# Web Application Code Document

This instruction document is for all the Web Application related code, where we have codes that needs to run before to access Web Application. Also, this folder contains different HTML5 + CSS3 codes for Web Pages. In this project, we have used the Python Flask API module.

Flask is a web framework that provides you with tools, libraries, and technologies to build a web application. This web application can be some web pages, a blog, a wiki, or go as big as a web-based calendar application or a commercial website.

Flask is part of the categories of the micro-framework. Micro-framework is usually a framework with little to no dependencies on external libraries that have pros and cons. The advantage would be that the frame is light. There is a low dependency to update and watch for security bugs; a disadvantage is that sometimes you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins.

**Note:** This project is developed on a Linux System (Ubuntu), so it is advisable to use Linux System.

### System Set-Up

There are a couple of Python packages that need to validate before running any above program in this folder. Please go to the Installation Documents folder and follow all the steps.

### Pre-Requisites

- A system running on Windows/Ubuntu APP/Ubuntu OS
- A user account with sudo/administration privileges
- Access to a terminal window/command-line
- Validate Flask Module (Flask comes with Python installation in Ubuntu, but if you do not have Flask installed on your computer, please visit here for installation)

Before continuing with this tutorial, make sure you are logged in as root or a user with sudo or administration privileges and completed all the necessary steps from the Installation Documents folder**.**

In this tutorial, we will explain all the Web Application Codes written in Python3 and Flask on Ubuntu.

1. Web_Application_Frontend.py
2. Web_Application_Backend.py
3. templates (HTML5 codes directory)

```
somak@VivoBook:~/Robotic-Greeter/Development_Code/Web_Application_Code$ ls -lrt
total 28
drwxrwxr-x 2 somak somak  4096 Jul  5 02:08 templates
-rwxrwxr-x 1 somak somak  6686 Jul 11 22:09 Web_Application_Frontend.py
-rwxrwxr-x 1 somak somak 10291 Jul 11 22:09 Web_Application_Backend.py
-rw-rw-r-- 1 somak somak  2867 Jul 11 22:23 README.rst
somak@VivoBook:~/Robotic-Greeter/Development_Code/Web_Application_Code$
somak@VivoBook:~/Robotic-Greeter/Development_Code/Web_Application_Code$
```

**Web Application Code**

Web Application codes are written in Python and Flask as follow:

1. **Web_Application_Frontend.py**

   Web_Application_Frontend.py is the main Python program that needs to run to access Web Application. This program will first call home.html to show to the home page in the Web Application. Based on the user's selection, this program will call another HTML program on the home page in Web Application. Also, this program will call the Web_Application_Backend.py process if the user wants to do a specific process.

   This program contains a specific parameter that needs to validate and check before running this program.

```python
import os
import subprocess
from flask import Flask, render_template, request
```

```python
if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=4000)
```

```python
app = Flask(__name__)
app.config['SECRET_KEY'] = 'secret-key'
```

2. **Web_Application_Backend.py**

   Web_Application_Backend.py program will be call from Web_Application_Frontend.py based on user's selection. Web_Application_Frontend.py will pass an input argument to this process, and based on the input argument; this process will call different Python programs such as Customer_Insert.py. Customer_Update.py, Customer_Search_Main.py, Main_Proces.py, Speech_Emergency_Evacuation_Procedure.py, etc.

   This program has one function called **process_parameter_set(),** which contains the value of main_directory. It would be best to make sure that all the folder of Database_Code,

Face_Recognition_Code, Speech_Recongnition_Code, Main_Process, and Web_Application_Code are present in this following directory. If you want another specific folder, then you need to make a change here.

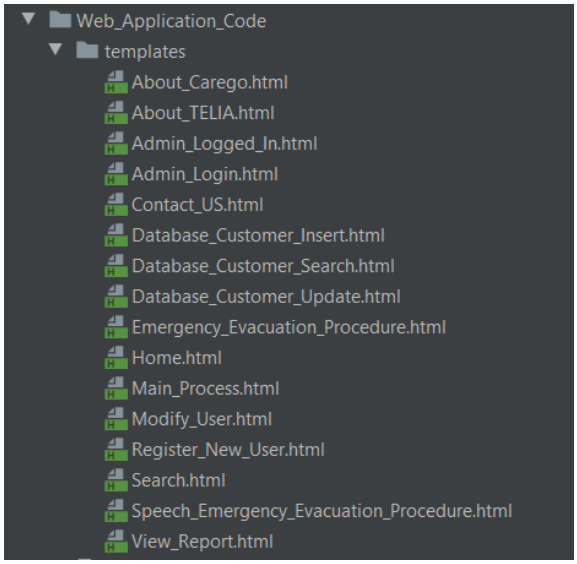
```
def process_parameter_set():
    """
```

```
region = "DEV"
main_directory = "/home/somak/Robotic-Greeter/Development_Code"

return region, main_directory
```

### 3. templates

Template folder contains all the HTML5 + CSS3 codes.




**Execution:**

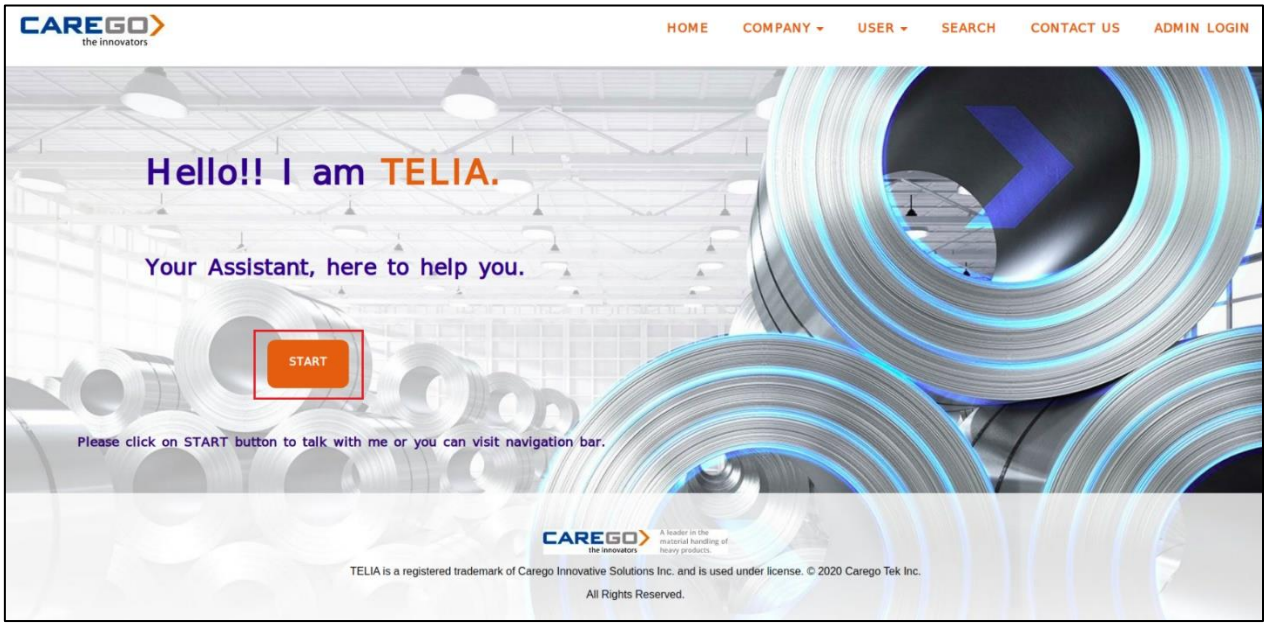
- To run this program, run the below command from your command prompt.

```
$ python3 Web_Application_Frontend.py
```

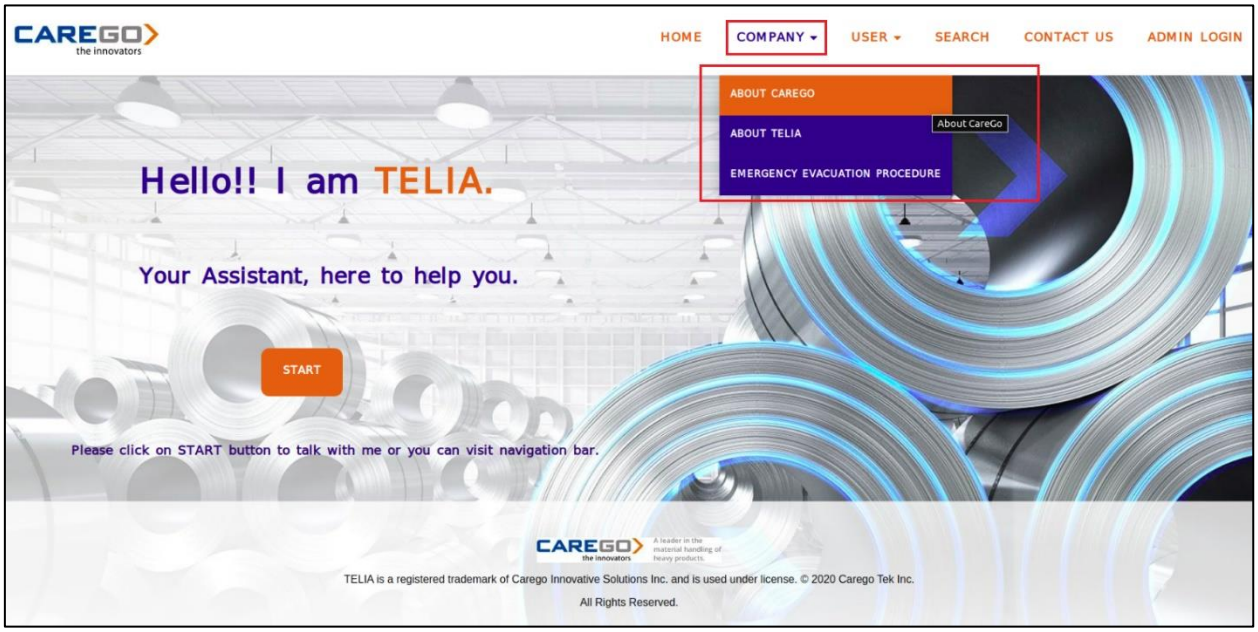
```
somak@VivoBook:~/Robotic-Greeter/Development_Code/Web_Application_Code$ python3 Web_Application_Frontend.py
 * Serving Flask app "Web_Application_Frontend" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:4000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 242-430-158
```

- Copy and Paste the above link on your browser, and you see the home page as below.

- The home page has various navigation bar. Also, on the home page, you can click on the START button to take directly with the Robot. If you click the START button, Web_Application_Frontend.py will call Web_Application_Backend.py with one argument, and Web_Application_Backend.py will call Main_Process.py (Which is the primary backend process, combining all database, face_recognition and speech_recognition codes)

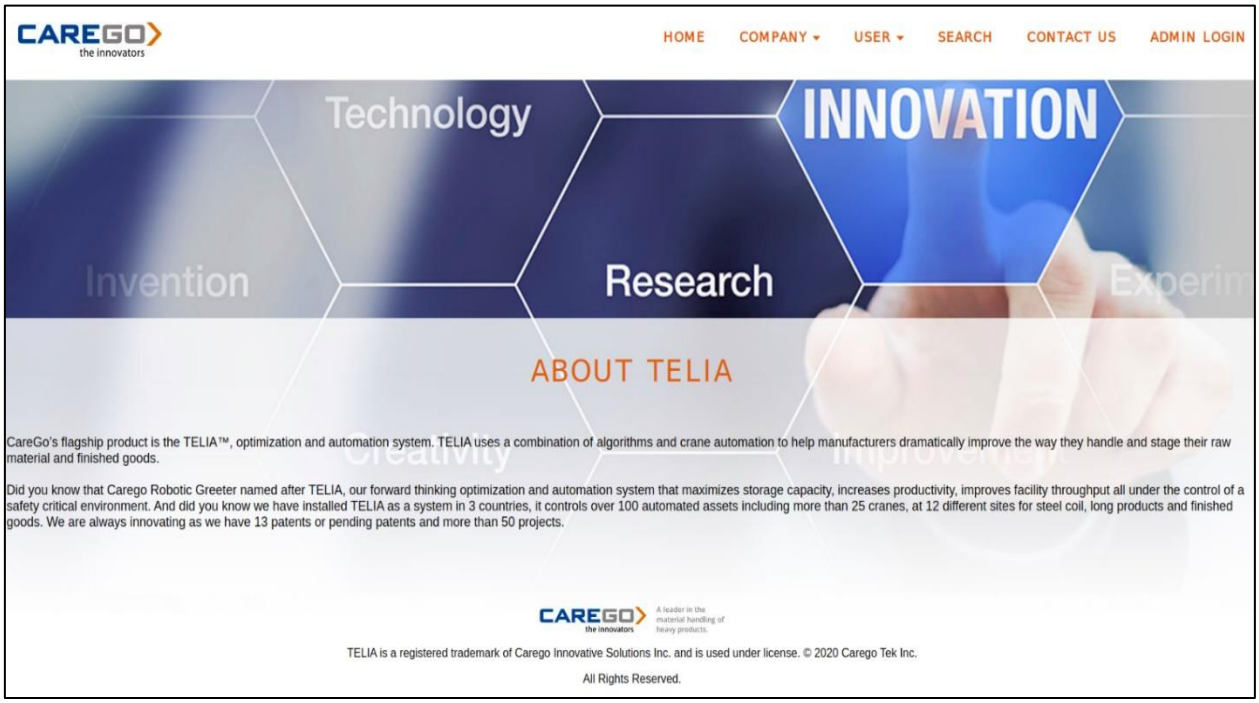



- Go to the navigation bar and click on COMPANY. In the COMPANY tab, you will ABOUT CAREGO, ABOUT TELIA and EMERGENCY EVACUATION PROCEDURE.



- Click on ABOUT CAREGO


- Click on ABOUT TELIA



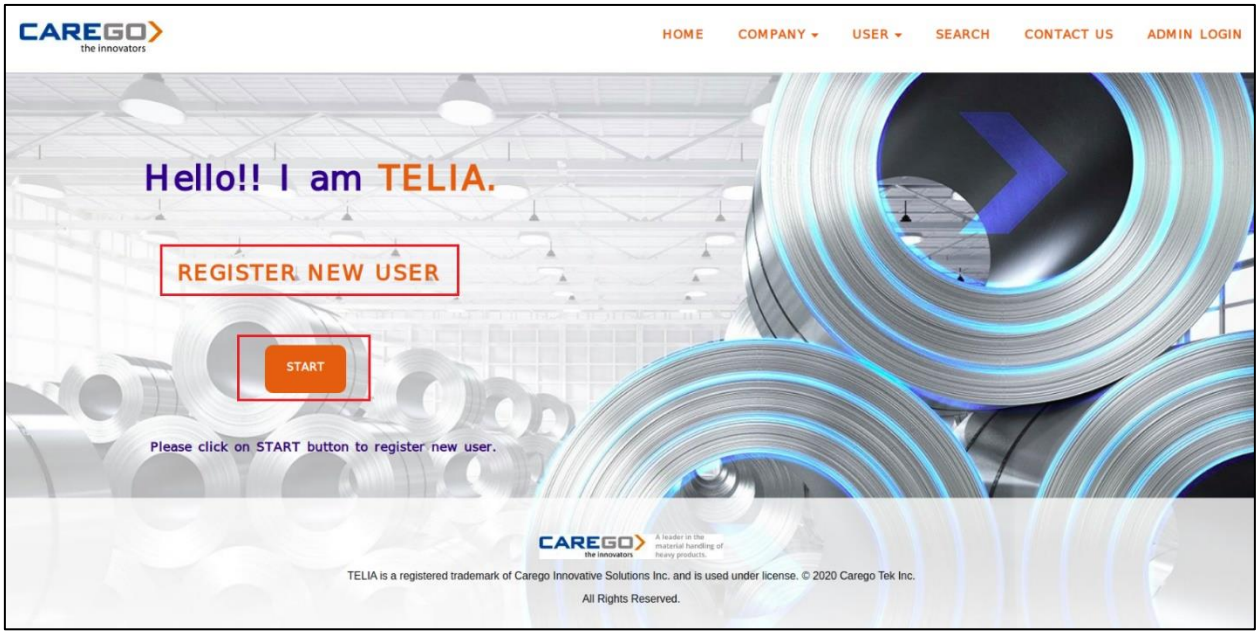

- Click on EMERGENCY EVACUATION PROCEDURE. On the emergency evacuation procedure page, you will see one START button. If you click on the START button, Robot will show all the emergency evacuation procedures.

When you click on the START button, Web_Application_Frontend.py will call Web_Application_Backend.py with one argument, and Web_Application_Backend.py will call Speech_Emergency_Evacuation_Procedure.py


- In the user tab, you will find an option for REGISTER NEW USER and MODIFY USER.

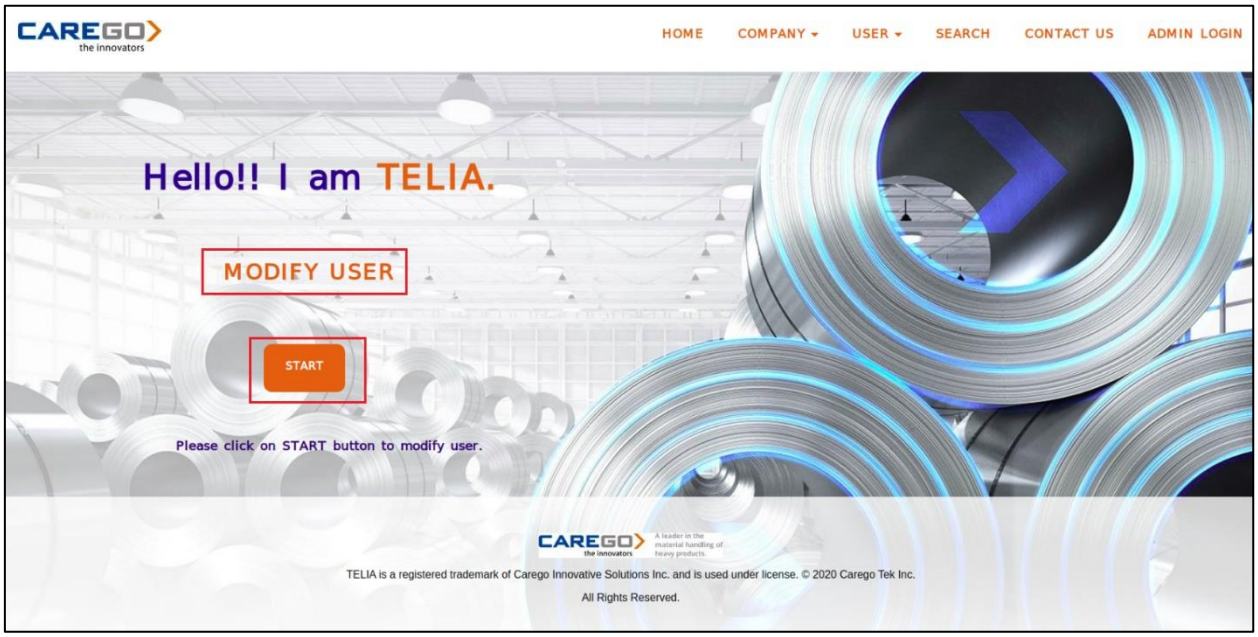

- Click on REGISTER NEW USER. On the register user page, you will see one START button. If you click on the START button, Robot will pop-up the registration screen in front of you.

When you click on the START button, Web_Application_Frontend.py will call Web_Application_Backend.py with one argument, and Web_Application_Backend.py will call Customer_Insert.py



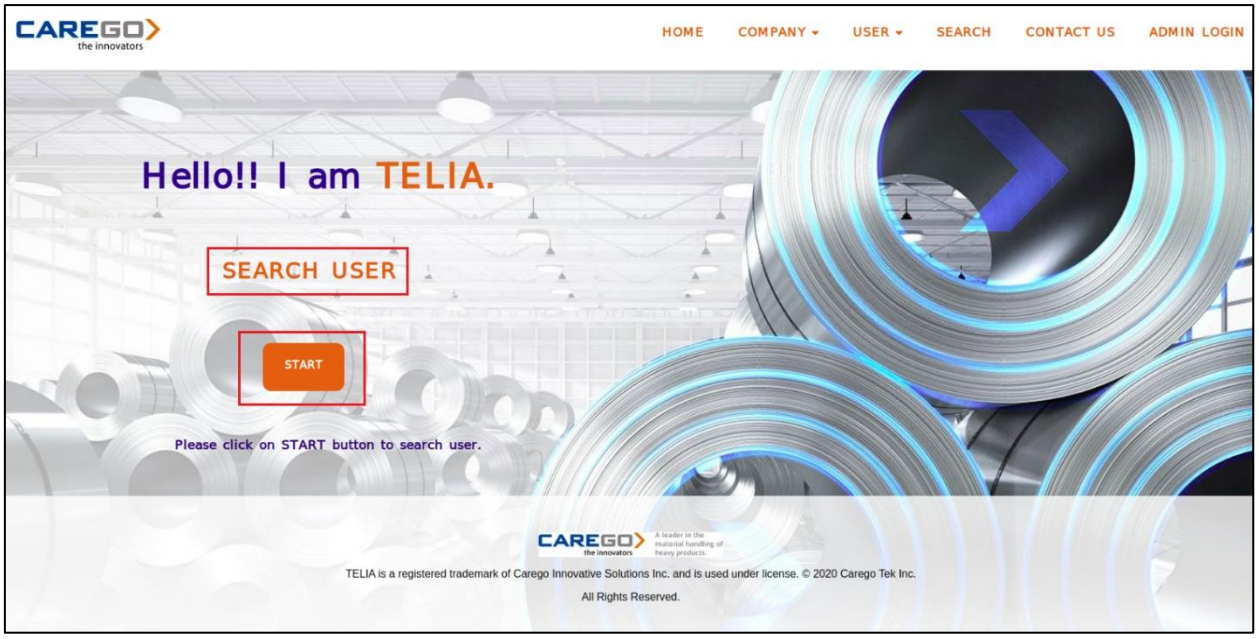

- Click on MODIFY USER. On the modified user page, you will see one START button. If you click on the START button, Robot will pop-up the update screen in front of you.

  When you click on the START button, Web_Application_Frontend.py will call Web_Application_Backend.py with one argument, and Web_Application_Backend.py will call Customer_Update.py

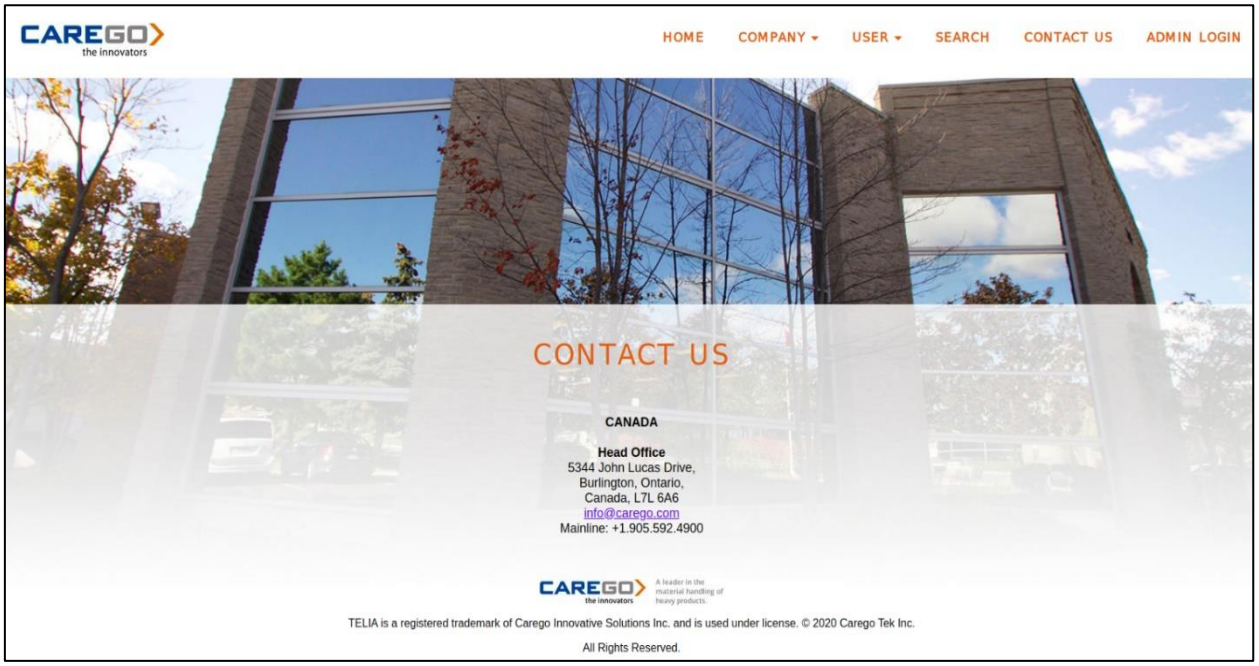


- Click on SEARCH USER. On the search user page, you will see one START button. If you click on the START button, Robot will pop-up the search screen in front of you.

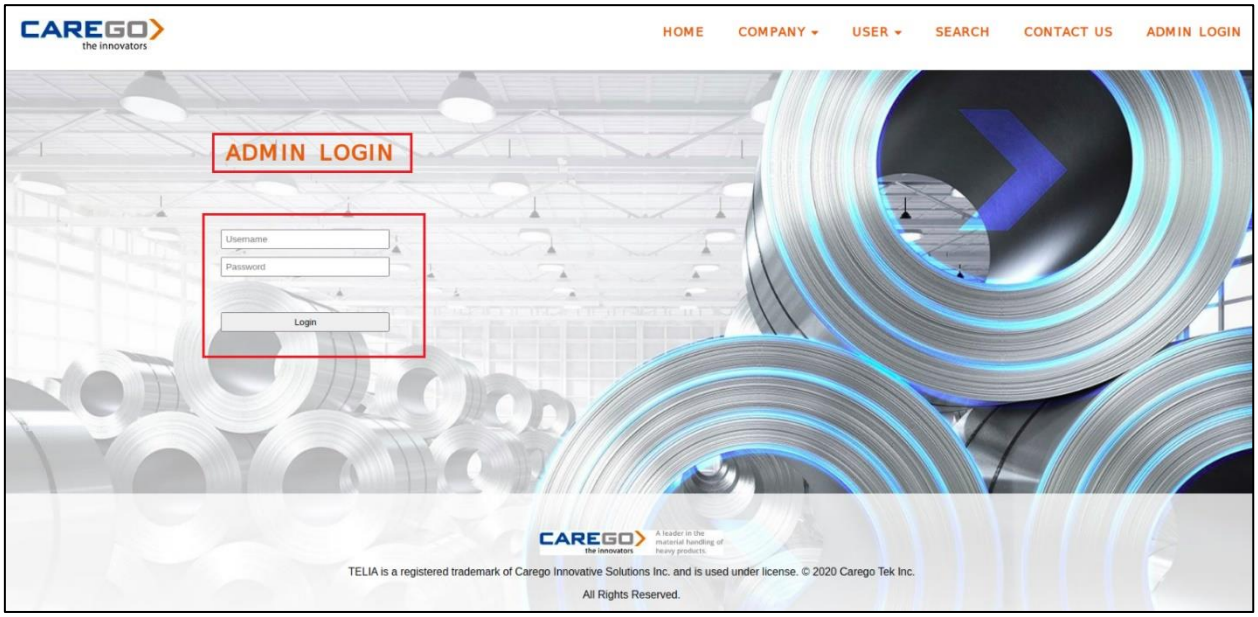
  When you click on the START button, Web_Application_Frontend.py will call Web_Application_Backend.py with one argument, and Web_Application_Backend.py will call Customer_Search_Main.py



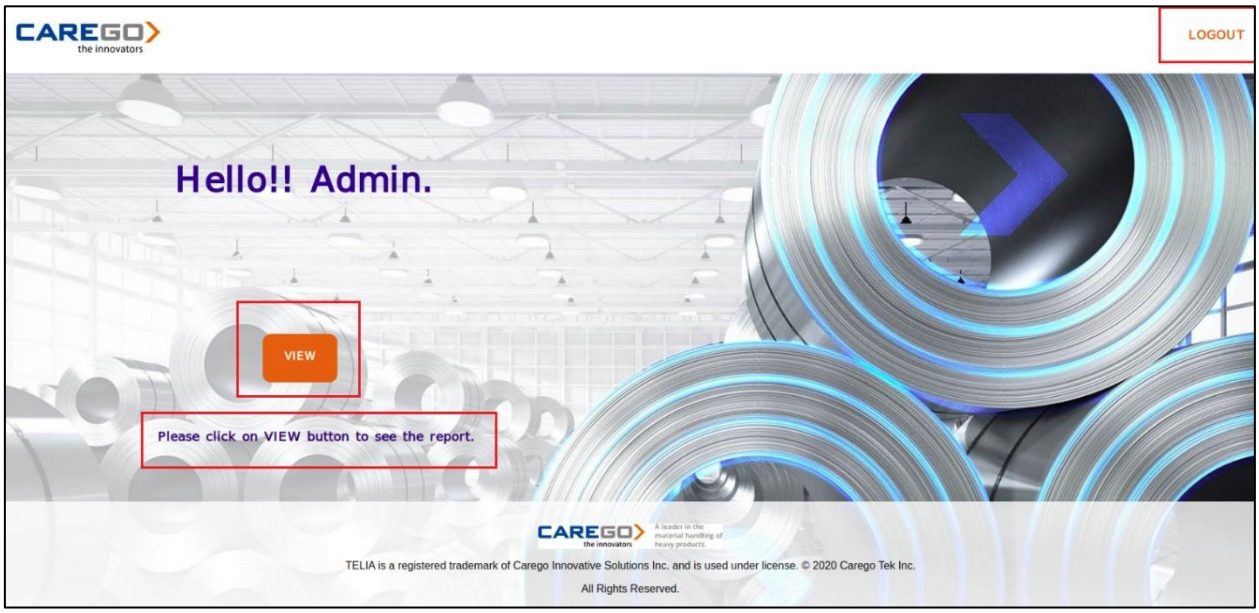

- Click on CONTACT US to see the contact information.



- Click on ADMIN LOGIN. On the admin login page, you need to give your (admin's) username and password for login.




- Admin can click on the VIEW button to see the report and click on LOGOUT to log off.

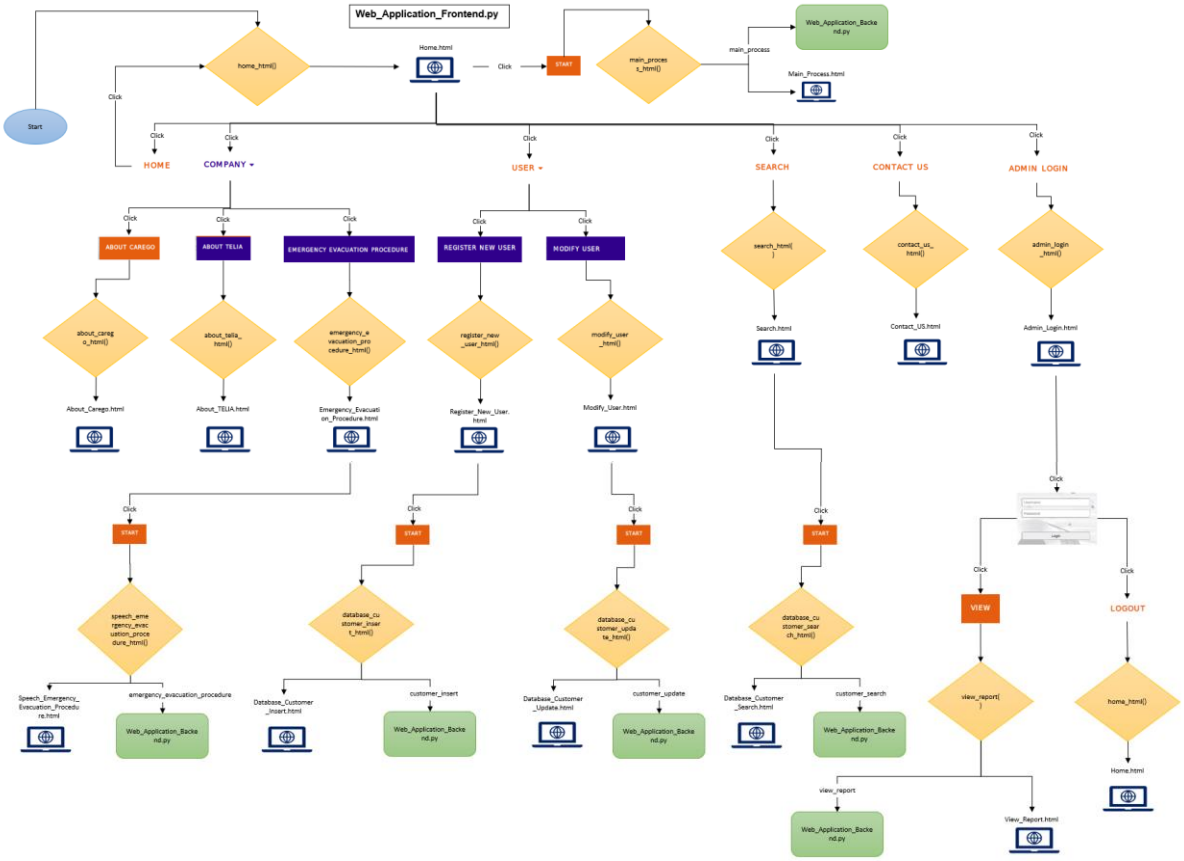



- The report will be generated on a different path. Please go to the following folder to see the report.

  **/Robotic-Greeter/Reports/**

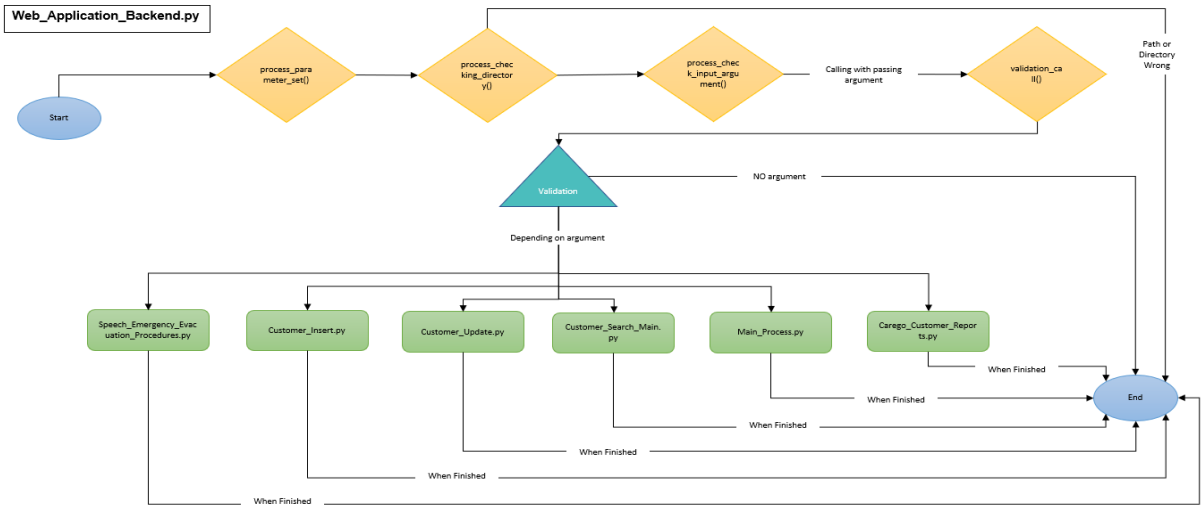
## Flowchart

Following is the flowchart architecture of the Web_Application_Frontend.py program.




## Flowchart

Following is the flowchart architecture of the Web_Application_Backend.py program.


Web_Application_Backend.py
Start
process_parameter_set()
process_checking_directory()
process_check_input_argument()
Calling with passing argument
validation_call()
Path or Directory Wrong
Validation
NO argument
Depending on argument
Speech_Emergency_Evacuation_Procedures.py
Customer_Insert.py
Customer_Update.py
Customer_Search_Main.py
Main_Process.py
Carego_Customer_Reports.py
When Finished
When Finished
When Finished
When Finished
When Finished
When Finished
End