# Introduction

Our problem statement for the project was to design and create a distributed OS dashboard. A **dashboard** is defined as (in the context of information systems) "an easy to read, often single page, real-time user interface, showing a graphical presentation of the current status (snapshot) and historical trends of an organization’s or computer appliances key performance indicators to enable instantaneous and informed decisions to be made at a glance.[[1]](#footnote-1)"

We kept this definition of what a dashboard is and what functions it provides when designing our own. Certain realities had to be faced: for example, due to our limited knowledge in UI development, it would be near-impossible to create a user interface that is similar to those available in commercial markets. However, we designed our dashboard with one key component: cross-OS client status retrieval.

We realized that, with this project, our target demographic, in the event of a public release, would be any user with multiple operating systems hoping to gather important statistics regarding their system(s) with nothing but a cursory glance at an application. As such, despite the amount of backend data present, we had to present the user with just the details he or she would consider the most important.

Another important factor to consider is portability. There might be situations where a user is running an intensive task on his or her system when they are physically separated from it. As such, it would become important for them to continuously gather data regarding their system to ensure certain conditions i.e. there is still memory present on a specific drive for installation, or that packets are still being sent through the wireless network adapter to ensure an internet connection is present. In these situations, the most convenient way for the user to view the data they require would be through the use of an application.

# Proposed Implementation

We targeted Windows and Linux operating systems, and aim to create a dashboard that displays information regarding the operating system, the memory and the network. To be more specific, we display the following:

1. Operating System
2. OS Name
3. OS Version Number
4. System Name
5. Memory
6. Physical disks present
7. Total/Free space on each disk
8. Percent usage
9. Network
10. Network Interface Cards present
11. Total bytes sent/received by each NIC

There will be three separate phases within the implementation: the code for the information retrieval from the client system, the establishment of a client-server model and the development of an application for the end-user to operate. We will designate the phases in the aforementioned order as phases A, B and C.

## Phase A

The first phase of our project can be interpreted and run locally. The purpose of this phase is firstly, to create modules that return the information we require and, secondly, to transform this data into a JSON object that can be passed to the server. The libraries to be used are os, platform and psutil.

The following table displays the methods within each library that are used along with their respective functions.

|  |  |
| --- | --- |
| Library  psutil | |
| Method | Purpose |
| psutil.disk\_partitions(all=False) | Returns a list containing all partitions. If the **all** argument is set to ‘False’, only physical drives are displayed and memory partitions are ignored. |
| psutil.disk\_usage(path) | Returns a tuple containing the total, used and free memory space in bytes, followed by the percentage usage. The data is returned only for the device that is defined by the variable **path**. |
| psutil.net\_io\_counter(pernic=False) | Returns a list with data regarding network I/O statistics, such as bytes sent/received, packets sent/received, etc. If **pernic** is set to True, the data is returned as a dictionary. Each NIC is defined as a key with the values stored for the respective NIC. |
| Library  os | |
|  |  |

1. https://en.wikipedia.org/wiki/Dashboard\_%28management\_information\_systems%29 [↑](#footnote-ref-1)