# SOFTWARE DEVELOPMENT OFFICER – PROJECT MRA

By Panjie Mziya

# Table of Contents

# UNDERSTANDING REQUIREMENTS

With respect to the MRA's need to increase the number of registered Taxpayers, development of a system to help was proposed.

The proposed system will need to fulfil Four (4) fundamental functional requirements with another one (1) non-functional yet fundamental requirement.

The system will have to consume an exposed webservice.

## FUNDAMENTAL FUNCTIONAL REQUIREMENTS.

1. **Registering Taxpayers**. The Goal of the whole system primarily rests on fulfilling this requirement. The System must allow users to register Taxpayers.
2. **Viewing registered Taxpayers**. The System must allow users to view the registered Taxpayers.
3. **Editing Taxpayers**. Users must be allowed to edit Taxpayers details after having registered these taxpayers.
4. **Deleting Taxpayers**. Users must be allowed to delete registered Taxpayers.

## FUNDAMENTAL NON-FUNCTIONAL REQUIREMENTS.

1. **Security**. The System should have Authentication and Authorisation mechanisms in place to boost the security of the system. An example of this could be the need to login to the system before being able to perform any operations on the system. The system must also allow the user to logout of the system whenever necessary.
2. Albeit non-functional, Data validation is also needed fundamental requirement.

## USE CASE DIAGRAM

Use case Diagrams help system developers understand proposed systems alongside respective user interactions with the proposed system.

Below is a use case diagram depicting the interactions of the User; in this case an MRA Officer; with the system.

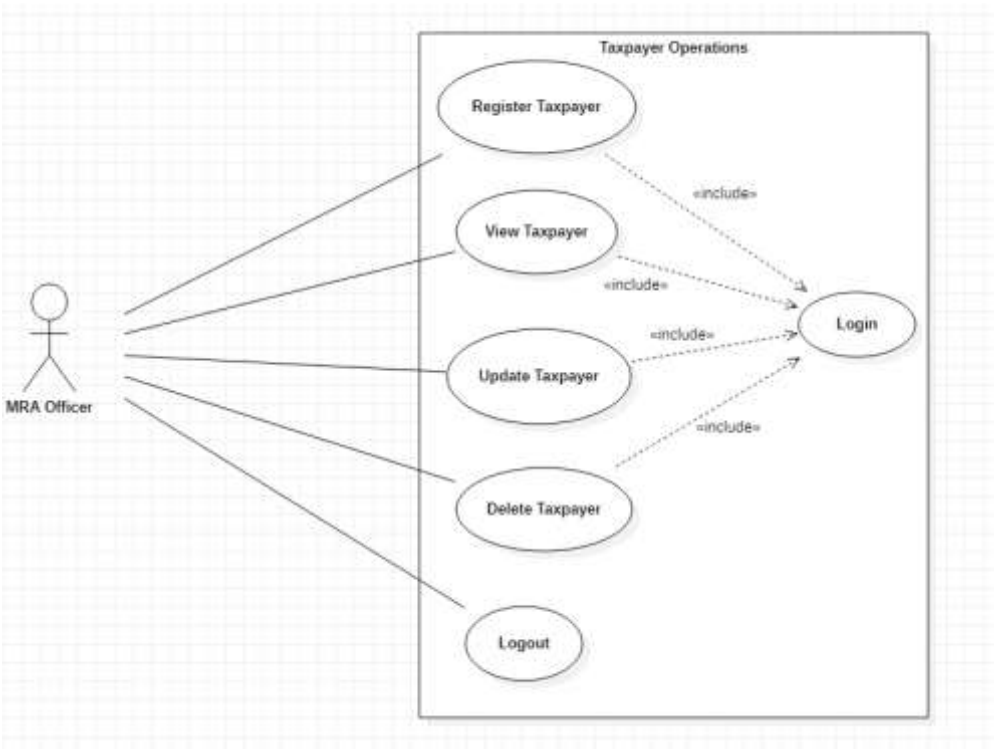


Figure 1.

Below is a brief description of the above use case.

| Actor | User case | Description |
|---|---|---|
| MRA Officer | Login | MRA Officer has to login to use the system. Every other operation apart from the logout operation will have to include an authorization mechanism. |
| | Register Taxpayer | MRA Officers can register new Taxpayers. The MRA officer will have to login to system first. |
| | View Taxpayer | MRA Officers can view all registered Taxpayers. The MRA officer will have to login to system first. |
| | Update Taxpayer | MRA Officers can update (edit Taxpayer details) for registered Taxpayers. The MRA officer will have to login to system first. |
| | Delete Taxpayer | MRA Officers can delete registered Taxpayers. The MRA officer will have to login to system first. |
| | Logout | MRA Officers can logout of the system. |

# TECHNOLOGY USED

### FRON END

JAVA is the preferred programming language for the front end.

Below are reasons for using:

- Java is platform independent meaning Java systems can run on any operating system that has the Java Virtual Machine (JVM) installed.
- Java is more secure and more robust meaning Java systems are more reliable.

Eclipse is the IDE to be used.

### BACK END

MRA exposed a webservice that has to be consumed. The Front end will have to interact with the webservice to be fully functional.

# THE DEVELOPED SYSTEM

The system was developed with all the requirements fulfilled. JAVA programming language was used to develop the new system and successfully consumed the exposed webservice.

### SYSTEM ARCHITECTURE

Being a system that has been built around CRUD alone with no extra operations required, the classes for the system were spilt only in 3.

The GUI class that consists of Variables, Objects and Methods that define the Graphical User Interface. This forms the fundamental building block for the front end.

The Main class consisting of only the main method where point of execution begins.

Lastly the Webservices class. This class contains all variables, objects and method used to consume the webservice. It forms the link between the backend (Webservice) and the front end (the GUI Class).

There were some External Java Libraries that were used to successfully link the front end with the webservice. Please refer to the table below for details relating to the used external libraries.

| External Libraries | Reasons for usage |
|---|---|
| Jersey lib | This was used to provide a link with the webservice and facilitated understanding between the two entities. POST and GET with authorization headers was made possible with this lib. |
| Json Simple -1.1 and Json lib – 2.4 | These libraries provided methods that made is possible for easy understanding of responses from the webservices. Responses were being received in JSON. These also prepared data that was sent to the webservice in Json as is required when sending data to the Webservice. |

Functions to validate data before being sent to the webservices were implemented. These validations were implemented after carefully analysing the sample responses. For example, an email address should at least have the "@" special character. If the user attempts to input something that does not qualify to be an email address, entry shall be denied until this requirement is fulfilled. Testing of these and the fundamental requirements of the system can be rereferred to in the PowerPoint presentation of the system

## SYSTEM INTERFACE

Please refer to the PowerPoint presentation of the system for this section.

## PROBLEMS EXPERIENCED AND SOLUTIONS

1. Inaccurate links to both the Login and Logout resources

    - This was resolved by adding the missing path between the challenge and auth paths. The following are the correct and used links to the login and logout resources respectively:

        ✓ https://www.mra.mw/sandbox/ /programming/webservice/challenge/auth/login

        ✓ https://www.mra.mw/sandbox/ /programming/webservice/challenge/auth/logout

2. Flawed response from the Delete taxpayer resource. Both returned true when attempting to delete registered and unregistered Taxpayers. Ideally when attempting to delete a taxpayer that does not exist the response must have been different from the one returned when attempting to delete a Taxpayer that does exist in the system.

    - The Solution implemented was to have TPINs retrieved from the webservice, then compared to the TPIN entered to delete a taxpayer, if the entered TPIN matches any of the TPINs retrieved from the webservices, the deletion process will then begin.

Although in sample response it was show as something that is possible, please note that when updating Taxpayers, TPINs cannot be changed. This is one detail that cannot be changed. When attempting to edit Taxpayers a current TPIN has to be provided to specify which Taxpayer details have to be updated and data on all fields has to be entered even those that do not necessarily need updating. This has been implanted like this due to the use of the POST and not PUT methods when updating registered users. Please also note that when registering a new taxpayer all details will have to be entered as a way of ensuring quality data entry.

# CONCLUSION

In conclusion the system has been developed successfully. The Java Programming language has been used to successfully develop the system.

System requirements include at least an Intel Celeron or higher processor with at least 1.5GHZ processing speed. At least 1 GB RAM and 1 GB free storage space, a keyboard and mouse.

Either a Windows, Linux or MAC OS system with the JAVA Virtual Machine installed.

An internet connect to enable connection to the webservice.