**FUNCTION GENERATION TOOL GUIDELINE**

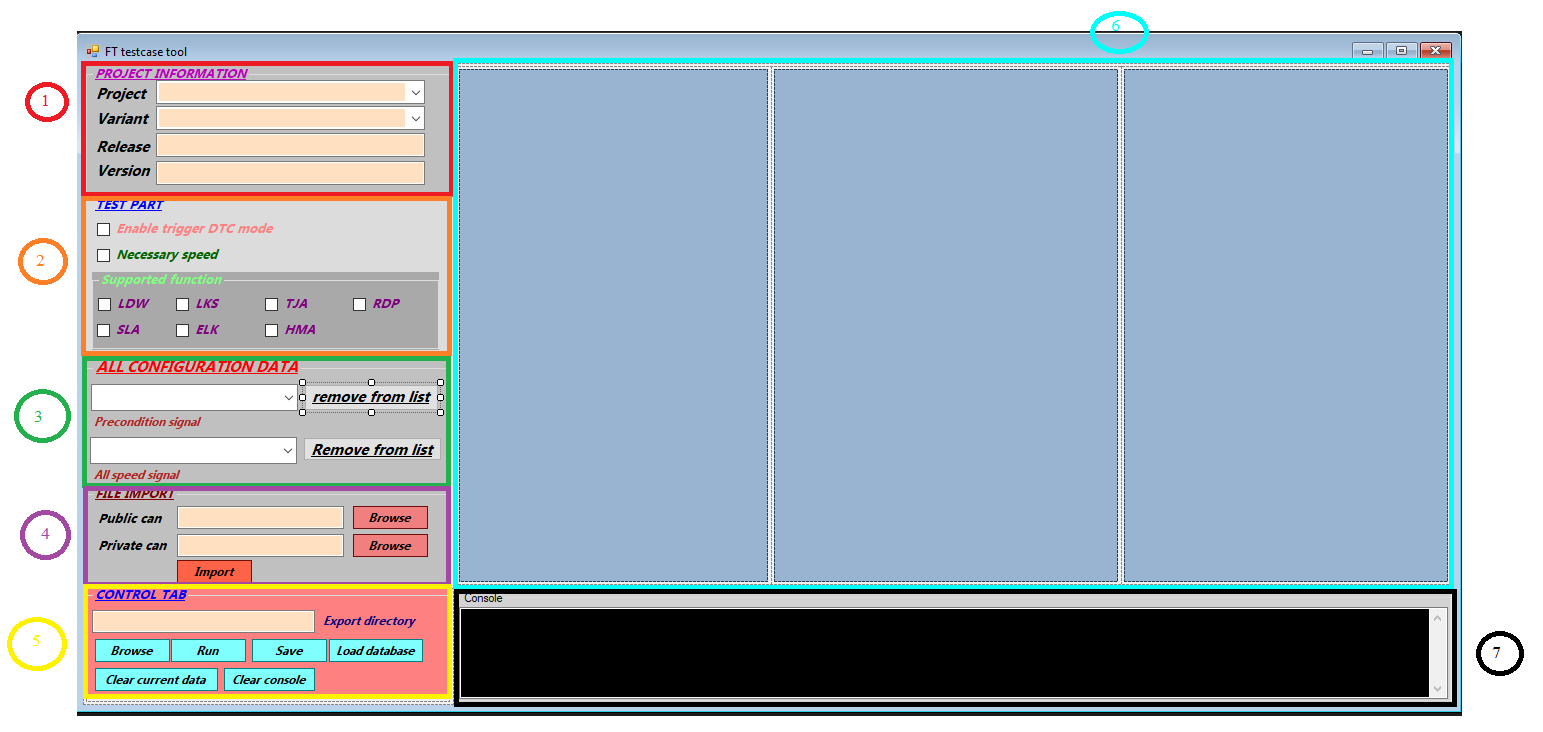
**Author: Ha Tien Tai – EDA23**

Contents

[**I.** **Overall view** 2](#_Toc54092841)

[**II.** **FILE STRUCTURE** 4](#_Toc54092842)

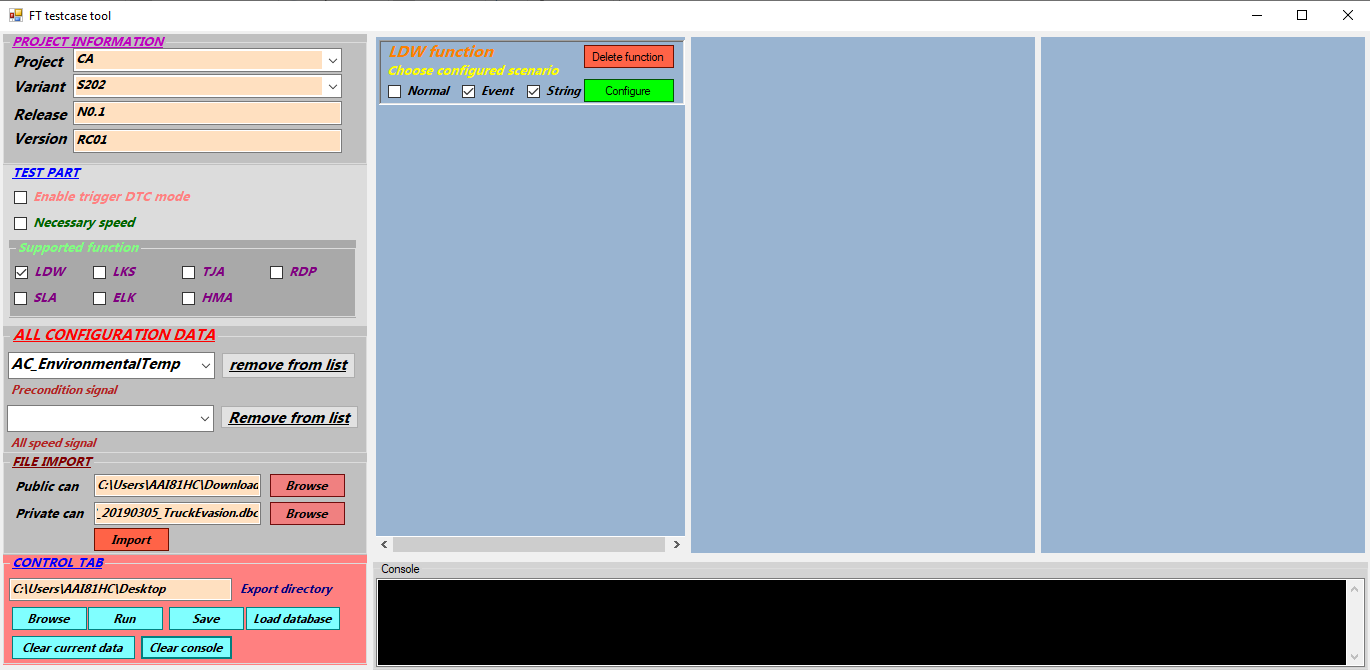
1. **Overall view**



* **(1) Project Information: Provide all textbox to input the mandatory information about the project**
* **(2) Test Part: Include all the relevant checkbox when performing function test, content for each checkbox will pop up a small window in section 6 when they are ticked**
* **(3) All Configuration Data: These scroll down lists include all pre-condition signals and speed signals. About function data, they will be display when the relevant pop up window show up**
* **(4) File Import: The location to input the directory of public-can and private-can dbc file**
* **(5) Control tab:**
* **Export directory: the output directory of your xml file, the name of the xml will have the format project\_variant\_release\_version\_labT.xml**
* **Run button will execute the procedure outputting the xml file with all the current configured data**
* **Save button will save all current data to the csv file, the file structure for this tool operation will be mentioned in the part II**
* **Load Database button will load all the data in csv file that relevant to the current project and variant**
* **Clear current data button: this one is too obvious ☺**
* **Clear console: so is this one**
* **(6) Content window: This window contents all the pop up window when you tick a checkbox in Test Part section**
* **Column 1: Content the function scenario as show below**

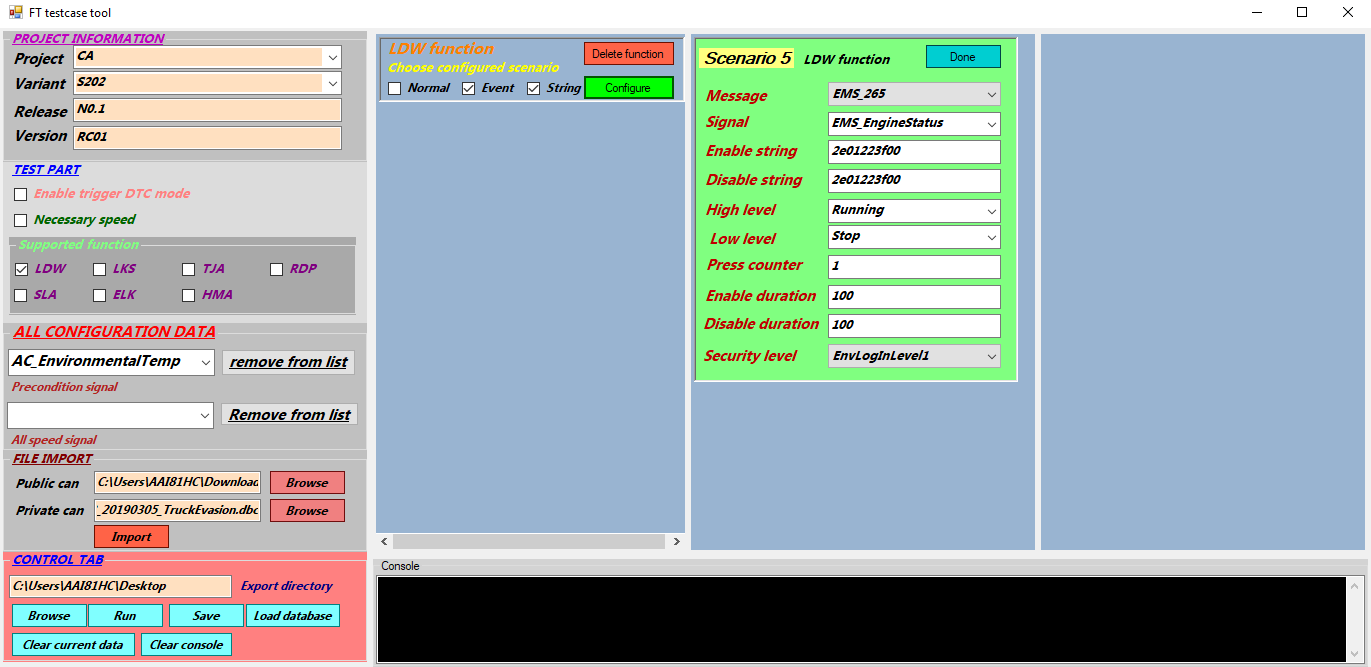
****

* **If there’s any pre-configured scenario for that function, the relevant checkbox will be ticked**

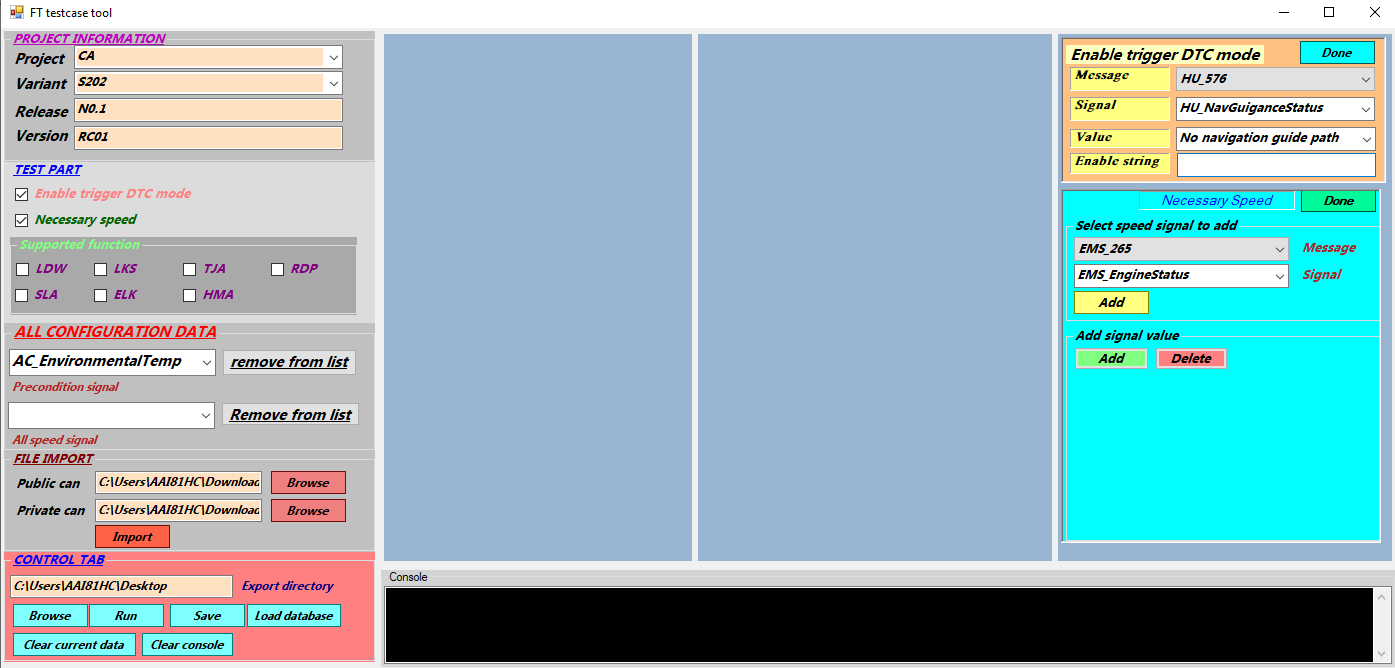
****

**Example for LDW function with scenario 5 (event and string)**

* **Column 2: Content the configuration window relevant to that scenario when you press Configure**

****

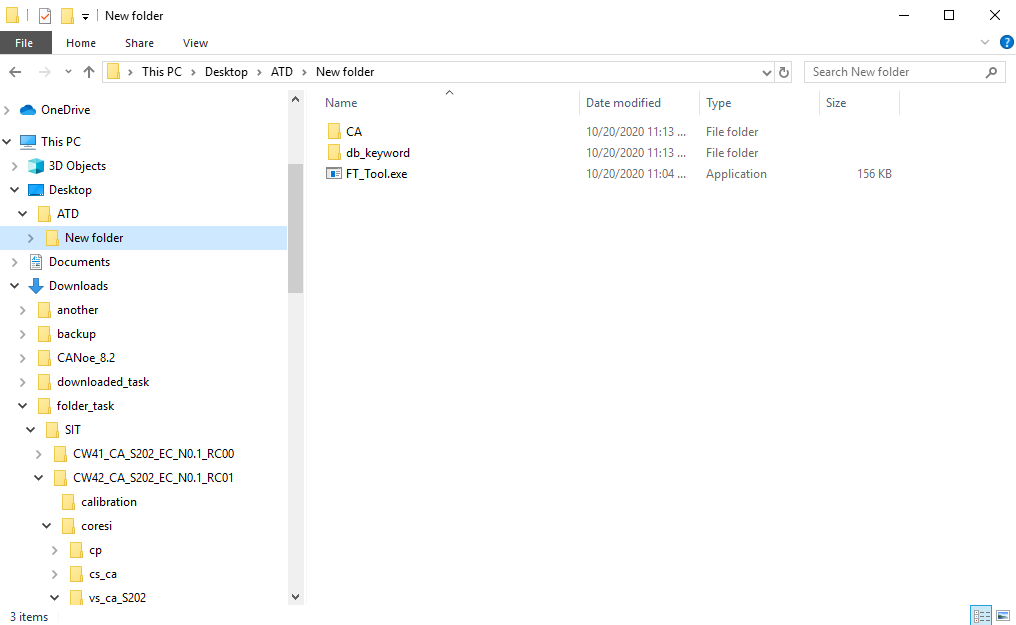
* **Column 3: Content the configuration window for trigger dtc mode and necessary speed**

****

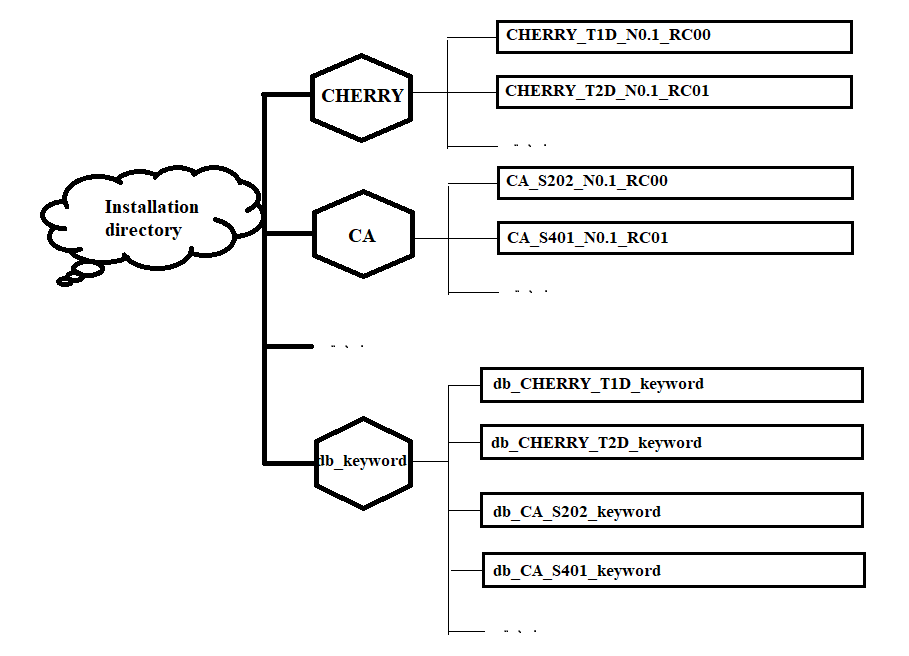
* **(7) Console window: print out all error and configuration status throughout setting procedure**

1. **FILE STRUCTURE**

* **For each particular project and variant, this tool will generate two csv file. One is for all the setting data, the second is for the necessary keyword when importing dbc file**
* **All the file is available in the same directory with exe file**

****

* **The example file structure can be summarized in the below picture**

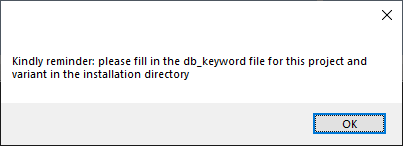
****

1. **EXAMPLE SETTING**

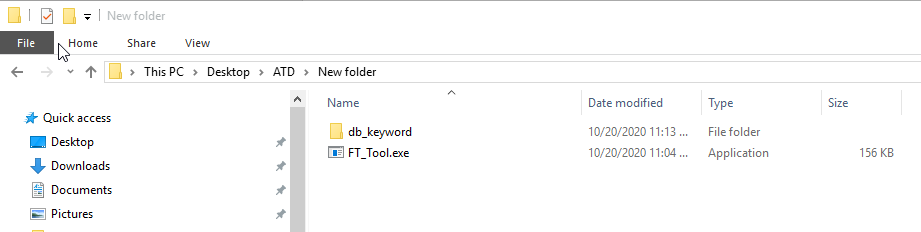
* **In this part we will perform the example setting to get the general idea of this tool**
* **We use project CA\_S202\_N0.1\_RC01 for example**
* **First fill in all the textbox in project information section**



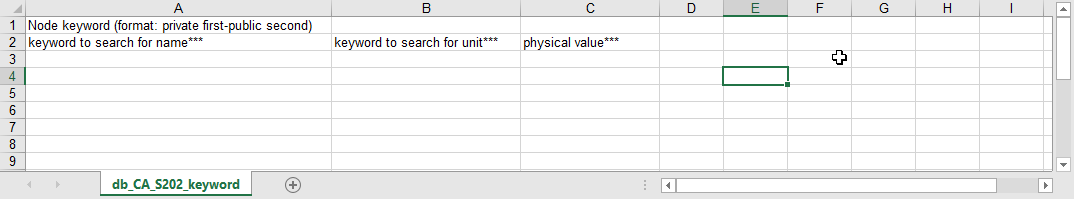
* **Next we choose the directory of the public and private can, you can use browse button to set the link or just paste the link if you have it already. Notice the small popup window like this will appear**



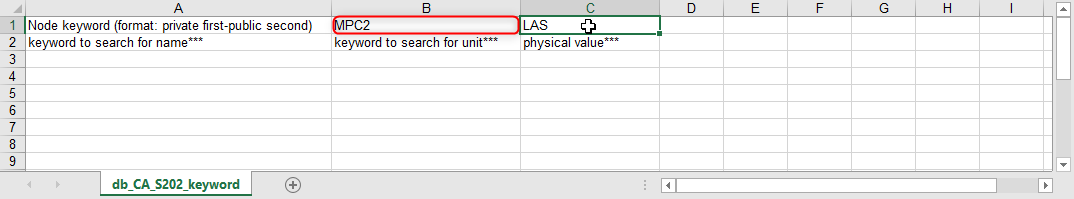
* **After this state, the tool has generated the keyword file for this project and variant**



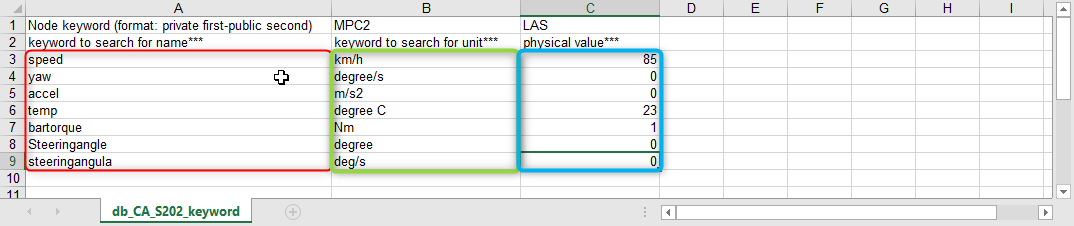
* **Go in db\_keyword folder and open file db\_CA\_S202\_keyword.csv. It has the default format like this**

****

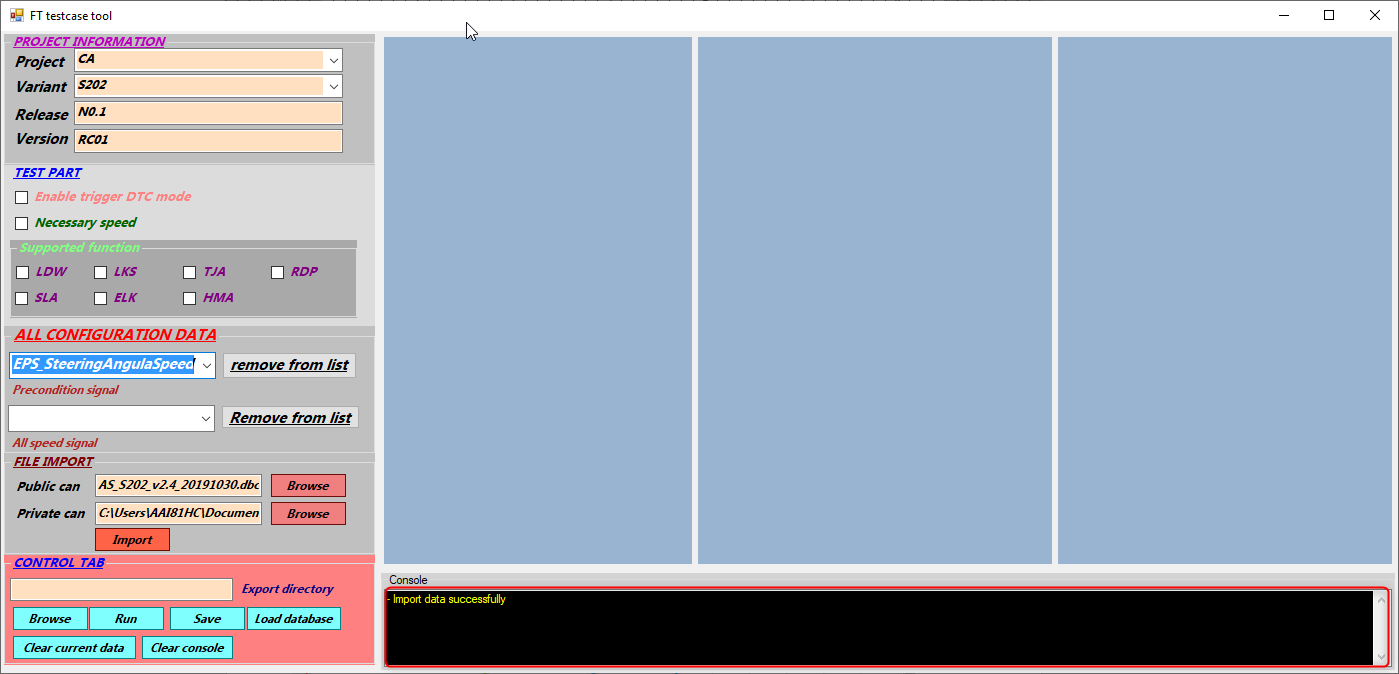
* **The main purpose of this file is to list out which ECU node we want to read in the dbc file and automatically select some precondition signal. Because the name of the MPC node may be different for each project and variant so these node keyword is mandatory**
* **At B1 we fill in the node in the private-can dbc and C2 for the public one**



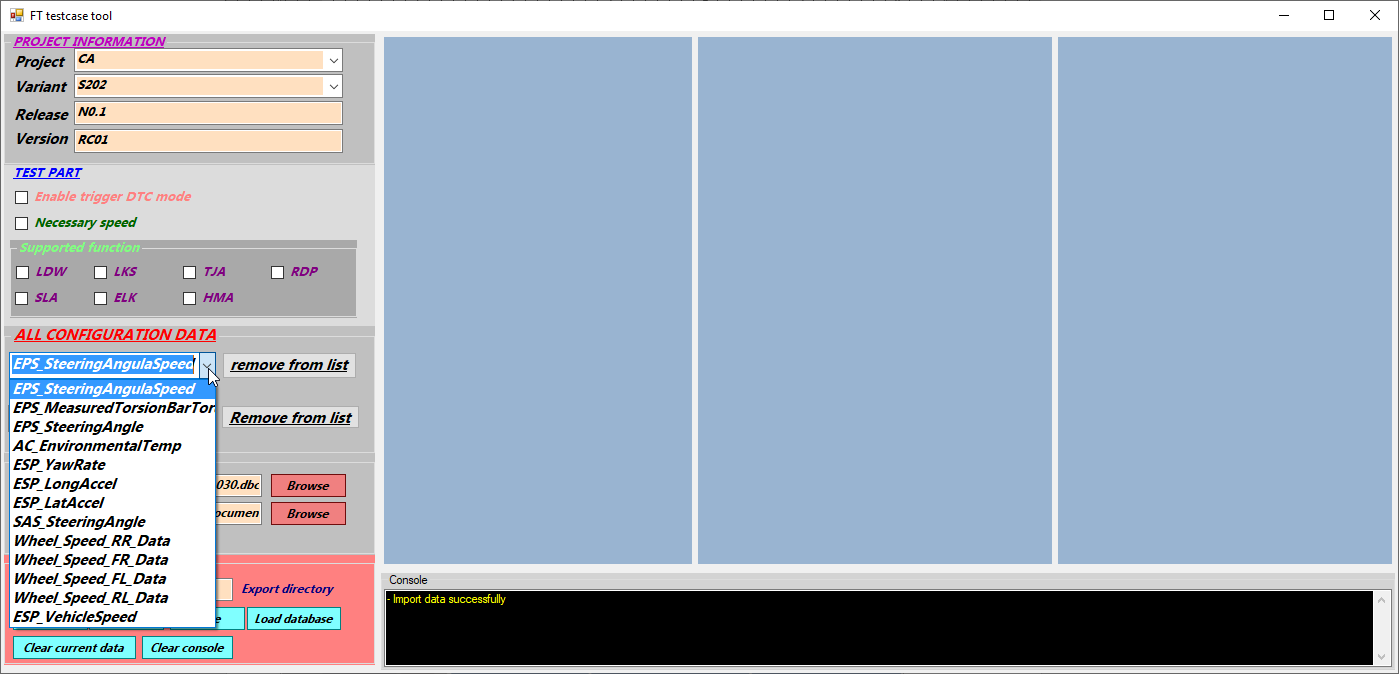
* **Now we need to specify keyword for precondition signal so that the tool can automatically filter it out. Example like this**



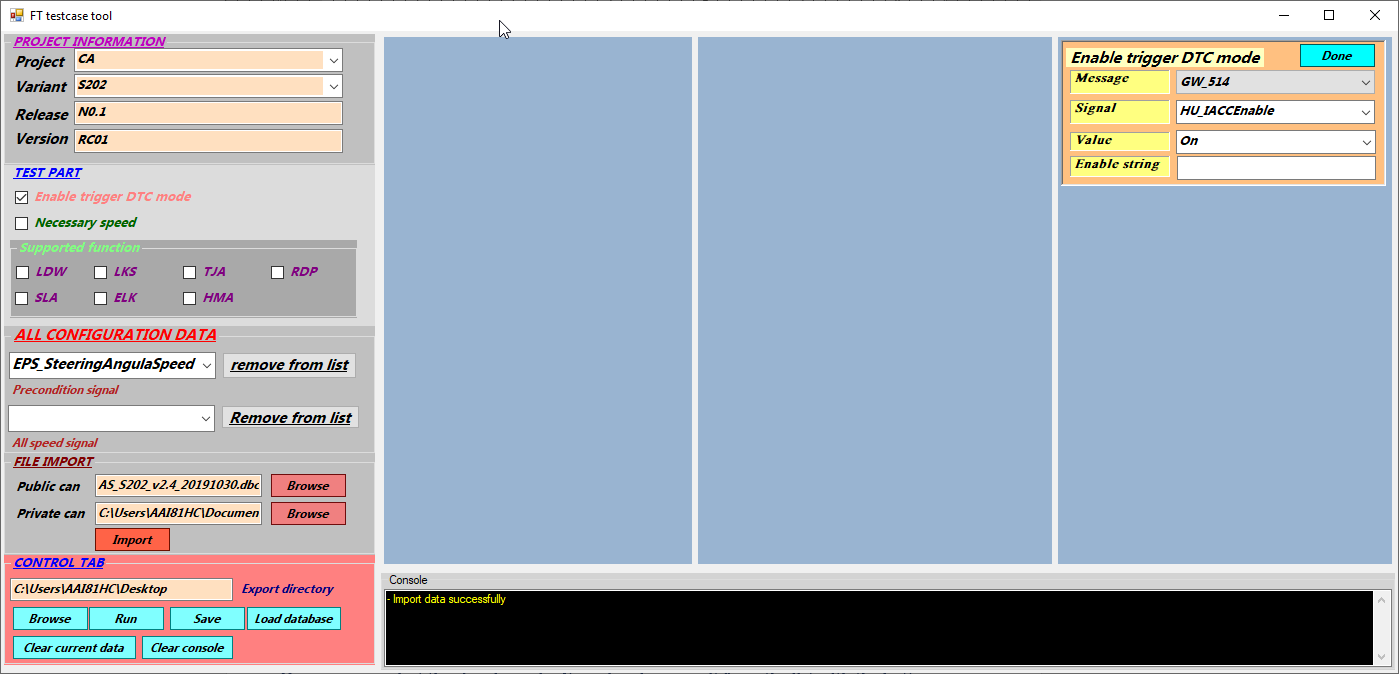
* **You don’t have to remember exactly the signal name, you can fill in a keyword in that signal. This also apply for unit keyword. If a signal fulfill both requirement about name and unit, it will be selected at precondition signal. Save the file and back to the tool**
* **After you fill in the directory for public and private can press import. If succeed, it will print a small notification on console**

****

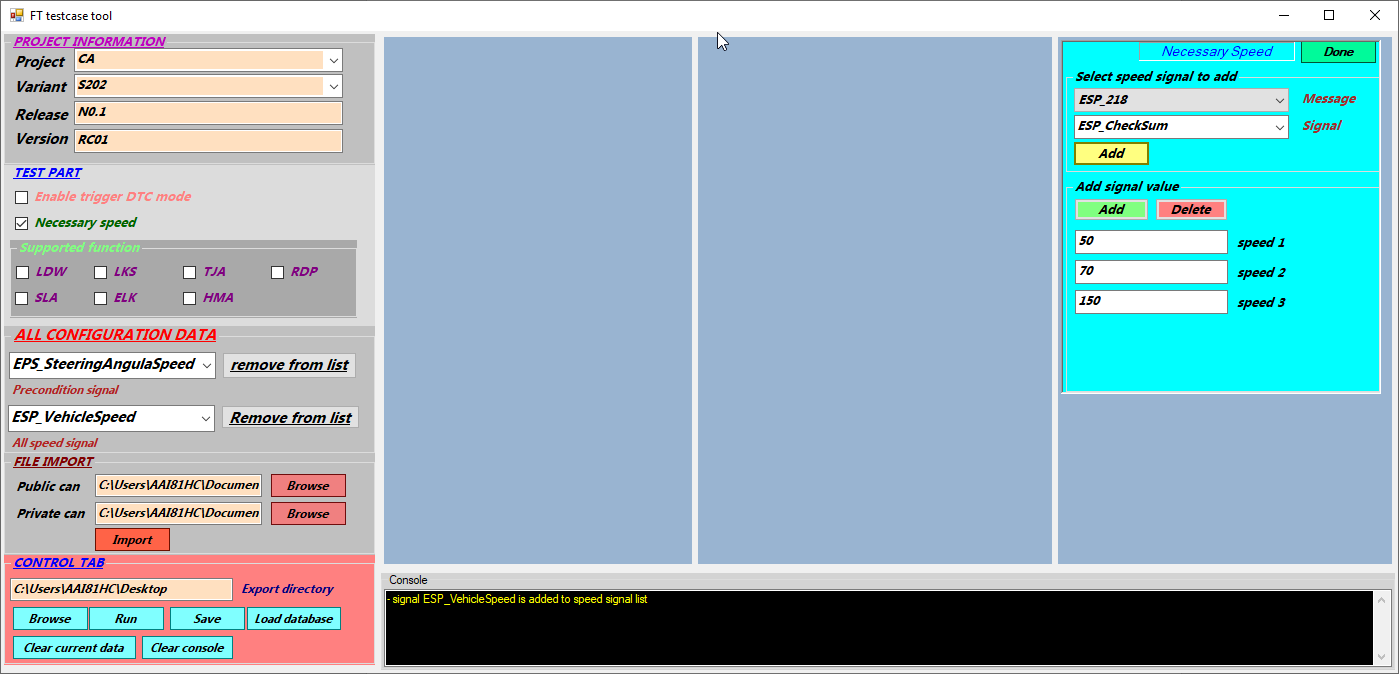
* **Now drop down the precondition list you can see multiple signals which not quietly what you want. Because the may be some signal that can both fulfill the name and the unit keyword**

****

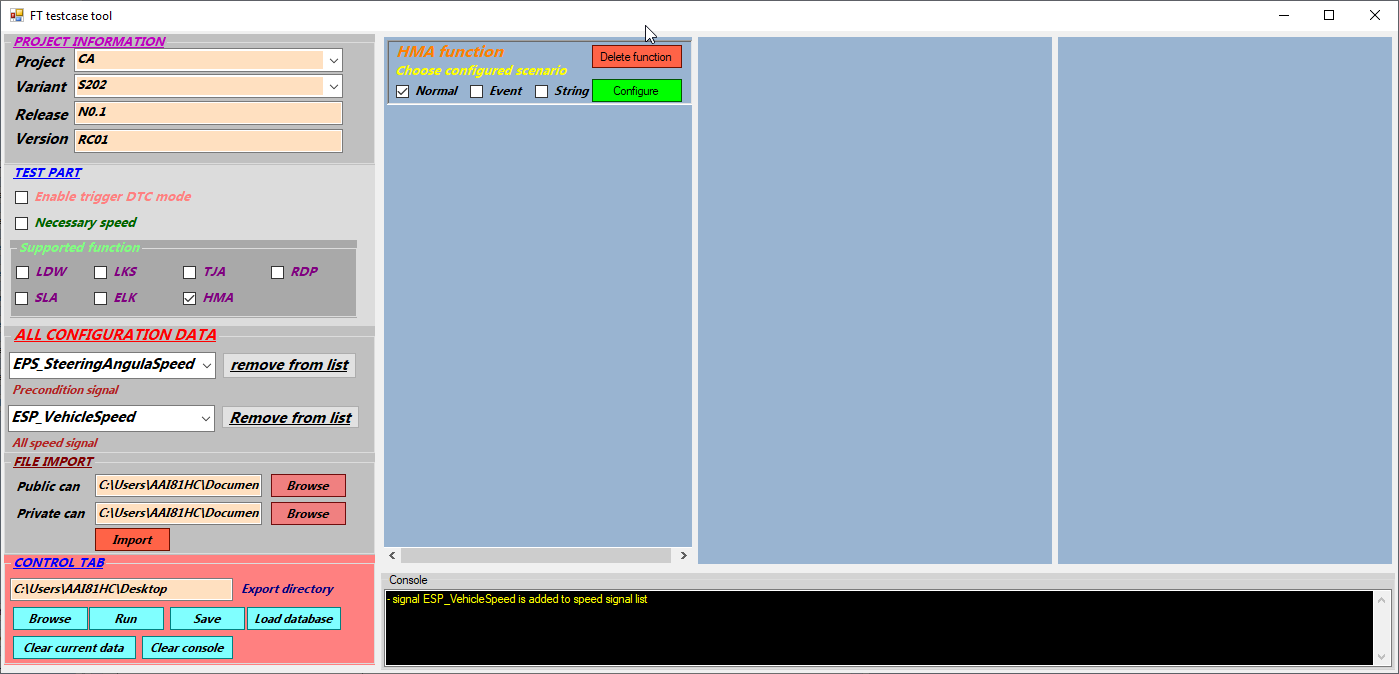
* **Here you can select the signal you don’t need and remove it from the list with the button on the right. So to get the best result, you need to specify the name and unit at much at possible**
* **We head to the test part. Tick the enable trigger DTC mode**

****

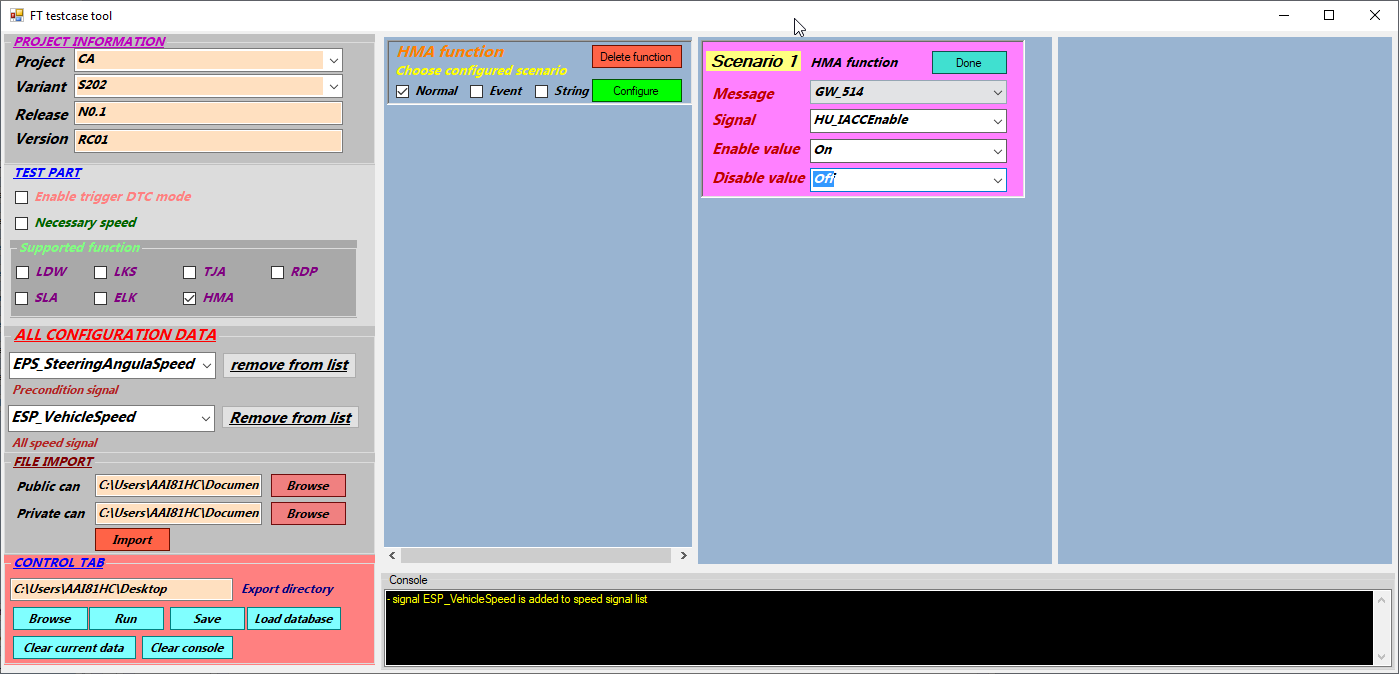
* **At here you have two drop down list for message and signal so that you can find the signal easier. *Small tip is that you can paste the signal name in signal drop down list and press enter, it will automatically search that signal for you.***
* **After you finish configuring that signal, you can press done**
* **In the necessary speed tag, you need to add speed signal. It will be updated in all speed signal drop down list. After that you have to specify all the speed value you want to set for that signal**

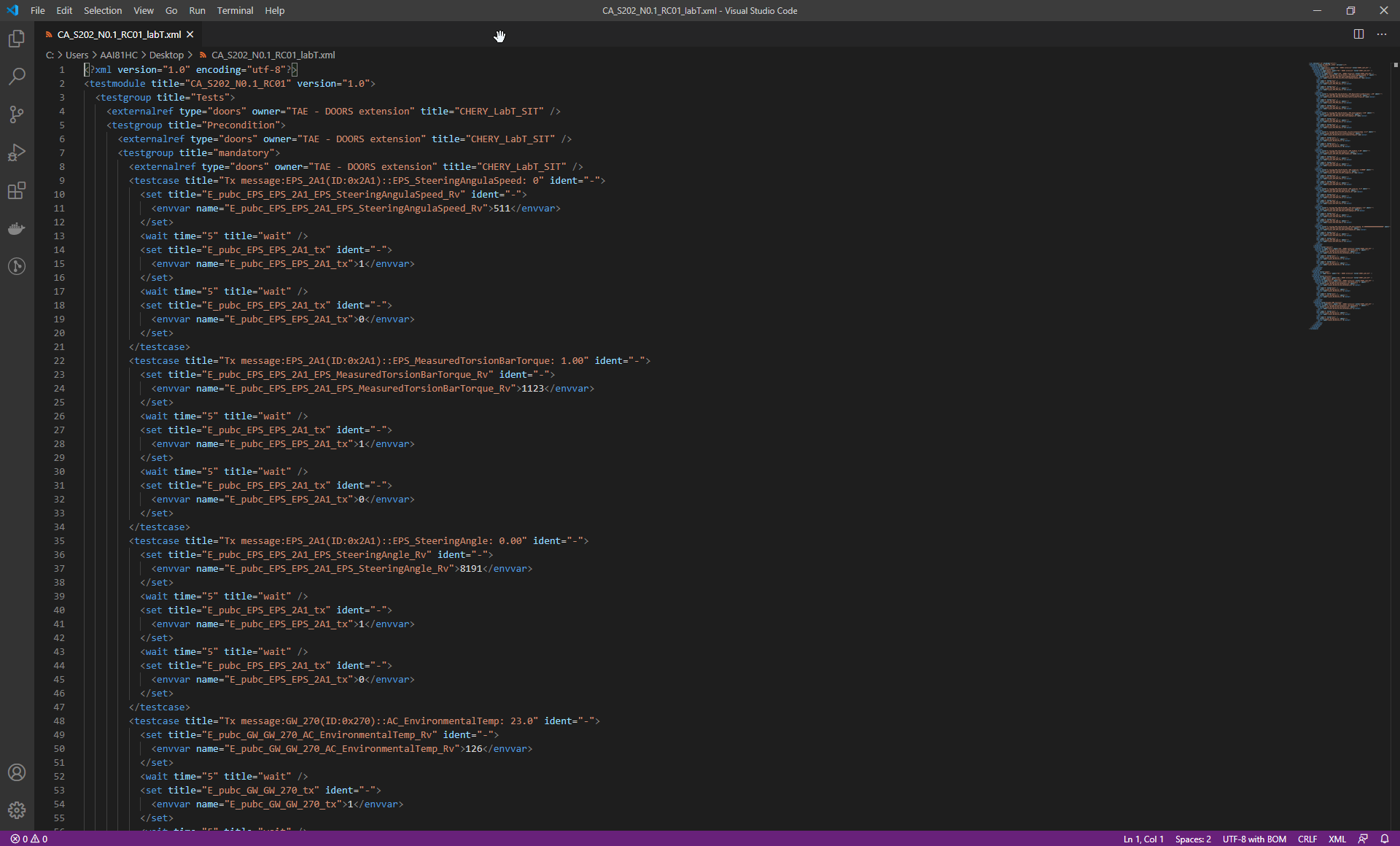
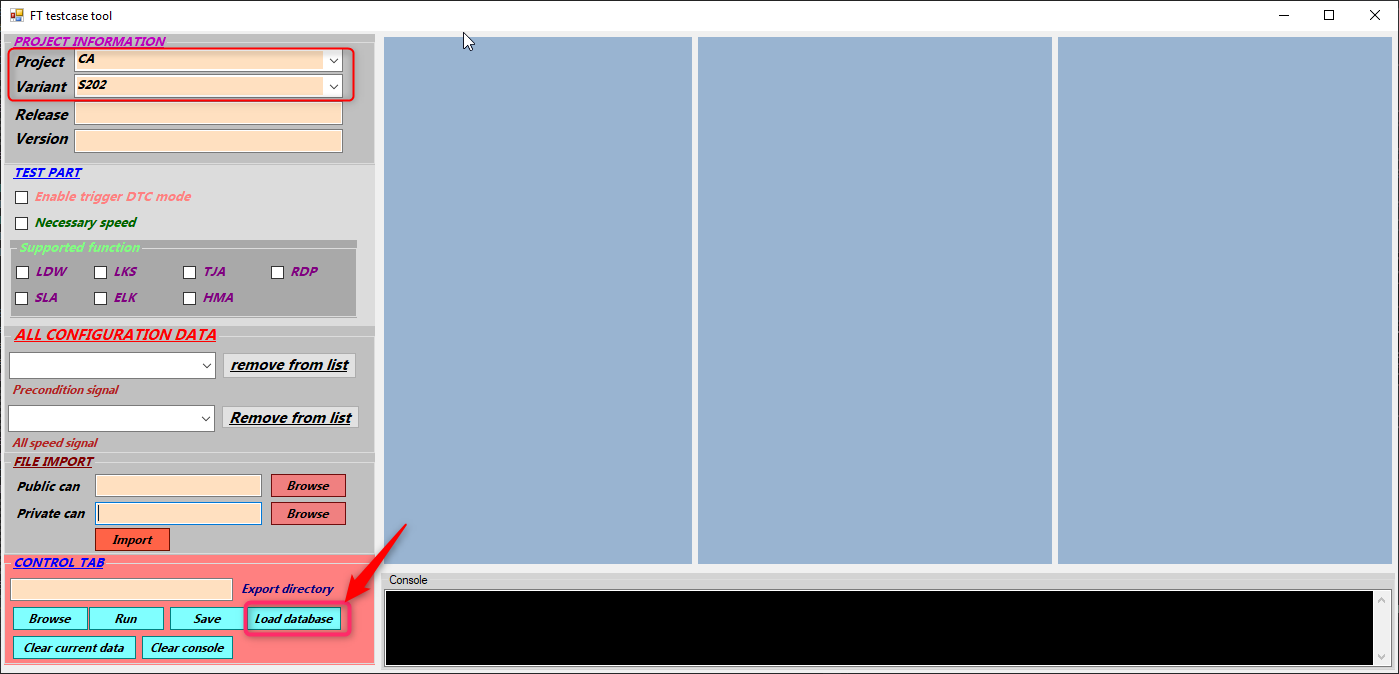
****

* **After you finish press done**
* **Next select the supported function you want to turn on and off and select the scenario for that function**

****

* **Press configure, find the signal you want and set all the necessary data in the textbox. The search feature is also worked here**

****

* **After that press done**
* **At this point, you should press save button so that all you data can be available for the next time. Or it will also save it you press run. Remember that all the necessary information must be filled before saving**
* **At the export directory after you hit run, you will have your script file, put this to coresi and run it**
* ****
* **In the next time when you run the tool you just need to select the project and variant, after that press load database and all the data that you previously configure will be available**
* ****