**MRA TAXPAYER SOLUTION**

**TECHNOLOGIES USED**

For the development of the solution, I used the following tools:

* XAMPP server
* Visual studio code
* PHP, bootstrap, HTML

NOTE: use the readme.txt file included for procedure on how to setup this system on your machine.

For the validation it is possible to use JavaScript but I did not implement it that way in this solution, also it would have been efficient creating the solution around a framework like Laravel (PHP framework) which has a lot of security features baked in but I used vanilla PHP as requested.

**PROCEDURE**

1. **AUTHENTICATION**

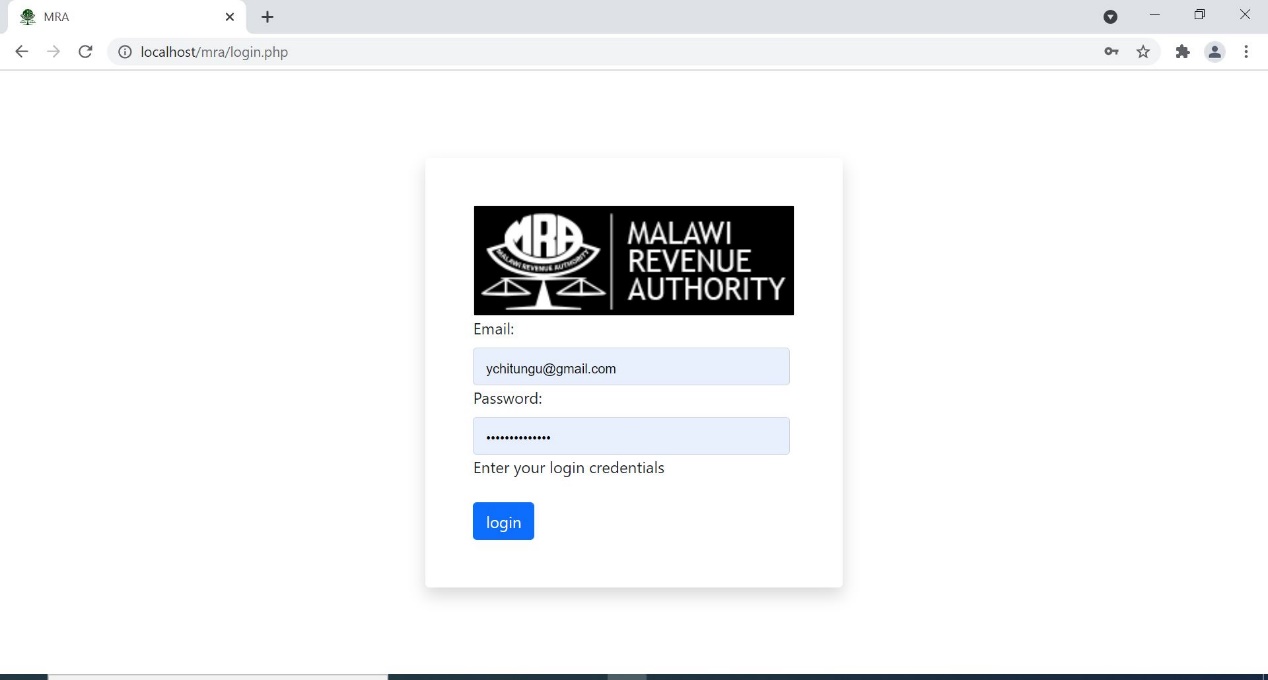


Figure 1

1. **LOGIN**

To get into the site on the development server go to <http://localhost/mra/> . Here the application attempts to go direct to the homepage, but before it does that it checks if a user is logged in by monitoring the session(“session status()”). If there is no active session it means no user is currently logged in and it redirects to the login page. On the login page the user will provide their assigned credentials to get into the system, both the email and password input boxes have been set to be mandatory as in the page will not attempt to authenticate if the fields are blank. The email field has been set to check if it has been provided with a valid email format (“string here” @” string here”. “string here”). Once the proper input formats are provided and the login button is clicked then the form data is posted to the “auth.php” file for authentication, in the “auth.php” file the “email” , “password” and the “login” variables are checked if they are set and if not then the page redirects pack to the login page. If the fields are set then they go through another layer of validation on the php side before attempting to submit the data to the webservice provided. The data is put in an array and encoded into json format then provided to the webservice using the client URL library(curl) where the method is set to “POST”, the header is provided with content type, candidateid and apikey. Then the request is executed and the result stored back into the $result variable where it is then decoded from the json format and the required values are extracted into dedicated variables, once that is done then the system checks if the returned values are a match and if so, sets the session values for the token, email, first name and last name then after redirects to the home page. If the credentials provided are not correct the system displays a message telling them they provided the wrong credentials on the login page.

1. **LOGOUT**

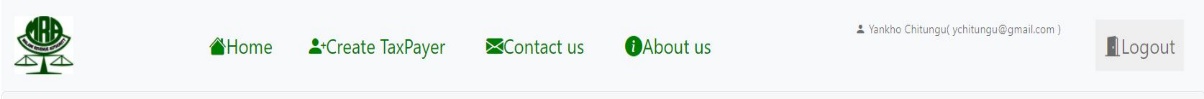


Figure 2

To log out, there is a button on the menu which when clicked calls the “auth.php” where it checks if “email” and “logout” variable are set, if they are set then the email is fed to the “logout” endpoint of service and if the result code returned is “1” then it unsets the session and redirect user to login page.

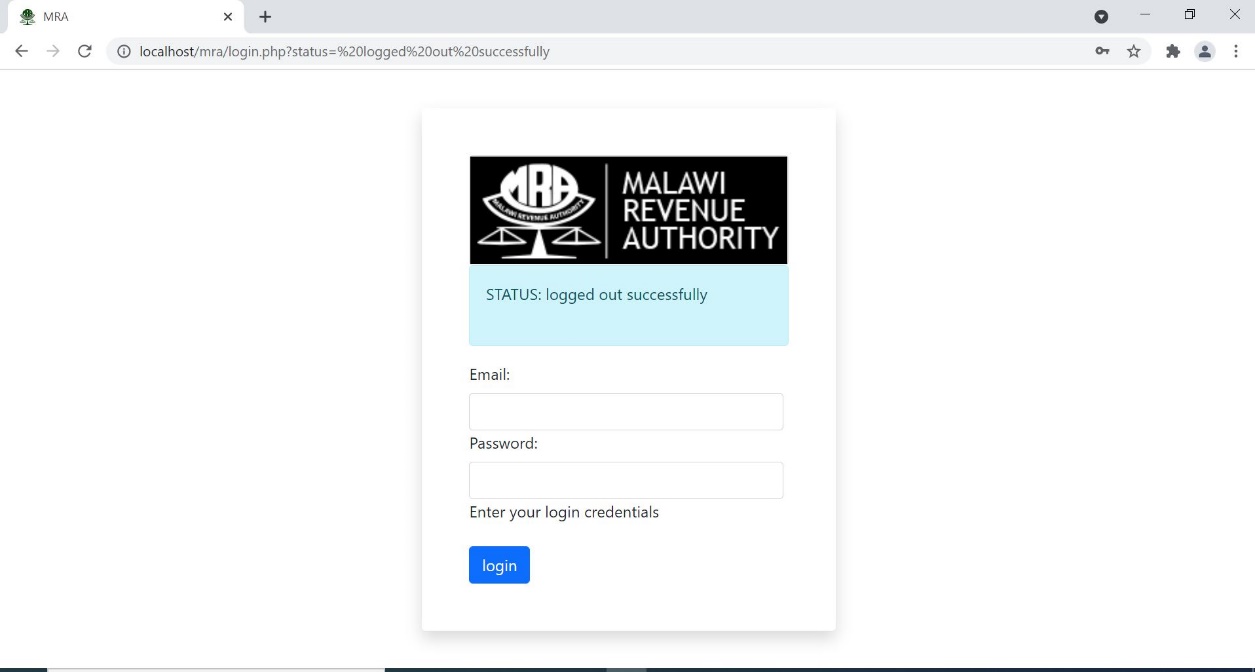


Figure 3

1. **HOME PAGE**

The home page and all other pages have a menu which has been implemented in a file “menu.php” and is included on all other pages for easy navigation through the system. The home page has been developed to consume the “getAll” end point when the page loads. The file “taxpayer.php” is included where a function get\_all\_taxpayers() is called and it consumes the “getAll “end point and returns the data which the logged in user has in the system, if the user has not uploaded any taxpayers in the system if shows a message telling them that they have no entries in the system.

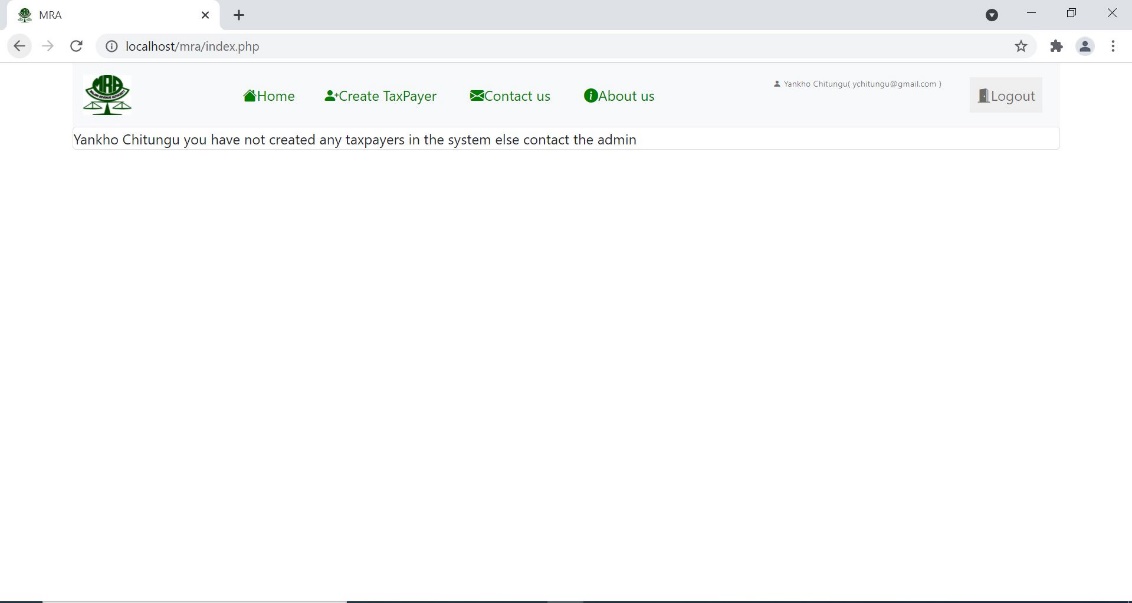


Figure 4

The retrieved data for current user is populated in a table showing TPIN, business certificate number, trading name, date of registration, phone number, email and location. On the home page, buttons have been added to allow the user to either delete or edit any of the entries shown.

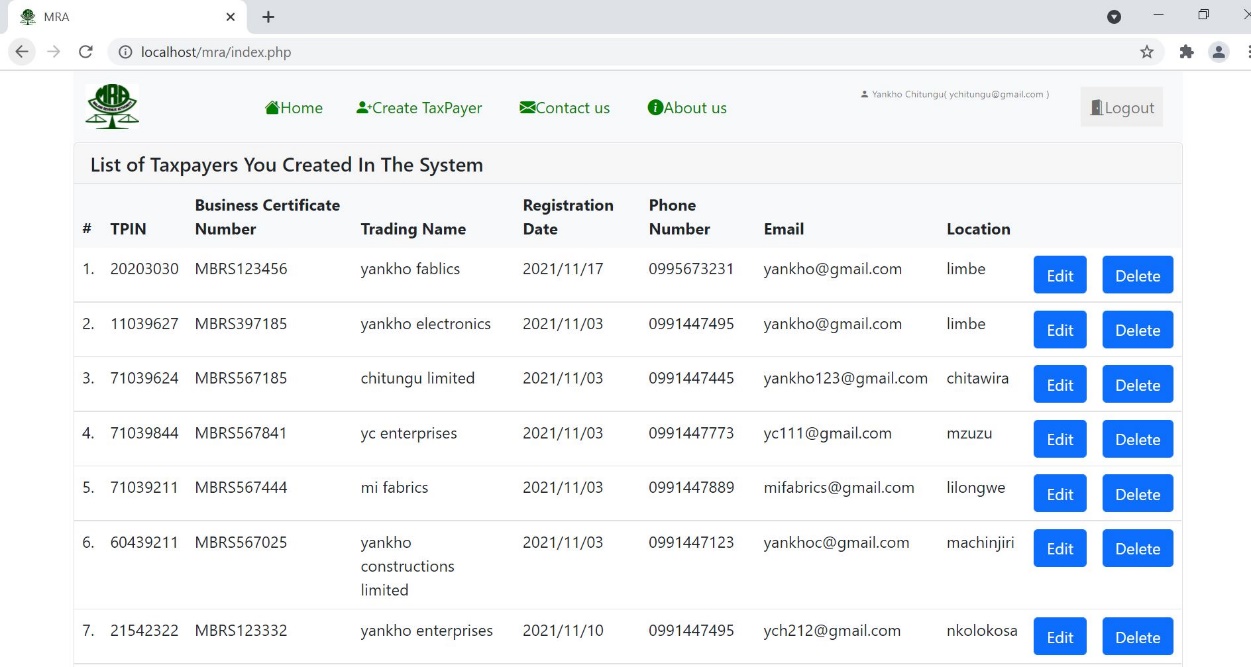


Figure 5

1. **CREATE TAXPAYER**

Create taxpayer menu item has been provided on the menu (navigation bar) and when clicked it takes user to a page with a form for creating new taxpayer. The page has been provided with all the necessary validation for the entries so that they capture the right data, where:

* The email field only takes emails format input.
* Location, trading name, business certificate number take generic text input as they can contain both numbers and words.
* Phone number and TPIN takes numbers only.
* Date of registration field allows user to select date.

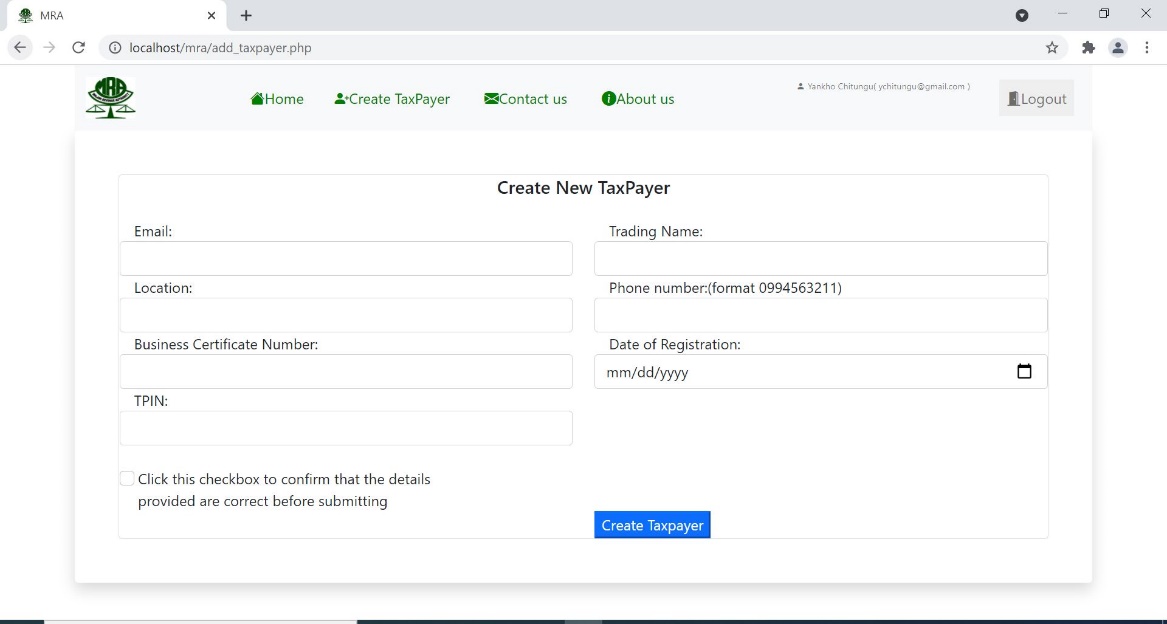


Figure 6

Before submitting the form there is a confirmation box which the user must tick first else the form won’t submit and when submitting the form also contains the email of the logged in user, once the form is submitted the “taxpayer.php” script is called again where it checks if the “confirm\_add\_taxpayer” value is set and if so, it checks also if all the fields are set and after that validates all the fields.

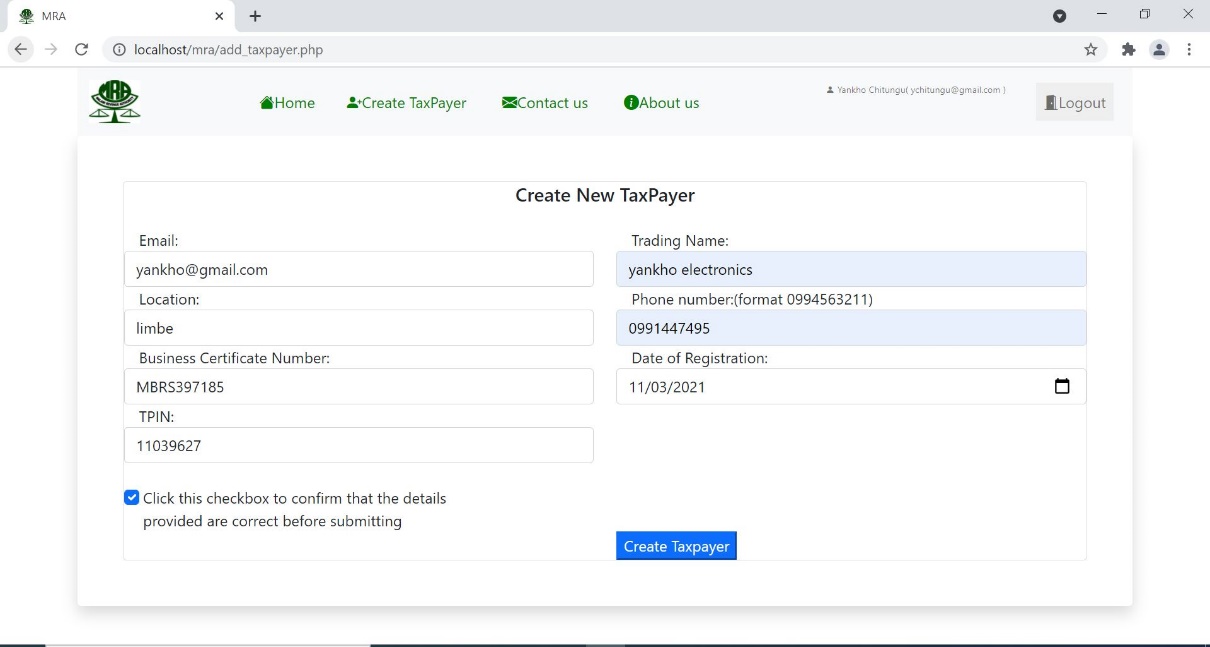


Figure 7

Once the fields are validated then the data is put in an array and together with the URL and $action variable are then fed to the api\_caller() function, in the function the data is json encoded and sent to the “add” endpoint of the webservice and from there an http reply is provided and depending on the response called it shows a status message on the same add taxpayer page.

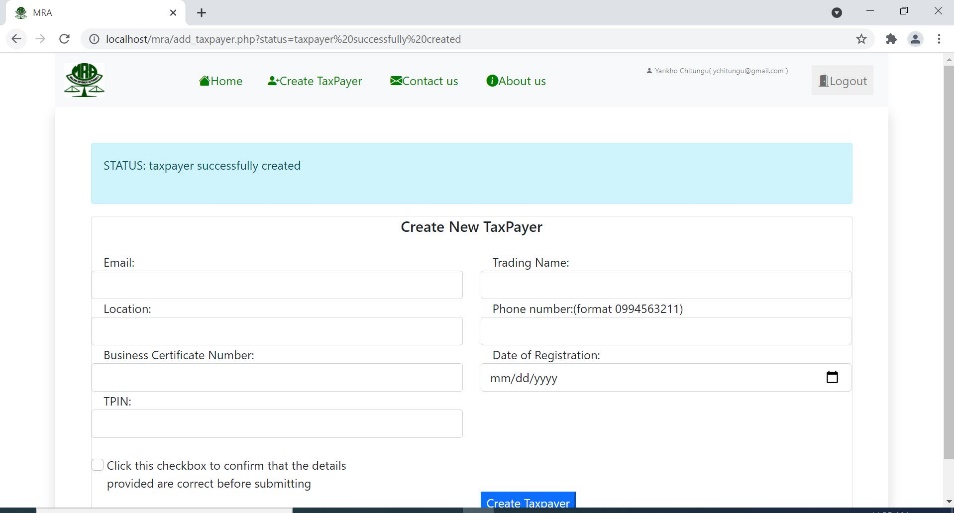


Figure 8

1. **EDIT TAXPAYER**

To edit a taxpayer the edit button on the homepage is used, this calls the “edit\_taxpayer.php” and gets all the information of the selected entry to that page. In this page the TPIN input box has been disabled from editing as this value never has to change, else the rest of the page functionality is similar to the one for adding new taxpayers.

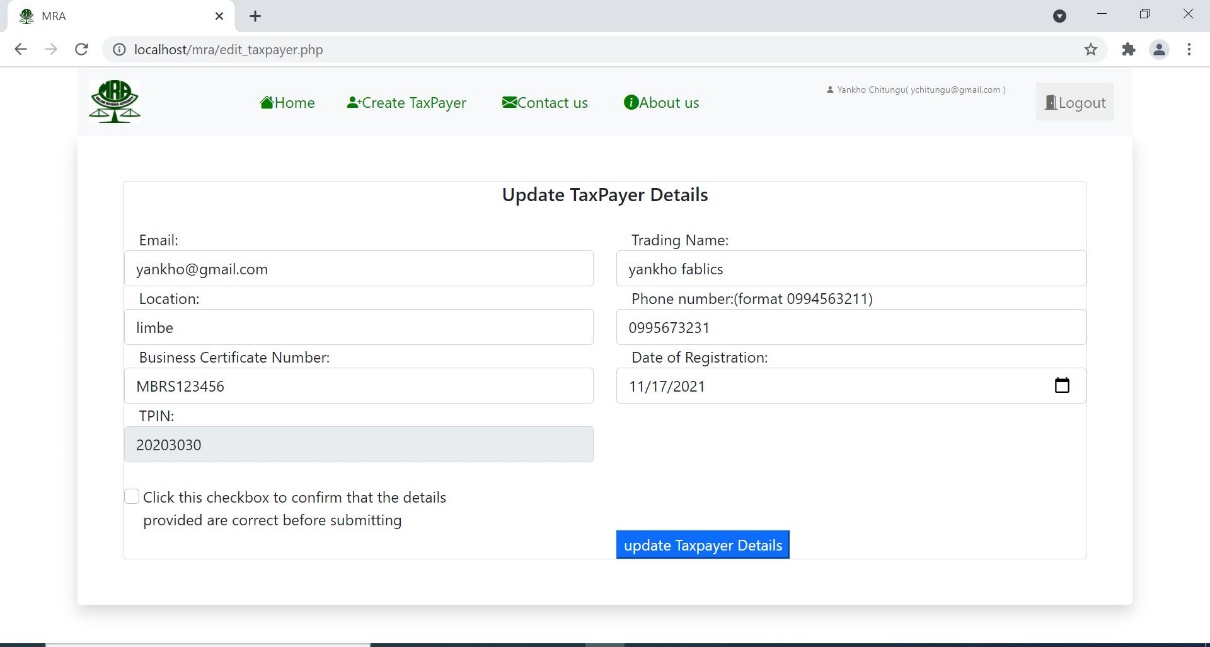


Figure 9

On submit it also goes through similar process at the “taxpayer.php” file as adding new taxpayer but this one checks for the confirm\_edit\_taxpayer value which is set when user confirms on the checkbox and also the $action variable is set to “edit\_taxpayer”. Once the function is executed the system redirects back to the homepage where the status messages is displayed according to the http response code.

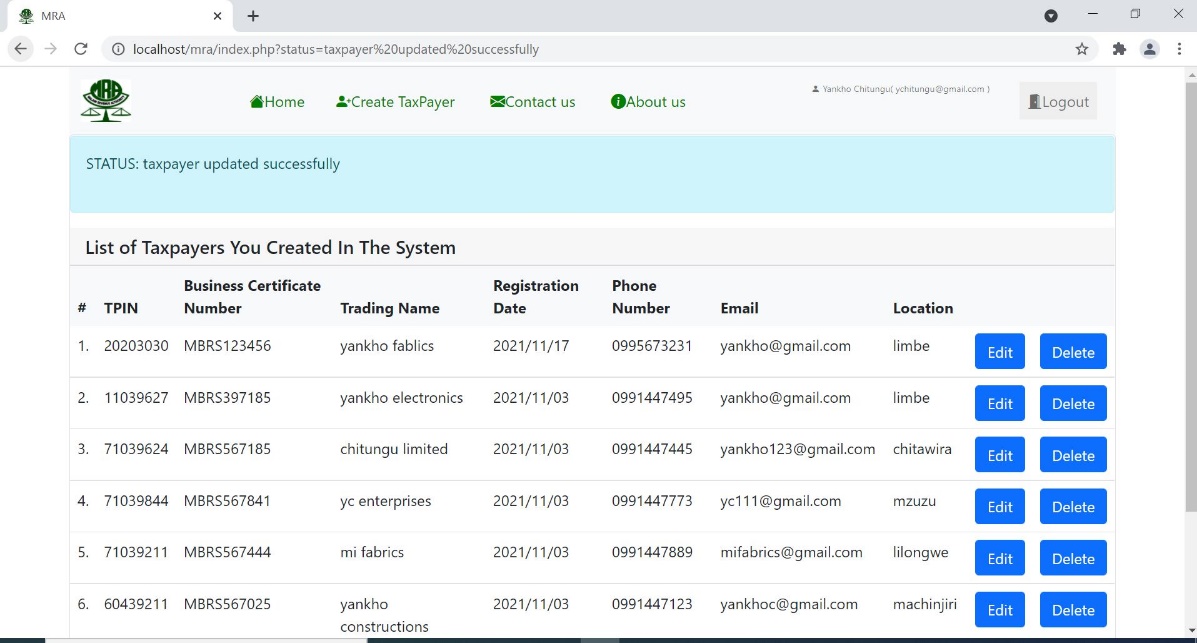


Figure 10

1. **DELETE TAXPAYER**

To delete a taxpayer from the system then a delete button is used on the home page. When the button is clicked a confirmation, box pops up which shows the TPIN of the taxpayer the user wants to delete and so that they confirm it’s the right one.

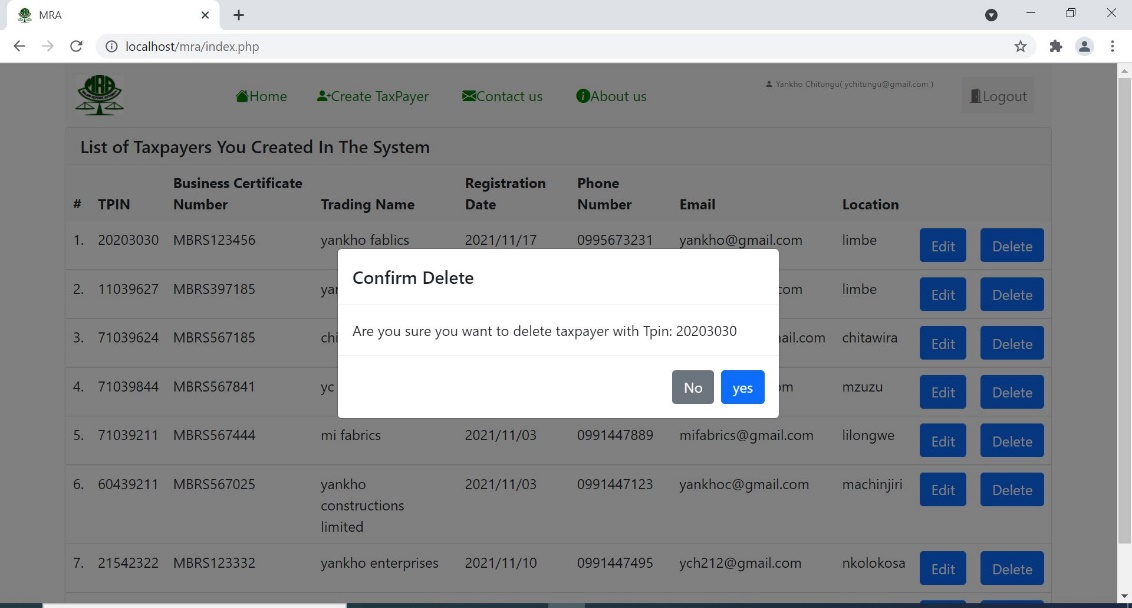


Figure 11

Then when the user clicks yes, the TPIN is submitted to the “taxpayer.php” where it checks if the “delete\_confirm” and “delete\_confirm\_tpin” values are set and executes the code inside the if condition. Here the TPIN provided is validated then put in an array, after that the $action variable is set to “delete\_taxpayer” and the api\_caller() function is called with the necessary data. After execution of the function then the system shows a status message depending on the response http code on the home page.

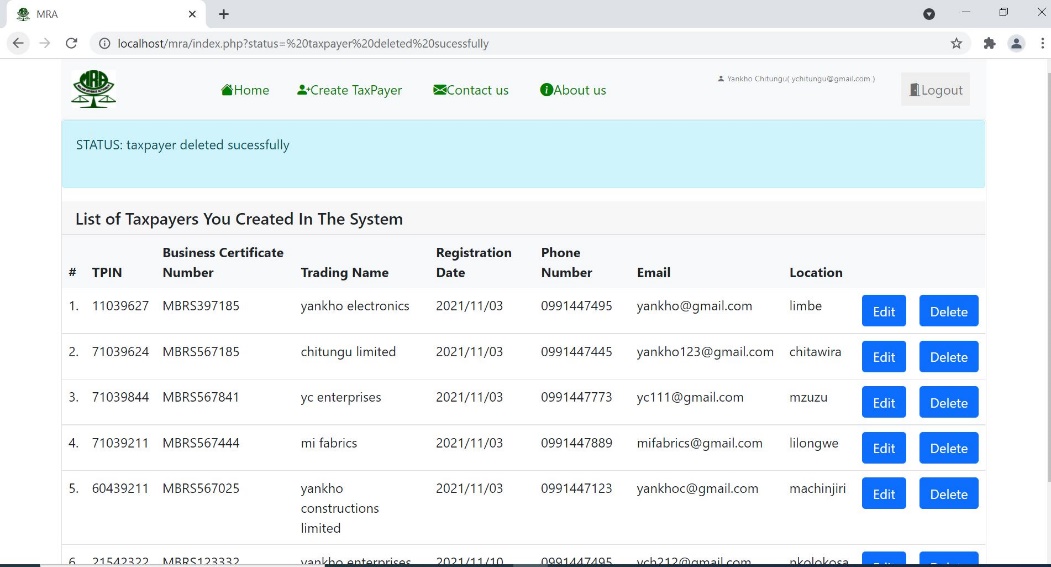


Figure 12

**END OF DOCUMENT**