**Code Refactoring Report for**

**Java Refactoring Test Project**

**Created By:**

Sajjad Ashraf

ssajjadashraf@yahoo.com

**Date:**

09 Oct, 2018

**Table of Contents**

[1. Introduction 3](#_Toc526770775)

[2. Steps for Improvements: 3](#_Toc526770776)

[ Java docs, Code formatting : 3](#_Toc526770777)

[ Handle Concurrent Requests 3](#_Toc526770778)

[ Removing element while Iterating through Array List 3](#_Toc526770779)

[ Update, Delete and Find user operation operate on the base of user’s email 3](#_Toc526770780)

[ Design pattern Singleton 4](#_Toc526770781)

[ User porpr HTTP method UserResource 4](#_Toc526770782)

[ Introduce logging. 4](#_Toc526770783)

[I. 4](#_Toc526770784)

[7. References 4](#_Toc526770785)

# 

1. Introduction

The task is to refactor the ‘java-test’ project in a way that it would break any existing functionality, test cases and improve the code readability, extensibility and maintainability.

2. Steps for Improvements:

* Java docs, Code formatting :

I have added Java docs in every class it will help to understand the code easily. I have also formatted the code (code indentation) in every class of the project so that it looks consistent and easy to read. I also remove unused variable and imports.

* Handle Concurrent Requests

To facilitate concurrent requests, I have introduced ‘**synchronized**’ code block where we are adding, updating and deleting a user in ‘UserDao’ class.

|  |
| --- |
|  |

We can also handle concurrent request by using following code Collections.synchronizedList(new ArrayList<>());

It is preferable to synchronize only part of a method which can cause a concurrency issue.

* Removing element while Iterating through Array List

‘deleteUser’ method of ‘UserDao’ has been updated to avoid java.util.ConcurrentModificationException .

* Update, Delete and Find user operation operate on the base of user’s email

As given email is unique that why these operations must work on the basis of email, previously they were working on name. An error will be thrown if user does not provide email address or at least one role while creating the user.

* Singleton Design pattern for ‘UserDao’

The way we are using ‘UserDao’ we have to apply a creational design pattern for this class, in current situation Singleton Design pattern best suits for ‘UserDao’.

* Use proper HTTP method in ‘UserResource’

I have updated HTTP Method type to its correct type according to CRUD operations.

addUser to POST.

updateUser to PUT.

deleteUser to DELETE.

getUsers and find findUser to GET.

* Introduce logging.



7. References

* User Interface Prototyping [http://agilemodeling.com/artifacts/uiPrototype.htm]
* Online Tool for prototype designing. [https://proto.io/]
* Lecture from slides and book.