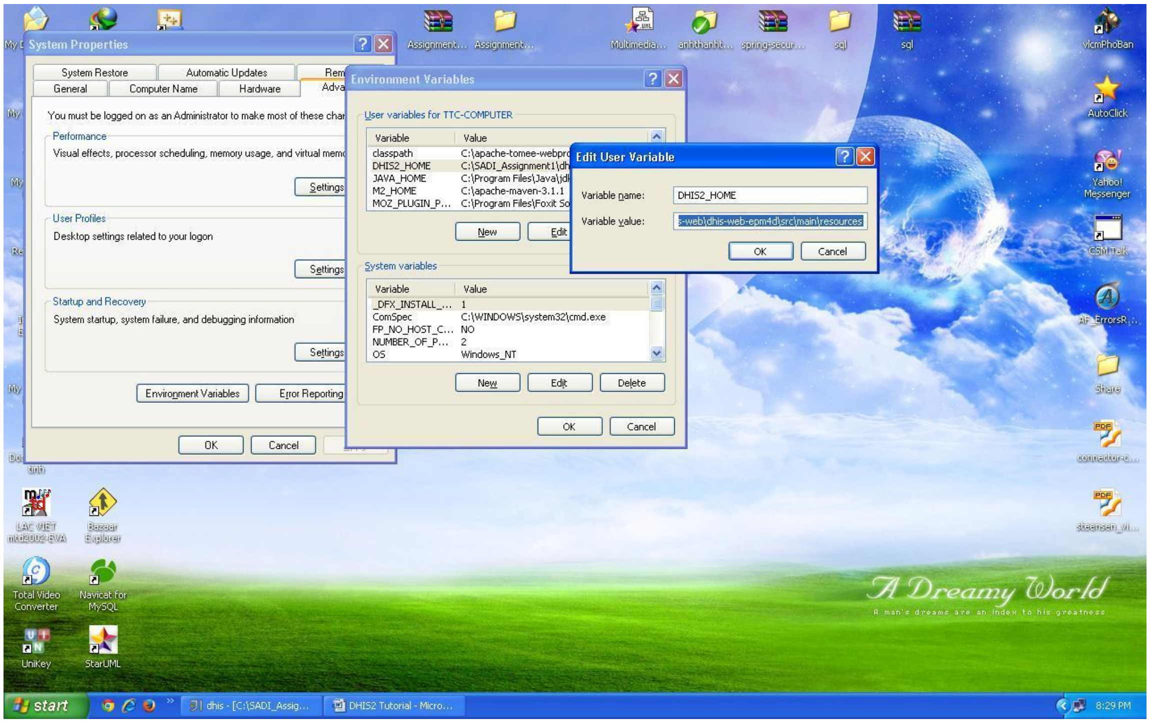
- DHIS2 has its own hibernate configuration
- It's hibernate-default.properties locates in: *dhis-2/dhis-support/dhis-support-hibernate/src/main/resources*
- By default dhis2 use h2 database to store databases

➔ Now we need to override this by:

  - Under our module/src/main/resources (or any safe place in your computer), create *hibernate.properties* file
  - Content of this file:

hibernate.dialect = org.hibernate.dialect.MySQLDialect
hibernate.connection.driver_class = com.mysql.jdbc.Driver
hibernate.connection.url = jdbc:mysql://localhost/tutorial
hibernate.connection.username = root (your MySQL username)
hibernate.connection.password = dhis (your MySQL password)
hibernate.hbm2ddl.auto = update

- Create classpath to this file: create new classpath named DHIS2_HOME in variable environment points to above *hibernate.properties*



System Properties

You must be logged on as an Administrator to make most of these char

Performance: Visual effects, processor scheduling, memory usage, and virtual memo

User Profiles: Desktop settings related to your logon

Startup and Recovery: System startup, system failure, and debugging information

Environment Variables

User variables for TTC-COMPUTER

Variable	Value
classpath	C:\apache-tomee-webpro
DHIS2_HOME	C:\SADI_Assignment1\dh
JAVA_HOME	C:\Program Files\Java\jdk
M2_HOME	C:\apache-maven-3.1.1
MOZ_PLUGIN_P...	C:\Program Files\Foxit So

System variables

Variable	Value
_DFX_INSTALL_...	1
ComSpec	C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	2
OS	Windows_NT

Edit User Variable

Variable name: DHIS2_HOME

Variable value: s-web\dhis-web-epm4d\src\main\resources

3.3 Integrate new module into DHIS2

1. Open dhis-2/dhis-web/pom.xml:
In tag modules, make sure our module existed like this

```
<modules>
   <module>dhis-web-commons</module>
   <module>dhis-web-commons-
resources</module>
```

```
    <module>dhis-web-api-mobile</module>
    <module>dhis-web-api</module>
    <module>dhis-web-api-fred</module>
    <module>dhis-web-maintenance</module>
    <module>dhis-web-dataentry</module>
    <module>dhis-web-importexport</module>
    <module>dhis-web-validationrule</module>
    <module>dhis-web-reporting</module>
    <module>dhis-web-mapping</module>
    <module>dhis-web-visualizer</module>
    <module>dhis-web-pivot</module>
    <module>dhis-web-dashboard-
integration</module>
    <module>dhis-web-caseentry</module>
    <module>dhis-web-light</module>
    <module>dhis-web-mobile</module>
    <module>dhis-web-sms</module>
    <module>dhis-web-portal</module>
    <module>dhis-web-epm4d</module>
  </modules>
```

2. Open dhis-2/dhis-web/dhis2-web-portal/pom.xml:
Add new dependency (our module) like this

```
<dependencies>
…………..
…………..
<dependency>
      <groupId>org.hisp.dhis</groupId>
      <artifactId>dhis-web-maintenance-appmanager</artifactId>
      <version>${project.version}</version>
      <type>war</type>
    </dependency>

<!-- new module -->
    <dependency>
```

```
        <groupId>org.hisp.dhis</groupId>
        <artifactId>dhis-web-epm4d</artifactId>
        <version>${project.version}</version>
        <type>war</type>
      </dependency>
    </dependencies>
```

3. Open dhis-2/dhis-web/dhis-web-commons/src/main/resources/META-INF/dhis/beans.xml
Add our module into list of web-portal like this

```
  <bean id="org.hisp.dhis.webportal.module.ConfigurableModuleComparator"
    class="org.hisp.dhis.webportal.module.ConfigurableModuleComparator">
    <property name="order">
      <list>
        <value>dhis-web-maintenance-appmanager</value>
        <value>dhis-web-maintenance-dataadmin</value>
        <value>dhis-web-maintenance-datadictionary</value>
        <value>dhis-web-maintenance-dataset</value>
        <value>dhis-web-maintenance-mobile</value>
        <value>dhis-web-maintenance-organisationunit</value>
        <value>dhis-web-maintenance-patient</value>
        <value>dhis-web-maintenance-settings</value>
        <value>dhis-web-maintenance-user</value>
        <value>dhis-web-appmanager</value>
        <value>dhis-web-dashboard-integration</value>
        <value>dhis-web-dataentry</value>
        <value>dhis-web-validationrule</value>
        <value>dhis-web-visualizer</value>
        <value>dhis-web-mapping</value>
        <value>dhis-web-importexport</value>
        <value>dhis-web-caseentry</value>
        <value>dhis-web-light</value>
        <value>dhis-web-mobile</value>
        <value>dhis-web-pivot</value>
        <value>dhis-web-reporting</value>
```

```
                <value>dhis-web-sms</value>
                <!-- New value -->
                <value>dhis-web-epm4d</value>
              </list>
            </property>
      </bean>
```

4. Dependencies

- add some necessary dependencies into the project pom.xml file such as dhis-api, dhis-web-commons, dhis-web-commons-resources, dhis-service-core… These dependencies contain other necessary dependencies for our project.

5. Create web.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
  <display-name>EDUCTION PROFILE MANAGEMENT</display-name>
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath*:/META-INF/dhis/beans.xml</param-value>
  </context-param>
  <context-param>
    <param-name>automaticAccessType</param-name>
    <param-value>ghostAdmin</param-value>
  </context-param>

  <!-- FILTERS -->

  <filter>

    <filter-name>RedirectFilter</filter-name>

    <filter-class>org.hisp.dhis.servlet.filter.HttpRedirectFilter</filter-class>

    <init-param>

      <param-name>redirectPath</param-name>

      <param-value>dhis-web-epm4d/index.action</param-value>

    </init-param>

  </filter>

  <filter>

    <filter-name>OpenSessionInViewFilter</filter-name>

    <filter-
class>org.springframework.orm.hibernate4.support.OpenSessionInViewFilter</filter-class>

  </filter>

  <filter>

    <filter-name>springSecurityFilterChain</filter-name>

    <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>

  </filter>

  <filter>

    <filter-name>Struts</filter-name>

    <filter-
class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>
```

```
</filter>
<filter>
    <filter-name>encodingFilter</filter-name>
    <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
    <init-param>
        <param-name>encoding</param-name>
        <param-value>UTF-8</param-value>
    </init-param>
    <init-param>
        <param-name>forceEncoding</param-name>
        <param-value>true</param-value>
    </init-param>
</filter>
<!-- FILTER MAPPING -->
<filter-mapping>
    <filter-name>encodingFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
    <filter-name>RedirectFilter</filter-name>
    <url-pattern>/</url-pattern>
</filter-mapping>
<filter-mapping>
```

```
    <filter-name>OpenSessionInViewFilter</filter-name>
    <url-pattern>*.action</url-pattern>
  </filter-mapping>
  <filter-mapping>
    <filter-name>OpenSessionInViewFilter</filter-name>
    <url-pattern>/api/*</url-pattern>
 </filter-mapping>
  <filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>

  <filter-mapping>
    <filter-name>Struts</filter-name>
    <url-pattern>*.action</url-pattern>
  </filter-mapping>
  <listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>
  <listener>
    <listener-class>org.hisp.dhis.system.startup.StartupListener</listener-class>
  </listener>
  <!-- Web API -->
  <servlet>
    <servlet-name>webapiServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>classpath*:/META-INF/dhis/servlet.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>webapiServlet</servlet-name>
    <url-pattern>/api</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>webapiServlet</servlet-name>
    <url-pattern>/api/*</url-pattern>
  </servlet-mapping>
</web-app>
```

Now our new module is visible in browser when running *mvn jetty:run-war* in *dhis-web-portal*

4. EPM4D - Design, Structure and Implementation

4.1 EPM4D - Design and Functionalities

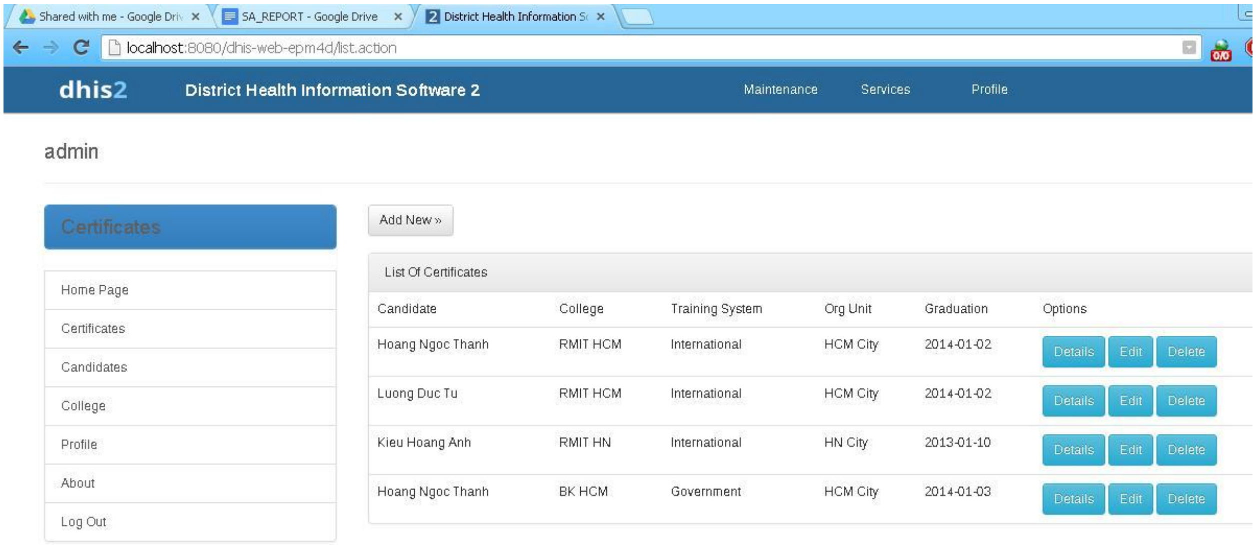
The project designed based on 3-tiers layer structure.

For the presentation layer we define 3 classes' certificate, candidate, and college and their service interfaces in dhis-api. These classes also act like our model

For the service layer, we implement these interfaces above in dhis-services/dhis-service-core/src/main/java

For the persistent layer, we build the mapping hibernate file in dhis-services/dhis-service-core/src/main/resources

The functionalities, we just implement simple CRUD to the data model.



List Of Certificates

Candidate	College	Training System	Org Unit	Graduation	Options
Hoang Ngoc Thanh	RMIT HCM	International	HCM City	2014-01-02	Details Edit Delete
Luong Duc Tu	RMIT HCM	International	HCM City	2014-01-02	Details Edit Delete
Kieu Hoang Anh	RMIT HN	International	HN City	2013-01-10	Details Edit Delete
Hoang Ngoc Thanh	BK HCM	Government	HCM City	2014-01-03	Details Edit Delete

4. 2 Framework

We inherit the existed framework in DHIS-2 system- Hibernate by mapping our objects to the SQL database through defining the hbm.xml file

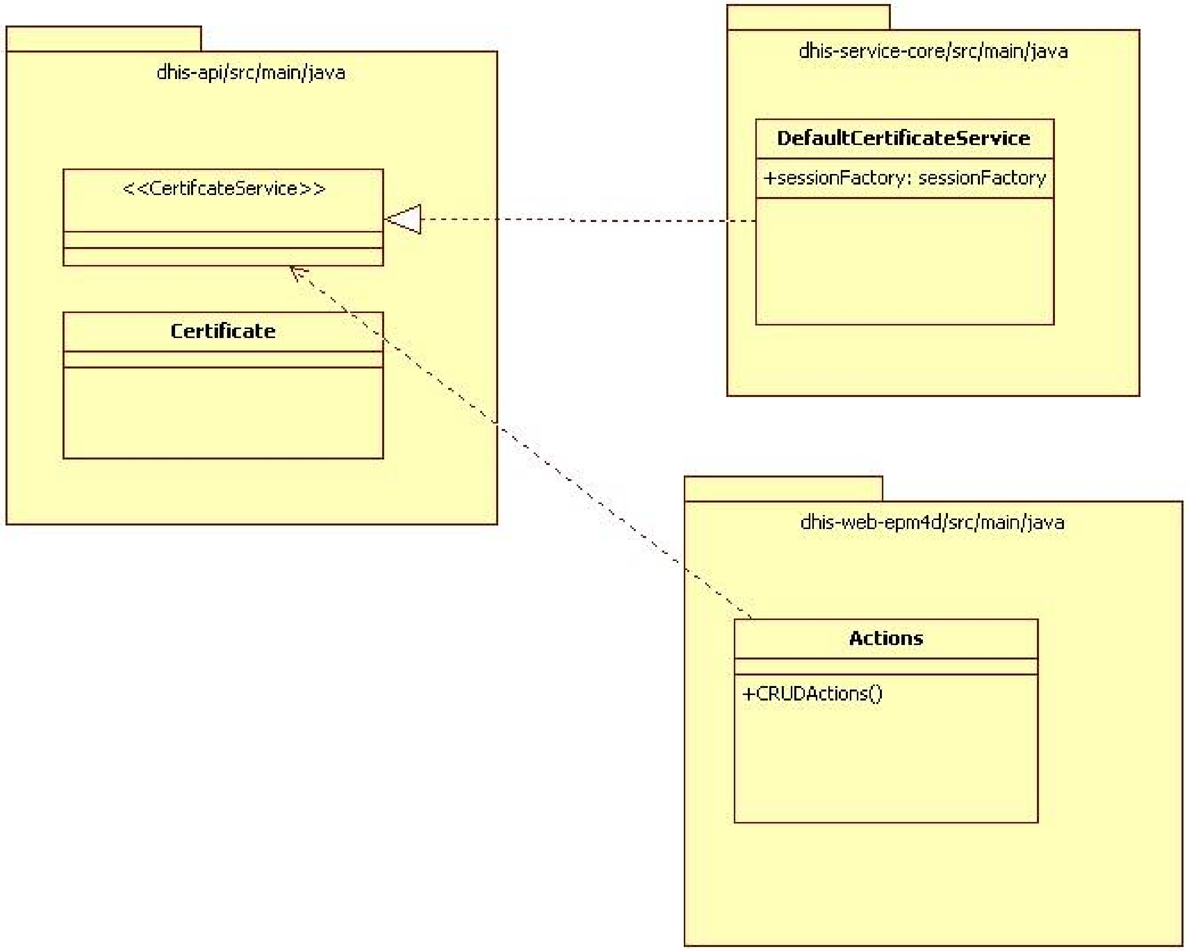
Using Session Factory from DHIS-2 system

- In dhis-services/dhis-service-core/src/main/resources/META-INF/dhis/beans.xml we reference the session factory to user services, and use it in the implementation classes.

```
<!-- Certificate -->

  <bean id="org.hisp.dhis.certificate.CertificateService"
class="org.hisp.dhis.certificate.DefaultCertificateService">

    <property name="sessionFactory" ref="sessionFactory" />

  </bean>

  <!-- Candidate -->

  <bean id="org.hisp.dhis.candidate.CandidateService"
class="org.hisp.dhis.candidate.DefaultCandidateService">

      <property name="sessionFactory" ref="sessionFactory" />

  </bean>

  <!-- College -->

  <bean id="org.hisp.dhis.college.CollegeService"
class="org.hisp.dhis.college.DefaultCollegeService">

        <property name="sessionFactory" ref="sessionFactory" />

  </bean>
```

4.3 Implementation

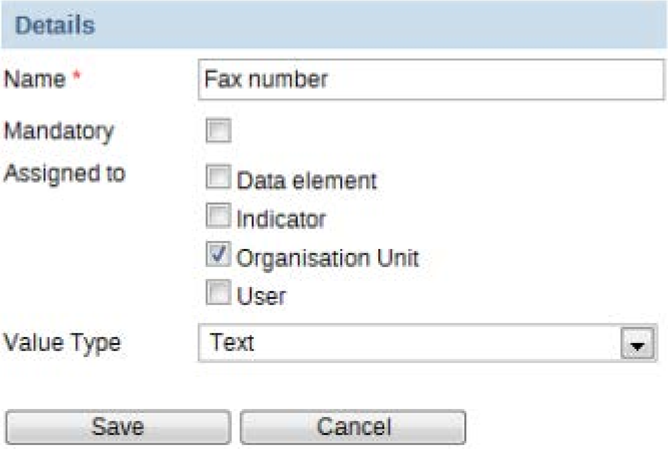
This is the class diagram of implementing one object in the project


dhis-api/src/main/java
<<CertifcateService>>
Certificate
dhis-service-core/src/main/java
DefaultCertificateService
+sessionFactory: sessionFactory
dhis-web-epm4d/src/main/java
Actions
+CRUDActions()

5. EPM4D - Improvement Based On Built-In DHIS2 Service

5.1 Attributes

Dynamic attributes can be used to add additional information to a specific object. To add a new dynamic attribute to an object, select "Maintenance->Data administration" from the main menu, then "Attribute" from the left side panel, and press the "Add new" button[2].



Details

Name *	Fax number
Mandatory	□
Assigned to	□ Data element □ Indicator ☑ Organisation Unit □ User
Value Type	Text

To create a new attribute, assign it a name. Each attribute should have a unique name. Check the tick-box "Mandatory" if the object should always have the dynamic attribute. Next, select which object (or objects) the attribute should be assigned to[2]. Lastly, select the value type. You can choose from "Text", "Yes/No", "Date", "Number", "Integer", "Positive integer" and "Negative integer". If the value supplied for the attribute does not match the value type, an error will result. Finally, click "Save" to save the attribute. The dynamic attribute will now be present in the object which you assigned it to in the respective "Edit" screen of each the object[2]. By default, new classes which create object will not show up in the object list when you create a new attribute, to make it happen, we have analyzed and found out the below necessary steps:

1. dhis-api/main/java/org.hisp.dhis/certificate/Certificate.java

   - declare attribute values as a hash set

```
private Set<AttributeValue> attributeValues = new HashSet<AttributeValue>();
```

2. dhis-api/main/java/org.hisp.dhis/attribute/Attribute.java

- declare boolean certificateAttribute

- declare boolean isCertififcateAttribute()

- setMethod for certificateAttribute

3. dhis-api/main/java/org.hisp.dhis/attribute/AttributeStore.java

- declare Set<Attribute> getCertificateAttributes()

4 dhis-api/main/java/org.hisp.dhis/attribute/AttributeService.java

- declare Set<Attribute> getCertificateAttributes();

5. dhis-services/dhis-service-core/main/java/org.hisp.dhis/attribute/DefaultAttributeService.java

- implement set<Attribute> getCertificateAttributes();

6. dhis-services/dhis-service-core/main/java/org.hisp.dhis/attribute/hibernate/HibernateAttributeStore.java

- implement public Set<Attribute> getCertificateAttributes()

7. dhis-support/dhis-support-system/main/java/deletion/DeletionHandler.java

- declare public void deleteCertificate(Certificate certificate)

8. dhis-services/dhis-service-core/main/java/org.hisp.dhis/attribute/AttributeValueDeletionHandler.java

- implement deleteCertificate(Certificate certificate)

9. dhis-services/dhis-service-core/main/resources/org.hisp.dhis/attribute.hibernate/Attribute.hbm.xml

- add <property name="certificateAttribute" not-null="false" />

10. Create GUI

dhis-web/dhis-web-maintenace/dhis-web-maintanace-dataadmin

- webapp/attribute.vm --> add new label for in detailsArea, innerHTML

- webapp/addAttributeForm.vm --> add new check box for new attribute

- action/AddAttributeAction --> add new functions for new attribute

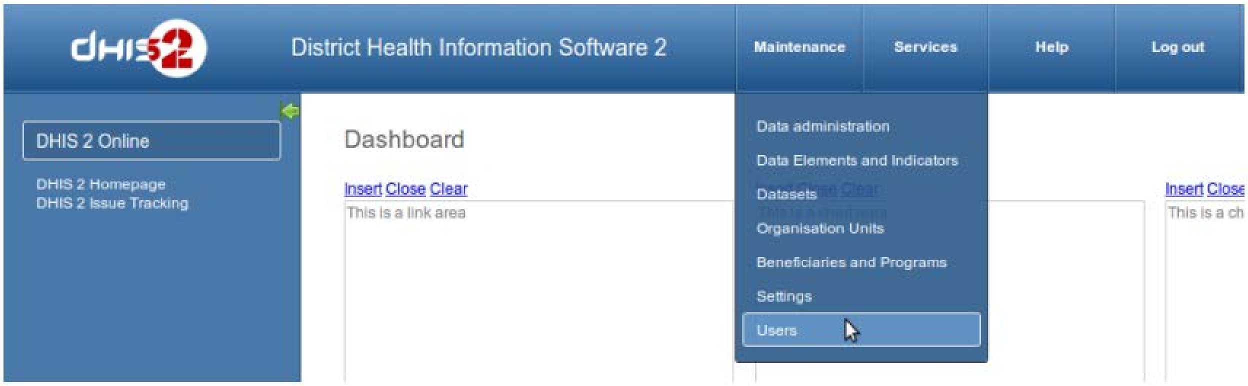
- webapp/updateAttributeForm.vm --> add new check box for new attribute

- action/UpdateAttributeAction --> add new functions for new attribute

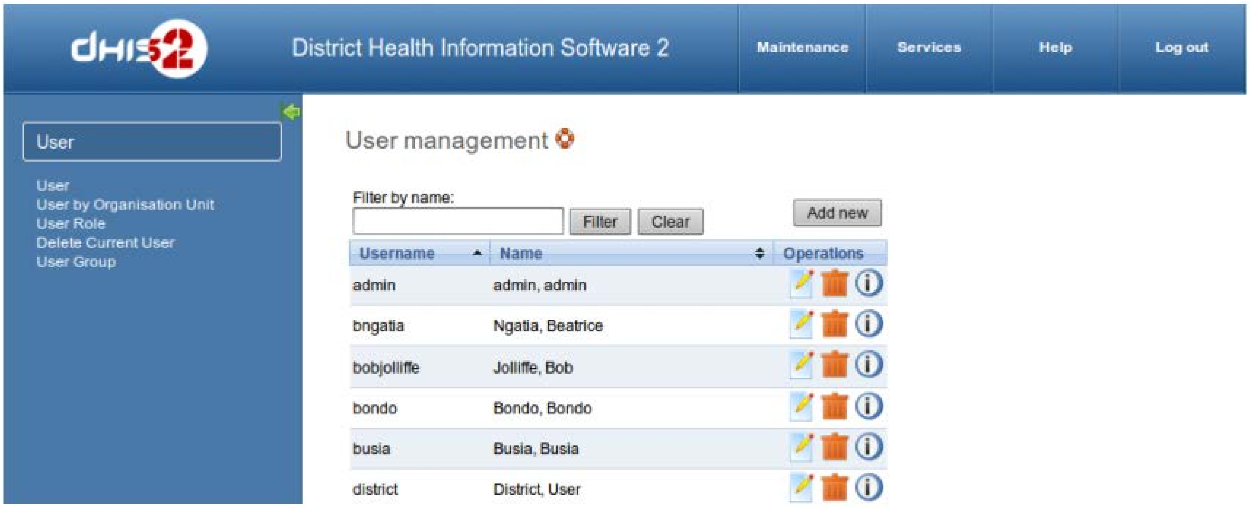
- jsonAttribute.vm

5.2 User and User Role

In this bound of project, we reuse the User and User Role functions in DHIS2. To create or find a user begin with clicking on the 'user' module displayed in the drop down menu of the Maintenance module located on the main tool bar on the top part of the displayed screen[2].

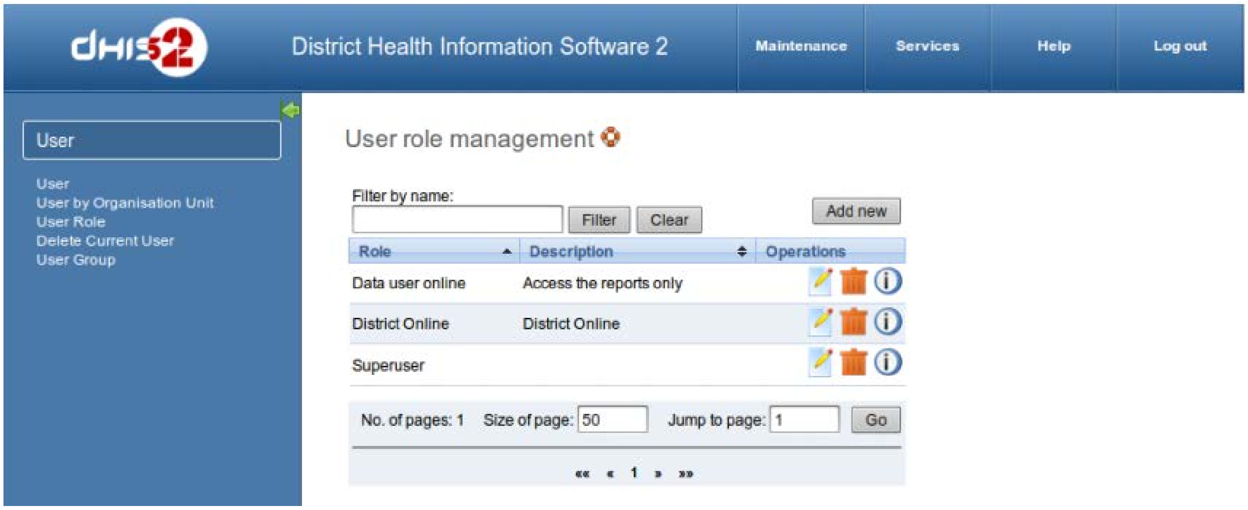


It will show up the registered users with CRUD functions.



Username ▲	Name ⇕	Operations
admin	admin, admin	✎ 🗑 ⓘ
bngatia	Ngatia, Beatrice	✎ 🗑 ⓘ
bobjolliffe	Jolliffe, Bob	✎ 🗑 ⓘ
bondo	Bondo, Bondo	✎ 🗑 ⓘ
busia	Busia, Busia	✎ 🗑 ⓘ
district	District, User	✎ 🗑 ⓘ

Before you can create a new user, you must define a new role. Do so by clicking on the ‘user role’ appearing on the left side of the displayed screen. This will lead you to the Role Management page where you will have to click on Add new to create a new role.



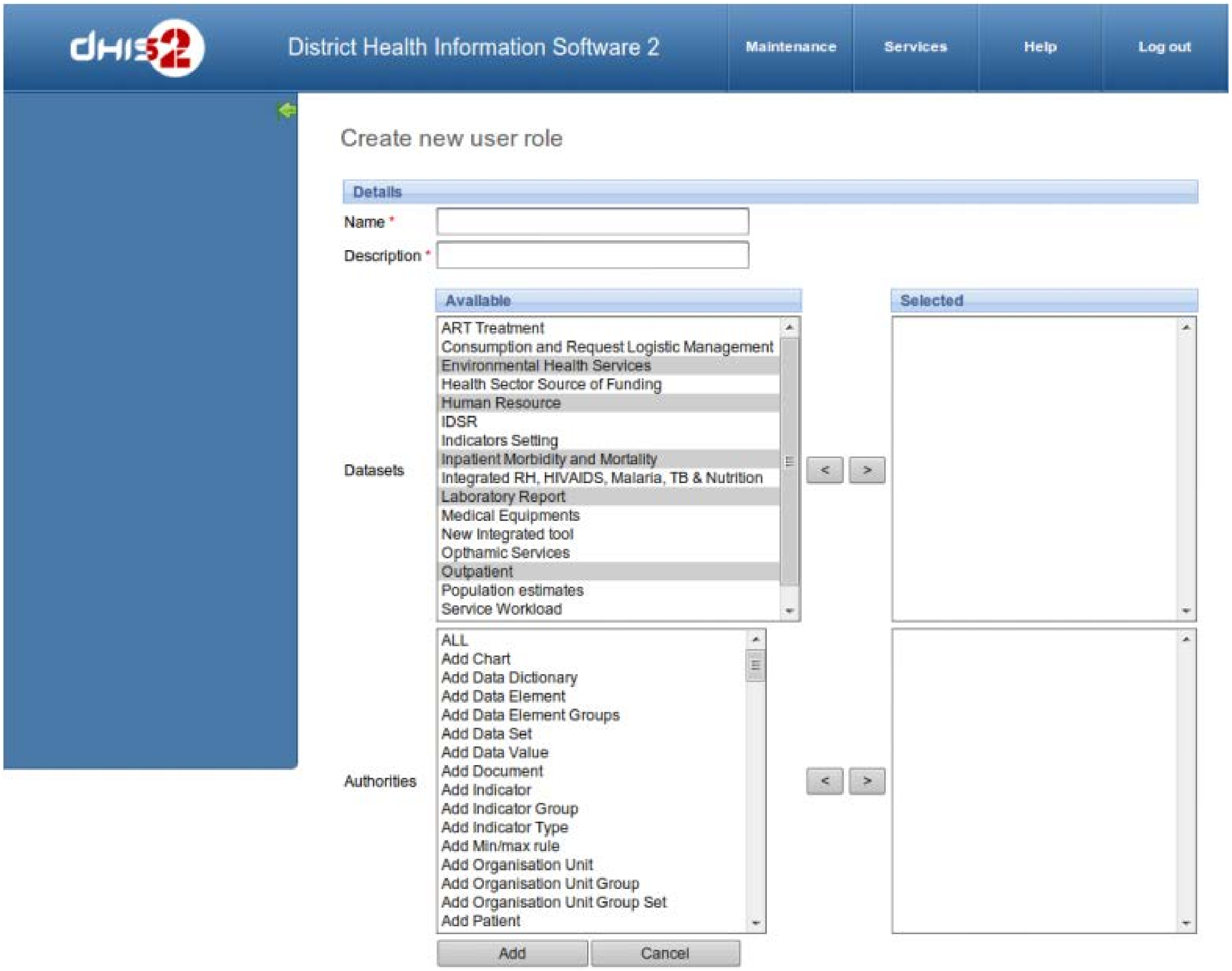
User role management

Filter by name:

Role	Description	Operations
Data user online	Access the reports only	
District Online	District Online	
Superuser		

No. of pages: 1 Size of page: 50 Jump to page: 1

When clicking in “Add new” button, a new window will appear as below:



Create new user role

Details

Name *

Description *

Datasets

Available:
- ART Treatment
- Consumption and Request Logistic Management
- Environmental Health Services
- Health Sector Source of Funding
- Human Resource
- IDSR
- Indicators Setting
- Inpatient Morbidity and Mortality
- Integrated RH, HIVAIDS, Malaria, TB & Nutrition
- Laboratory Report
- Medical Equipments
- New Integrated tool
- Opthamic Services
- Outpatient
- Population estimates
- Service Workload

Selected

Authorities

Available:
- ALL
- Add Chart
- Add Data Dictionary
- Add Data Element
- Add Data Element Groups
- Add Data Set
- Add Data Value
- Add Document
- Add Indicator
- Add Indicator Group
- Add Indicator Type
- Add Min/max rule
- Add Organisation Unit
- Add Organisation Unit Group
- Add Organisation Unit Group Set
- Add Patient

Name field is the name of the role, like User, Admin or Moderator. Description field should store the brief description about that role, something like “Access the Certificate only”.

Next you will specify the particular data set(s) that are to be made available to the particular role. You will also need to specify the type of ‘authority’ to be given to the particular user. For each of the three options namely Datasets, Reports and Authorities user can select multiple options from the scroll down menu provided against each field. A user can choose multiple options either by moving them one-by-one[2].

In order for particular users to be able to enter data, you must add them to both a dataset as well as an organizational unit level. You can also select multiple datasets individually by pressing the Ctrl key on the keyboard and clicking on individual datasets [2].

Finally when you have entered the required fields click on “Save” which is located on the lower part of the displayed screen. The desired user role and related authorization will be saved to the database, and can then be assigned to a particular user [2].

Now we can move back to creating a new user. To add a new user, just follow these steps[2]:

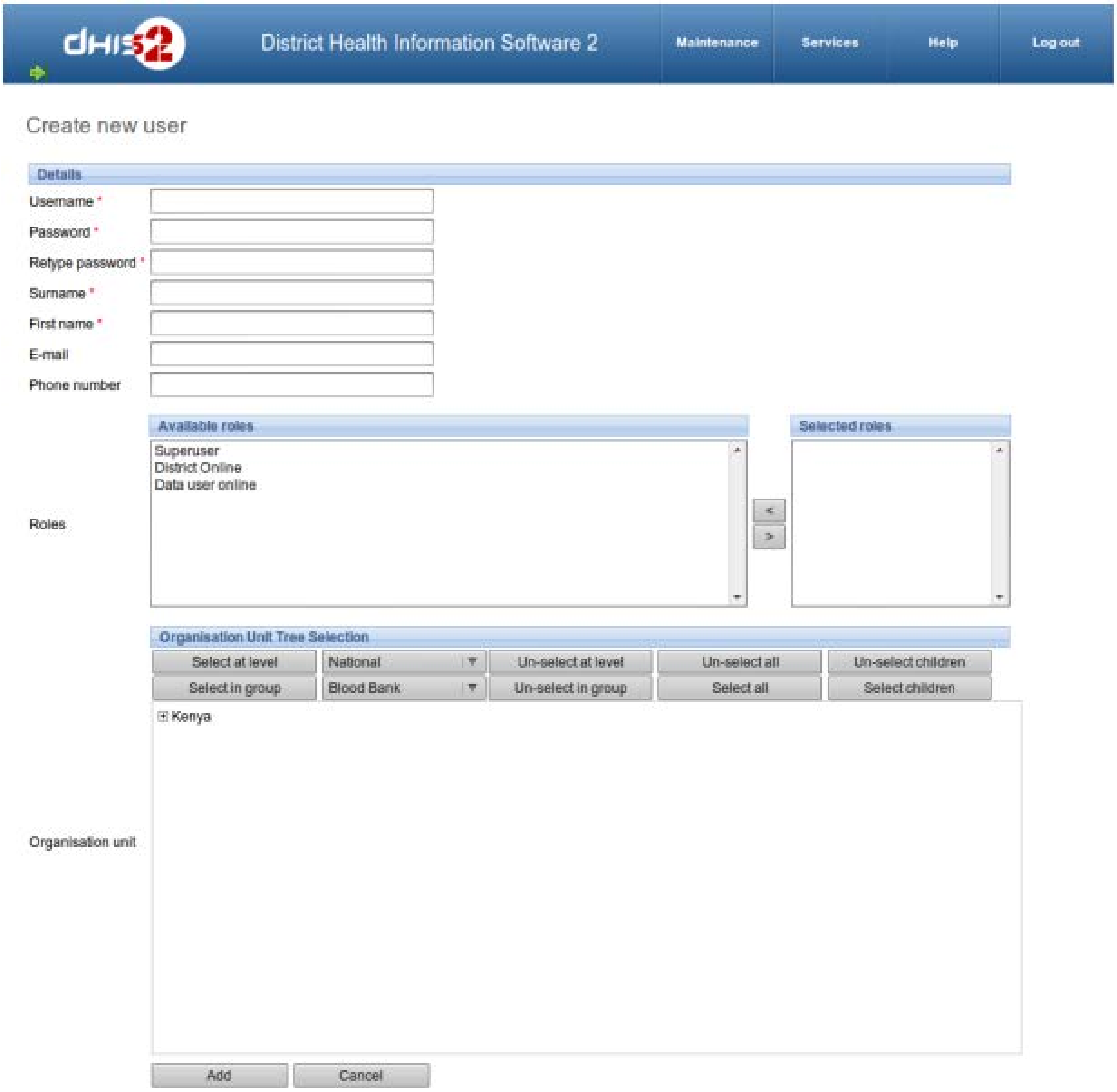
- Click on the Add New button.

- Enter New User details like Username, Password, Confirm password, Surname, First name and Email in new user’s option tabs.

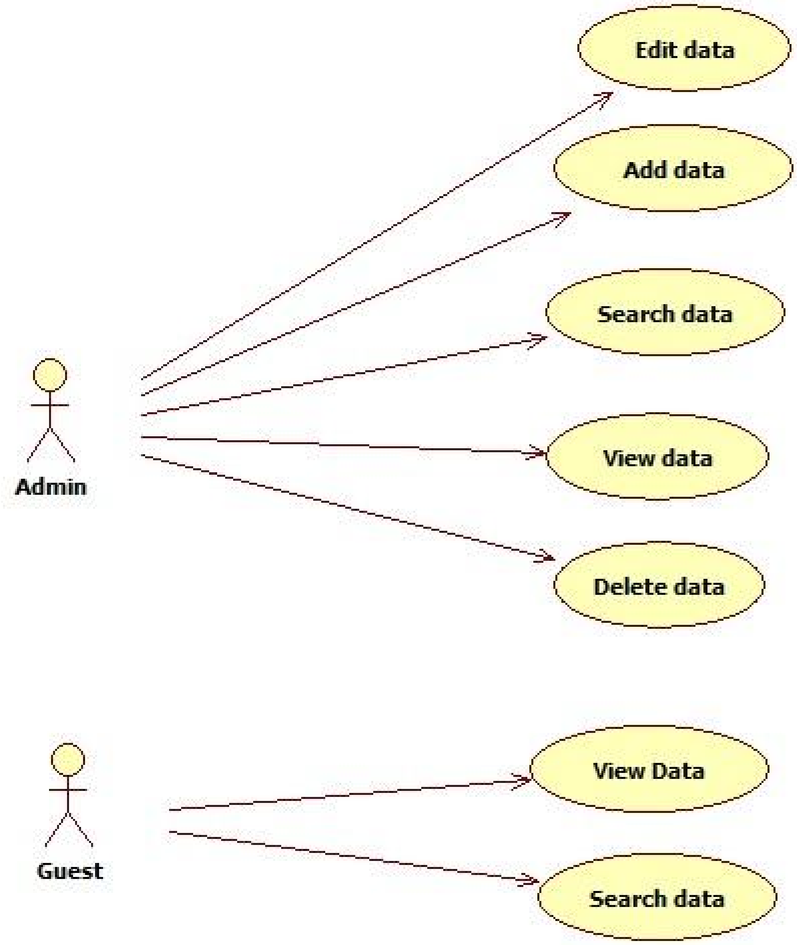
- Click on Add button for confirmation of new user details and follow the user error while creation of new user.

- The recently created new user can be seen in main’ User management Screen

- You can edit (like password, surname….etc.) and delete the details of new/old users by selecting corresponding User’s Edit and Delete Buttons.



Assign a role for new user, one user can have multiple roles. In this project, we did not implement the organization unit. The use case diagram should be as below:



6. Compiling & Running

Re-import all dependencies in Intellij IDE

Run the cmd: *mvn clean install* at

- dhis2/dhis-2
- dhis2/dhis-2/dhis-api
- dhis2/dhis-2/dhis-services
- dhis2/dhis-2/dhis-web

Run the cmd: *mvn jetty:run-war* in *dhis2/dhis-2/dhi2-web/dhis-web-portal*

7. Free & open-source development tools

Using open-source software for development requires a comprehensive research before adapting, in order to ensure stability and maturity. There is a large amount of open-source software in the market such as Netbeans IDE, Eclipse Software Development Kit, Intellij IDE, etc. In the bound of this project, we use Intellij IDE as recommended in class. After a while of using this IDE (community version), we have figured out some advantages:

> The Auto-completion function is actually very good, not only autocompleting based on Java library, but it can also predict and suggest the solution in most cases.

> It supports a large number of different frameworks (Spring framework for example) with multiple languages.

> It is also well-known for keeping up to date with the Java platform

8. Reference

1. DHIS2 home page, 2013, “DHIS2 Development”, viewed Dec 31, 2013, <http://www.dhis2.org/development>

2. DHIS2 User Manual, 2014, version 2.14, viewed Jan 3, 2014, download at <http://www.dhis2.org/documentation>

3. Launchpad, 2014, homepage, Canonical, Ltd (copyright 2004 - 2014)

4. Oracle, 2014, homepage, Oracle Company, <www.oracle.com>

5. Bazaar, 2014, homepage, Canonical, Ltd, <wiki.bazaar.canonical.com>

6. Maven, 2014, homepage, The Apache Software Foundation, <maven.apache.org>