SYSTEMS DEVELOPMENT OFFICER  
**PROJECT**

**DUE DATE: Friday, 26 NOV 2021 midnight**

Submitted To: MRA

For: Junior Programmer

**Submitted by: Kings Lincoln Mkumbwa**

**(Candidate Developer)**

**Languages used: PHP & HTML**

**Description of Document**

This document outlines steps taken to develop the solution. The target audience is “Junior programmer” who wants step by step process to create a  
similar project.

Contents

[1. Problem 4](#_Toc88832605)

[2. Key Objectives 4](#_Toc88832606)

[3. App specifications 4](#_Toc88832607)

[4. Summary of procedure: 5](#_Toc88832608)

[5. Requirements: 6](#_Toc88832609)

[6. Procedure: 6](#_Toc88832610)

[7. App installations: 6](#_Toc88832611)

[8. Building the skeleton: 7](#_Toc88832612)

[9. Html Form validations 7](#_Toc88832613)

[10. Implementing the php code. 7](#_Toc88832614)

[11. THE PHP logic: 9](#_Toc88832615)

[12. Testing 10](#_Toc88832616)

[13. Buffing up of the UI. 10](#_Toc88832617)

[14. Uploading to GitHub 11](#_Toc88832618)

[15. Conclusion 11](#_Toc88832619)

## Problem

MRA intends to widen its tax net by registering new tax payers. One way to do this is to send officers to recruit new taxpayers who are operating businesses yet they have not registered for taxes.  
The author has been identified as a developer to develop an application (that will consume the provided webservice) to be used by the field officers to register tax payers.

## Key Objectives

1. Building a front end with validations to ensure quality data submission.
2. Consuming the webservice and handle all response types

## App specifications

* The application should allow field officers to login using a username and a password
* The application should allow field officers to create a new taxpayer record with the attributes specified in the registration web service.
* **NOTE**: All validations are up to the developer of the app to Analyze the fields exposed by the webservice and make sure your application includes all necessary validations to ensure quality data is submitted*.*
* The application shall allow view of all taxpayers registered by the logged in user.
* Field officers should be able to create, edit and delete taxpayers.
* NOTE: Web methods for all the specified functions have been provided by the webservice. The

## Summary of procedure:

1. Install web server (xampp), php, java.
2. Plan layout and flow of pages to achieve goals of the app. Thus, Login page, viewing all records Page, editing records page, Deleting records page in HTML.
3. Create the html pages with .php file extensions and upload to localhost.
4. Study endpoint data formats and requirements
5. Formulate logic for retrieving and send data in a compatible manner.
6. Add php code for authentication, data retrieval and data posting to webservice.
7. Use php to pass variables for the headers (token and username) and session management.
8. Used curl functions to carry out the POST and GET methods when getting and retrieving data from database using the webservice.
9. Test the app,
10. clean the code,
11. Upload to git hub.
12. Submit Documentation and Deployment procedures.

## Requirements:

* 1. Knowledge of PhP, html, JSON
  2. Php web server with curl and js installed/ enabled.
     + Used xampp web server application
  3. The endpoint points from which the web app was going to consume
     + Provided by MRA
  4. Post man, an app for testing http CRUD (read, create, delete, update) methods.
     + Downloaded from the internet
  5. Text editor,
     + Downloaded and user Notepad Plus and Sublime
  6. Web browser: Firefox or chrome,
     + All found in many laptops/ computers as default web browsers.
     + Can be downloaded for free.
  7. Windows/Linux machine,
     + Used a dell laptop i7 with min of 8gb ram for optimal performance
     + used Windows 10 O.S environment
  8. Internet connectivity, since the endpoints are available over the internet on MRA servers.
     + Used Airtel and TNM internet by means of hotspot from mobile smart phones
  9. Json To Php Array Converter.
     + This is a freely available online tool that was used in setting up variables and array indexes and associations when passing variables, (encoding php arrays and decoding Json data Responses) into and from json format and php array formats.

## Procedure:

## App installations:

1. First installed XAMPP, and enabled the php server

* I also checked that curl has been enabled by using command prompt code *curl –version*

1. Installed notepad plus + text editor and postman , sublime
2. Installed chrome and Firefox
3. Installed JS node, java

## Building the skeleton:

1. After installed necessary applications above I started building the skeleton of the app in basic html

Whereby I produced the:

* Login page
* Home page
* New/ register taxpayer page
* Edit tax payer page,
* View all taxpayer’s page.

## Html Form validations

* Specified data types that can be enter in text field
* Other fields have been disabled for editing when updating records. E.g.,
  + TPIN (the edit method in the webservice won’t allow it),
  + username (is current user, value passed in the background),
  + id (is auto assigned and hence no need to input or edit. may arise on conflict)
* Deleting a record asks for confirmation before deleting
* Email validation makes sure at least there is @ and .com etc. characters that form an email
* Date input set to only accept current day as maximum allowable date, used php to echo current date as max date in input attributes.

## Implementing the php code.

After building the basic html pages I started now adding the required php code for:

* + **Login page**: to verify session and post login credential to endpoint for authentication and authorization.
  + Added proper validation of data and made it not to allow user to login with wrong credentials as authenticated by web service.
  + Created session variables such as http headers (Session usernames and APikey ) that would be passed together with the http requests that the app will be making to achieve its goals.
  + Used curl functions in the php for POST and Get methods used at the provided end points.
  + **Home page**: Added php to display session user as the current user and some html welcoming message, a few html descriptions of the app and what is does and how user can navigate. All has links to other pages.
  + **New/ register taxpayer page**: this page does as it says.it has html forms and html validations to ensure that clean meaningful data is entered. For instance, limiting the type of data that can be entered on a field i.e., limiting to numbers only on a phone number input field. Etc. codded the submit actions such that when submitted, values are passed to a php array which is passed to the curl post method that sends the request to the webservice. Success messages were added and link to view the records in the db.
  + **View all taxpayer’s page**: this is the trickiest page to make because all records will be retrieved and shown in table format and from here it should be possible to delete a record and choose any record to edit: this was implemented by:

**Html**: making an html table with defined headers ie TPIN, EMAIL, PHONE etc, but no records. Made sure to skip columns like user name because they have no importance on this front end.

**Php**: curl was used to make a get all request to the web service, the curl response was the passed on to an array after being decoded.

This particular array then was **while looped** with each value echoing html code that places it in rows and columns ending up with a table.

* + **Edit tax payer page:**  this was implemented by:

**html**:a link to a form that like the new tax payer for with passed values for the row number. On this form php is used to echo values on the row number into the editable form fields where one can see and choose where to edit.

Necessary form validation has to be added to make sure users put it acceptable data.eg not to put date that is in the future.

* + **logout page**: redirects to the login page, terminating the sessions and also sending a post method request using the username for the headers

## THE PHP logic:

All these pages were linked up in proper logic

* + ***When user logs*** in the email and user name entered are collected in php array that is then encoded to json format and sent to the \*…/auth/login\* method for authentication.
  + Using curl, the response from the auth method is decoded and saved to an array from which it is determined whether authentication was successful or not.
  + If successful the user details ie. Username and token are retrieved and passed on to session variables $\_SESSION [username] and $\_SESSION [apikey] so that they can be used in the methods (post, get) when sending requests to the web service.
  + ***When viewing records***: I used the session variables in the headers of the get http request using curl. The cur response was decoded and passed on to an array that was looped through into an html table.
  + ***When deleting records****:* in the html table I set variables which are the indexes of records of taxpayers in the array such that a delete link is placed on far right of every record this delete button passes the index of the record to the delete method by retrieving the necessary record details (the TPIN of that particular taxpayer) from decoded curl response from **get all method** in view taxpayers page and passing them to the curl function.
  + ***When updating/ editing taxpayers records****:* similarly, to deleting the edit button is also placed at the end of each record in the table in VEW taxpayer’s page.
    - This button Is set to pass index and associated value of the record (from the decoded curl response on get all method in view page) to the edit .php file which retrieves these values and echo is used to set them as values in the html form.
    - ***Validation:*** only taxpayer details that are eligible for editing are displayed in the form e.g trade name and physical location etc. but not the TPIN, user name etc.
    - the other taxpayer details are just passed along to the post method.
* ***When logging out***. The session user variable is passed to the delete method using curl Post, and then the session variables are unset, session destroyed and token destroyed on the webservice. The page out redirects to the login page.

## Testing

* + Testing was being carried out throughout the development process, tools such as the postman were being used to diagnose errors, or post methods and get methods that were going to be implemented in the app using curl.
  + Web server XAMPp was being used to simulate / operate the app in a real online situation.
  + Response times were also being monitored
  + The app was then tested for validation whereby I currently input wrong data to see if it accepts (which is wrong and needs to be addressed) or not(which is good means form validation works )

## Buffing up of the UI.

* + Bootstrap CSS was downloaded and was used in the html pages to make them presentable.
  + Prototype of designs and screen provided in the attached PowerPoint presentation

## Uploading to GitHub

Created git hub account and

Created a repository Mrataxes

Made the repository public

Uploaded files for the app in the repository.

## Conclusion

The developed solution would enable officers to manage their taxpayer’s data that is, register new taxpayers, update existing records, and also delete existing records. The app would have to be deployed on a web server and so as to be accessible to authorized/ authenticated users through a link over the internet using a web browser.