Hardware requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Author | Description | Reason | Version |
|  |  |  |  |  |
| 23/05/2017 | Vinicius | Initialization of report | Draft to be presented as a state report | 1.0 |
| 23/07/2017 | vini | Update information\revision | Further explanation provided for each solution | 1.1 |

This section aims to evaluate the hardware necessary in order to host the application in a local environment. As there are many options available, our goal is to recommend adequate hardware to satisfy the client's needs.

It is crucial to understand that this research was undertaken as an individual component of the project, and it disregards any information about the venue’s current infrastructure such as networking or non-physical solutions, like a cloud-based system.

Any pricing provided was acquired through the research process on (23/05/17), all prices are subject to change.

The research was done based on the requirements of the system, see below:

* Project goal is to create a **real-time** scoreboard application or website.
* Can handle 100 users(judges) updating the database simultaneously.
* Can handle at least 400 users(audience members) or requests simultaneously.
* OS was not specified as client required an open source system. AUT could also provide assistance with software.

*Please see* appendix A for *hardware specifications and quotation and appendix C for definitions.*

Solution 1

In order to host a back-end system for either a web-based-application or for a mobile phone application We recommend the system to be divided in three servers: Reverse Proxy, Application Server and Database server, please see figure 1 for a high-level network draft. For reliability and performance, it is recommended that you separate them. (*See distributed system reference for more info*).

The use of a reverse proxy server would, among many other benefits, distribute the load from incoming requests. It will also protect against common web-based attacks, like DoS or DDoS and malware. Additionally, it would reduce the load on its origin servers by caching static content.

The application server would be responsible for handling requests from the user or proxy server. It would inquire the database for and serve it back to users, as well as handle all processing required by each request.

The database is responsible for managing and storing all data of the application, and serving it back to the application server when requested.

Machine generated alternative text:
RESTful request 
Internet/ 
Response with ISON 
Application 
Figure 1 : 
Senaer Side 
Reverse Proxy server (NGINX Server) 
Application server 
(Tomcat server) 
Database server (Postgresql) 
Discussed framework 

Solution 2

A more compact solution that would also impact on the cost of the system is to take use of virtualization (*See virtualization in reference for more info*). The use of this strategy would allow us to co-populate 2 servers on one host, that is to say that one server would have role of two. In this case the application server and the reverse proxy server will be located in one server. This strategy would eliminate the need of one of the servers. However, some enhancements could be required, such as a higher RAM capacity.

Beyond the cost benefits, using virtualization is a more efficient use of the processing power of a server. Taking into consideration that most of the requests will be **I/O bound** rather than **CPU bound**, the hardware recommended would not have problem responding to the demand. ( *See Request-per-second in reference* )

However, there is consequential reliability on the server in discussion, creating greater risks. For the same reason, we recommend that the database server is a separate system, in simple words, this is where all your data will be stored.

Solution 3

A simpler approach is to have one sever that works as a database and application server, to eliminate the need for a proxy server. This is common practice for small applications or businesses.

This set up could provide a working environment to the application, however, it faces potential issues such as slowness, no response at all, security issues, and a much higher risk of the entire system crashing.

Therefore, we discourage this solution until the criticality of the system is fully assessed. System failure could lead to a major disruption in the competition.