Python Host Scanner – (Project2: CGI, Networking, GUI, Database)

For this assignment, you need to use Python 3.x to create a CGI/GUI program (using ttkinter) that can be used for scanning open ports at any IP address. The program design should be object-oriented with **5** classes: Python\_Host\_Scan, Scanning\_Update, Switch\_Result, HostScannerDAL, GUI\_Display. The Scanning\_Update() class must include the following properties:

1. min\_port: the smallest port number to scan, the default value is 0.

2. max\_port: the largest port number to scan, the default value is 1023.

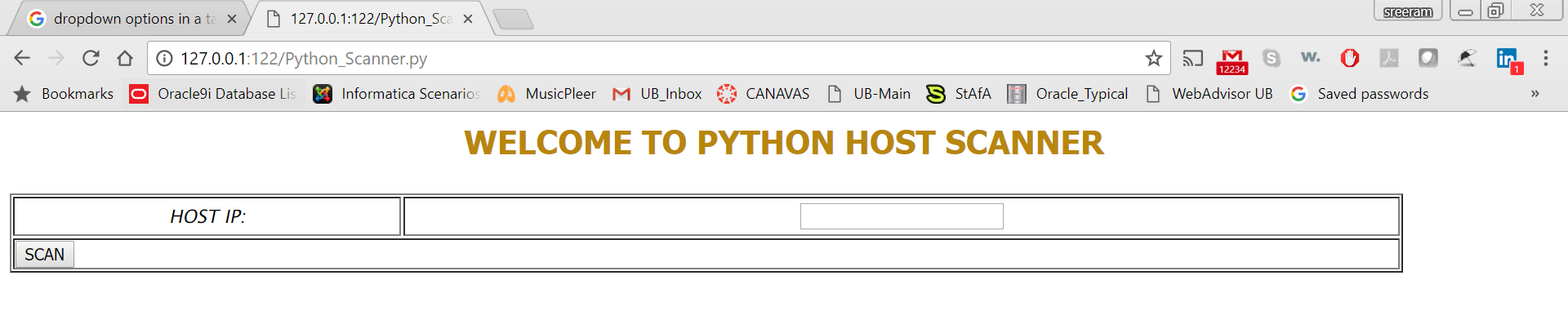
3. IP\_address: this object will change based on the user input in the webform.

4. dal: is an object of the type HostScannerDAL.

And the following Classes-Methods:

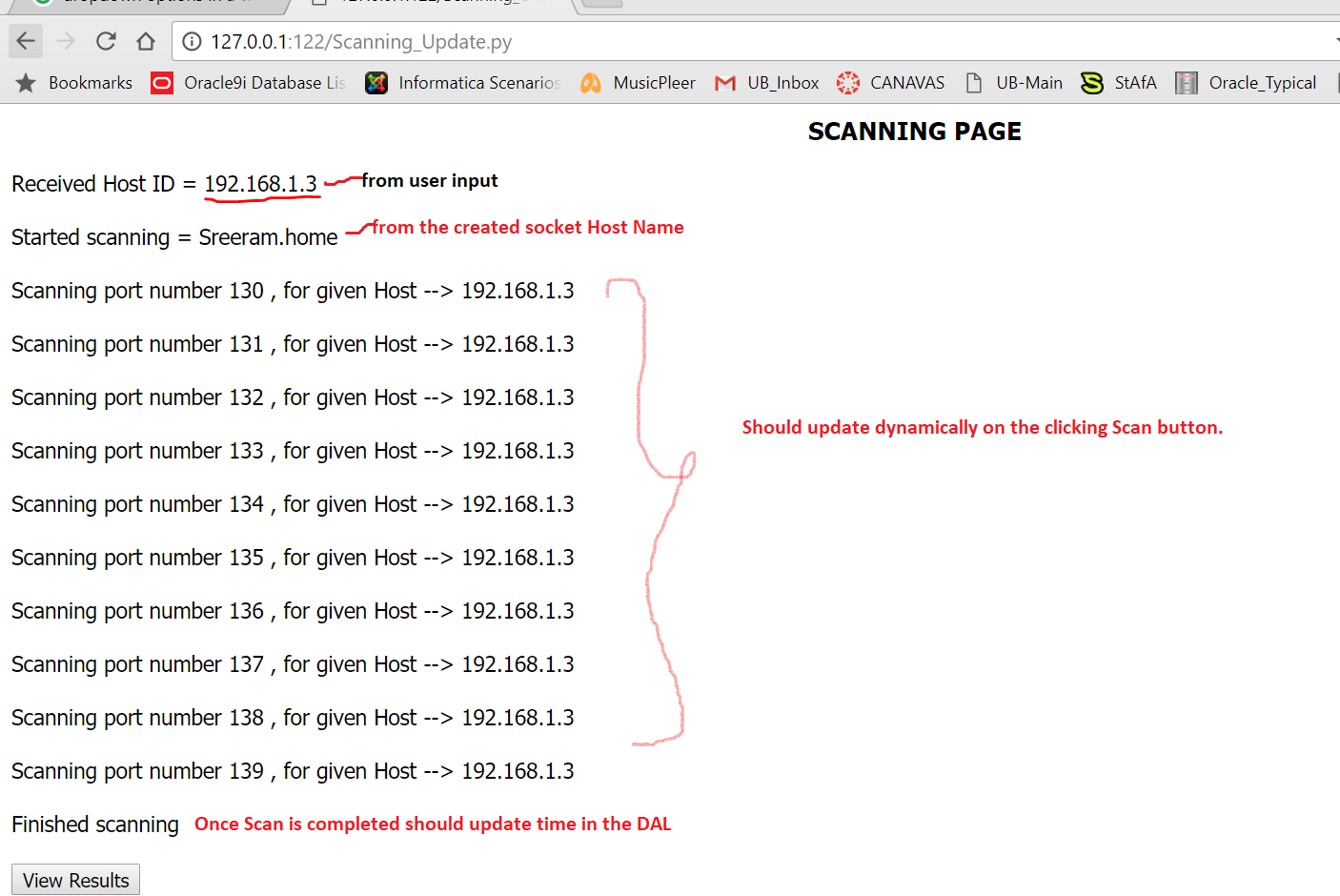
Class **Python\_Host\_Scanner()**:

1. \_\_init\_\_(self): Should Initialize the attributes for the webpage.
2. init\_WebForm(self): Should create the webpage and should take the input ip-address from user and pass value to the “Scanning\_Update.py” on the click of SCAN button.

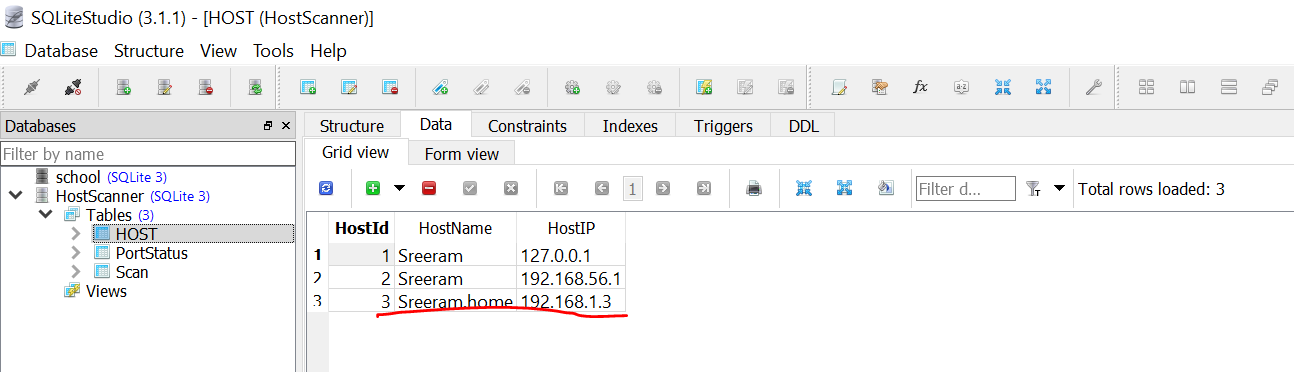


Class **Scanning\_Update**():

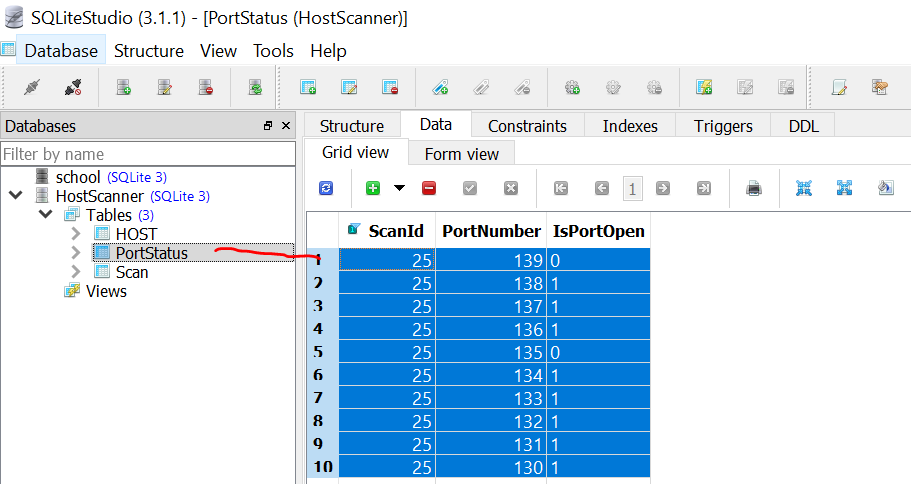
1. \_\_init\_\_(self): Get the Host-IP from the PythonScanner.py using the cgi.Fieldstorage() method and check the HostName for the Host-IP received, using socket.gethostbyaddr() method in socket class
2. update\_host\_name(self): display the HostName on the webpage and call the start\_scan().
3. start\_scan(): Creates an entry for the Host-IP received from user in the Host Scanner-Data Access(**HSD**) Layer, calls the method Create\_Host() of the HSD and the Create\_Scan() method, then displays the scanning port numbers and calls the method scan\_port(min\_port, max\_port)with the maximum, minimum port range.
4. scan\_port(self, min\_port, max\_port): Create a simple socket using python socket module, and then use the method connect\_ex to connect to the machine at IP\_address. Then add the connection result and the PortStatus table by calling the DAL method create\_port\_status.



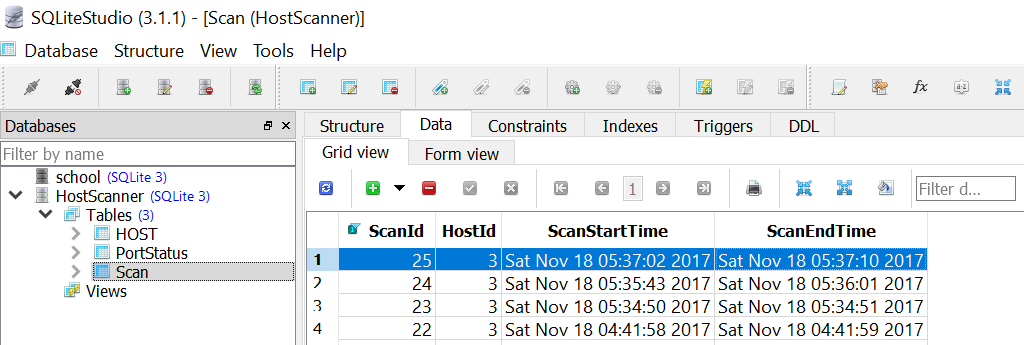
New IP inserted into the table HOST if IP doesn’t exist:



Number of Port’s Scanned are inserted into in the Database with the SCAN ID and the status 1-Close and 0-Open.



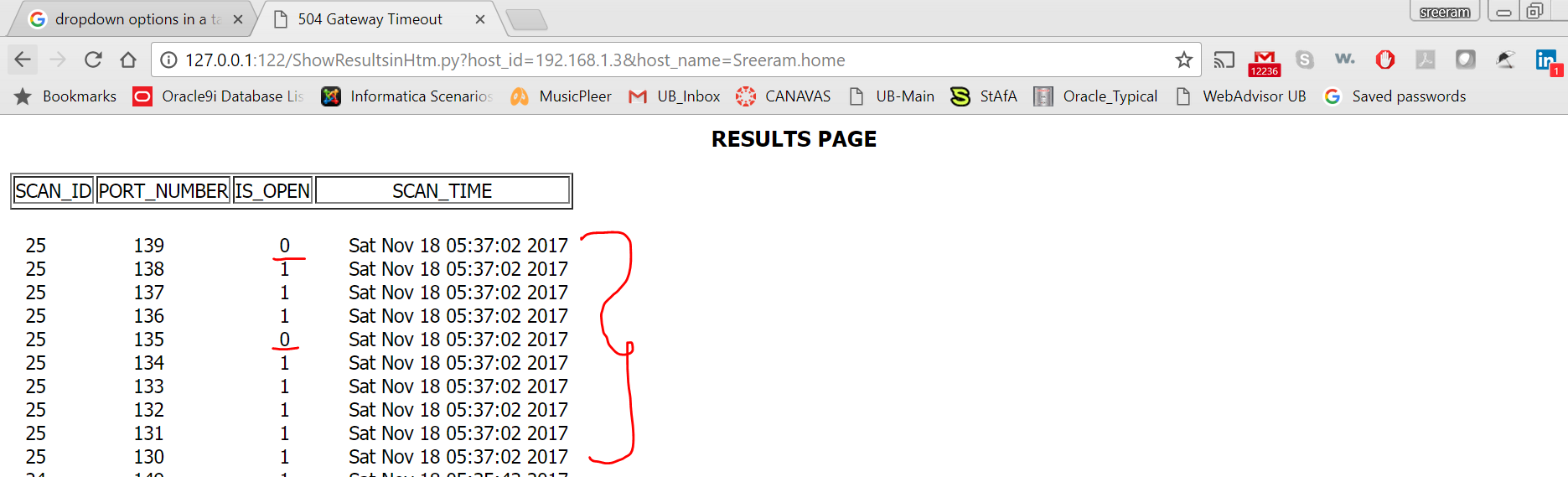
Latest Scan ID start time and end time:

Once after the View results been clicked on the Webpage it needed to be redirected to the “Switch\_Result.py” class.

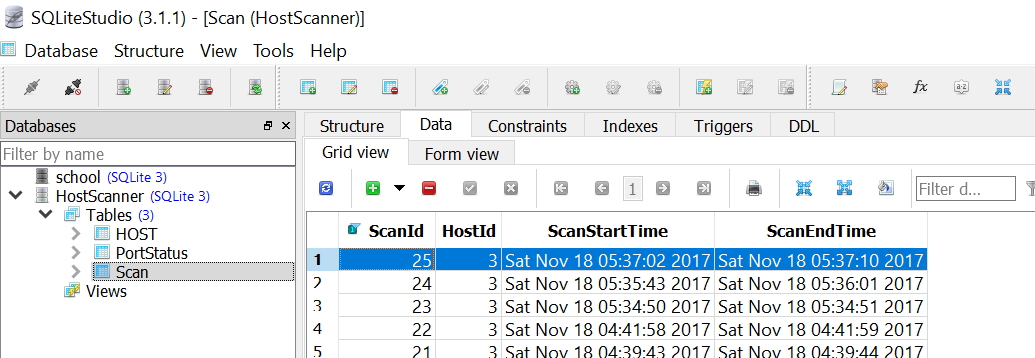
Class **Switch\_Result** ():

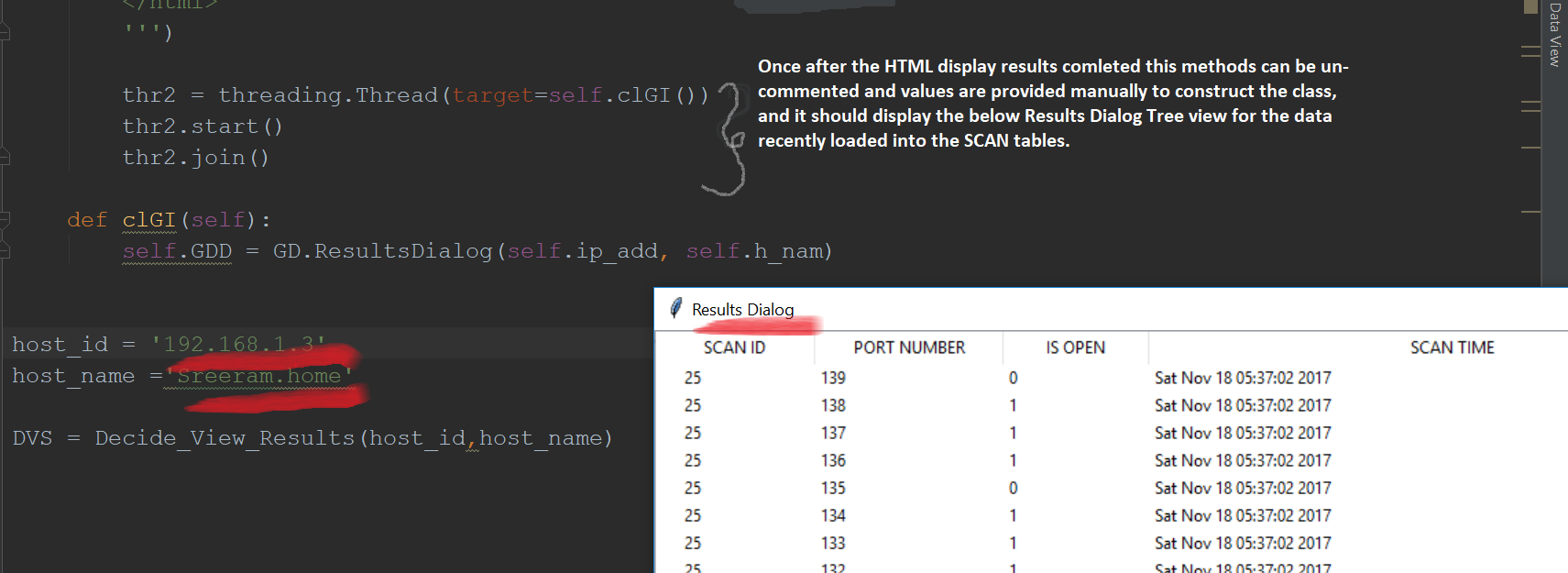
1. def \_\_init\_\_(): Get the values for the HOST-IP, Host name and calls the method Switch\_Results().
2. Switch\_Results(): Creates the SQL query search query and gets the data in form of result-set, displays the data on the Html page and once after the HTML page uploading is completed, it creates a separate thread for displaying data in GUI, by calling the CGUI() method. These methods can be commented-out and should be enabled only after the display of the HTML is completed.
3. CGUI(self): Creates an object entry for the ResultsDialog() class which shows results in form of ttk tree.

Results displayed on the HTML page:



In database:





Class **ReusltsDialog()**:

1. \_\_init\_\_(self, host\_ip, host\_name): should get initialise the base window for the Tree view, for data display, creates the connection for the HSD and start the main loop and call the gui\_init().
2. gui\_init(): should construct the tree-view using the TTKinter module and display results accordingly, by fill the data from database by calling the method \_\_update\_grd\_().
3. \_\_update\_grd\_():Create the local data calling method and populate the data accordingly.

Class **HostScannerDAL()**:

This class should create connection with the database, initialise the variables, create different methods for data retrieval, data insert, data updating for the records which are created newly or already existing.

This class should also dispose the connection on the end. Detail of each method is provided in Skeleton.