Task 4 - GUI App

Graphical user interface (GUI) has been created to help everyone understand what is been communicated between Server and Client.

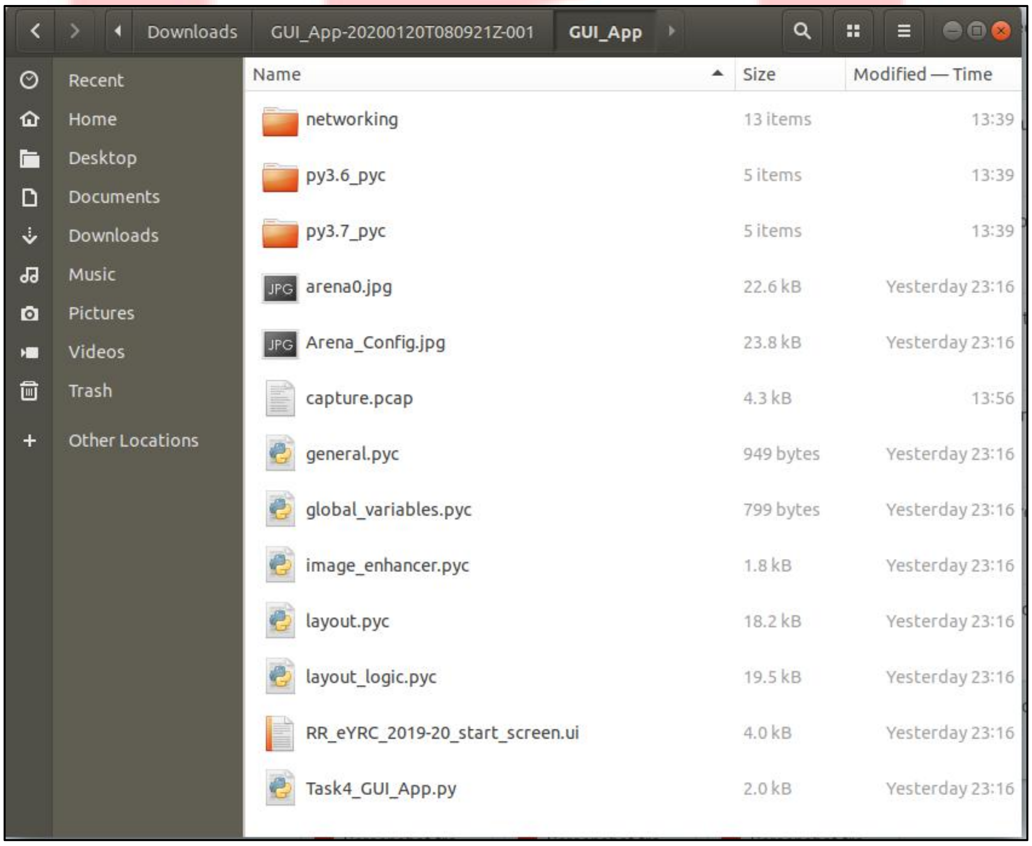
It will also give the values of other parameters as specified in the Scoring Formula and an arena image to indicate the Firezone/s, highlight the Path and position of Obstacle/s.

Prerequisites:

- **<u>GUI Application</u>:** The **Task4_GUI_App** folder provided in Task 4 consists of a file: **GUI_App_drive_link.txt** which consists of a **link** to the shared Google Drive folder. This folder consists of all the files required to run the **GUI**.

  Download the complete shared Google Drive folder. It will be downloaded as a **.zip** file.

  Extract it in any directory of your choice. Let's say the name of the extracted folder is **GUI_App**. This folder will consists of the files and folders as shown in the Figure 1.



Name	Size	Modified — Time
networking	13 items	13:39
py3.6_pyc	5 items	13:39
py3.7_pyc	5 items	13:39
arena0.jpg	22.6 kB	Yesterday 23:16
Arena_Config.jpg	23.8 kB	Yesterday 23:16
capture.pcap	4.3 kB	13:56
general.pyc	949 bytes	Yesterday 23:16
global_variables.pyc	799 bytes	Yesterday 23:16
image_enhancer.pyc	1.8 kB	Yesterday 23:16
layout.pyc	18.2 kB	Yesterday 23:16
layout_logic.pyc	19.5 kB	Yesterday 23:16
RR_eYRC_2019-20_start_screen.ui	4.0 kB	Yesterday 23:16
Task4_GUI_App.py	2.0 kB	Yesterday 23:16

Figure 1: Content of the shared Google Drive folder

NOTE: You are not allowed to open / edit / rename / move any files from this folder. Kindly do not touch it.

- **PyQt5 installed in Python 3.6 or 3.7:** Follow each of these steps below carefully and make sure no errors are occurred while performing them.
  - Open a Terminal and type: "**python3 -V**". Check whether the output is: **Python 3.6.x** or **Python 3.7.x** (**x** can be any number).
  - If the output of above command is **not Python 3.6.x** or **Python 3.7.x**, then follow these steps:
    - If your system is **Ubuntu 16.04**, type the following commands in a sequence to install **Python 3.7**:
      - Start by updating the packages list and installing the prerequisites:
        - "**sudo apt update**"
        - "**sudo apt install software-properties-common**"
      - Next, add the deadsnakes PPA to your sources list:
        - "**sudo add-apt-repository ppa:deadsnakes/ppa**"
        - When prompted, press **Enter** to continue.
      - Once the repository is enabled, install Python 3.7 with:
        - "**sudo apt install python3.7**"

      At this point, Python 3.7 is installed on your Ubuntu system and ready to be used. You can verify it by typing: "**python3.7 --version**". You will get output as: **Python 3.7.x**.
    - If your system is **Ubuntu 16.04**, type the following commands in a sequence to install **Python 3.6**:
      - Start by updating the packages list and installing the prerequisites:
        - "**sudo apt update**"
        - "**sudo apt install software-properties-common**"
      - Next, add the deadsnakes PPA to your sources list:
        - "**sudo add-apt-repository ppa:deadsnakes/ppa**"
        - When prompted, press **Enter** to continue.
      - Once the repository is enabled, install Python 3.6 with:
        - "**sudo apt install python3.6**"

At this point, Python 3.6 is installed on your Ubuntu system and ready to be used. You can verify it by typing: "**python3.6 --version**". You will get output as: **Python 3.6.x**.

- If your system is **Ubuntu 18.04**, type the following command to install **Python 3.7**: "**sudo apt install python3.7**"
- If your system is **Ubuntu 18.04**, type the following command to install **Python 3.6**: "**sudo apt install python3.6**"

- You have to set the **Python 3.6** or **Python 3.7** as default interpreter for **python3** command. For the same, follow from **Step 2** of this link: https://www.itsupportwale.com/blog/how-to-upgrade-to-python-3-7-on-ubuntu-18-10/. You can verify by typing "**python3**" on Terminal and check
- Upgrade the **pip** in your system with command:

  "**sudo python3 -m pip install --upgrade pip**"

  Verify the updated version of **pip** installed using command: "**pip -V**".
- Install **PyQt5** using this command:

  "**sudo python3 -m pip install pyqt5**"
- Verify its installation by typing these commands:
  - "**python3**" (Python interpreter will get invoked in the Terminal)
  - "**from PyQt5 import QtCore**"

  If everything has been installed successfully, you won't encounter any error.

NOTE:

- If you have **Python 3.6.x** installed, then you have to **copy and replace** following files from **py3.6_pyc** folder to the folder shown in Figure 1:
  - **general.pyc**
  - **global_variables.pyc**
  - **image_enhancer.pyc**
  - **layout_logic.pyc**
  - **layout.pyc**

  Make sure all these **5 files** are replaced correctly. Now, follow the steps described below to run the GUI.

- But, if you have **Python 3.7.x** installed, then you need not worry; you can continue with running the GUI as described in steps below. Those **5 files** already present are compiled binaries for **Python 3.7**.

Running the GUI:

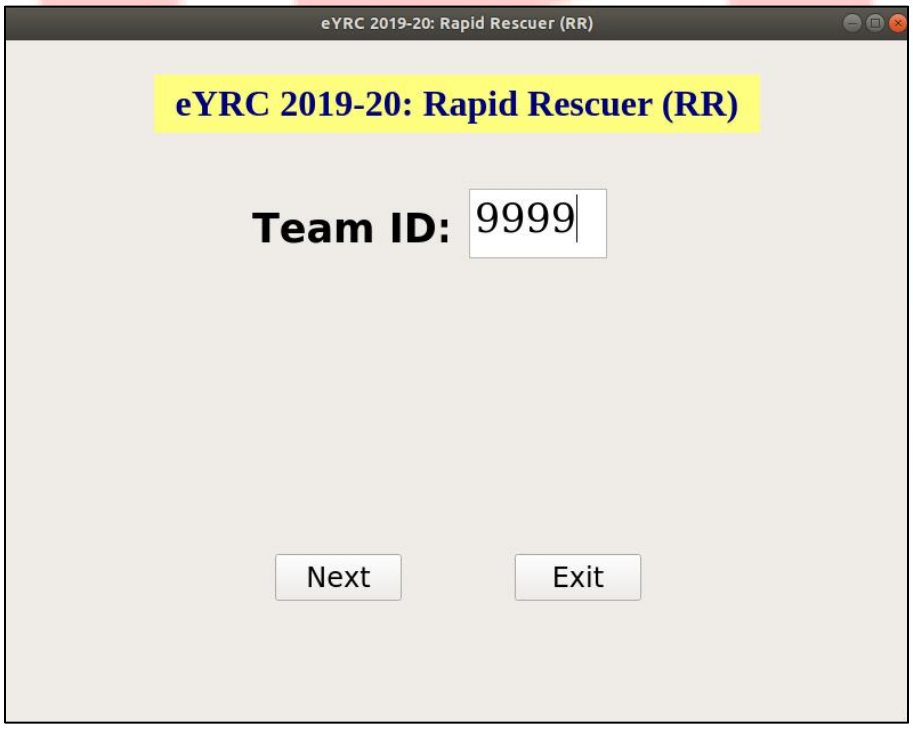
Now, right click on the **GUI_App** folder and Open in Terminal.

Steps to be followed:

- Turn **ON** the Robot and Connect your PC/laptop to the Wireless Access Point of your Robot.
- Run the GUI using the following command inside the Terminal:

  " **sudo python3 Task4_GUI_App.py** "

- Window shown in Figure 2 will appear. Enter the Team ID in the text box provided. Click on **Next** button.

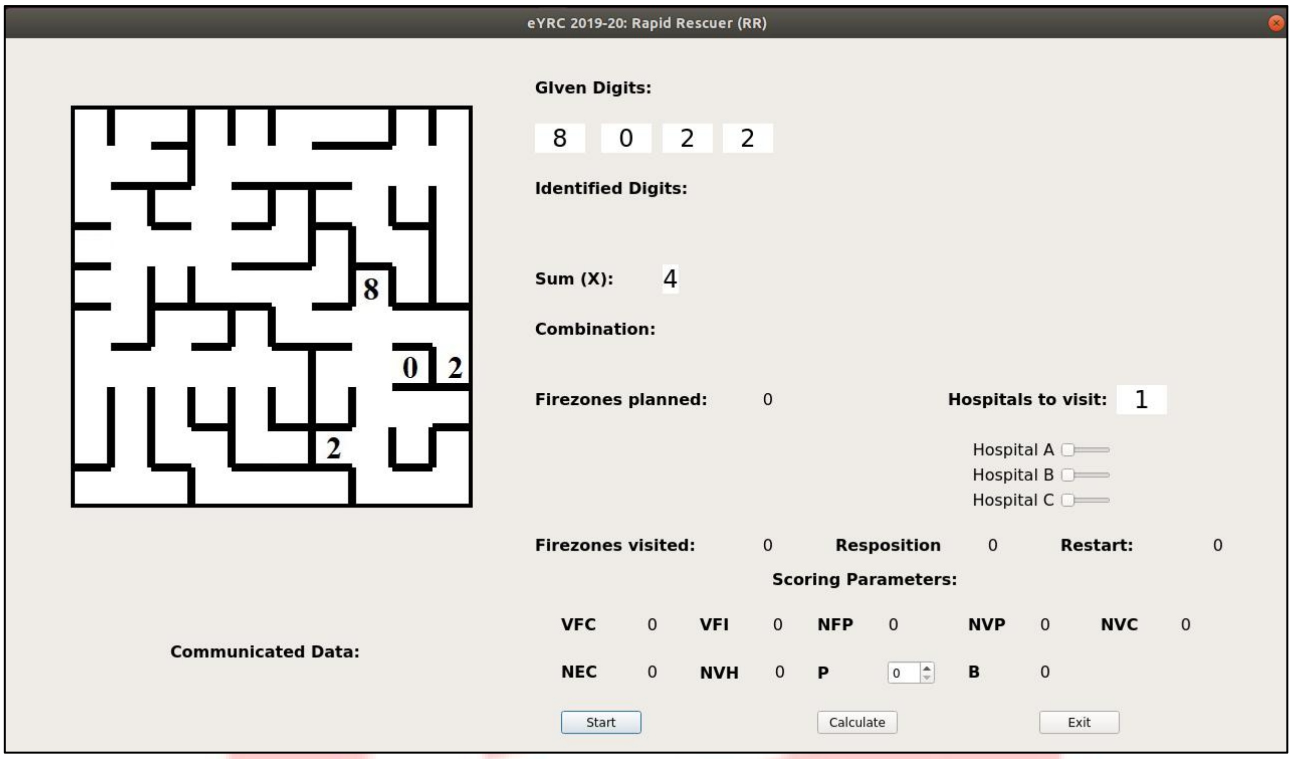

eYRC 2019-20: Rapid Rescuer (RR)

eYRC 2019-20: Rapid Rescuer (RR)

Team ID: 9999

Figure 2: Screen 1

- Window shown in Figure 3 will appear.



Given Digits: 8 0 2 2

Identified Digits:

Sum (X): 4

Combination:

Firezones planned: 0 Hospitals to visit: 1

Hospital A
Hospital B
Hospital C

Firezones visited: 0 Resposition 0 Restart: 0

Scoring Parameters:

VFC 0 VFI 0 NFP 0 NVP 0 NVC 0

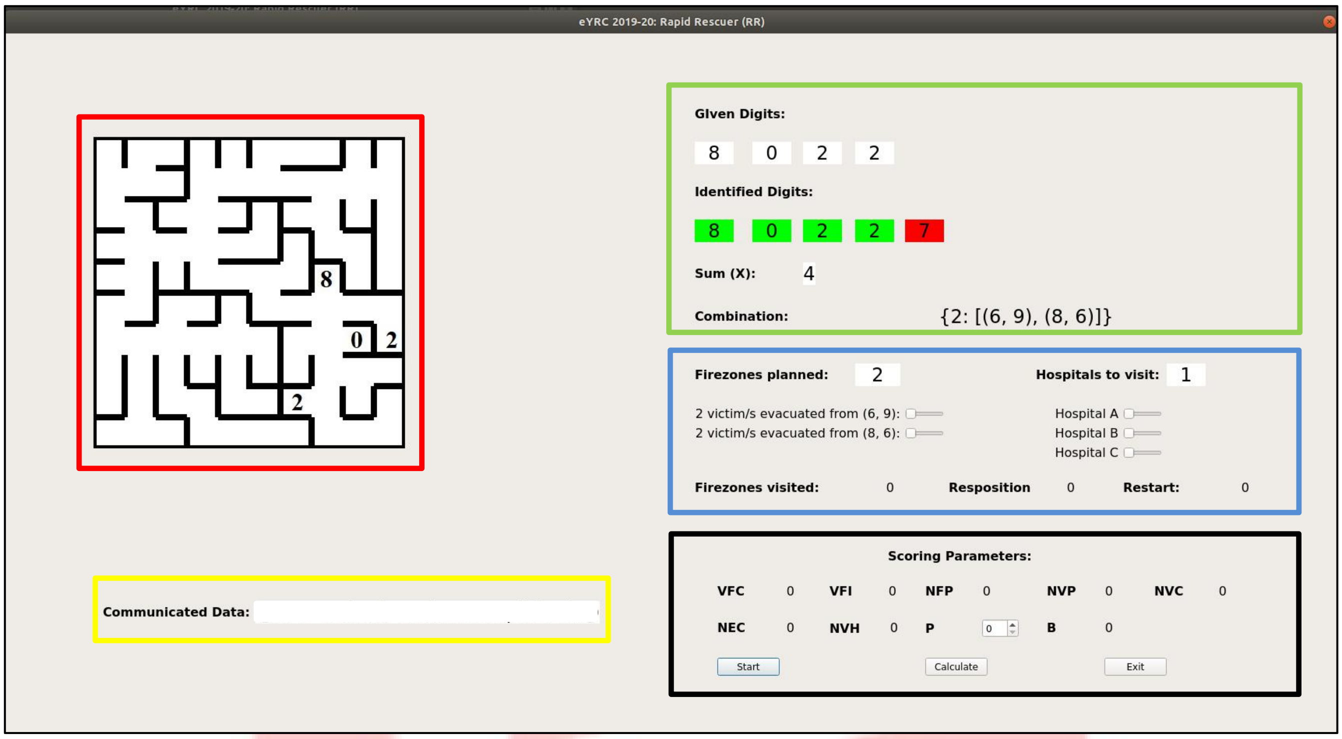
Communicated Data:

NEC 0 NVH 0 P 0 B 0

Figure 3: Screen 2

- Click **Start** button on GUI (present at bottom).
- Run the Python Client script for Task 4: **python task_4.py** (as stated in Problem Statement) in another Terminal. **NOTE:** Make sure that your Conda environment is activated before running **task_4.py**.
- The Python Client script should first communicate the information: **digits_list** and **combination** wirelessly to the robot using socket (refer to Problem_Statement.pdf). At the instant this is communicated, the GUI will update itself.
- You will see the output as shown in Figure 4. Consider Figure 4 as just an example. The output may vary according to your script **task_4.py**.
- When the **USER_SW** is pressed on the robot and **@started@** is received by the Python Client script, the GUI will start the Timer at its backend. This will be considered as the start of a run.

Explanation of sections in Screen 2:


eYRC 2019-20: Rapid Rescuer (RR)

Given Digits:

8 0 2 2

Identified Digits:

🟩 8 🟩 0 🟩 2 🟩 2 🟥 7

Sum (X): 4

Combination: {2: [(6, 9), (8, 6)]}

Firezones planned: 2 **Hospitals to visit:** 1

2 victim/s evacuated from (6, 9):
2 victim/s evacuated from (8, 6):

Hospital A
Hospital B
Hospital C

Firezones visited: 0 **Resposition** 0 **Restart:** 0

Communicated Data:

Scoring Parameters:

VFC 0 **VFI** 0 **NFP** 0 **NVP** 0 **NVC** 0

NEC 0 **NVH** 0 **P** 0 **B** 0

Start Calculate Exit

Figure 4: Start pressed

Red box indicates the arena image (that will vary depending on the given configuration). It shows the all the Firezones and later on communication, position of Obstacle/s and the shortest path sent from Python Client to ESP32's server will be shown.

Green box indicates the following:

Given Digits: Displays all digits present in maze image

Identified Digits: Green colored box indicates correctly identified Digits whereas Red one indicates incorrectly identified Digits or extra identified

Sum (X): Displays total number of vacancies available in Hospital

Combination: Displays combination of digits for sum with their locations

Blue box indicates the following:

Firezones planned: Displays the number of Firezones planned to be visited depending on the combination chosen

Hospitals to visit: Display the number of Hospitals to be visited (1 in this case, ideally 3 as per Theme)

All the above parameters will be displayed **automatically on communication**. But there are few parameters that require **manual entry**.

Following **two** parameters should be manually enabled as explained below:

- Below **Firezones planned** part, **slider/s** are present. Team should move slider towards **right** (as shown in Figure 5) **ONLY** when:

  “ **robot has visited the planned Firezone, faced towards it, glown Red LED for 1sec and communicated appropriate data (as stated in Problem Statement and Rulebook)** ”

  This will repeat for the number of Digits (Firezones) in the chosen combination.

  Depending on the number of Digits in the combination, number of sliders will vary.

  Click on **Calculate** button to update the parameters and observe the changes in the **Scoring Parameters** section.

Similarly,

- Below **Hospitals to visit** part, **three sliders** are present. Team should move slider towards **right** (as shown in Figure 5) **ONLY** when:

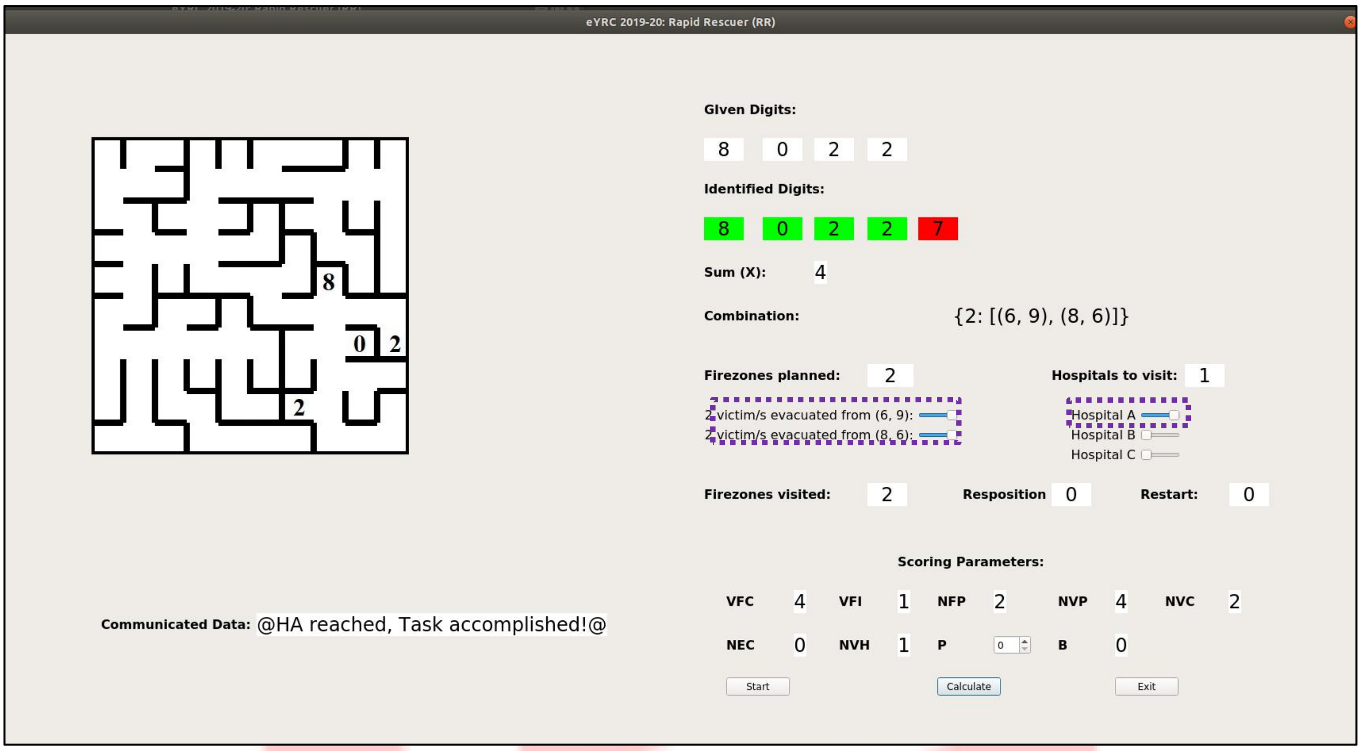
  “ **robot has visited the Hospital, faced towards it, glown Green LED for 1sec and communicated appropriate data (as stated in Problem Statement and Rulebook)** ”

  This will repeat for the number of Hospitals (**1** in this Task 4; **3** as stated in Rulebook).

  Depending on the number of Digits in the combination, number of sliders will vary.

  Click on **Calculate** button to update the parameters and observe the changes in the **Scoring Parameters** section.

- For this Task 4, when **@HA reached, Task accomplished!@** is communicated from the robot to the Python Client script, the GUI will stop the Timer and note down the Total Time of the run at its backend.



Given Digits: 8 0 2 2

Identified Digits: 8 0 2 2 7

Sum (X): 4

Combination: {2: [(6, 9), (8, 6)]}

Firezones planned: 2

Hospitals to visit: 1

2 victim/s evacuated from (6, 9):

2 victim/s evacuated from (8, 6):

Hospital A

Hospital B

Hospital C

Firezones visited: 2 Resposition 0 Restart: 0

Scoring Parameters:

VFC	4	VFI	1	NFP	2	NVP	4	NVC	2
NEC	0	NVH	1	P	0	B	0		

Communicated Data: @HA reached, Task accomplished!@

Figure 5: Sliders enabled

Firezones visited: Displays the count of firezones visited correctly (as planned) as well as incorrectly (extra if any)

Reposition: Explained below

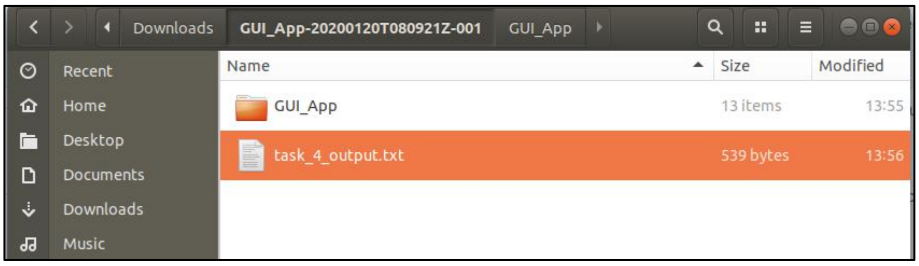
Restart: Explained below

Black box (in Figure 4) indicates the Scoring Parameters as stated in the Rulebook. All the parameters will be automatically updated (only on clicking **Calculate** Button each time) depending on the above scenario except **Penalty** (**P**).

NOTE: P is the **manual entry** to be done by teams. It should be incremented for each object in the arena that the robot **dashes against or displaces** during the run. For **each reposition / restart**, once respective " **& / %** " is given as the input on the Python Client Terminal, **P** as well as **B** will be updated automatically.

Yellow box (in Figure 4) displays all the data communicated from robot to the Python Client.

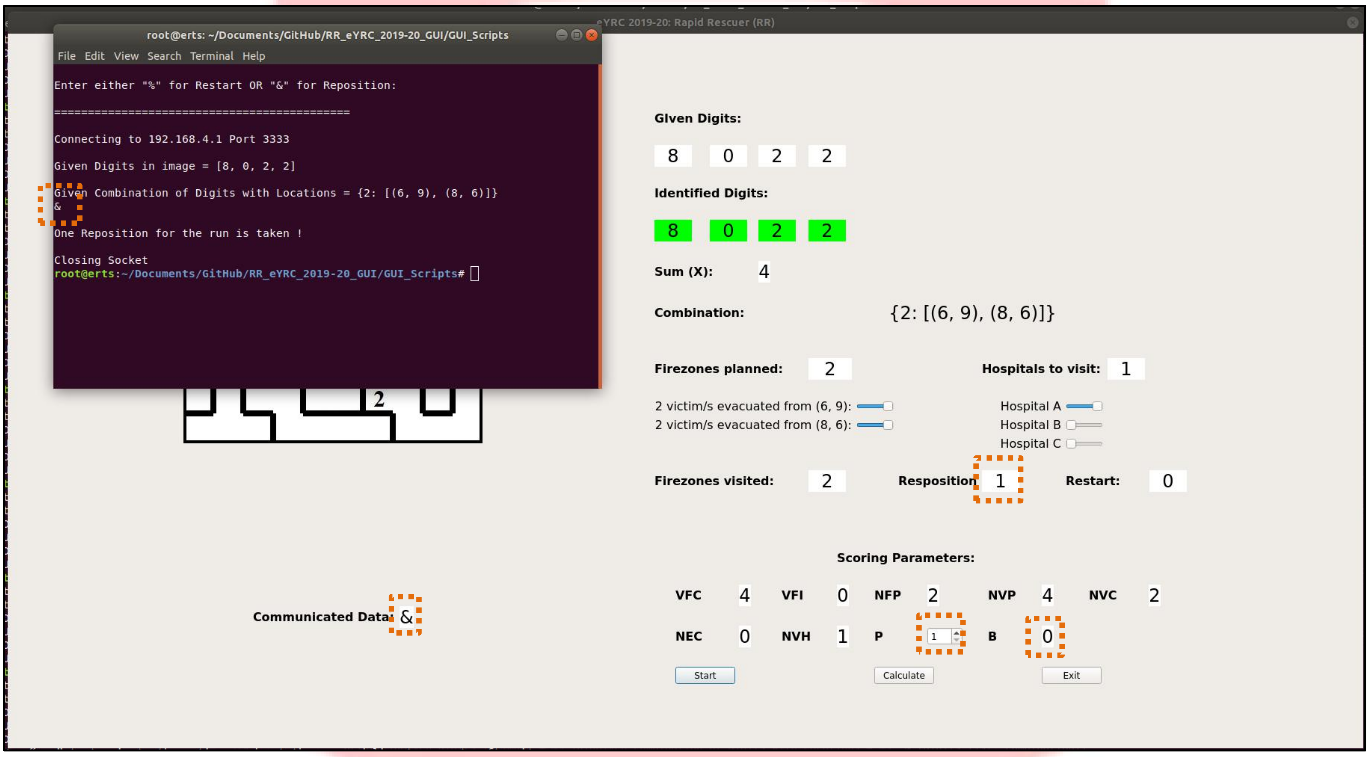
Exit button: Once done with the run and all the parameters are updated on pressing **Calculate** button, press **Exit** button (right bottom corner). This will generate encoded **task_4_output.txt** file with all the parameters including the Total Time of Run and Team ID. It will be saved outside the **GUI_App** directory as shown in Figure 6. **Remember:** only when **Exit** button is used to close the GUI, the **task_4_output.txt** file will be generated with appropriate data. If the GUI is closed by **Alt + F4**, the **task_4_output.txt** file will be generated but with no data.



Name	Size	Modified
GUI_App	13 items	13:55
task_4_output.txt	539 bytes	13:56

Figure 6: task_4_output.txt file generation

Reposition: If team wishes to take Reposition, Team member should enter "**&**" on Python Terminal of **task_4.py** script. The parameter **P** (Penalty) and **B** (Bonus) will automatically update (refer Figure 7).



```
root@erts: ~/Documents/GitHub/RR_eYRC_2019-20_GUI/GUI_Scripts

Enter either "%" for Restart OR "&" for Reposition:

============================================

Connecting to 192.168.4.1 Port 3333

Given Digits in image = [8, 0, 2, 2]

Given Combination of Digits with Locations = {2: [(6, 9), (8, 6)]}
&

One Reposition for the run is taken !

Closing Socket
root@erts:~/Documents/GitHub/RR_eYRC_2019-20_GUI/GUI_Scripts#
```

eYRC 2019-20: Rapid Rescuer (RR)

Given Digits: 8 0 2 2

Identified Digits: 8 0 2 2

Sum (X): 4

Combination: {2: [(6, 9), (8, 6)]}

Firezones planned: 2 Hospitals to visit: 1

2 victim/s evacuated from (6, 9):
2 victim/s evacuated from (8, 6):

Hospital A
Hospital B
Hospital C

Firezones visited: 2 Resposition 1 Restart: 0

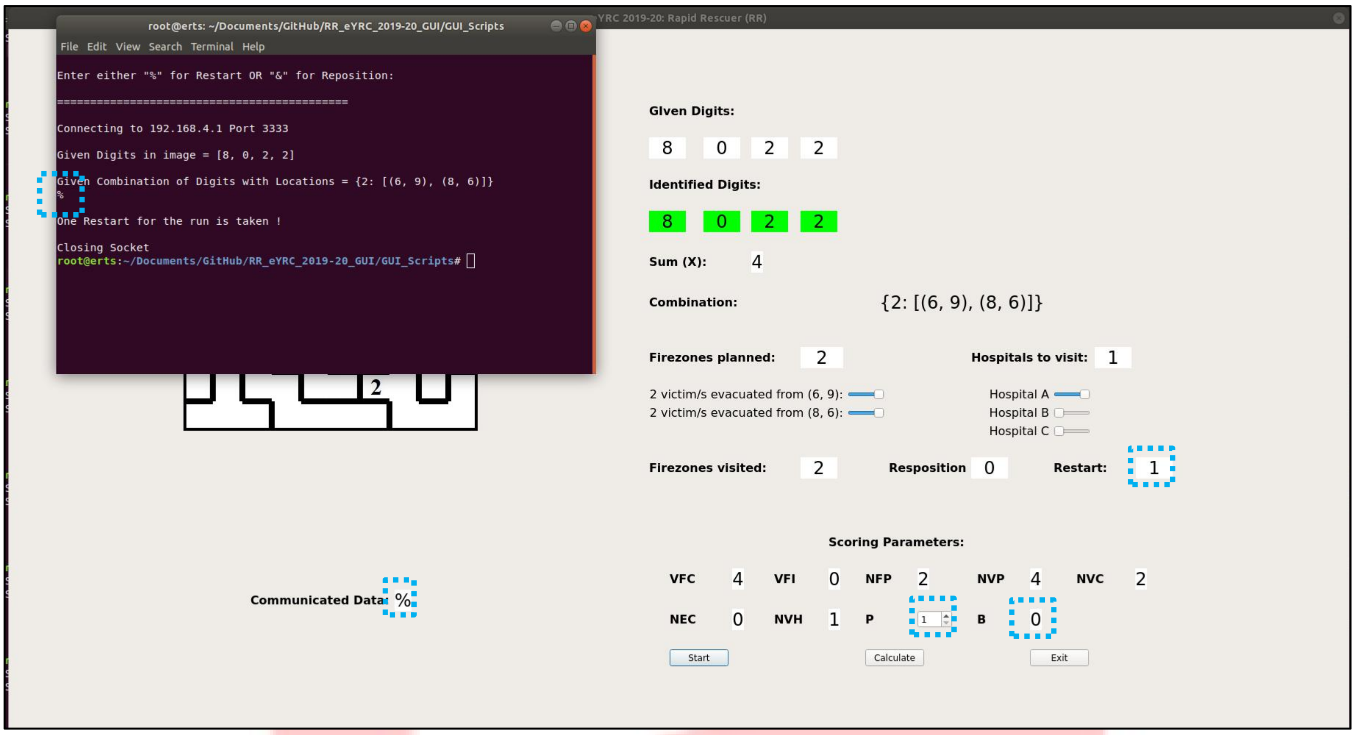
Communicated Data: &

Scoring Parameters:

VFC	4	VFI	0	NFP	2	NVP	4	NVC	2
NEC	0	NVH	1	P	1	B	0		

Figure 7: Reposition

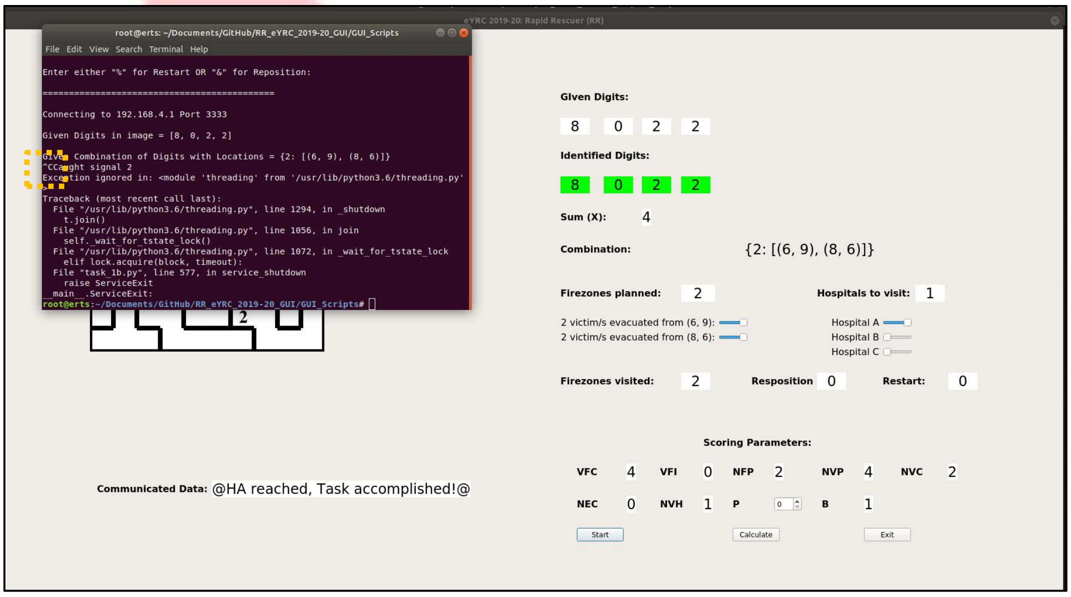
Restart: If team wishes to take Restart, Team member should enter "**%**" on Python Terminal of **task_4.py** script. The parameter **P** (Penalty) and **B** (Bonus) will automatically update (refer Figure 8).


```
root@erts: ~/Documents/GitHub/RR_eYRC_2019-20_GUI/GUI_Scripts
File  Edit  View  Search  Terminal  Help
Enter either "%" for Restart OR "&" for Reposition:

==============================================

Connecting to 192.168.4.1 Port 3333

Given Digits in image = [8, 0, 2, 2]

Given Combination of Digits with Locations = {2: [(6, 9), (8, 6)]}
%

One Restart for the run is taken !

Closing Socket
root@erts:~/Documents/GitHub/RR_eYRC_2019-20_GUI/GUI_Scripts#
```

```
eYRC 2019-20: Rapid Rescuer (RR)

Given Digits:        8  0  2  2
Identified Digits:   8  0  2  2
Sum (X):             4
Combination:         {2: [(6, 9), (8, 6)]}
Firezones planned:   2                 Hospitals to visit:  1
2 victim/s evacuated from (6, 9):      Hospital A
2 victim/s evacuated from (8, 6):      Hospital B
                                       Hospital C
Firezones visited:   2    Resposition  0    Restart:  1

Scoring Parameters:
VFC  4   VFI  0   NFP  2   NVP  4   NVC  2
NEC  0   NVH  1   P    1   B    0

Communicated Data: %
```

Figure 8: Restart

If the Team doesn't want to take any Reposition or Restart, they can enter "**Ctrl+C**" on Python Client Terminal either before pressing **Exit** button or after pressing **Exit** button (refer Figure 9).


```
root@erts: ~/Documents/GitHub/RR_eYRC_2019-20_GUI/GUI_Scripts
File  Edit  View  Search  Terminal  Help
Enter either "%" for Restart OR "&" for Reposition:

==============================================

Connecting to 192.168.4.1 Port 3333

Given Digits in image = [8, 0, 2, 2]

Given Combination of Digits with Locations = {2: [(6, 9), (8, 6)]}
^CCaught signal 2
Exception ignored in: <module 'threading' from '/usr/lib/python3.6/threading.py'
>
Traceback (most recent call last):
  File "/usr/lib/python3.6/threading.py", line 1294, in _shutdown
    t.join()
  File "/usr/lib/python3.6/threading.py", line 1056, in join
    self._wait_for_tstate_lock()
  File "/usr/lib/python3.6/threading.py", line 1072, in _wait_for_tstate_lock
    elif lock.acquire(block, timeout):
  File "task_1b.py", line 577, in service_shutdown
    raise ServiceExit
__main__.ServiceExit:
root@erts:~/Documents/GitHub/RR_eYRC_2019-20_GUI/GUI_Scripts#
```

```
eYRC 2019-20: Rapid Rescuer (RR)

Given Digits:        8  0  2  2
Identified Digits:   8  0  2  2
Sum (X):             4
Combination:         {2: [(6, 9), (8, 6)]}
Firezones planned:   2                 Hospitals to visit:  1
2 victim/s evacuated from (6, 9):      Hospital A
2 victim/s evacuated from (8, 6):      Hospital B
                                       Hospital C
Firezones visited:   2    Resposition  0    Restart:  0

Scoring Parameters:
VFC  4   VFI  0   NFP  2   NVP  4   NVC  2
NEC  0   NVH  1   P    0   B    1

Communicated Data: @HA reached, Task accomplished!@
```

Figure 9: END

Final score will not be displayed on GUI.