School of Science and Engineering

ITS SYSTEMS STATUS PAGE

Submitted in

Fall 2018

By

Youssef M’rabet

Supervised by

Dr. Hamid Harroud
Capstone Report

I attest that I have applied ethics to the design process and in the selection of the final proposed design. And that I have held the safety of the public to be paramount and have addressed this in the presented design wherever may be applicable.

Youssef M’rabet

Approved by the Supervisor
Acknowledgements

I would like to acknowledge Dr Hamid Harroud for proposing this project to me, as well as for his assistance and availability throughout the course. I have learned a lot under his guidance. I also thank my family and friends for the support they showed me during this learning experience as I got to achieve a lot thanks to them. Finally I would like to thank the volleyball team for showing the utmost support for this semester at AUI.
List of Figures and tables

Table 1: Comparative Study between Cachethq and Statusfy .................................................9

Table 2: Schedule ..................................................................................................................11-12

Table 3: comparative study between ping and HTTP request .................................................17

Figure 1: ITS systems status page architecture .................................................................16

Figure 2: index.js code ........................................................................................................18

Figure 3: variable set up in client.js ....................................................................................19

Figure 4: ping interpretation in client.js .............................................................................20

Figure 5: partial index.html code .......................................................................................21

Figure 6: server pinging IP addresses ...............................................................................21

Figure 7: ITS STATUS PAGE HEADER .............................................................................22

Figure 8: ITS STATUS SERVICES ......................................................................................23
# Table of Contents

1. Abstract .......................................................................................................................... 6  
2. Introduction .................................................................................................................... 6  
3. Steeple Analysis .............................................................................................................. 7  
4. Methodology ................................................................................................................... 8  
   4.1. Opensource Research ............................................................................................... 8  
   4.2. Software Engineering Model ................................................................................... 10  
5. Feasibility Study ............................................................................................................. 10  
   5.1. Feasibility Study and Analysis ................................................................................ 10  
   5.2. Schedule .................................................................................................................. 11  
6. Requirement Specification ............................................................................................... 12  
   6.1. Functional Requirements ......................................................................................... 12  
   6.2. Non-Functional Requirements ................................................................................ 13  
7. Technology Enablers ....................................................................................................... 14  
8. Architecture .................................................................................................................... 16  
9. ICMP echo request vs HTTP request ............................................................................ 17  
10. Implementation ............................................................................................................. 18  
11. Results .......................................................................................................................... 21  
12. CONCLUSION AND FUTURE WORK ...................................................................... 24  
13. References ..................................................................................................................... 25
1. Abstract

The purpose of my Capstone is to analyze, design, and implement an ITS systems status page for AUI. The page will monitor different systems managed by the ITS of Al Akhawayn University. The analysis is mainly going to be focused on monitoring critical services provided by the ITS for a better support and maintenance. Concrete examples of services that can be monitored are the network (within campus either via WIFI or cable, outside campus), availability of printers, number of unoccupied computers in labs, phone services…

This report will show you the different phases that were undertook to implement the web page.

2. Introduction

Nowadays, the use of application is becoming unavoidable in helping with the efficiency of different manual tasks. Moreover, IT solutions are even more crucial in educational systems. It would be inacceptable for these IT services to be down without being notified. That is why the ITS SYSTEMS STATUS PAGE is a web application that allows any client via a web browser, to check the status of different services provided by the ITS. Previously that information was only available to administrators. Thanks to a webpage available to everyone. People won’t have to rely on calling the ITS or checking is a service is down it will. The website will permanently provide information on a list of critical statuses such as the portal, webmail, auiConnect, Aui Guest, phone services, administrative services.
3. Steeple Analysis

The STEEPLE analysis was very important in the sense that it gave me a unique insight in understanding the macro-environmental factors that my capstone project affects as well as how it affects them.

- **Socio-Cultural**

The ITS systems status page will have a positive impact on the AUI community as it offers students, faculty and staff a resource that is accessible 24/7 to check if a system is down. It also helps reduce the report of the system that is down allowing the ITS to fix the problem much faster.

- **Technological**

This web application allows clients to always know the status of a set of statuses offered by the ITS. The web application uses existing technologies such opensource frameworks for the back and front end.

- **Economic**

This web application has a good economic potential as this service is provided in many institutions. These institutions usually pay monthly fees for services like this one.

- **Environmental**

This application doesn’t have any Environmental impact.

- **Political**

Because of the nature of this web application, it doesn’t have any political impact as well.
• Legal

Free opensource frameworks, libraries and node package managers were used in the development of this web application.

• Ethical

This web application helps the community it serves by spreading meaningful information on a timed schedule making it very effective.

4. Methodology

4.1. Opensource Research

The purpose behind this research was to not reinvent the wheel and try to use existing resources to implement my web applications. This research led me to use new technologies that I wasn’t necessary familiar with. At first, I looked at installation guide for ready to use status page. Only to discover that these guides require a very good understanding of the technologies that were used to develop the concerned application. The two main opensource status page that I tried to work on were Cachethq.io and Statusfy.

The following table is a comparative study on the differences between the two opensource pages.

<table>
<thead>
<tr>
<th>comparator</th>
<th>Cachethq.io</th>
<th>Statusfy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framework</td>
<td>Laravel</td>
<td>Node.js</td>
</tr>
<tr>
<td>database</td>
<td>MySql</td>
<td>No database</td>
</tr>
<tr>
<td>caching</td>
<td>Redis</td>
<td>No caching database</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Server side</th>
<th>PHP</th>
<th>Serverless</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client side</td>
<td>Vuejs</td>
<td>Vuejs</td>
</tr>
<tr>
<td>Incident handling</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Push-based notification</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Pull-based notification</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Configuration &amp; Deployment</td>
<td>Limited</td>
<td>Compatible</td>
</tr>
<tr>
<td>Npm Compatibility</td>
<td>No</td>
<td>yes</td>
</tr>
<tr>
<td>Extensibility</td>
<td>Not extensible</td>
<td>Extensible</td>
</tr>
</tbody>
</table>

Table 1: Comparative Study between Cachethq and Statusfy

The conclusion I got from this comparative study is that both opensource use very interesting technologies. However, given the time constraint the configuration of cachethq was just not possible because of all the technologies that I needed to be acquainted with. I did not even manage to load a page in Cachet after spending quite some time on it.

On the other Statusfy was very easy to configure and install. The only technological constraint was to learn what was the node.js framework and all the subsequent services it provides. After loading the page, I discovered that Statusfy only had the least critical functional requirements I needed.

Although these two resources weren’t what I was expecting to get. They helped me understand better the architecture needed to get my website going. So instead of using opensource installation pages I decided to use open source frameworks as I already got to use them and find the best way to make it all work. This leads me to my next point which is the software engineering model I was going to adopt.
4.2. Software Engineering Model

As mentioned in the section above, I got to manipulate critical technologies such as javascript frameworks and node.js. Thanks to the Opensource research I decided to use the prototyping model.

In fact, It consists of developing a prototype with different technologies to have a web application that is as close as possible to the final system. Then we keep on reworking and trying new modules until the prototype is acceptable.

5. Feasibility Study

5.1. Feasibility Study and Analysis

The main objective of this project is to develop a responsive design web application to be used under different clients, platforms, and locations.

The web application is going to provide knowledge about the status of the IT systems provided to its community. These statuses will be categorized as follows: The system is UP and therefore operational. The system is DOWN and therefore non-operational. The system is DEGRADED means that it is up however it is not working properly (internet being slow for instance).

Critical Systems to be monitored include mail, phones, portal, network, printers, lab availability, point of sales, online payment, Turnitin etc…

There must be an announcement section next to the status list that provides an explanation related to non-operational systems.

The last feature the website will provide is subscriptions for a different type of notification system. In fact, by default the website provide a pull-based notification method where the user must enter the website and look for the data he is looking for. However, thanks to a subscription,
the user chooses a push-based notification method which allows the website to send notifications to the user via mail or SMS whenever a system is non-operational.

Many Institutions all over the world offer a “check status” feature in their official website. Therefore, an Initial analysis and a comparative study will have to be made to determine the most appropriate approach for such project.

Because of the nature of the web application I will be implementing, it will require the use of different tools and platforms related to web technologies such as JavaScript frameworks, node.js.

Based on this initial description, this project is feasible within the allocated time. Since this web application is going to improve the daily life students, staff and faculty, as they will have crucial information on the status of many services. Therefore, it is expected go through a test phase in order it to be deployed starting the following semester.

5.2. Schedule

According to the following Schedule, the project is deemed feasible.

<table>
<thead>
<tr>
<th>Task</th>
<th>Deadline</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Specification</td>
<td>September 14\textsuperscript{th}</td>
<td>3</td>
</tr>
<tr>
<td>Feasibility Study and Analysis</td>
<td>September 21\textsuperscript{st}</td>
<td>7</td>
</tr>
<tr>
<td>Diary 1</td>
<td>September 24\textsuperscript{th}</td>
<td>3</td>
</tr>
<tr>
<td>Diary 2</td>
<td>October 1\textsuperscript{st}</td>
<td>7</td>
</tr>
<tr>
<td>Diary 3</td>
<td>October 8\textsuperscript{th}</td>
<td>7</td>
</tr>
<tr>
<td>Diary 4</td>
<td>October 15\textsuperscript{th}</td>
<td>7</td>
</tr>
<tr>
<td>Interim Report</td>
<td>October 19th</td>
<td>4</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
<td>---</td>
</tr>
<tr>
<td>Diary 5</td>
<td>October 22nd</td>
<td>3</td>
</tr>
<tr>
<td>Diary 6</td>
<td>October 29th</td>
<td>7</td>
</tr>
<tr>
<td>Diary 7</td>
<td>November 5th</td>
<td>7</td>
</tr>
<tr>
<td>Diary 8</td>
<td>November 12th</td>
<td>7</td>
</tr>
<tr>
<td>Diary 9</td>
<td>November 19th</td>
<td>7</td>
</tr>
<tr>
<td>Final Report</td>
<td>November 26th</td>
<td>7</td>
</tr>
<tr>
<td>Updated Final Report</td>
<td>December 12th</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 2: Schedule

6. Requirement Specification

6.1. Functional Requirements

- The web application shall display dynamically the status of systems.
- The web application shall display statuses by 3 categories (up, down or degraded).
- The web application shall display the round-trip time of ping requests.
- The web application shall display the last time a system was monitored.
- The web application shall be able to report incident messages.
- A client may subscribe to receive a notification when a service becomes down.
6.2. Non-Functional Requirements

• Distribution Requirements

The application shall be distributed on the internet as well as the intranet for it to properly monitor all AUI’s internal systems.

• Security Requirements

The application shall be able to preserve the security of the targeted server during the monitoring process.

• Confidentiality Requirements

The application shall maintain the confidentiality of the monitored IP addresses.

• Integrity Requirements

Clients shall be assured they are connected to the right application.

• Availability Requirements

The application shall be accessible when needed, with no single point of failure.

• Access Requirements

The application shall be accessible from any web browser.

• Extensibility Requirements

Because the project is based on opensource modules. Extensibility is easily met when it comes to increasing the functionalities of the application.
• Performance Requirements

The application shall provide low pinging intervals for systems.

The application shall be able to monitor a high number of systems at once.

• Scalability Requirements

The application shall be able to maintain its performance when the number of servers being monitored increases.

7. Technology Enablers

Node.js is an open source server environment. The latter uses JavaScript on the back-end. It is very useful in the context of my project as it can generate dynamic page content, create/open/delete/read/write, close files on the server. Note that it can also manipulate data in a database.

Express.js is a web application framework that runs on top of Node.js. It is useful server-side
Vue.js is a JavaScript framework that helps build user interfaces. It is useful client-side.

Socket.io is a JavaScript library for Realtime web applications. It enables communication between web clients and servers.

Bootstrap is an opensource tool that helps development with HTML, CSS and JS as it offers templates for buttons, tables etc...
HTML was used to create the web application as it is the standard Markup Language for implementing web applications.

8. Architecture

Figure 1: ITS systems status page architecture

The architecture of the application is as follows:

- It is composed of 2 tiers: A client side and a server side.
- The server side can ping different server addresses and communicate with different clients.
- The server side is implemented using express.js which is built on top of node.js
• The client side uses VueJS to build the user interface and bootstrap for the design templates in CSS and html.
• Both tier communicate via HTTP and Socket.io.
• Socket.io has a client-side library that runs inside the browser and a server-side library for node.js

9. ICMP echo request vs HTTP request

The next step was to figure out the more suited request for the web application. ICMP echo requests and HTTP requests work differently and serve different purposes. That is why a comparative study was made to determine the most efficient method to use.

<table>
<thead>
<tr>
<th></th>
<th>PING</th>
<th>HTTP REQUEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTOCOL</td>
<td>ICMP</td>
<td>HTTP</td>
</tr>
<tr>
<td>PROCESS</td>
<td>- A packet is sent</td>
<td>- Request is made to a specific domain</td>
</tr>
<tr>
<td></td>
<td>- The server replies with the packet that was asked for</td>
<td>- That request is then processed, executed and rendered</td>
</tr>
</tbody>
</table>

Table 3: comparative study between ping and HTTP request

According to the table the ICMP is straightforward and presents not as much overhead as opposed to HTTP requests. That is why I opted for the first option.
10. **Implementation**

As presented in the previous section. The architecture of this project is a 2-tier architecture. The logic is located 3 main files. Index.js which contains the logic for the server side or back end. Client.JS which contains logic for the client side or front end. Finally, Index.html which is an html file that supports bootstrap for the look and socket.io for the dynamic communication between the 2 tiers.

![Image of code](image)

**Figure 2: index.js code**

This screenshot shows the code contained in the server side. It basically pings periodically a list of IP addresses then updates them to the client side via socket.io.
Figure 3: variable set up in client.js

This screenshot illustrates the variable setup in the client side. We can manually add different types of services under different types of category. Each service contains an IP address, a name, an initial round trip time set to 0, a status initially set to secondary which is the equivalent of unknown status (colored in grey) before the first ping is issued, and a date which is later filled automatically to accommodate the last time a service was monitored.
```javascript
aui_network.forEach(function (e) {
    if (e.ip == data.ip) {
        e.rtt = data.rtt
        e.date = new Date().toLocaleString()
        if (e.rtt > 3000) {
            e.status = 'danger'
            vm.up = false
        } else if (e.rtt <= 150) e.status = 'success'
        else if (e.rtt > 150 && e.rtt <= 3000) {
            e.status = 'warning'
            vm.up = false
        } else e.status = 'secondary'
    }
})

wireless_network.forEach(function (e) {
    if (e.ip == data.ip) {
        e.rtt = data.rtt
        e.date = new Date().toLocaleString()
        if (e.rtt > 3000) {
            e.status = 'danger'
            vm.up = false
        } else if (e.rtt <= 150) e.status = 'success'
        else if (e.rtt > 150 && e.rtt <= 3000) {
            e.status = 'warning'
            vm.up = false
        } else e.status = 'secondary'
    }
})
```

**Figure 4: ping interpretation in client.js**

This screenshot illustrates the interpretation of the ping requests under client.js. Two variables are updated for each type of category of variable. The round-trip time on the one hand. In fact, depending on its value, a different status is generated. Danger corresponds to the “red down”, warning corresponds to “yellow degraded” and success corresponds to “green up”. On the other hand, the variable date is also updated to give the last time the status was pinged and processed.
This snapshot shows some partial html code from the index.html file. We can see here the different status types for the AUI Network services category.

11. Results

Figure 6: server pinging IP addresses
The above screenshot shows the command prompt with the query to start the server. Once index.js is launched, it automatically pings the list of ip addresses present in your page.

Figure 7: ITS STATUS PAGE HEADER

The webpage contains a header containing the title of the page, an AUI logo on the left and a subscribe to notification button on the left. Below the header is help section explaining the purpose of the page as well as contact information. In addition to a local notification section. This section is either green if all services are operational, yellow if not all of them are operational or red if all the services are down. It also contains a “last updated” feature which gives the last time the services were monitored.
Figure 8: ITS STATUS SERVICES

The webpage contains drop down sections which contains the different services sorted by categories. As explained earlier in the implementation. A service is either “green UP “, “yellow Degraded” or “red Down”.

<table>
<thead>
<tr>
<th>Academic Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jenzabar portal</td>
</tr>
<tr>
<td>Turnitin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Administrative IT services</th>
</tr>
</thead>
<tbody>
<tr>
<td>mynag.aui.ma</td>
</tr>
<tr>
<td>Sage Suite Financière</td>
</tr>
<tr>
<td>Points of sales</td>
</tr>
<tr>
<td>Agirh</td>
</tr>
<tr>
<td>Online Payment</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IT Support Services</th>
</tr>
</thead>
</table>

23
12. CONCLUSION AND FUTURE WORK

Thanks to this project I was able to learn so many new technologies. Most of them are critical in the job market. The time constraint didn’t allow me to implement every functionality we discussed during the early stages of the project although the most important one is completed.

During the next few days, I will be working on some of the functional requirements:

- Implement the logic behind the subscribe. The latter button will prompt for a client email and will send him notification email when a system stops working.
- Add an incident handling section that can affect the status of systems such as phone services or scheduled maintenance on other services.
13. References


Chat. (2018, October 31). Retrieved from https://socket.io/get-started/chat?fbclid=IwAR0wDZO9AFIrEZ30T7oN2wQyeNbAkYI4ZdaEMKbk1Dc1yegdZjv9sZJKWrw#Introduction


