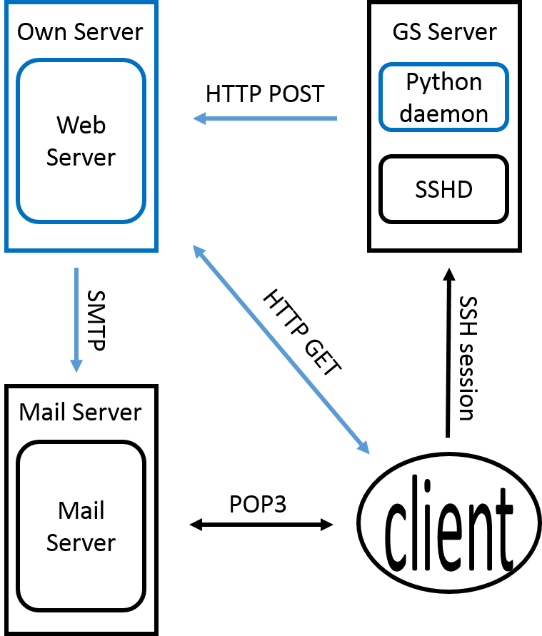
**EC4205 project design //5043손선일 5142이정호**

**Project Name: Conan.py**   
**–codename: Newton’s server**

**How it works?**  


**Python modules used  
subprocess:** invoke external process  
**json:** create, write, manage json files  
**smtplib:** sending mail  
**simpleHTTPServer:** running http server

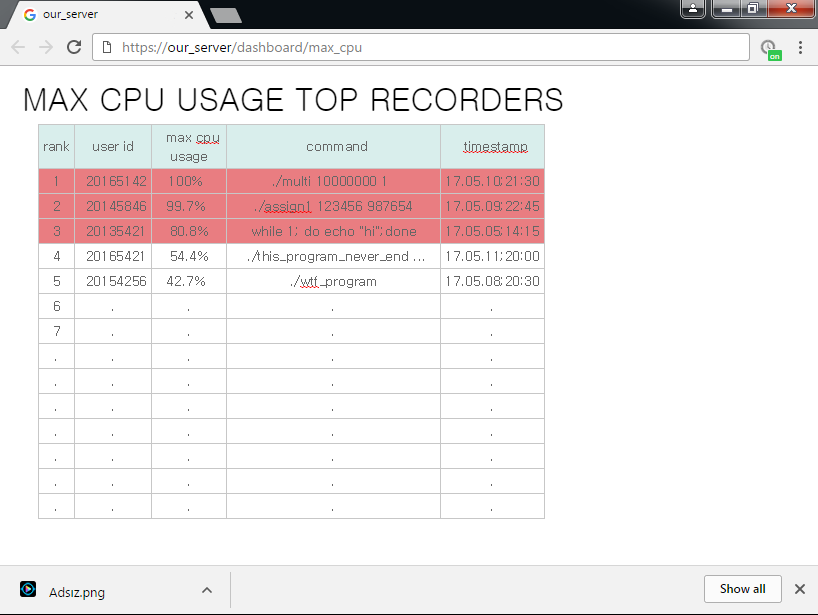
**Components & Details**—modules & functions to be implemented

**1. Python Daemon (Conan.py)—A niced process**

**Json invokeTop()**# Invoke ‘top’ process  
# parse result of ‘top’ into json format  
**Json invokeWho()**  
# Invoke ‘who’ process  
# parse result of ‘who’ into json format  
**Json mergeJson(whoJson, topJson)**# merge two json files which contain parsed result of ‘who’ and ‘top’ if there is a user doing nothing, he/she will be added with blank entries, and system users will be ignored for readability  
**checkHeavyUser(Json)**# if an excessive user exists**,** `write` warning message to the user’s terminal. **saveJson(Json)**# append Json data to maintained file. **sendJson()**# report maintained file to our own server  
# using HTTP POST method  
# once per week (reporting interval surely adjustable)  
**examineSystem()   
#**main() like function for the daemon  
(1)invokeTop() & invokeWho()  
🡪(2)mergeJson() & checkHeavyUser()  
🡪(3)saveJson() & sendJson()   
# called regularly (once per 5 min)

**2. Web Server  
2.A.Daemon Listner:** Listening to HTTP POST request from python daemon on newton(GS) server. On request, server works as follows.  
**updateData()**# update data with received json file  
# load up new json data into memory  
**analyzeData(Json)**  
# sort users according to three legend.  
# Using time, max cpu usage, avg cpu usage  
**generateHTML(Analysis)**# generate dashboard page from analysis.# barely containing title and tables **sendMail(Analysis)**# send mail of congratulations to the top 3 resource bruisers (for each scoring standard in **analyzeData()**)# assume we collects email when registering users

**2.B.Client listener:** Listening to HTTP GET request for dashboard page. HTML file already provided by daemon listener, just send the file to client.

**Appendix Dashboard example:  
JSON format:**Object {  
user:string,  
cpu:float,  
process\_name:string,  
timestamp:string}