**TAXPAYER DEVELOPMENT MANUAL (STEP BY STEP PROCESS)**

To develop the application to register new tax payers, I have used PHP programming language together with HTML, CSS bootstrap and JavaScript to create my view pages.

In my project folder

1. The **main** **folder** is taxpayer and it contains an assets, exec and layout folders that I created. The taxpayer folder also holds view pages of the system and these include:
2. **Index.php**

* This is the login display page.
* Contains Html that displays the login form with email and password textbox and sign button.
* It also contains php script to submit email and password value to login.php found in the exec folder. This happens after a click on a sign button on the form.
* Php script on this file is also used to perform other tasks for example include or require pages, echo error messages e.t.c.

1. **Taxpayerview.php**

* This is view of all taxpayers and a submenu of the system.
* Contains Html that displays a taxpayer table and a submenu of a logout button, Add Taxpayer button, Edit taxpayer button and a Delete taxpayer button.
* It also has php script that retrieves taxpayer data which is displayed in the taxpayer table.
* Php script is also used to perform other tasks for example include or require pages, echo error messages e.t.c.

1. **Addtaxpayer.php**

* This Contains Html that displays a form with textboxes to enter taxpayer’s details. This also has a save, cancel and logout button.
* Contains Php script used to perform other tasks for example include or require pages, echo error messages e.t.c.

1. **Edittaxpayer.php**

* This Contains Html that displays a form with textboxes to enter taxpayer’s details. This also has a save, cancel and logout button.
* Contains Php script used to perform other tasks for example include or require pages, echo error messages e.t.c.

1. **Auth.php**

* Contains php script to check user login session if it is active else it redirects to index.php.
* It also holds a function to check passed values to another page are not manipulated on the far end side.

1. **Layout folder** contains a header and footer that are included in the view pages e.g. addtaxpayer.php, taxpayerview.php and edittaxpayer.php. The header is included on top of each page while footer is included on bottom.
2. **Header.php**

* This file contains a head tag of the pages that includes it. In the head tag, it holds all Meta information, title and link tag of the pages.
* I have assigned a variable to a session value of the logged user and echo it on the navigation panel using php.
* The file also contains the beginning of the body tag of html and a nav tag to provide navigation links for the pages.

1. **Footer.php**

* This file contains script tag used to embed a client-side script (JavaScript).

1. **Assets folder** contains all the bootstrap CSS and JavaScript. The css files are used for styling of the pages and the javascript files are used to give interactive purposes to the pages. The following are contents of assets folder:
2. **css folder**

* bootstrap.min.css
* jquery.dataTables.min

1. **js folder**

* jquery.min.js
* jquery.dataTables.min.js
* bootstrap.min.js

1. **Exec folder** holds all the php execution script pages that interact with API, assign values to variables e.t.c. These includes:
2. **Apiconnect.php**

* This page holds a function that connects with the MRA API using php client Uniform Resource Location connector (php -cURL).
* The function passes three parameters namely $ttype(transaction type) which is either POST or GET, $field which is a list of field name and its value, $urlink which is the sub domains of link to the API. E.g. **$apiscript($ttype, $fields, $urllink);**
* This page is almost required in all pages to have connection to the API.

1. **Apiconfig.php**

* This file is called in the apiconnect.php file by requiring it. It provides the apiconnect function with the api connect configurations. E.g. password, apikey e.t.c.
* It consist of an array.

1. **Login.php**

* This page calls the function that connects to the API and compares passed user credentials entered on the index page with those retrieved from the API to determine a login authorization. (Email and password credentials).
* Submit the transaction type (POST/GET), taxpayer’s email and api url to the api connect function.
* Submit the transaction type (POST/GET), taxpayer’s user credentials (Email and Password) and api url to the api connect function.
* It then redirects to another page (taxpayerview.php) when credentials match or displays an error message when credentials do not match.

1. **Logout.php**

* This page checks the login user session against the logged user at the API side and unset and destroy the session if they match. It then redirects to index.php page.
* Submit the transaction type (POST/GET), taxpayer’s email and api url to the api connect function.

1. **Savetaxpayer.php**

* Assigns posted taxpayer’s details to variables and pass them to the API call function with other parameters.
* Submit the transaction type (POST/GET), taxpayer’s details and api url to the api connect function to save taxpayer’s details.
* And finally assigns messages to session and redirect the page to addtaxpayer.php.

1. **Edittaxpayer.php**

* Assigns posted taxpayer’s details to variables and pass them to the API call function with other parameters.
* Submit the transaction type (POST/GET), taxpayer’s details and api url to the api connect function to edit taxpayer’s details.
* And finally assigns messages to session and redirect the page to edittaxpayer.php.

1. **Deletetaxpayer.php**

* This page checks the login user session against the logged user at the API side using Email and deletes the record if they match. It then redirects to taxpayerview.php page.
* Submit the transaction type (POST/GET), taxpayer’s email and api url to the api connect function.