



­

Assignment Document:

Core Spring

Version: Core Spring Practice Case study 5/ASSIGNMENT/1.0

Date: 11-05-2015

Cognizant

500 Glen Pointe Center West

Teaneck, NJ 07666

Ph: 201-801-0233  
[www.cognizant.com](http://www.cognizant.com)

Contents

[CSR Fund Collection 2](#_Toc475630916)

[1. Infrastructure Section 2](#_Toc475630917)

[i. Hardware, Software Specification 2](#_Toc475630918)

[2. Case Study Assignments 4](#_Toc475630919)

[3. Pre- requisite steps 4](#_Toc475630920)

[4. Database Setup 5](#_Toc475630925)

[5. Donate Amount for CSR: 6](#_Toc475630928)

[6. Technical Specifications: 6](#_Toc475630930)

[7. Business Rules& Validations: 7](#_Toc475630931)

[8. Sequence Diagram 10](#_Toc475630932)

[9. View Donated Details: 10](#_Toc475630933)

[10. Technical Specifications: 11](#_Toc475630934)

[11. Business Rules & Validations: 12](#_Toc475630935)

[12. Sequence Diagram 12](#_Toc475630936)

[13. Solution: 13](#_Toc475630937)

[14. Evaluation Rubrics 13](#_Toc475630938)

[15. Summary of this Case Study: 13](#_Toc475630939)

# CSR Fund Collection

1. Infrastructure Section

i. Hardware, Software Specification

This section captures the hardware and software specifications for the effective delivery of the course.

1. **Hardware Specification**

|  |  |
| --- | --- |
| **Server Specification** | Windows 7 - 32bit  4GB RAM |
| **Desktop / Client Specification** | Admin Rights |

**b. Software** **Specification**

| **#** | **Name of the software (s) to be installed** | **Version** | **License available in RAMS?**  **(Yes/No)** | **License available in RAMS for onsite?**  **(Yes/No)** | **Description of any patch to be installed** | **Can be used through Tool Wire / SoftGrid?** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | JDK | 7.0 | Yes | Yes |  | Yes |
| 2 | SDE 7.0 | 7.0 | Yes | Yes |  | Yes |
| 3 | Tomcat 6.0/7.0 | 6.0/7.0 | Yes | Yes |  | Yes |
| 4 | MySQL Workbench | 6.0.8 | Yes | Yes |  | Yes |
| 5 | Netbeans 6.9.1- Complete Installation (Including Derby database and Tomcat Server) | 6.9.1 | Yes | Yes |  | Yes |
| 6 | Eclipse | 3.6 | Yes | Yes |  | Yes |

**Instruction for installing the software in the “Tool Wire/SoftGrid” environment *(if the response is ‘Yes’ in the last column in the above table)*:**

| **#** | **Name of the software** | **Instruction** |
| --- | --- | --- |
| 1 | Spring Jars 3.0 | <http://www.java2s.com/Code/Jar/s/Downloadspringweb310RELEASEjar.htm> |
| 2 | Mysql-connector-java-5.1.12-bin | <http://www.java2s.com/Code/Jar/m/Downloadmysqlconnectorjava5123binjar.htm> |
| 3 | spring-hibernate3-2.0.8 | <http://www.java2s.com/Code/Jar/s/Downloadspringhibernate3208jar.htm> |
| 4 | Hibernate 3.5 jars | <http://www.java2s.com/Code/Jar/h/Downloadhibernate353jar.htm> |

Installation Details:

Refer the below link for JDK and JRE installation :

<http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase6-419409.html>

Refer the below link for SE download and installation :

<https://gto.cognizant.com/javacoe/SitePages/SDEDownload.aspx>

**Test Cases to check the installed software’s:**



1. Case Study Assignments

Estimated Completion Time: xx Minutes

Objective:

Cognizant Facilities Team wants to automate the process of donating amount for CSR activities for its employees who are interested and wants to donate funds for Corporate Social Responsibilities activities. Employees can donate through the system and the CSR admin can generate a report of the fund collections for different units/activities.

They would like to develop the below components.

**Service1: Donate for CSR**

**Service2: View Donated Amount Details**

**Control flow is Tester -> Manager -> Façade->BO->Dao.**

1. Pre- requisite steps

* Please make sure that JAVA\_HOME is set your JDK installation folder, otherwise please set as following: *C:\Program Files\Java\jdk1.7.0\_51*
* Unzip the below content and place it in your hard disk



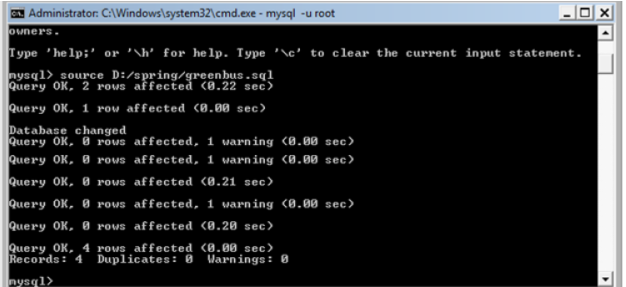
* Open Eclipse IDE or SDE and import the below project using “Import -> Existing Projects into workspace” option
* This acts as your code base to begin the case study development

1. Database Setup

* Unzip the below content and place it in your hard disk



* Open MySQL Command Line and run the script using the command:
  + Example: source D:/Javaca/library.sql (after unzipping the sql file please run this command by giving the location where the .sql file is placed)
  + Queries should execute without any errors



Note: if you want to drop and re-create any tables or re-insert data for testing purposes, please run the same .sql script again as given in step 2 to reset your database setup.

1. Donate Amount for CSR:

The service is used to automate the process of donating amount for CSR activities. This service is going to become a part of different other modules in the system so have been asked to develop the Service as an independent module so that it can be plugged into other modules easily.

This service accepts the employee details and amounts to be donated for CSR activities. If the below business rules are satisfied, then persist the data and will return an Integer value (registrationId). If any of the business rules is not satisfied, throw an appropriate user defined exception as given in the below table

1. Technical Specifications:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Method Name** | **Input** | **Output** | **Exception** |
| Tester | **displayEnglish()**  Read the key value pair from the messages\_en\_US.properties |  | void |  |
| Tester | **displayFrench()**  Read the key value pair from the messages\_fr.properties |  | void |  |
| CSRManager | **donateAmount()** | **RegistrationVO registrationVO** | ReturnsInteger value  registrationId | InvalidEmpNameException, InvalidEmailException,  InvalidPhoneNoException,  InvalidUnitCodeException,  These Exceptions to be caught and thrown back to Tester class |
| CSRFacade | **donateAmount()** | **RegistrationVO registrationVO** | ReturnsInteger value  registrationId | InvalidEmpNameException, InvalidEmailException,  InvalidPhoneNoException,  InvalidUnitCodeException,  These Exceptions to be caught and thrown back to CSRManager class |
| CSRBO | **donateAmount()**  \*\* check for the following business rules given below | **RegistrationVO registrationVO** | ReturnsInteger value  registrationId | InvalidEmpNoException InvalidNameException,  InvalidLocationIdException  InvalidEmergencyNoException,  InvalidEmployeeException These Exceptions to be caught and thrown back to CSRFacade class |
| CSRDao interface | **getUnitDetails()** | Integer unitCode |  |  |
| **donateAmount()** | **RegistrationVO registrationVO** |  |  |
| CSRDaoImpl | **getUnitDetails()** | Integer unitCode | return the corresponding Unit object for the unitCode if valid, null if invalid |  |
| **donateAmount()** | **RegistrationVO registrationVO** | You need to update the **totalAmount** of CSRUnit table for the unitCode which you are donating  Persist the data in the Registration Table and returns**Integer** value  registrationId |  |
| csr\_config.xml | Contains all the xml configurations related to Service1 |  |  |  |

1. Business Rules& Validations:

In the **CSRBO class** check for the following business rules given below

|  |  |  |
| --- | --- | --- |
| **Rule**  **No.** | **Rule Description** | **User Define Exception to be thrown** |
| 1. | **empName** entered should contain only alphabets and spaces and should not contain any digits and special characters | If empName is invalid  **InvalidNameException** |
| 2. | **empEmail** entered should be valid | If empName is invalid  **InvalidNameException** |
| 3. | **unitCode** entered should be valid [ getUnitDetails(Integer unitCode) method of CSRDao. Method returns CSRUnit object if locationId is valid , null if valid] | If **locationId** is invalid  **InvalidLocationIdException** |
| 4. | **phoneNo** entered should contain only 10-digit numbers | If **phoneNo** is invalid  **InvalidPhoneNoException** |

* If any of the business rules is not satisfied, throw an appropriate user defined exception as given in the above table

**Note: The RegistrationVO, CSRUnit, Registration class will be provided with the required getter and setter methods which should not be modified. No changes should be done to the skeletons provided (Especially the name and method signature)**

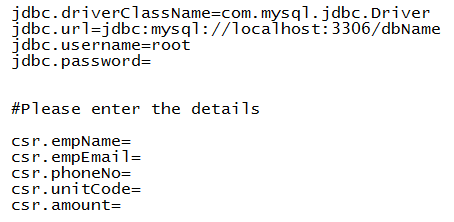
**Limitations and Constraints**

1. All the above mentioned java classes to be declared as beans in spring configuration file **csr\_config.xml.**
2. **CSRFacade** should be integrated **CSRManager** using Setter Based Dependency Injection.
3. **CSRBO** should be integrated with **CSRFacade** using Constructor based dependency injection
4. **CSRDao** should be integrated with **CSRBO** using Setter Based Dependency
5. Use ONLY **csr\_config.xml** for all configurations related to Service1
6. Create two properties files messages\_en\_US.properties, messages\_fr.properties for English and French locale
7. Read the **input data** and **Data Source** details from the **constants.properties**
8. Enable transactions by configuring transaction manager

**<SQL Connection details will be provided to the associate as below>**

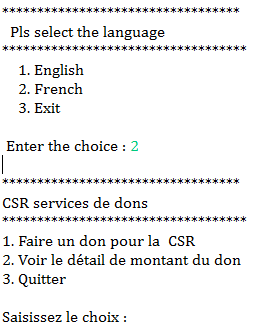
**Create a constant.property file with the following details. Please set the value for DBName, password and also set the values to donate Fund for CSR activities.**

Please enter the below details

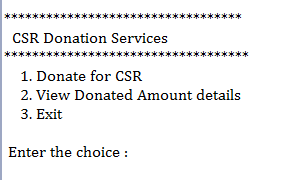


**Output**

**French locale**

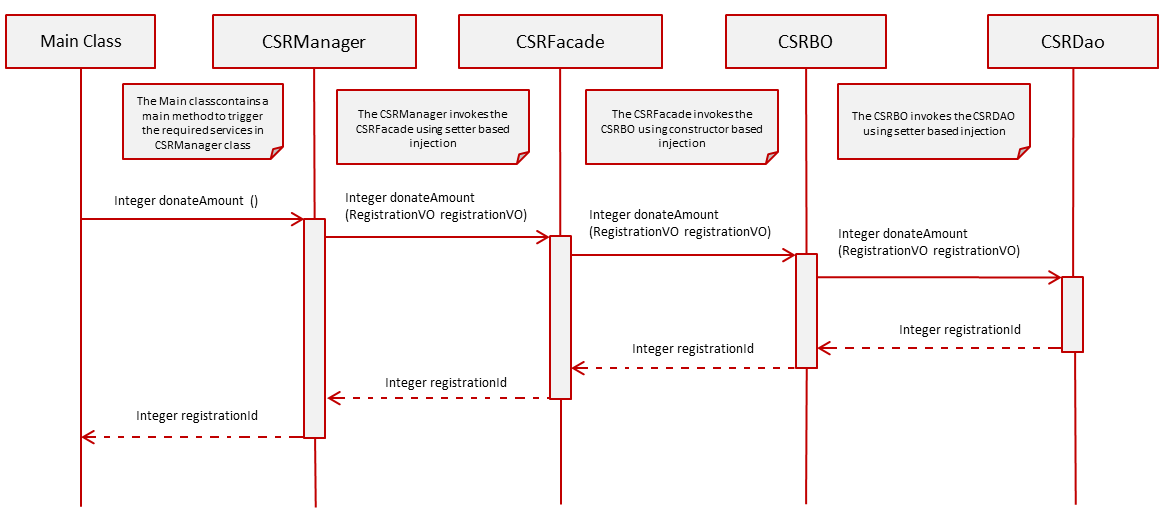


**English locale**



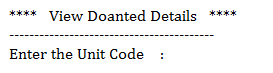


1. Sequence Diagram

****

1. View Donated Details:

* When the user selects Option 2**,** viewDonatedDetails () **method of Tester** is invoked.
* The CSR admin will input the **unitCode.**
* The **View Donated Details** menu in the viewDonatedDetails ()method of Tester class looks like below



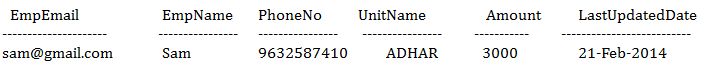
Control flow is **Tester -> Manager -> Façade->BO->Dao.**

* In the **Tester** Class,
* Invoke viewDonatedDetails (Integer unitCode)from Tester to **CSRManager** following the Control flow mentioned above. This method returns a list of type **DetailsVO** objects.
* Write code to display the output in the below format by using the returned values if the business rules are satisfied, otherwise display the appropriate exception message

**[Hint:** While displaying, to display lastUpdatedDate in proper Date Format (dd-MMM-yyyy),

Use **DateUtility.getStringFromDate (Date date)** to convert lastUpdatedDate from Date to String.**]**

**OUTPUT**



1. Technical Specifications:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Method Name** | **Input** | **Output** | **Exception** |
| CSRManager | **viewDonatedDetails()** | Integer unitCode | List<DetailsVO> detailVoList | NoRecordException,  This Exception to be caught and thrown back to Tester class |
| CSRFacade | **viewDonatedDetails()** | Integer unitCode | List<DetailsVO> detailVoList | NoRecordException,  This Exception to be caught and thrown back to CSRManager class |
| CSRBO | **viewDonatedDetails()** | Integer unitCode | List<DetailsVO> detailVoList | NoRecordException,  This exception to be caught and thrown back to CSRFacade class |
| CSRDao interface | **viewDonatedDetails()** | Integer unitCode | List<DetailsVO> detailVoList |  |
| CSRDaoImpl | **viewDonatedDetails()** | Integer unitCode | List<DetailsVO> detailVoList |  |
| csr\_config.xml | Contains all the xml configurations related to Service1 |  |  |  |

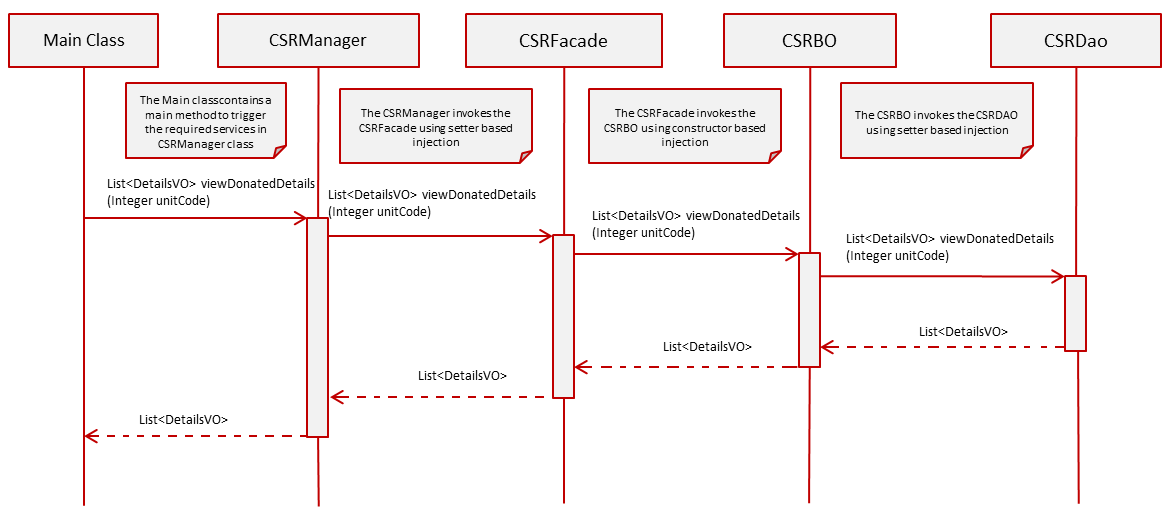
1. Business Rules & Validations:

In the **CSRBO class** check for the following business rules given below

|  |  |  |
| --- | --- | --- |
| **Rule**  **No.** | **Rule Description** | **User Define Exception to be thrown** |
| 1. | List<DetailsVO> returned by the method **viewDonatedDetails** () of CSRDao class should not be empty. | If list is empty  **NoRecordException** |

* If any of the business rules is not satisfied, throw an appropriate user defined exception as given in the above table
* **Note: The DetailsVO, class will be provided with the required getter and setter methods which should not be modified. No changes should be done to the skeletons provided (Especially the name and method signature)**

1. Sequence Diagram

****

1. Solution:



1. Evaluation Rubrics

|  |  |
| --- | --- |
| Parameters | Weightage |
| 1. Completeness |  |
| 1. Accuracy |  |
| 1. Clarity of understanding |  |
| 1. Presentation |  |
| Total |  |

1. Summary of this Case Study:

You have just learnt:

* Spring Frame Work and IOC container
* Spring Dependency Injection
* Spring JDBC framework
* Spring Transaction management
* ResourceBundleMessageSource