Steps taken to develop the consumer. The document is for someone who has a little bit knowledge of Laravel

1. Firstly, I had to identify what type of API
2. Choose REST or SOAP mode for consuming the API
3. Then I chose Laravel for solution development as it already comes with a good authentication library which is secure rather that configuring from scratch
4. Downloaded Laragon here <https://laragon.org/download/> and visit here <https://laragon.org/docs/index.html> on how to use laragon which is local development server for rapid and secure development
5. Then I had to install Laravel from the Laragon by following instructions on <https://www.wikihow.com/Install-Laravel-Using-Laragon>. Just copy the link and paste it on browser url and then press enter
6. Then I created Laravel project named “webapp” and then I opened it in my text editor “Sublime”
7. Since Laravel comes with several folders of which some we tackle and some we don’t
8. Then I went to **C:\laragon\www\webapp\app\Http\Controllers** of my computer where the project files are located and create a controller called ***Usercontroller***
9. In this controller I created all my functions. These include

a. Authenticate which handles login by sending request to <https://www.mra.mw/sandbox/programming/challenge/webservice/auth/login> and wait for the response.

**b.** gettaxpayer which is responsible for displaying all the tax payers by sending and getting response from <https://www.mra.mw/sandbox/programming/challenge/webservice/Taxpayers/getAll>

**c.** putdata for adding a new tax payer by interacting with <https://www.mra.mw/sandbox/programming/challenge/webservice/Taxpayers/add>

**d.** getedit is responsible for populating the edit for after clicking the edit button. It interacts with to get a single taxpayer using the tpin <https://www.mra.mw/sandbox/programming/challenge/webservice/Taxpayers/getAll>

**e.** edit updates the changes made in the edit form by sending request to <https://www.mra.mw/sandbox/programming/challenge/webservice/Taxpayers/edit>

**f.** delete is for deleting the selected tax payer as long as delete button is clicked by sending request to <https://www.mra.mw/sandbox/programming/challenge/webservice/Taxpayers/delete>

1. Since we are interacting with the api, every function has to send request and then expect response. The request is sent in the json format and also response can either come in json format or different format. The response can be success or failure depending the status code displayed.
2. All the above function call different routes. Which are responsible for taking you where you want depending on where you have clicked on the app.
3. I then created a file in **C:\laragon\www\webapp\routes** called **web.php** which will contain all the routes responsible for calling, in this route different views.
4. Route::get('/', default route. It loads login page if the user is not yet logged in and loads the view that show all the taxpayers if the user if logged in and it is called by authenticate function in UserController
5. Route::post('login', loads a login view called welcome.blade.php called by authenticate function in UserController
6. Route::get('logout' for logout and called by logout function in UserController
7. Route::get('getTaxpayers', it is called by gettaxpayer function in UserController for displaying the showall.blade.php view
8. Route::get('edit/{tpin}', edit route called by edit function in UserController which handles loading and populating of edit.blade.php view
9. Route::post("putdata”, called by putdata function in UserController for actual saving into api database
10. Route::post('update', called by edit function for actual updating of the tax payer into api database
11. Route::get('delete/{tpin}', called by delete and is for actual delete of the tax payer from the api database
12. Route::get("add", responsible for displaying the add.blade.php view
13. Then I went on to create all the views which I might need in **C:\laragon\www\webapp\resources\views** as follows
14. edit.blade.php for editing tax paper
15. add.blade.php for adding tax payer
16. welcome.blade.php for login
17. showall.blade.php for diplaying the taxpyers
18. when all is done, I go t cmd and navigate to project folder and running **php artisan serve** which starts the laravel server and displays the url on cmd on which you can access the app.