6-22-2020

Use of Internet of Things for Remote Laboratory Settings

Abul K.M. Azad
Northern Illinois University

Use of Internet of Things for Remote Laboratory Settings

Prof. Abul K. M. Azad, Northern Illinois University

Abul K. M. Azad is a Professor and Associate Dean with the College of Engineering and Engineering at Northern Illinois University, US. He has been in academics 30+ years, and his research interests include remote laboratories, mechatronic systems, mobile robotics, and educational research. In these areas, Dr. Azad has over 130 refereed journal and conference papers as well as 5 edited books. So far, he has attracted around $2.6M of research and development grants from various national and international funding agencies. He is a member of the editorial board for a number of professional journals as well as an Editor-in-Chief of the International Journal of Online Engineering. Dr. Azad is active with remote laboratory field and is the President of the Global Online Laboratory Consortium (GOLC) as well as the Vice-President of the International Association of Online Engineering (IAOE). Dr. Azad is also active with few other professional organizations like- IEEE, IET, ASEE, ISA, and CLAWAR Association and served as Chair and Co-Chairs of numerous conferences and workshops. He was a program evaluator for the ABET and is active in evaluating research and development projects for various national and international funding agencies in US, Europe, and Australia.

Use of Internet of Things for Remote Laboratory Settings

1. Introduction

It is vital to provide laboratory activities to maximize learning in STEM disciplines. Traditionally, students perform experiments by being present in a laboratory and working with physical systems. However, when considering the financial involvement, manageability, and accessibility, this arrangement is not always effective. Traditional laboratory classes are scheduled only for a set time period. Given the mixed ability level of students, the allocated time is often not enough for all students to complete their tasks effectively and also achieve intended outcome [1, 2]. Sometimes students like to perform additional experiments beyond their assigned tasks. It is usually difficult to accommodate any extra time due to the lack of available resources to keep the laboratories open. Additionally, laboratory facilities are often inaccessible to the students of other departments within the same institution because of their geographical location. At the same time too much laboratory equipment lies idle during most of its usable lifetime. Only a remote experimentation facility can provide cost effective and unlimited access to experiments and maximize the utilization of available resources [3, 4, 5]. Moreover, this will allow inter-laboratory collaboration among universities and research centers by providing research and student groups access to a wide collection of expensive experimental resources at geographically distant locations.

One of the major limitations of the existing distance-learning courses is their failure to deliver the laboratory courses [6, 7]. While simulation and multimedia provide a good learning experience, for effective and complete learning, especially in STEM programs, a mixture of theoretical and laboratory sessions are needed. Currently, students from distance learning programs have to visit a campus to perform the laboratory sessions for a limited period of time [8, 9], which is usually insufficient to allow them to complete their learning cycle [1, 10]. Making the remote experimentation facility accessible through the Internet would address this problem. Given the need for an Internet-based remote experimentation and emergence of Internet technologies there is a number of initiatives to develop remote experimentation facilities [11, 12, 13, 14, 15, 16, 17, 18].

Academics and researchers have been working on design and developing remote experimentation systems for a considerable period of time and are making remarkable breakthroughs utilizing cutting edge technologies and current understanding of educational and learning strategies. After all of these developments, the popularity of remote experimentations is still very limited and does not have the attention of the academic community to incorporate them as a part of their regular curriculum. There are a number of factors that hinder acceptance of remote laboratories as a part of a curriculum. These are: Integration of a number of disciplines into remote experimentation design; Modularity in designs; Readily available commercial products; Integration of learning management system; Maintenance and training; Administrative awareness and support; and Industry applications. The first part of the paper will discuss these issues and will highlight how we can move forward in a coordinated manner so there will be a viable remote experimentation infrastructure with a high degree of acceptance.

Historically remote laboratories are developed by utilizing personal computers or workstations as the main controller unit on the experiment side and a local server for database and user management. However, given the emergence of new technologies, embedded processors are becoming more powerful, faster, and resourceful. It is now possible to replace personal computers or workstation with embedded processor boards.

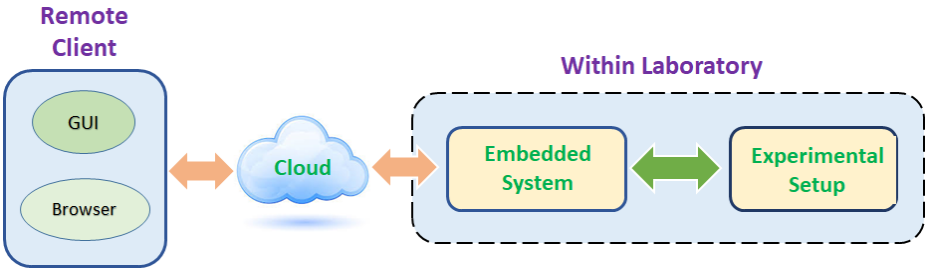

Figure 1: Concept diagram of a remote testbed

The author has developed a number of remote experimentation facilities utilizing IoT infrastructure. To explore technologies involved in the development, Figure 1 shows a conceptual structure of a remote testbed. The main components can be identified as the experimental setup, local computer/server, Internet cloud, and remote clients. The experimental system is connected with a local computer/server, which plays the role of a gateway between the experiment and the remote computer for clients. There should be some middleware that facilitates the information exchange between the local and remote computers.

These facilities utilized embedded processors for accessing multiple experiments, manipulating experimental setup from remote locations, integrated assessment, and real-time learning management features. The developed facilities are used for delivering a number of laboratory courses, while gathering data in terms of achieving learning outcomes and assessing the effectiveness of the system in terms of system designs. The paper will describe the development and implementation of remote laboratory systems in terms of design philosophy, system design implementation, pedagogical design, and evaluation outcome. The paper will be concluded with a discussion considering on what was presented.

2. State of Remote Experimentation

This section provides a discussion involving the current state of remote experimentation. The main focus is to identify the shortcoming and stumbling blocks towards the progress of remote experimentations. This is followed by an outline of potential strategies to address those issues.

2.1 Limitations

Performing physical laboratory experiments over the Internet is a relatively new concept. Researchers are pursuing this problem in an abrupt manner and are not yet coming up with a sustainable solution that can popularize the use of remote laboratories. The major issues are integration of a number of disciplines into remote laboratory design, lack of modular approach in designs, absence of readily available commercial products, lack of integration of

learning management system, maintenance issues, and insufficient administrative support. This section will discuss each of these underlying issues.

Any development with remote experimentation warrants expertise from a number of disciplines. This includes computer interfacing, data acquisition and control, web application development, computer networking, web security, and real-time control. Considering our compartmental arrangement of disciplines in education and research, it is usually difficult for a person or even a research/academic group to gather all this expertise, unless one deliberately forms a group with these capabilities, which is usually difficult for most of the research/academic institutions. Also the nature of experiments that need to be accessed via Internet varies from discipline to discipline. Having all these issue, it was difficult for most of the remote laboratory designers to consider the modularity in system design. In the academic/research areas, there are various kinds of experiments with a variety of inputs and outputs (in terms of frequency and voltage levels) along with need for data presentations. Due to lack of modularity in design as well as an absence of any common framework, each development initiative starts from a scratch. This introduces a difficulty in transferability of one component of a system to another or integration between the systems. This makes it difficult for an Internet-based remote laboratory to adapt with additional experiments or to interface with another remote laboratory system. To accommodate new experiments one has to redesign the system. In this respect, some of the limitations of remote laboratories are reported by some researchers [19].

In recent years both the hardware and software technology has been developed extensively, which has enabled the remote experimentation designers to come up with more powerful systems that wear not possible in the past. However, most of the cases of integration of these available technologies are a major issue and warrant some level of expertise that is not readily available to an institution. There are no commercial products that are designed with an Internet-based remote laboratory in mind that one can design a system with off the shelf. In addition, there is a desire to have an integrated learning management system so an academic can manage the learning process while delivering a remote laboratory course. Most of the remote laboratory designs do not have a learning management system incorporated with an existing learning management system such as Blackboard.

Considering the complexity of Internet-based remote laboratory system, it is difficult to find a maintenance technician who can understand and address the need for a system of this nature. In most cases, the developed system is maintained by graduate students or entry level researchers who are usually a moving entity. Very few institutions/laboratories can afford to have permanent staffs to maintain these systems.

Another major drawback is the lack of administrative support for the remote laboratory developments and their maintenance. Not many administrators in academic institutions realized the potential of Internet-based remote laboratories. As a result remote laboratory development efforts are not getting enough attention from administration and solely depend upon external grants, which is not always easy to acquire.

2.2 Potential Discourse

After having the above discussion and looking from an overall perspective, it is true that there is a major technological progress in remote laboratory developments. However, most of the cases these are used for testing and validation purposes and very few systems were integrated

into our regular educational infrastructure. So, one can ask a question- Are we moving too fast? Is it not important for us to address few additional issues before we proceed further? With this notion and scenario the author would like to present the issues that need to be addressed to make the use of remote experimentation widespread and sustainable.

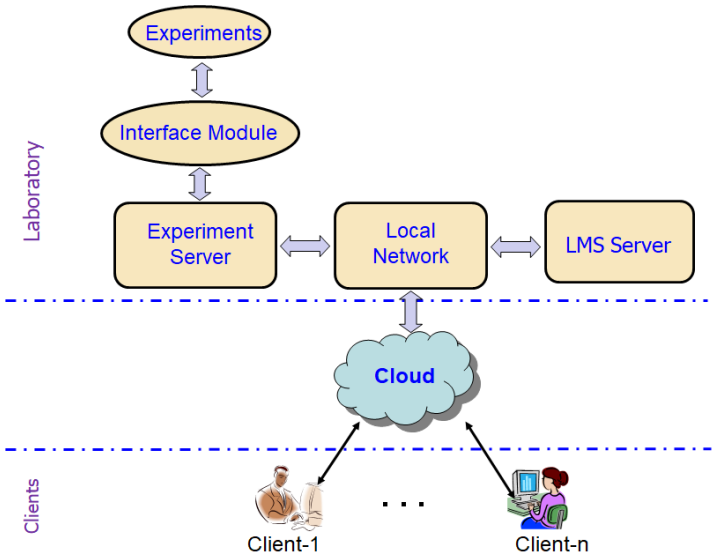

Figure 2: Concept of modularity in design of remote laboratories.

A concept of modularity in design is presented in Figure 2. The main modules are Interface module, Experiment server, Learning management system (LMS) server, and Local network. To have a level of flexibility in operation, future expansion, and collaboration, there should be an understanding within the community for an interfacing standard between the modules. In this regard, recently, MIT took an initiative along with researchers and academics around the world to form a consortium to discuss major developments in this area and come up with a broader framework for remote laboratory development. This will streamline future developments through a modular design of system components with pre-determined inputs/outputs for each module. This kind of approach will foster easy interfacing between the systems, ability to handle different kinds of experiments, and capacity to share resources to maximize the use of available resources.

In terms of development of commercial products, there should be collaboration between academia and industry so they can launch projects that will design and prototype custom products that can be utilized in design of remote laboratories. In this regard one can look for federal, state, and industry funding. In U.S. there are funding for small business initiatives from federal agencies. To get attention, it is important to highlight the necessity and potential of Internet-based remote laboratories. It is important for the remote laboratory community to arrange lectures and presentations in various forums so that it can get attention of administrators and decision makers. I believe that the International Association of Online Engineering (IAOE) and other national and regional forums can take a leading role in this effort.

3. IoT Structure for Remote Experimentation

A traditional remote laboratory system involves a full-scale computer system along with associated interfacing and web hosting technologies. This involves a significant overhead for

the initial commissioning and subsequent maintenance of a remote experimentation system [21, 21]. To address this issue, this initiative utilizes an embedded processor for the development of a remote experimentation facility. The embedded processor performs the function of the computer and server used for traditional remote laboratories [22]. In terms of embedded processors, this project utilized a Raspberry Pi as the system controller as well as the web server. Raspberry Pi, a small-scale computing system with its own operating system, replaced a full-scale computer/server, thereby reducing the cost and complexity of the remote laboratory design [23, 24]. The developed experiments can be accessed remotely over the web using a suitable graphical user interface (GUI). The GUI is accompanied by a live video feed, using a Raspberry Pi Camera, so the user can have a real time video of the experimental facility. The system includes a provision so users can download data to a local system for offline analysis. Figure 3 illustrates this layout with Raspberry Pi at its core. Raspberry Pi is a cheap yet a very efficient solution to set up a remote laboratory system. As a single board computer, it functions as a web server through which remote users can access the laboratory through the internet. It also functions as an embedded processor to interact with the hardware of the experiment modules residing in the laboratory.

It should be noted that though we have used Raspberry Pi in this project, any other single board computer, such as BeagleBone Black, can be used equally to develop such a system. Such computers, also referred to as microprocessor breakouts, at times may even be preferred over Raspberry Pi if higher capacity or performance is sought.

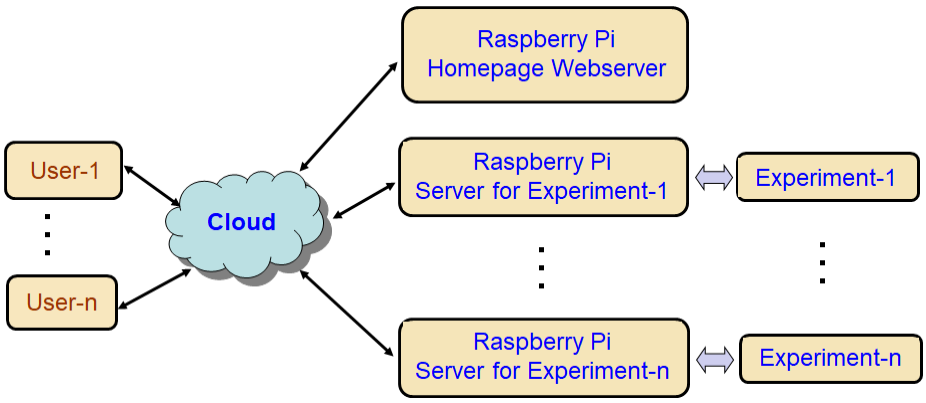

Figure 3: General layout of the remote laboratory.

A number of Raspberry Pis are used in this project, each of which runs its own circuit and controls an experiment. An additional Raspberry Pi was used in this configuration to run a webserver that posts an entry webpage to the remote laboratory. This was an initial stop point or welcome page for remote users. Then the webpage served as a gateway to direct a remote user to the webserver of the experiment as desired by a client.

As the main programming language, Python was used to develop the required software for this project. Python is one of the most widely-used high-level programming languages, a large knowledge base is available for novice programmers to start developing their programs. Additionally, it has a simple and easy to understand syntax that emphasizes readability and thereby reduces the cost of program maintenance. It is easy to develop internet applications with the aid of open source modules and packages developed for Python.

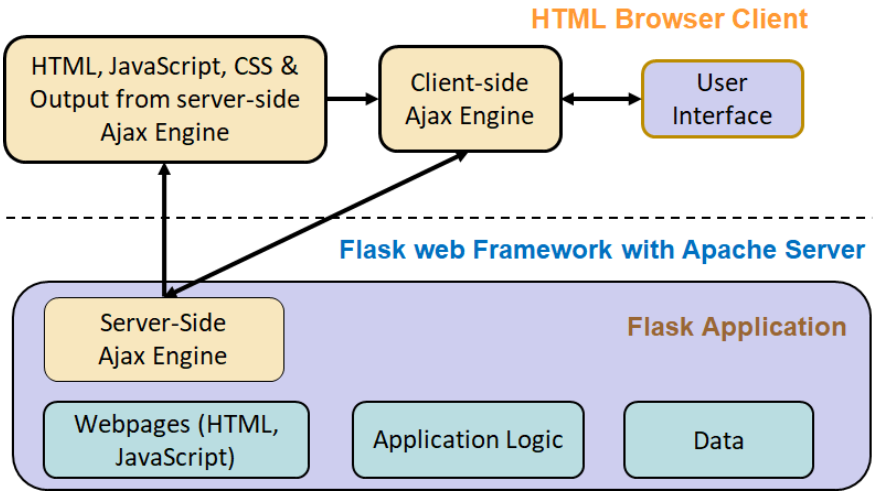

Figure 4: Software block diagram and information flow.

The operating system being used on Raspberry Pi is Raspbian, a Debian Linux based operating system customized for Raspberry Pi. Each Raspberry Pi runs its own web server, a built-in Apache webserver. This allows Raspberry Pi to render webpages for the client-side graphical user interfaces (GUIs) and facilitate remote interaction with hardware attached to the Raspberry Pi. Python is used to program the web server through Flask to facilitate this communication and execution procedure. Flask is one of the popular web frameworks, such as Django, that expedites development of web applications using Python by providing a solid core with basic services. Moreover, Flask supports databases, web forms, authentication, and other high-level tasks by the application of its extensions. Flask extensions are readily available to integrate with the core packages, and not all extensions need to be installed when developing an application. The web technologies used in this project are illustrated in Figure 4.

Bootstrap, a front-end component library, was used to develop the GUIs on the client side. Bootstrap is an open source toolkit provided by Twitter for developing with HTML, cascading style sheets (CSS), and JavaScript. When a remote client requests a specific function using a GUI on his browser, JavaScript sends the request to the server through JSON (JavaScript Object Notation) data. The request is redirected to the Flask application, which executes the Python function mapped to return the request from the given URL.

4. Experimental Systems

This section provides some details of three remote experimentation facilities developed utilizing the Raspberry Pi as the system controller. The first one is the remote vacuum cleaner, where one can program an embedded processor to control the vacuum cleaner. The second experiment is the remote experimentation with an embedded processor system. This involves the development of experiments where students used Python to program a Raspberry Pi over the web to work with the sensors and actuators. The third experiment is an Internet of Things (IoT) enabled structural health monitoring system to provide an automated real-time diagnosis of the structural health of an infrastructure. For this study a laboratory scale suspended-bridge was used along with accelerometers mounted for data collected.

4.1 Remote Vacuum Cleaner

The first testbed is a self-navigated mobile platform fitted with a vacuum cleaner. The mobile platform is wirelessly connected to a server for operation and control. The system is fitted with a number of sensors to implement self-navigation around obstacles and has an onboard microcontroller system for control. A GUI was developed to facilitate the interaction between the system and a remote user.

The remote vacuum cleaner consists of a mobile platform fitted with a dc vacuum cleaner along with a number of sensors for navigation. A general block diagram of the system is shown in Figure 5.

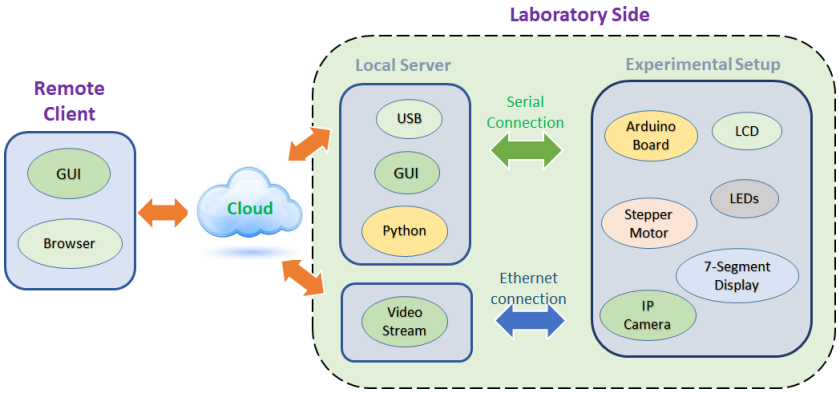

Figure 5: System diagram for remote vacuum cleaner

The system has three main components: a mobile platform as the remote testbed system, a local server is the gateway between the testbed and remote clients and the remote client. The mobile platform consists of a drive system, sensors for navigation, an embedded processor (Arduino board) for local control and data management, an XBee for wireless communication with the local server, and an IP camera for real time video. The IP camera has its own communication route via a WiFi channel. The video is then embedded within the GUI for user monitoring. Images of completed mobile platform are shown in Figure 6.

(a) Overall system.

(b) Close up view of electronics.

Figure 6: Images of the mobile platform

Designing a GUI to provide users with excellent visual composition is a vital part of remote testbed designs. The goal is to improvise and enhance the visual experience between the human eye and computer. Considering the issues for an effective GUI, HTML was used as a software tool for testbed GUI development. Along with HTML, Cascaded Style Sheets (CSS) were

provided to improve the overall visual experience. The GUI contains all the input and output selection buttons and variables, which can be adjusted by a client. Figure 7 shows an image of the developed GUI showing all the functionalities. There is an on/off switch provided at the top right corner of the page. Just to the right side of this, there is a mode selection switch. The system can be operated in two modes: auto and manual. On the top right corner there is a vacuum on/off switch for that can activate and deactivate the vacuum cleaner. In the manual mode one can use the arrows for motion control and the red square button to stop. In the auto mode the system travels on its own guided by a set of IR sensors for wall detection and a pair of ultrasonic sensors for obstacle detection. In addition, users can adjust the speed and allow proximity to an obstacle. On the mid left, there is an arrow to determine the distance from the nearest obstacle at any point. The distance will display at the bottom. At the bottom right, the safety status is displayed to show the danger from a nearby obstacle.

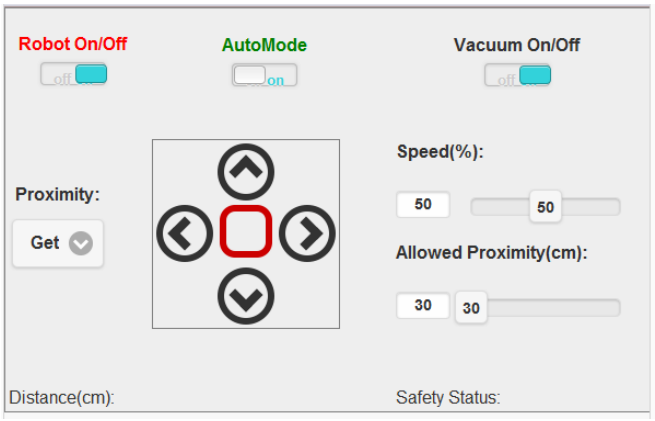

Figure 7: Images of the GUI used for the remote vacuum cleaner.

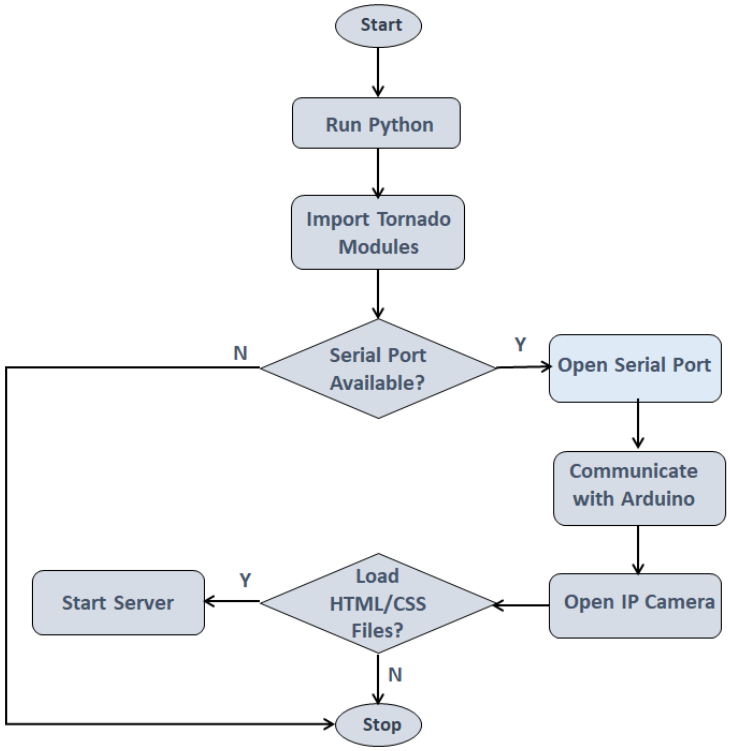

Figure 8: Program flow for the server.

Figure 8 is a flowchart showing a high level view of the program execution cycle for the server system. To start the server, the Tornado Python script should be executed [25, 26, 27]. At the beginning, Python takes in all the modules required for operation using import method. It intiates the opening of a serial port. There will be an error message if the serial port is already being used or is not avaialble. The server communicates with the Arduino board once it opens the serial port. The server script requires all of the HTML/CSS files when a request is made from the client side. The files are loaded before starting the server, and whenever a request is made, it will transfer the files to the client's browser as a webpage. In case of any irregularity in the procedure, it will exit the loop and will display an error message.

4.2 Programming of Embedded Processor

This section illustrates a remote experimentation where one can program an embedded processor system over the Internet. Within this facility multiple input and output devices are connected to a Raspberry Pi, which serves as an embedded processor to manipulate these devices. The goal of the facility is to provide students with a platform to learn how to develop an embedded system using Raspberry Pi as the embedded processor and Python as the programming language. Figure 9 shows the general layout of this facility.

A student can remotely access this module through the webpages broadcast from the webserver on the Raspberry Pi. The student can watch a live video stream that shows the entire module as well as observe the state of the devices attached to the module through custom displays on the GUIs of these webpages. The student can manipulate the state of these devices through the controls provided in the GUIs and immediately observe the changes made.

The Python codes that perform the manipulation of the devices are downloadable from the webpages together with relevant background information needed to program the Raspberry Pi for these manipulations. The student can download these code files and study them to learn how they work. Then the student can write his own Python program to manipulate these devices, upload his code file through a GUI on the webpage, execute his own code remotely on Raspberry Pi, and immediately observe the results. Thus, the main design aspect of this system is that it can be programmed from anywhere at any time.

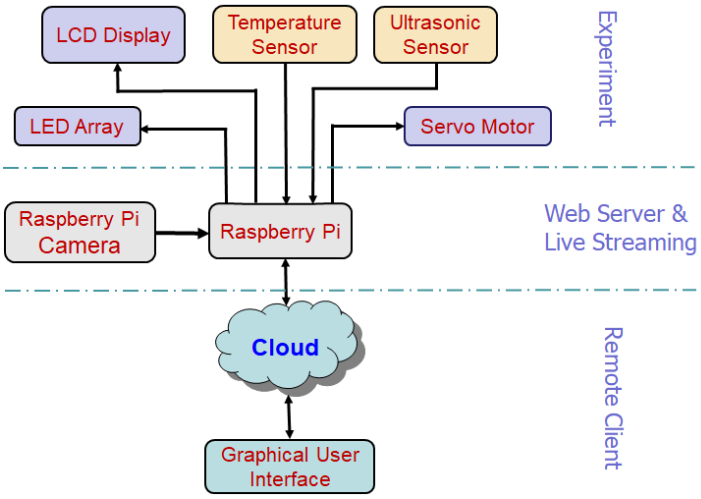

Figure 9: Block diagram of remotely programmable Raspberry Pi.

In addition to a live video camera attached to the Raspberry Pi, the setup includes two input devices: an ultrasonic sensor and a temperature sensor. There are three types of output devices: LEDs, a 16x2 LCD, and a servo motor.

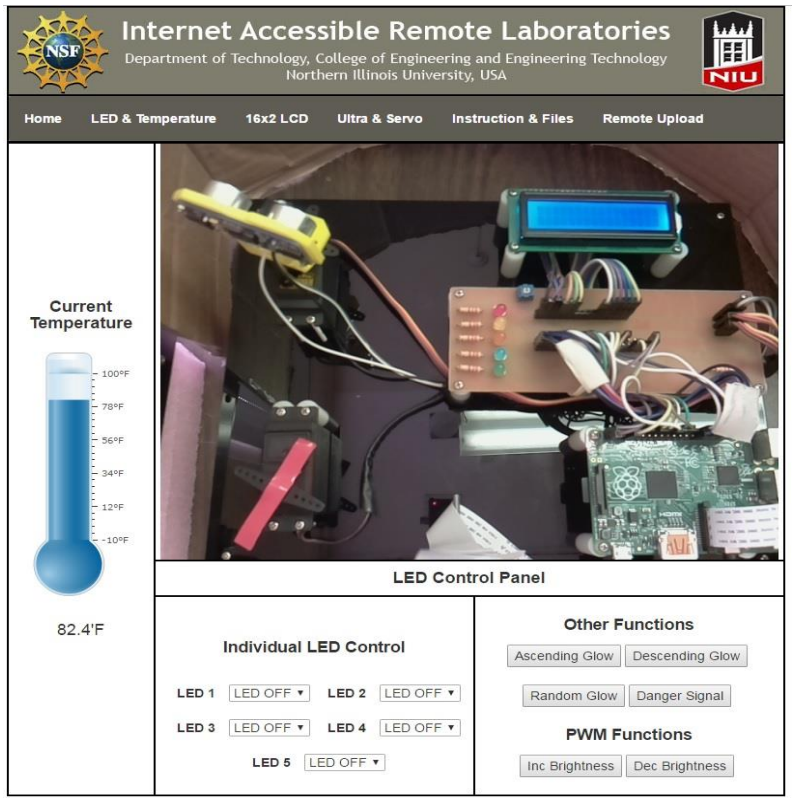

Figure 10: GUI for LEDs and temperature sensor.

Figure 10 shows an image of the webpage for LED control and temperature monitoring. The horizontal menu on the page has links to custom pages for other devices in the module. The picture on the image is the live video of the module from the camera, which shows the ultrasonic sensor at the top left, servo motor at the bottom left, LEDs at the center of the circuit board, and the LCD display above and Raspberry Pi below the circuit board. The GUI on the left of the page shows the temperature in the laboratory room using a graphical display and in text. The GUI at the bottom shows the state of the LEDs and provides controls to manipulate their states. Five LEDs are connected to the system, and LEDs can be controlled individually as well as together using specific functions, such as making LEDs glow in ascending or descending order or increasing and decreasing brightness levels by implementing pulse width modulation (PWM) through the controls on the GUI. Another page, accessible through the "12x2 LCD" link on the horizontal menu, displays the text on the LCD and lets the user change it remotely by entering his own text in a text field on the GUI and sending it to the display by clicking on a button. Users can see the change on the LCD live on the video stream. The LCD module is integrated with the Raspberry Pi using the Adafruit LCD library.

Another important feature of this system is remote upload and execution. The user can upload his own program to Raspberry Pi using the GUI shown in Figure 11. The upload form accepts only *.py format for it to execute the program file once it is uploaded. Once the "Execute" button is clicked, the uploaded file is executed and either the error or the program output is displayed in the "Execution Output" tab in the GUI. Response to the execution can be observed live in the video stream as well. The two graphs (Figure 11) show data from the

temperature and the ultrasonic sensor captured over time. These data are logged into a file and are available for download by remote user to perform offline analysis of his experiment.

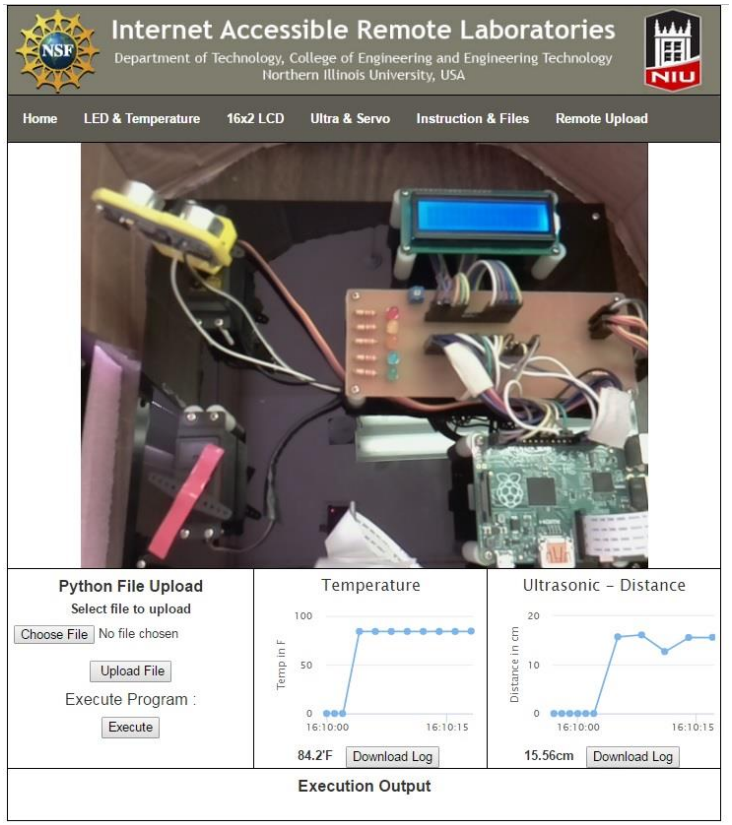

Figure 11: GUI for remote upload and data log.

4.3 Structural Monitoring

This section reports the design and development of an IoT enabled structural health monitoring system to provide an automated real-time diagnosis of the structural health of an infrastructure. For this study a laboratory scale suspended-bridge was used along with accelerometers mounted for data collected. Figure 12 shows the system diagram. The sensors are connected with an embedded processor system for data collection and pre-processing. Processed data are simultaneously passed to the cloud as well as a remote server for client access.

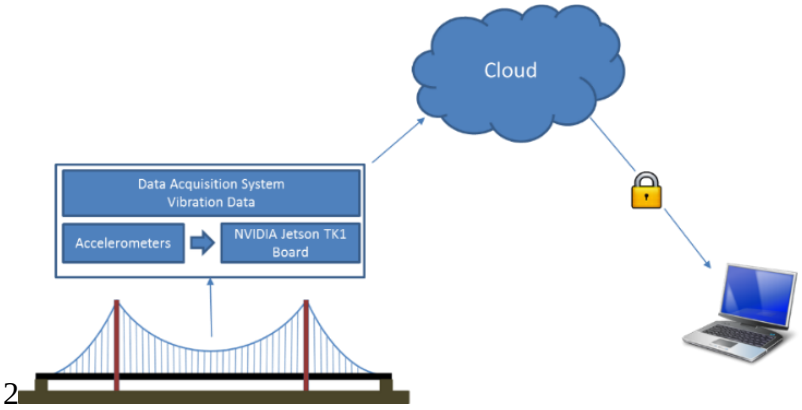

Figure 12: System diagram of the developed monitoring system.

The center of the cloud integration is the NodeJS server program running Heroku cloud. This server is connected with the NVIDIA TK-1 board as well as a local web server and Datastore (Figure 13). The Heroku cloud accepts/requests data from all the sensors, TK-1 board's, supporting programs, and other users [28, 29]. Sensor data are stored in Elasticsearch (Lucene) which is running in the local Datastror. Elasticsearch is a Non-SQL Datastore, which helps provides faster indexing, fuzzy searching, and analytics of very large data. It is highly scalable compared to SQL [30].

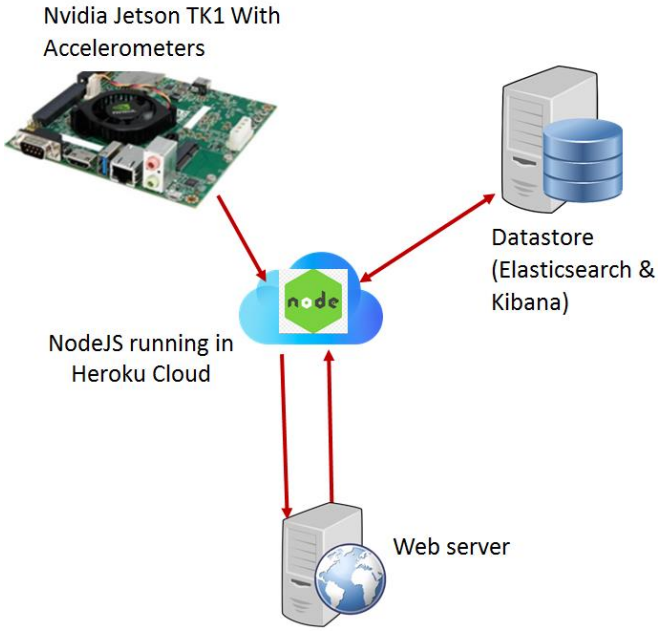

Figure 13: System interaction using Cloud.

The NodeJS server program in cloud provides Representational State Transfer (REST) end point to TK-1 board and push the data to the Datastore. This architecture makes it highly scalable to connect multiple monitoring sensors across multiple infrastructures. The collected data is pushed to Elasticsearch by the NodeJS program after receiving it from Python program running on TK-1 board. The central NodeJS server perform tasks which require high availability and computation such as serving multiple devices, clients such as web users, triggering alert, pushing the data to Datastore, and spectral analysis.

Considering limited storage capacity on TK-1 board, data is never stored locally but is pushed to the external server where large volume storage is possible through central NodeJS server. In addition to transferring data between the servers there is an additional service running in the web server, which trigger alerts such as e-mail, when there is critical breach in set threshold value. NodeJS in Heroku cloud automatically checks this threshold in real-time.

Data collection is done through python script running in TX-1 board. Multiple scripts run in parallel in multi-threading mechanism for each sensor (Figure 14).

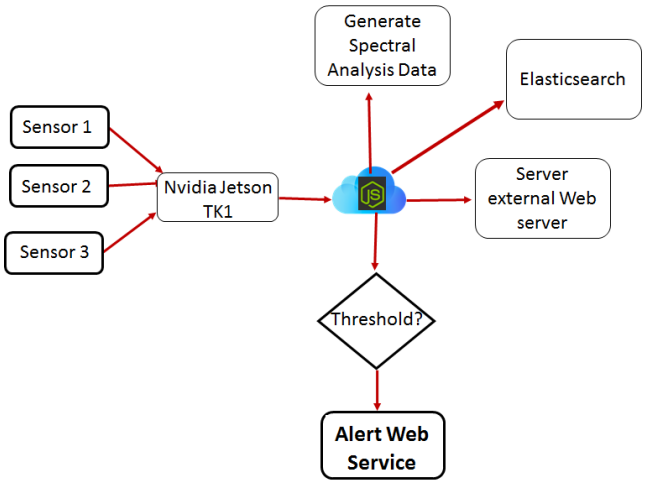

Figure 14: Data collection threads.

There are multiple services running in NodeJS. It fetches and pushes data from Elasticsearch, perform spectral analysis, provides service to running program to trigger threshold alert, supports web application such as reading and updating threshold. Node.js is a JavaScript runtime that uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. NodeJS runs on a single thread but its event driven approach makes it highly efficient with increased throughput [30].

There are three accelerometers fitted with the bridge, each providing data for three axes (x, y, and z). Data from each accelerometer will have two graphs, one is the time series and the other is the spectral analysis. Graphs are plotted using Kibana, which is an open source analytics and visualization platform designed to work with Elasticsearch [31]. Kibana graphs are embedded with the web page using 'IFrames'. Kibana also provides advanced filtering options in the graph. In addition, it can run Elasticsearch Query DSL within the graph to apply custom filter.

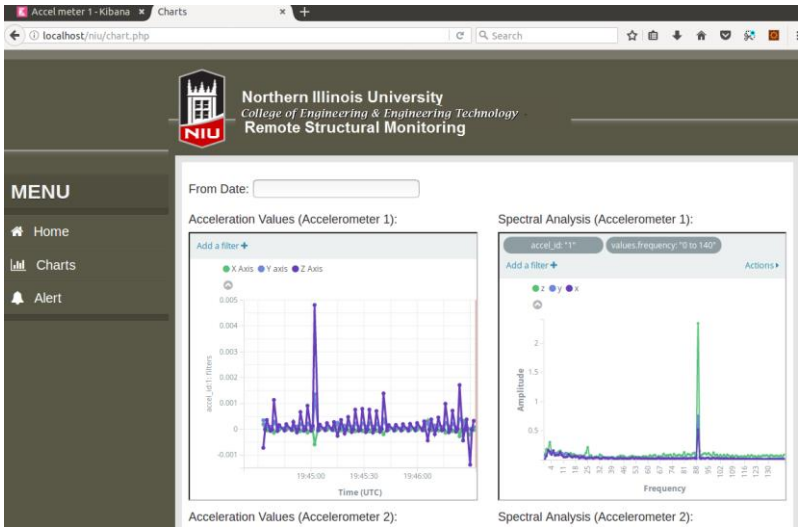

Figure 15: Accelerometer data (time series and spectral analysis).

Figure 15 shows two graphs for one accelerometer that will be displayed within a web browser. The left-hand side graph is showing time series for one axes and the right-hand side graph is showing its spectral analysis. Within the web page the users can choose the start date for plotting. The time series graphs are dynamic and fetches new data from the

Elasticsearch Datastore every 5 seconds. Spectral analysis is done by NodeJS server running in the cloud and pushed into Elasticsearch every minute. Each time NodeJS takes recent 2048 data points from time-series data to perform spectral analysis and pushes them to Elasticsearch to be plotted by Kibana via the graphical user interface (GUI).

5. System Utilization

To assess students' learning behavior in terms of the access time and duration of use, the developed facility has built-in capacity to collect students' login and logout times along with the time taken to perform each experiment. These data allowed the facilitator to know the level and timing of facility use and hence provided a broader understanding of the students' behavior in terms of use of the facility.

These data allowed comparing the learning efficiency of the control group and the test group and also the students' behavior in terms of the use of the facility. It was found that there were statistically significant differences between the test and the control groups in their time spent on the laboratory tasks, with the test group spending 69% less time than the control group on average. It can be interpreted that the test group learned more efficiently than the control group. Where the control group is performing experiments physically inside the laboratory. In terms of access time to the facility, it was found that the time of the day when students in the test group performed their laboratory tasks ranged between 9am and 2am of the next day, which indicates great flexibility and convenience for students that is otherwise impossible because of the cost and administrative limitations under a traditional laboratory configuration. Figure 16 shows the access profile to the remote laboratory experiments in terms of percentage of access.

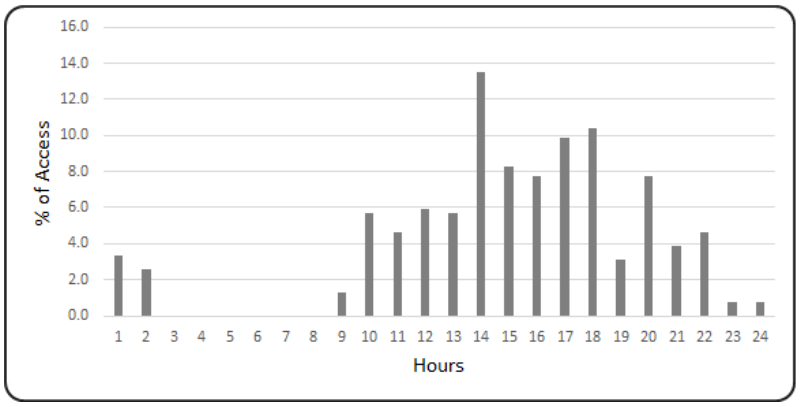

Figure 16: Shows the access profile to the facility in terms of time of the day.

6. Conclusions

The paper provides a discussion on the issues that need to be addressed to make the Internet-based remote experimentation facility sustainable and acceptable by the academic community. The main factors are modularity issues in design, commercial products and systems that can support the development process, and gather support from administration. The second part of the paper presents three remote experimentation facilities utilizing the embedded processor as the main controller. One is a remotely controlled vacuum cleaner with remotely programmable embedded processor. One is a sensor actuator system the user can manipulate to perform certain laboratory experiments. The second system complements

teaching Python programming for a Raspberry Pi over the web. Students can write their programs and upload them on the remote experimental system. The sensors and actuators are an array of light emitting diodes, a temperature sensor, a liquid crystal display, a servo motor, and an ultrasonic sensor. A suitable GUI was also developed so remote users can manipulate the controlled entities with little difficulty. The third system provides an automated real-time diagnosis of the structural health of an infrastructure. For this study a laboratory scale suspended-bridge was used along with accelerometers mounted for data collected.

7. References

[1] V. Terzis, and A. A. Economides, "The acceptance and use of computer based assessment," *Computers in Education*, vol. 56, no 4, pp. 1032-1044, 2011.

[2] Z. Papamitsiou, and A. A. Economides, "Learning Analytics and Educational Data Mining in Practice: A Systematic Literature Review of Empirical Evidence," *Journal of Educational Technology & Society*, vol. 17, no. 4, pp. 49-64, 2014.

[3] A. Miguel, J. F. PradaJuan, A. Serafín, G. Sergio and Manuel, D. "Challenges and solutions in remote laboratories. Application to a remote laboratory of an electro-pneumatic classification cell," *Computers & Education*, vol. 85, pp. 180-190, July 2015.

[4] D. Lowe, P. Newcombe and B. Stumpers, B. "Evaluation of the Use of Remote Laboratories for Secondary School Science Education," *Research in Science Education*, vol. 43, pp. 1197-1219, 2013.

[5] E. Mitsea and A. Drigas, "A Journey into the Metacognitive Learning Strategies," *International Journal of Online and Biomedical Engineering*, vol. 14, no. 14, pp. 4-20, 2019.

[6] S. Appanna, "A Review of Benefits and Limitations of Online Learning in the Context of the Student, the Instructor and the Tenured Faculty," *International Journal on E-Learning*, vol. 7, no. 1, pp.5-22, 2008.

[7] N. Croft, A. Dalton and M. Grant, "Overcoming Isolation in Distance Learning: Building a Learning Community through Time and Space," *Journal for Education in the Built Environment*, vol. 5, no. 1, pp. 27-64, 2015.

[8] D. Gillet, H. A. Latchman, Ch. Salzmann and O. D. Crisalle, "Hands-On Laboratory Experiments in Flexible and Distance Learning," *Journal of Engineering Education*, vol. 90, no. 2, pp. 187-191, 2013.

[9] L. F. Z. Rivera and M. M. L. Petrie, "Models of Collaborative Remote Laboratories and Integration with Learning Environments," *International Journal of Online Engineering*, vol. 12, no. 9, pp. 14-21, 2016.

[10] T. Tsiatsos, S. Douka, A. Mavridis, S. Tegos, A. Naddami, T. Zimmer and D. Geoffroy, "Evaluation Plan and Preliminary Evaluation of a Network of Remote Labs in the Maghrebian Countries," *International Journal of Online Engineering*, vol. 10, no. 5. pp. 15-20, 2014.

[11] I. Santana, M. Ferre, E. Izaguirre, R. Aracil and L. Hernandez, "Remote Laboratories for Education and Research Purposes in Automatic Control Systems," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp.547 – 556, 2013.

[12] R. Heradio, L. Torre and S. Dormido, "Virtual and remote labs in control education: A survey," *Annual Reviews in Control*, vol. 42, pp.1-10, 2016.

[13] A. Maiti and B. Tripathy, B. “Remote Laboratories: Design of Experiments and Their Web Implementation,” *Journal of Educational Technology & Society*, vol. 16, no. 3, pp. 220-233, 2013.
[14] R. Tirado-Morueta, R. Sánchez-Herrera, M. A. Márquez-Sánchez, A. Mejías-Borrero and J. M. Andujar-Márquez, “Exploratory study of the acceptance of two individual practical classes with remote labs,” *European Journal of Engineering Education*, vol. 43, no. 2, pp.278-295, 2018.
[15] W. Li and G. Bai, “Internet of Things System Based on Mobile Communication Network,” *International Journal of Online Engineering,* vol. 14, no. 11, pp.64-76, 2018.
[16] A. K. M. Azad, M. E. Auer and V. J. Howard [Editors], *Internet Accessible Remote Laboratories: Scalable E-Learning Tools for Engineering and Science Disciplines*, IGI Global, ISBN 13: 9781613501863, 2012.
[17] I. Fidan and N. Ghani, “Acquisition steps of a Remotely Accessible Rapid Prototyping laboratory,” *International Journal of Computer Applications in Technology (IJCAT), Vol. 30, No. 4, 2007*
[18] I. Fidan, A. Elliott, M. Cossette, T. Singer, E. Tackett “The Development and Implementation of Instruction and Remote Access Components of Additive Manufacturing’”. In: Auer M., Azad A., Edwards A., de Jong T. (eds) *Cyber-Physical Laboratories in Engineering and Science Education*. Springer, Cham, 2018.
[19] M. E. Auer, A. K. M. Azad, A. Edwards and T. d. Jong, [Editors], *Cyber-Physical Laboratories in Engineering and Science Education*, Springer, ISBN: 9783319769349, 2018.
[20] D. D. Magdum and R. D. Patane, “Raspberry Pi Based Remote Lab Implementation,” *International Journal of Innovative Research in Computer,* vol. 4, no. 8, pp. 15181-15185, 2016.
[21] M. Kalúz, L. Cirka, R. Valo and M. Fikar, “ArPi Lab: A Low-cost Remote Laboratory for Control Education,” *IFAC Proceedings Volumes (IFAC-PapersOnline),* vol. 47, no. 3, pp. 9057-9062, 2014.
[22] D. Fallon, (2013). “Survey of Existing Remote Laboratories used to Conduct Laboratory Exercises for Distance Learning Courses,” *in ASEE Annual Conference & Exposition*, Atlanta, 2013
[23] Y. Limpraptono, A. A. P. Ratna and S. Harry, “New Architecture of Remote Laboratories Multiuser based on Embedded Web Server,” *Remote Engineering and Virtual Instrumentation (REV), 2012 9th International Conference*, Bilbao, Spain, 2013.
[24] I. Angulo, J. García-Zubia, P. Orduña and O. Dziabenko, “Addressing low cost remote laboratories through federation protocols: Fish tank remote laboratory,” *Global Engineering Education Conference (EDUCON), 2013 IEEE*, Berlin, Germany, 2013.
[25] J. Gravier, B. Fayolle, M. A. Bayard and J. Lardon, “State of the Art About Remote Laboratories, International Journal of Online Engineering,” vol. 4, no. 1, pp. 19-25, 2008.
[26] Python, *Python Software Foundation*, 2015, https://www.python.org/ [Accessed January 15, 2020]
[27] Tornado, *Tornado Stable*, 2015, http://www.tornadoweb.org/en/stable/ [Accessed January 15, 2020]
[28] Arduino Mega, *Overview of Arduino mega*, 2015, http://arduino.cc/en/Main/Arduino BoardMega

[29] NodeJs, *About Node.js*, 2017, https://nodejs.org/en/about/ [Accessed January 15, 2020]
[30] Heroku, 2017, https://www.heroku.com/ [Accessed January 15, 2020]
[31] Elastic Search, 2017, https://www.elastic.co/ [Accessed January 15, 2020]