




Article

Towards a Supervised Remote Laboratory Platform for Teaching Microcontroller Programming

Manos Garefalakis ^{1,*} , Zacharias Kamarianakis ^{1,2}  and Spyros Panagiotakis ¹ 

¹ Department of Electrical & Computer Engineering, Hellenic Mediterranean University, Estavromenos, GR71410 Heraklion, Greece; zkamar@hmu.gr (Z.K.); spanag@hmu.gr (S.P.)

² Institute of Agri-Food and Life Sciences, Research & Innovation Center, H.M.U.R.I.C., Hellenic Mediterranean University, GR71410 Heraklion, Greece

* Correspondence: mgarefal@gmail.com

Abstract: As it concerns remote laboratories (RLs) for teaching microcontroller programming, the related literature reveals several common characteristics and a common architecture. Our search of the literature was constrained to papers published in the period of 2020–2023 specifically on remote laboratories related to the subject of teaching microcontroller programming of the Arduino family. The objective of this search is to present, on the one hand, the extent to which the RL platform from the Hellenic Mediterranean University (HMU-RLP) for Arduino microcontroller programming conforms to this common architecture and, on the other hand, how it extends this architecture with new features for monitoring and assessing users' activities over remote labs in the context of pervasive and supervised learning. The HMU-RLP hosts a great number of experiments that can be practiced by RL users in the form of different scenarios provided by teachers as activities that users can perform in their self-learning process or assigned as exercises complementary to the theoretical part of a course. More importantly, it provides three types of assessments of the code users program during their experimentation with RLs. The first type monitors each action users perform over the web page offered by the RL. The second type monitors the activities of users at the hardware level. To this end, a shadow microcontroller is used that monitors the pins of the microcontroller programmed by the users. The third type automatically assesses the code uploaded by the users, checking its similarity with the prototype code uploaded by the instructors. A trained AI model is used to this end. For the assessments provided by the HMU-RLP, the experience API (xAPI) standard is exploited to store users' learning analytics (LAs). The LAs can be processed by the instructors for the students' evaluation and personalized learning. The xAPI reporting and visualization tools used in our prototype RLP implementation are also presented in the paper. We also discuss the planned development of such functionalities in the future for the use of the HMU-RLP as an adaptive tool for supervised distant learning.

Keywords: remote laboratory; Arduino microcontroller programming; xAPI; LRS; assessment types; pervasive learning; supervised learning



Citation: Garefalakis, M.; Kamarianakis, Z.; Panagiotakis, S. Towards a Supervised Remote Laboratory Platform for Teaching Microcontroller Programming. *Information* **2024**, *15*, 209. <https://doi.org/10.3390/info15040209>

Academic Editors: Luis Borges Gouveia and Aneta Poniszewska-Maranda

Received: 16 February 2024

Revised: 22 March 2024

Accepted: 1 April 2024

Published: 8 April 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Remote laboratories (RLs) have been extensively studied in the literature, with a prominent highlighted issue being the lack of a common design pattern among online laboratory systems. This lack of standardization has constrained scalability, integration, interoperability, traceability, reliability, security, and privacy. Addressing this concern, the IEEE 1876-2019 standard [1] was developed to facilitate the design, implementation, and usage of online laboratories in education. The fundamental purpose of remote labs is to afford students the opportunity for practical experimentation (PEX) and laboratory experimentation (LEX), leveraging components and equipment typically found in university labs. Through remote labs, students can apply theoretical knowledge gained in the classroom,

accommodating their learning styles, at their convenience. Whether utilizing PCs for E-learning or smartphones for M-learning, students can engage with learning objects ranging from simple tasks to more complex scenarios, within the parameters set by the remote lab. This educational approach fills a crucial gap between simulated environments (full virtual labs) and real-world lab experiences, enabling the provision of distant courses focused on hardware equipment usage and software programming. The demand for remote access to educational lab facilities was further underscored during the COVID-19 pandemic, where such labs became essential for mobile learning, enabling students to practice and experiment with specialized equipment and tools from any location to acquire practical skills and fulfill course requirements.

In the paper [2], the classification of laboratories is presented. They are divided into local and remote laboratories. The remote laboratories are classified into real remote laboratories (remote laboratories), hybrid remote laboratories, and virtual laboratories. In this paper, we review and propose a real remote laboratory, where the user works at a distance on real equipment, especially on programming microcontrollers like Arduino boards. Working on an RL is different from working on a remote simulation, like Tinkercad [3], which also provides learning skills (circuit design, coding, etc.), but only to some extent. An RL can implement more experiments, more complex and with real components, working in real situations and not based on a mathematical model [4], because they can connect to an existing infrastructure such as LoRa Gateway, MQTT servers, etc. It must be highlighted that there is no discrimination between the two types of laboratories, real or virtual, but on the contrary, they are used in a complementary way to each other and both provide skills to the user.

The HMU-RLP is a project designed and implemented by the Sensor Network LAB of the HMU. It started from an Erasmus+ KA2 project named SYS-STEM, which included five partners from European universities and is described in the paper [4]. During SYS-STEM, a qualitative evaluation from teachers and students that used the SYS-STEM RL was organized [4]. The evaluation highlighted some technical issues with the SYS-STEM infrastructure, such as the need for more inherent interaction of the user with the RL as well as the need for loading an initialization sketch in the experiment controller at the end of each user session. Also, teachers of secondary education declared their willingness to run pilot testing of the SYS-STEM methodology and ARDLAB with their students, although they welcomed a tool providing analytics for their students' actions over the lab. From the logs of this evaluation, we noticed that students used ARDLAB for LAB exercises especially during the late hours and, also, that many students repeated the courses more than once, which potentially showed that it might be helpful for them or that they needed some help to complete the tasks. The experience acquired from SYS-STEM led to the design and implementation of a different platform that provides more capabilities and functions toward pervasive and supervised learning, concepts that require more than an RL on which the user only monitors the results of the code uploaded on the experimental microcontroller. A first presentation of the HMU-RLP was issued in the paper [5], where we described the ability of an RL to connect with a learning management system (LMS) and how the experiments it provides can be used as learning objects (LOs) within an LMS course.

In the present paper, we proceed one step further toward pervasive learning, describing how an RL can supervise user actions over its infrastructure for potentially providing more powerful analytics to the instructors and/or intelligent tutoring to the learners. The need for these features was revealed by the outcomes of [4] and is implemented in this paper with the three assessment types we introduced in the HMU-RLP.

This paper also implements a literature search for finding other remote laboratories (RLs) that are used for teaching microcontroller programming of the Arduino family. This search was underlined by our need to understand how education based on modern remote labs is built today and what new features could potentially be added as a complement. The literature search revealed similarities in the architecture of remote laboratories but did not show any type of assessment apart from monitoring the results of the coding of the

microcontrollers. The HMU-RLP is a use case for teaching programming in the Arduino ecosystem. The HMU-RLP implements three different types of assessment of the user's use of the RL and the uploaded code, which is used for further processing of adaptive learning concepts and aligns with the concept of pervasive and supervised learning. Furthermore, the experimental microcontroller is connected to many sensors and actuators, which aligns it with the concept of one RL for many experiments.

The literature search also revealed concepts that should be considered for implementation in the HMU-RLP and will be included in future work, such as hosting of the RL on a remote laboratory management system, remote H/W configuration, and user complements with augmented and virtual reality value-added concepts. Also, the need for an extended qualitative review of our RLP from teachers and students.

The paper is organized as follows: Section 1 presents an overview introduction of the paper. Section 2 includes the related literature search. Section 3 presents the architecture of the remote lab. Section 4 presents the remote lab assessment types. Section 5, Experience API Statements and Tools, presents the experience API used for the users' learning analytics reporting, while Section 6 presents the remote lab experiments. Finally, Section 7 presents the conclusion and future work.

2. State of the Art

Remote laboratories seem to be a necessity in recent years because they cover a gap between theory and hands-on laboratories, real on-site laboratories. The users work on real equipment remotely. Under this context, there are advantages and disadvantages. The advantage is that more users can practice on experiments from the comfort of their homes and at any time they wish, 24/7, and RLs promote inclusion and diversity, providing lab access to persons with special needs.

The disadvantages are that users do not acquire hands-on skills, and the experiments are dependent on network communication and the quality of video and audio streaming.

The literature search that was performed revealed the number of papers published per year. Queries were issued on the following databases: Google Scholar, Scopus, Semantic Scholar, and OpenAlex. The keywords used were "Remote Labs", "Remote Laboratories", "Remote experiments", "Online Labs", "Online Laboratories", and "Online Experiments". The keywords were limited to the paper's title only. The initial search returned 13,544 results and after removing duplicates, and again filtering keywords in the title for a second time, the number was reduced to 1615, as can be seen in Table 1.

Table 1. Results of analysis.

Data Source	Initial Search	1st Stage (Identification)	2nd Stage (Screening)
Google Scholar	2267	620	
Scopus	2152	842	
Semantic Scholar	5186	492	
OpenAlex	3939	16	
Total	13,544	1970	1615

In Figure 1, the number of papers related to remote laboratories published per year can be seen. What we can comment on in the graph is that, during the period of 2012–2015, there was a peak in the number of papers published related to remote laboratories. After the period of 2016–2019, there was a decrease in the number of papers on the topic of remote laboratories, which then increased again during the period of 2020–2023, probably due to the COVID-19 pandemic period, where remote laboratories were an important and required solution for teaching, apart from the fact that technologies were developed.

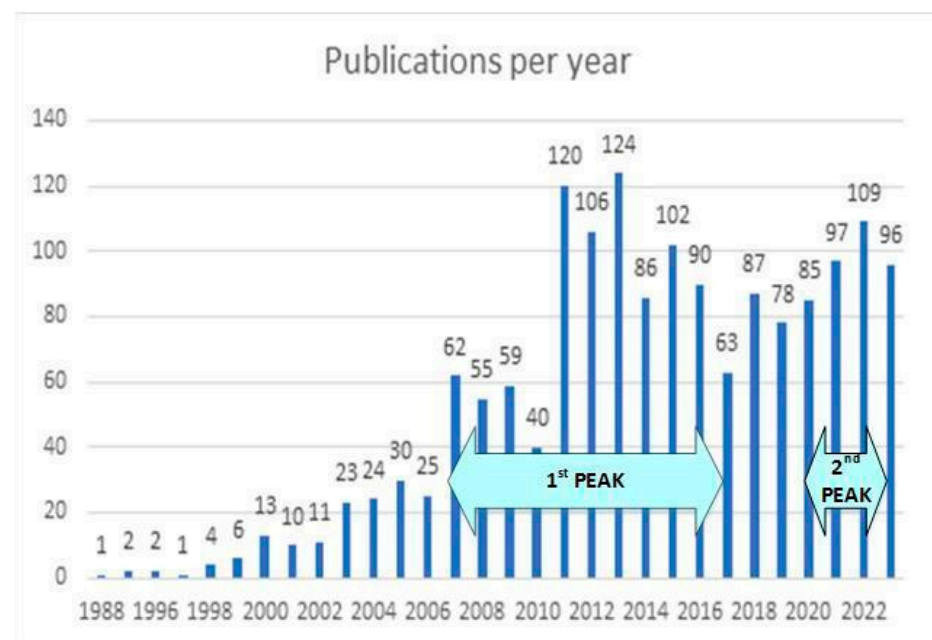


Figure 1. Papers on remote laboratories published per year.

Another literature research was performed to reveal the differences between the proposed RL platform and other relevant use cases. Since the proposed RL is about programming Arduino ecosystem boards, the keyword Arduino was added to the search keywords. For this particular literature research, a search in the Scholar Google database issued the following statement: “Remote Labs” OR “Remote Laboratories” AND Arduino.

The criteria set for the eligibility of a paper were the following:

- Recent papers that were published during the second peak period (see Figure 1), which included also the COVID-19 and post-COVID-19 period. Additionally, it revealed new generation RLs that included new technologies, e.g., complying with IEEE 1876-2019 [1].
- RLs related to teaching microcontroller programming, specifically Arduino, because the proposed RL is about teaching programming of Arduino boards.
- Access to the article using the institutional credentials of Hellenic Mediterranean University.
- Papers written in the English language.

The initial results were 1590 papers and, after applying the selection criteria, the returned number of results was reduced to 625 papers (query performed on 21 January 2024).

The screening was performed on the title and the abstract, and we present the selected papers in Table 2.

By reviewing the related literature listed in Table 2, we saw some common characteristics. These characteristics are the following: RL name, project website, programming board, programming language, RL controller, hosted in platform, direct access, assessment, GUI for access, statistics, monitoring, remote H/W reconfiguration, LMS integration, and logging learning analytics.

Some RLs have a name that can be found in the literature and a website for the project they belong to, for example, VISIR+, etc. The RLs are for programming microcontrollers, so knowing the board that is programmed is important (Arduino, STM, etc.) and also the programming language used. For example, in some cases of Arduino programming, it is C/C++ coding or visual programming.

Table 2. List of reviewed papers.

Reference No	Title	Publication Year
[5]	Integration of a Remote Lab with a Learning System for training on Microcontrollers' programming	2023
[6]	LabsLand Electronics Laboratory: Distributed, Scalable and Reliable Remote Laboratory for Teaching Electronics	2023
[7]	Remote Laboratory for the Development of Customized Low-Power Computing and IoT Systems	2023
[8]	ARM Distributed and Scalable Remote Laboratory for Texas Instruments Launchpad Boards	2023
[9]	Mobile Arduino Robot Programming Using a Remote Laboratory in UNAD: Pedagogic and Technical Aspects: Experience Using a Remote Mobile Robotics	2021
[10]	Learning CAN bus communication with a remote laboratory	2022
[11]	Learning Management Systems as a platform for deployment of remote and virtual laboratory environments	2022
[12]	ERPLab: Remote Laboratory for Teaching Robotics and Programming	2023
[13]	Fpga-based remote laboratory for digital electronics	2020
[14]	Practice Projects for an FPGA-Based Remote Laboratory to Teach and Learn Digital Electronics	2023
[15]	Teaching programming and microcontrollers with an arduino remote laboratory application	2023
[16]	Remote Experimentation Through Arduino-Based Remote Laboratories	2021
[17]	Remote laboratory for microcontroller programming course	2022
[18]	An Implementation of a Web Laboratory Converting Off-Line Experiments into Remotely Accessible Experiments	2022
[4]	Remote Arduino Labs for Teaching Microcontrollers and Internet of Things Programming	2022
[19]	Remote Laboratory Offered as Hardware-as-a-Service Infrastructure	2022
[20]	A Remotely Configurable Hardware/Software Architecture for a Distance IoT Lab	2021
[21]	Internet of things network infrastructure for the educational purpose	2020
[22]	IOT-OPEN.EU: Introduction to the IoT Practical Projects in English–IOT-Open	2024
[23]	μLAB A remote laboratory to teach and learn the ATmega328p μC	2020
[24]	Block. Ino: Remote lab for programming teaching and learning	2020
[25]	Training Laboratories with Online Access on the ITMO. cLAB Platform	2020
[26]	Design and development of remote laboratory system to facilitate online learning in hardware programming subjects	2020
[27]	Implementation of an Arduino remote laboratory with Raspberry Pi	2019
[28]	Improving the scalability and replicability of embedded systems remote laboratories through a cost-effective architecture	2019

RLs are either autonomous, where the user connects directly to the RL, the direct access characteristic, or they are hosted in platforms like remote laboratory management systems (RLMSs), massive online open courses (MOOCs), small private online courses (SPOCs), learning management systems (LMSs), repositories like remote lab management systems (RLMSs), etc., see the hosted in platform characteristic. These platforms handle user authentication and interoperability with the RL. There are also cases where the RLs can work in both cases autonomously or provided via a platform. All of these are suggested in the IEEE 1876-2019 Standard for Networked Smart Learning Objects for Online Laboratories [1], which defines methods for storing, retrieving, and accessing online laboratories. According to the standard, an online laboratory (RL) is defined as a lab as a service (LaaS).

The assessment type characteristic is whether the RL provides an assessment type and reports grading to the user profile, like it is presented in the HMU-RLP in the next sections.

The type of access to the RL is another characteristic, via a web interface, an application, or via remote desktop application. Another characteristic is the monitoring of the RL's experiment and equipment. In most cases, there is a web camera or an RPI camera module connected, but we also found a paper in which there was no monitoring but images of the board were displayed with the current status. In most of the use cases, the H/W experiments are ready connections, but in one case they suggested an online H/W reconfiguration to expand experiment scenarios.

Finally, the last two characteristics are very important in the concept of pervasive and supervised learning. The possibility of integration in a learning management system (LMS) and the logging of learning analytics on the user's learning experience. According to the paper [5], when an RL is integrated into an LMS, as a learning object (LO), then there are learning analytics that can be stored in the LMS user profile, for example, updating the user grade book, marking an LO as completed, etc. The use of these LAs can be used in updating and planning the learning path of a user. For example, imagine a user who cannot complete an experiment that has to turn on some LEDs. Then, the course path in the LMS can be updated to simpler experiments to help the user complete the learning task and acquire the learning skill. On the other hand, when a user archives directly to complete the experiment, then the LMS enables a more complex experiment to be assigned to the user.

All of the characteristics mentioned above are about working in an LMS and the RL is an LO. Now what about when the RL is not integrated into an LMS and works independently. Then, the learning analytics are stored in learning record storage (LRS), the database defined in the xAPI standard. With the use of tools, the instructor may follow the learning path of the user and accordingly assign experiments to the user. This process needs to be developed in such a way as to automate it and is currently under research by the team developing the HMU-RLP. The Table 3, presents the summary of the paper's review characteristics.

Table 3. Summary of the paper’s review. Reviewed papers’ characteristics.

RL Paper	RL Name	Project Website	Programming Board	Programming Language	RL Controller	Hosted in Platform	Direct Access	Assessment	GUI for Access	Statistics	Monitoring	Remote H/W Reconfiguration	LMS Integration	Logging Learning Analytics
[5]	HMU-RLP	[29]	Arduino UNO	C/C++	Raspberry Pi 3 B+	No	Yes	Yes	Web Interface	Yes	RPI Camera Module	No	Yes	Yes
[6]						LabsLand [30]				Yes	Yes		Yes	
[7]	RemoCLEC	[31]	NUCLEO-WB55RG			LabsLand [30]				Yes	Yes			
			Pololu Zumo 32u4 robot with Arduino	C/C++ or Visual programming Blockly		LabsLand [30]				Yes	Yes			
[8]	TIVA Remote Laboratory at the University of Washington	[32]	TIVA			LabsLand [30]	No			Yes	Yes		Yes	
[9]	Robotics Remote Lab of UNAD		ATmega 32U4 compatible with Arduino	C/C++	Raspberry Pi 3 B+	LabsLand [30]				Yes	Yes		Yes	
[10]	Public University of Navarra CAN Bus RL		Arduino Board			Go Lab [33] LabsLand [30]				Number of Accesses and time spent	Yes			
[11]	UbiLAB project					Weblab-Deusto [33]							Yes	
[12]	ERPLab Environment-Robotic-Programmin-Laboratory		Arduino Uno development board and Ethernet Shield W5100	C/C++		No					ESP32CAM		No	
[13]			FPGA Nexys 3								No camera			
[14]	RLAB University of Málaga	[34]	FPGA Nexys 3		Raspberry Pi 4		Yes				No camera			
[15]	RemoLab in Croatian schools		Arduino								WebCam			
[16]	University UNED		Arduino		Raspberry Pi						WebCam			
[17]	Riga Technical University		Texas Instruments (TI) MSP430						Remote Desktop		VLC			

Table 3. Cont.

RL Paper	RL Name	Project Website	Programming Board	Programming Language	RL Controller	Hosted in Platform	Direct Access	Assessment	GUI for Access	Statistics	Monitoring	Remote H/W Reconfiguration	LMS Integration	Logging Learning Analytics
[18]	WEB Laboratory at University of Kragujevac		Arduino UNO Arduino DUE						X2GO-Remote Desktop					
[4]	SYS-STEM Hub	[35]	Arduino UNO	C/C++	Raspberry Pi	SYS-STEM Hub [35]	No	No	Web Interface	No	RPI Camera module		No	No
[19]	Wrocław University of Science and Technology WUST		STM32 microcontrollers • Nucleo-L476RG • 32L476GDIS-COVERY • STM32F429I-DISC1		Raspberry Pi						RPI Camera module			
[20]			Arduino UNO		Raspberry Pi 3B+							Yes		
[21,22]	IOT-OPEN.EU VREL	[35]	Arduino Uno ESP 8266 (ESP-12E);	C/C++	Raspberry-Pi 2 & 3	VREL management server			Web Interface		RPI Camera module			
[23]	μLAB-Polytechnic Institute of Porto		ATmega328p		Raspberry Pi		Yes		Web Interface		Webcam			
[24,36]	Block.Ino -University of Santa Catarina, Brazil		Arduino	visual programming environment		RELLE [37]					Yes			
[25]	ITMO.cLAB-ITMO University		SDK-1.1M STM32F407VG microcontroller TFK-4.0U MA842 analog I/O module											
[26]			MK20DX128VLH5 by NXP Semiconductors						Chrome Remote Desktop					
[28]	ArduinoRL		Arduino			LabsLand [30]	No		Web Interface		Yes			
[27]	UNED		ATmega328p ATmega2560 MKR1000		Raspberry Pi				Web Interface		Yes			

3. The Architecture of the HMU-RLP

The literature search revealed similarities in the architecture of remote laboratories but did not show any type of user assessment apart from monitoring the results of the coding of the microcontrollers. The HMU-RLP attempts to fill this gap by implementing, as we will present, three different types of assessment of the user's interaction with the RL and of the code the user uploads. Furthermore, the experimental microcontroller is connected to many sensors and actuators, which aligns with the concept of one RL for many experiments. The literature search also revealed concepts that should be considered for implementation in the HMU-RLP and will be included in future work, such as hosting of an RL on a remote laboratory management system and user complements with augmented and virtual reality value-added concepts.

The presented platform for remote labs has a similar architecture to most RLs presented in Section 2. There is a main server, which is a Raspberry PI single-board computer, that runs an application developed in the Flask framework. The application provides the user interface and all necessary tasks needed for the function of the RL.

In most of the cases reviewed, the user connects to the RL via the Internet from anywhere the user wishes and by any device s(he) prefers (PC, laptop, tablet, or smartphone), exploiting distance learning positives. Such an architecture is proposed in [28].

The application provides a platform for the users to use the RL. There are tasks granted to administrators and users. Table 4 lists the main tasks of the application for users and administrator groups.

Table 4. Tasks per users and administrators.

Users:	Administrators:
<ul style="list-style-type: none"> • User creation • User profile update • User Arduino sketch storage • User sketch compilation and uploading • Remote lab monitoring • Remote lab user interaction • Activating activities • Monitoring activity status 	<ul style="list-style-type: none"> • Setting configuration parameters • Shadow controller check (Enable/Disable) • Booking system (Enable/Disable) • Clearing users' sessions • Setting user group (Student/Administrator) • Setting xAPI status (enable/disable) • Creating activities

Additionally, the application implements the learning tools interoperability (LTI), which allows the platform to act as an external tool in a learning management system like Moodle.

From the above discussion, the platform can work in two modes. The first mode is "LMS-Free", which handles everything, like users' creation and authentication, security issues, activities, learning analytics, etc. The second mode is the LMS external tool, where the platform is called from an LMS for a particular activity (learning object), and the user executes the activity according to the instructions and finally submits the activity. The performed activity is assessed by the RL and the grade is updated in the LMS users' grade book. Additionally, the users' learning analytics are stored in a learning record store (LRS) as part of the experience API standard.

This is what differentiates the remote lab presented from the other cases mentioned in Section 2. The RL can assess the uploaded code, but on the contrary, the other remote labs only stay in the phase of monitoring the results of the sketch uploaded on the Arduino board.

To summarize the above-mentioned information and visualize it, we can see Figure 2. The user logs in to the HMU-RLP via the Internet using a web browser. The HMU-RLP is located on the campus of the HMU, in the Sensor Networks LAB.

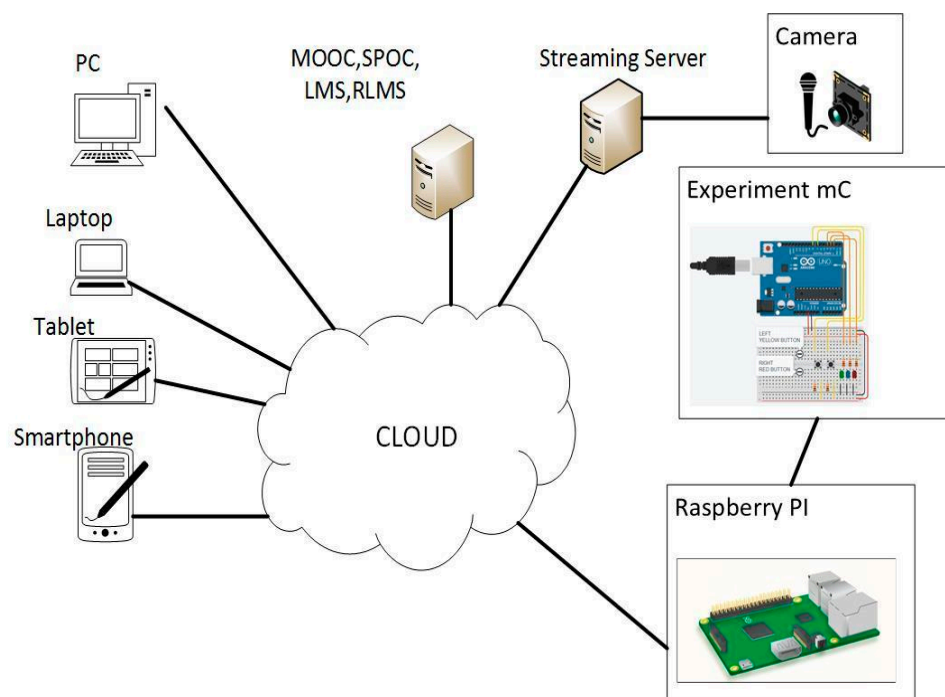


Figure 2. Generic architecture of remote laboratories.

HMU–RLP User Workflow

The user’s workflow of the HMU–RLP is presented in Figure 3. After the user logs in to the HMU–RLP, the user then has to activate an activity.

The main object in the RLP is the activity. The activity is the scenario or the experiment that will run on the RLP; everything begins with activity activation. The user activates an activity in the RLP and receives instructions about the scenario and also relevant documentation. After activation, the user creates the code required from the activity’s scenario in the Arduino IDE. It must be reassured that the code contains no syntax errors using the Arduino IDE Compiler. After verifying that the code is correct, the code is stored in the RLP users’ profile. The code must be compiled and uploaded in the Arduino experimental microcontroller.

When the new code is uploaded, then the user can remotely monitor the execution of the code via the camera/s of the RLP. Additionally, the user can interact with the experiment (turn on/off the lights, the heater, the FAN, or pressing the red or the yellow buttons); in this way, the activity’s scenario is tested. After testing the scenario, the user can check that they have followed the correct instructions from the activity status report.

When the user is ready, they can submit the activity for assessment. The activity submission concludes the experiment.

During the whole process, the user’s learning analytics profile is updated using xAPI statements.

As we mentioned, there are three types of assessments in the RLP. The logging of the path of the actions assesses the actions instructed to the user. The shadow microcontroller verification checks the code at a hardware level. And last, the machine learning check of the code, which uses a decision tree algorithm to verify that the code adheres to the instructions. All assessment test results are stored in the xAPI LRS as statements.

Finally, instructors can use reporting and visualization tools for further processing, the user’s learning path, and adaptive learning.

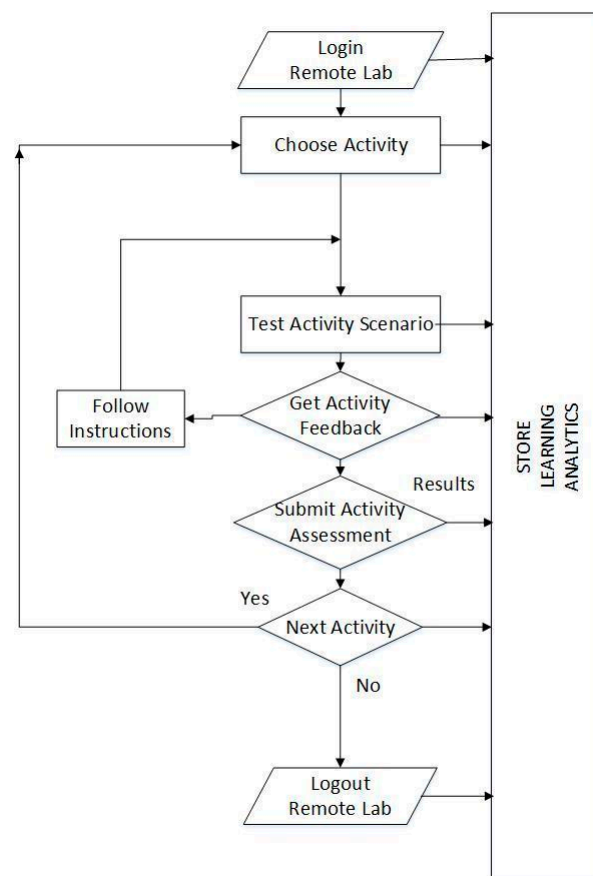


Figure 3. HMU-RLP user workflow.

4. Remote Lab Assessment Types

We will begin by explaining what we mean by the term assessment of experiment in the context of a remote lab. When a user is assigned an activity (experiment), the user needs to have the instructions for the activity, the relevant theory documentation, and the schematic of the connections on the microcontroller. The user has to make the required code and compile it to the Arduino IDE to verify that there are no syntax errors and store it in his profile sketches in the remote laboratory. Consequently, the user has to compile the code and upload it to the microcontroller via the remote laboratory. After this, the user has to check that the code created is doing what it is supposed to do, according to the activity instructions. This is where the assessment starts. How is the instructor informed that the code is correct and aligned with the activity's instructions? One way is to show the teaching assistant the results, but this cannot be performed since we are working remotely and there is not an available assistant. Another way is to send the code to the teaching assistant, which is time-consuming. Another way is to create a report, but also in this case there is no guarantee that the user is the one who performed the experiment or that the user did not copy the report of somebody else. So, the best way is for the code to be assessed automatically by the RL.

The remote lab has three types of assessment that can be implemented. The following subsections will present the HMU-RLP assessment types.

4.1. Actions Assessment

The first one is to monitor the “microactivities”, each action that the user performs on the RL, for example, login, logout, create, update, delete, compile, and upload a sketch, and interactions with the experiments, such as turn on/off the lights, the fan, the heating resistance, and change the position of the potentiometer.

Every action performed is logged and compared to the required “microactivities” path of the exercise. This type of assessment shows what exactly the user is doing on the RL. Also, it is very useful in a simulation scenario of an operator. In case there is a need to train an operator, when an alarm occurs indicating what sequence of action must be followed.

All of these “microactivities” are logged via xAPI statements to an LRS for learning analytics. Using xAPI reporting tools, the instructor can obtain conclusions about the user and how they performed in the experiment. For example, the number of experiment attempts, the duration of the experiment session, how many times the code was uploaded, what experiments the user performed, etc.

4.2. Shadow Microcontroller Assessment

The second type is to monitor the experimental microcontroller by another microcontroller called a “shadow” microcontroller. Ideally, the “shadow” microcontroller must be the same as the experimental microcontroller because the pins must be the same in both cases. All pins between the two microcontrollers are connected (exp. Mc A0-sha. Mc A0 ... exp. Mc Pin13-sha. Mc Pin13).

Using the shadow microcontroller, the instructor assesses the activity on a hardware level. The shadow microcontroller monitors what happens on the pins of the experimental microcontroller and reports it to the RPI.

The communication between the RPI and the shadow microcontroller is implemented via the I2C bus. We must mention that for protecting the I2C bus of the RPI, we must use a Bi-Directional Logic Level Converter due to different voltage levels between RPI (3.3 V) and the Arduino UNO (5 V).

There are two ways that we can implement the shadow microcontroller assessment. The “Activity Specific Firmware” and the “General Firmware”.

4.2.1. Activity-Specific Firmware

The “shadow” microcontroller is loaded with a specific firmware when an activity is activated. This firmware instructs the shadow microcontroller what to monitor from the experimental microcontroller. If the shadow microcontroller obtains the expected results, it reports to the RPI. The validity of the activity’s code is reported by the shadow microcontroller sending “success” or “failure” when it is inquired from the RPI. Then, the RPI sends an xAPI statement of the shadow controller report.

To elaborate on this case, we will present an example. Suppose that the activity instructs the user to create a code that will flash the green LED connected to pin 2 on the experimental microcontroller. As mentioned before, pin 2 of the shadow microcontroller is also connected with pin 2 of the experimental microcontroller (see Figure 4).

The shadow microcontroller is programmed to set pin 2 as the input and monitor the pulses that come to pin 2. The shadow firmware is created and stored from the instructor/creator of the activity (see Figure 5, section C), where the activity creation is enabled only for administrators/instructors.

The user, from their side, is instructed to create a code that flashes the green LED every 1 s. If the code is correct, then the green LED flashes every 1 s and the shadow microcontroller starts counting pulses from its side. Then, the shadow microcontroller reports “success” when it is inquired from the RPI. Now, in case the user made a mistake and instead of flashing the green LED flashes the red, which is connected to pin 4, which is a logical error, then the shadow microcontroller will not count any pulses on pin 2 and then will report “failure” when it is inquired from the RPI. The same result will occur if the user loads a different code, different than the code instructed.

As a second example, we have a scenario where the user is instructed to control the RL environment temperature. The user creates a code that monitors the temperature using pin A5 (see Table 5), and when the temperature exceeds the threshold value, then the user’s code must activate the fan connected to pin 11 (see Table 5). The shadow microcontroller firmware will monitor the value of the analog pin A5, and if it exceeds the threshold, pin

11 should be set to HIGH. If these conditions exist, then the shadow microcontroller will set the activity as successful; as long these conditions never occur, then the activity is set as failed.

It must be highlighted that the shadow microcontroller is transparent to the user, that the user does not know of its existence, so it is hidden in the RL. The drawback of this option is that when activating an activity, there is a delay in compiling and uploading the shadow microcontroller firmware, which may confuse the user.

Additionally, it must be clarified that the shadow controller-specific firmware is created by the instructor and creator of the activity. The code is in Arduino programming and it is based on a template of code that at the end sends the report SUCCESS or FAILURE to the RPI when it is requested from the RPI.

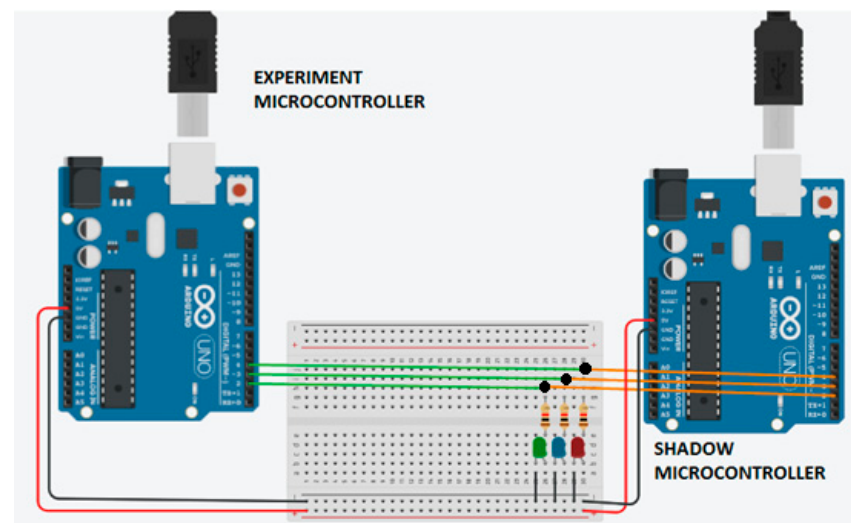


Figure 4. Experiment and shadow microcontrollers.

Table 5. Sensors and actuators connected to the experimental microcontroller.

1.	LED	Three LEDs connected to pins 2, 3, and 4
2.	Servomotor	One servo motor connected to pin 5
3.	Buzzer	One buzzer connected to pin 6
4.	Push Buttons	Two push buttons connected to pins 7 and 8
5.	H-Bridge	One HW95 H-Bridge connected on pins 9, 12, and 13
6.	Relay	One relay connected on pin 10, which turns on an LED stripe One relay connected on pin 11, which turns on a fan
7.	LCD Display	One LCD connected to A0 and A1
8.	Potentiometer	One potentiometer connected on pin A3
9.	Photoresistor	One photoresistor connected on pin A4
10.	Temperature sensor LM35	One temperature sensor connected on A5

Edit Activity

Activity Title

Turn on Red LED by pressing the red button. **A**

Please insert Activitys' title.

Activity Description

This activity requires the following to be done. The program does nothing until the user presses the red button. As long as the red button is pressed, the red LED will light up. **B**

The red LED is connected to PIN2, while the red button is connected to PIN8.

Please insert activitys' description.

Shadow microcontroller sketch.

NOTHING **C**

Please insert Shadow Controllers' code. If no needed fill 'NOTHING'.

Machine Learning Model

mycode[label]mandatory **D**

```
const int buttonPin = 8, ledPin = 2; void setup() { pinMode(buttonPin, INPUT); pinMode(ledPin, OUTPUT); } void loop() { bool myButtonState = digitalRead(buttonPin); if (myButtonState != LOW) { digitalWrite(ledPin, LOW); } else { digitalWrite(ledPin, HIGH); } }correct|buttonPin = 8,ledPin = 3,void setup()
```

Please insert Machine Learning model (optional).

Activitys' Image

Browse... No file selected. **E**

Please, select activitys' image file.

Microactivities path

LIGHTS_ON|CompileSketch|UploadSketch|RED_BUTTON_ON|RED_BUTTON_OF F|LIGHTS_OFF **F**

For creation of microactivities path, use the tool below.

Select microactivity

LIGHTS_ON
LIGHTS_OFF
CompileSketch
UploadSketch

Add Micro Activity Clear

Create Microactivities Path

Submit Cancel

Figure 5. Activity creation/update.

4.2.2. General Firmware

In the case of the general firmware, we upload a general firmware on the shadow microcontroller, which monitors all the pins of the experimental microcontroller. When there is a change in the pin status, the shadow microcontroller sends a report to the RPI of a string with the status of all of the pins (High/Low for digital pins and the value read from analog pins). Then, the RPI sends the pins' status reports as xAPi statements to the

LRS. The instructor, using the xAPI tools and filtering, can view the sequence of pin status in the specific activity and understand that the code is working as it should.

As can be understood, in this case, there is the option for bigger users' learning analytics and reporting. We will use the above two examples to elaborate on this case.

In the first example, where the user is assigned to flash (turn on/off) the LED connected to pin 2, each time that pin 2 changes status, the RPI is triggered and the pin status report is sent. Next, the RPI sends the xAPI statement to the LRS. Then the teacher can filter the users' profile statements for the specific activity and see the status of all of the pins. It will be visible that pin 2 flashes, changing from LOW to HIGH and from HIGH to LOW.

In the second example, where the user has to control the environment temperature, as the temperature increases, the value of pin A5 increases. As long as the value increases, the status of the pins changes, and for each change, the RPI receives the pins' status reports from the shadow microcontroller, which are sent as xAPI statements. When the value of pin A5 reaches the threshold, then pin 11 will be set to HIGH. Filtering the xAPI statements, the instructor will see that the value of pin A5 increases and after the value of A5 exceeds the threshold value, pin 11 will be set to HIGH.

The positive aspect of this method is that the shadow controller is loaded once with the generic firmware and there is no delay in loading the firmware each time an activity is activated. The drawbacks of this method are that there are a large amount of data stored in the LRS and that the instructor must check for the validity of the user's code, which is time-consuming. We are working on automating this process.

The report that the Arduino sends and is received from the RPI is in the following format: `{'digital_pin_statuses': [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0], 'analog_pin_values': [1023, 993, 929, 816, 25, 25]}`

The drawback in this case is that there are a lot of data sent due to unstable analog pins values, which is why in the code we have implemented a percentage threshold of difference. This method of assessment is a work in progress, and we suggest the use of activity-specific firmware.

4.3. Artificial Intelligence Assessment

The third type of assessment in the RL is to use artificial intelligence (AI) to assess the uploaded code. Before we start the presentation, we must highlight that this type of assessment is a broad topic that needs particular research and development. Different algorithms can be implemented, like neural networks, decision trees, etc., or even using Ghat-GPT. For this paper, we created a simple use case for proof of concept, which will be elaborated on in future works.

In this use case, we created an algorithm based on a decision tree. For each activity activated, there are stored defined accepted codes and wrong codes. When the user uploads the required code and submits the activity for assessment, the code is compared with the AI model and replies if it is successful or wrong.

The model is stored during the activity creation, as shown in section D in Figure 5. The model is created by the instructor and can be enriched gradually using users' successful or wrong submitted codes.

In the paper [38], they propose a generic AI-based technique for assessing student performance in conducting online virtual and remotely controlled laboratories. Their suggestion is based on the dynamic use of the mouse, which is different than our suggestion, but has a similarity in breaking an experiment into different stages or user steps, which in our case we call "microactivities".

Based on these three types of assessment, the RL can reply if the code submitted by the user aligns with the scenario of the experiment or not, and reply accordingly, updating the grade book or sending the relevant xAPI statements for the profile of the user.

4.4. Assessment Types Roles

For a better understanding, we must clarify that each assessment type is independent and has a particular role in the assessment process. Also, each activity's assessment type result is logged separately to obtain the final activity grade.

Action assessment: Each activity has a set of paths of actions to be followed. This is used for training purposes in a use case where a scenario has to be followed. For example, a user who receives an alarm or an indication has to perform a specific workflow. In this assessment, it is monitored that the users use the experiment and test it, and we can avoid cases where users log in to the RL and only log their session time.

Shadow microcontroller assessment: The role of the shadow microcontroller is to report the status of the experimental microcontroller. For example, if there is an activity where the user has to log and report some analog values on the experimental microcontroller, who will verify that the values reported are correct and not copied from another report? The comparison of the user's report and the statements of the shadow microcontroller will show the real activity testing.

Artificial intelligence assessment: This is used to verify that the code used by the user is according to the code expected. In this case, if the instructor wants to teach a specific programming method, for example, "while loops", and does not want the use of "for loop", this can be easily traced.

4.5. Activity Creation

In Figure 5, it can be seen how an activity is created in the platform. In the fields from A to F, we can see the information that needs to be filled in by the activity creator.

- A. Activity Title
- B. Activity Description–User Instructions
- C. Shadow Controller Code
- D. Machine Learning Models
- E. Activity Image
- F. Microactivities–User action path

5. Experience API Statements and Tools

In the paper, we have mentioned the experience API (xAPI), formerly known as Tin Can API. It is a specification for learning technology that allows for data collection about a wide range of a person's experiences (both online and offline). Developed by the Advanced Distributed Learning Initiative (ADL), xAPI provides a way to store, manage, and share data about learning experiences in a consistent format and can be used to integrate various learning tools and platforms. It is often seen as a successor to SCORM (Sharable Content Object Reference Model), offering more flexibility and capabilities for tracking learning experiences.

The basic components of xAPI are the following:

1. **Statements:** At the heart of xAPI are "statements" that record what a learner has done. A statement is usually formatted as "I did this" or "[Actor] [Verb] [Object]". For example, "Manos logged in the Remote Lab 1", "Manos activated activity No 1 on Remote Lab 1", or "Manos passed activity No 1 on Remote Lab 1".
2. **Actor:** The individual or group that the statement is about.
3. **Verb:** Describes the action taken by the actor.
4. **Object:** What the action is performed on.
5. **Result:** Additional data about the outcome (optional).
6. **Context:** Additional data to help understand the context in which the action occurred (optional).

xAPI statements can be generated by tools, simulations, quizzes, serious games, or learning environments. Depending on the learning analytics required, the designer decides

what kind of user activities need to be stored. The xAPI statements are stored in and retrieved from the learning record store (LRS).

The project uses an LRS, which is a learning locker [39] hosted in the site <https://lrs.nile.hmu.gr/> (accessed on 28 March 2024) and provided by the Natural Interactive Learning Games and Environments Lab (NILE) of the HMU. While the learning locker primarily serves as an LRS, it also offers some basic reporting and visualization features. Users can build custom dashboards and reports using the built-in reporting engine or by integrating with external visualization tools (see Figures 6 and 7).

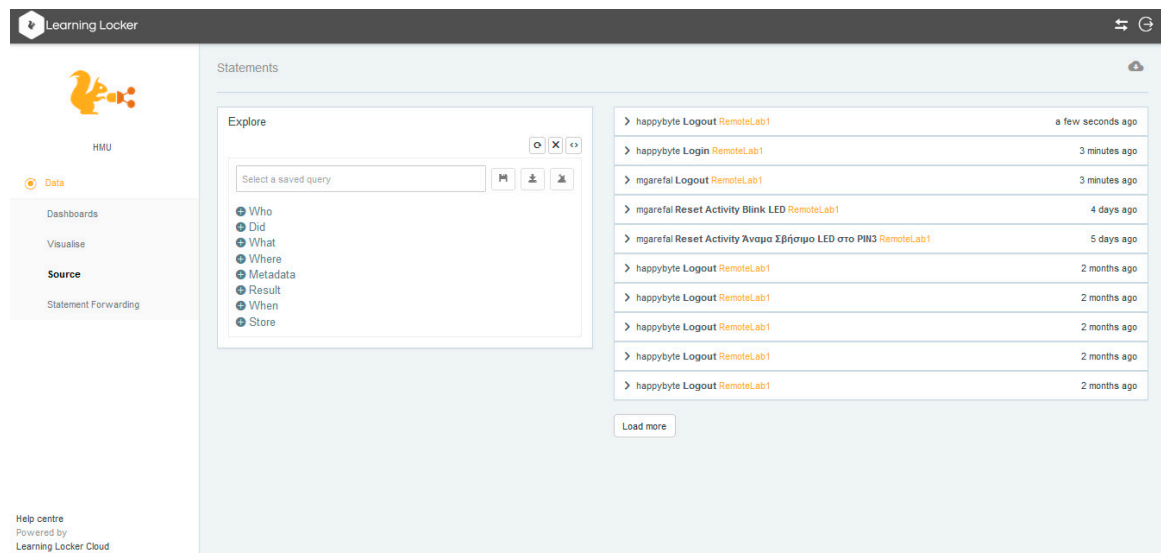


Figure 6. Learning locker reporting features.

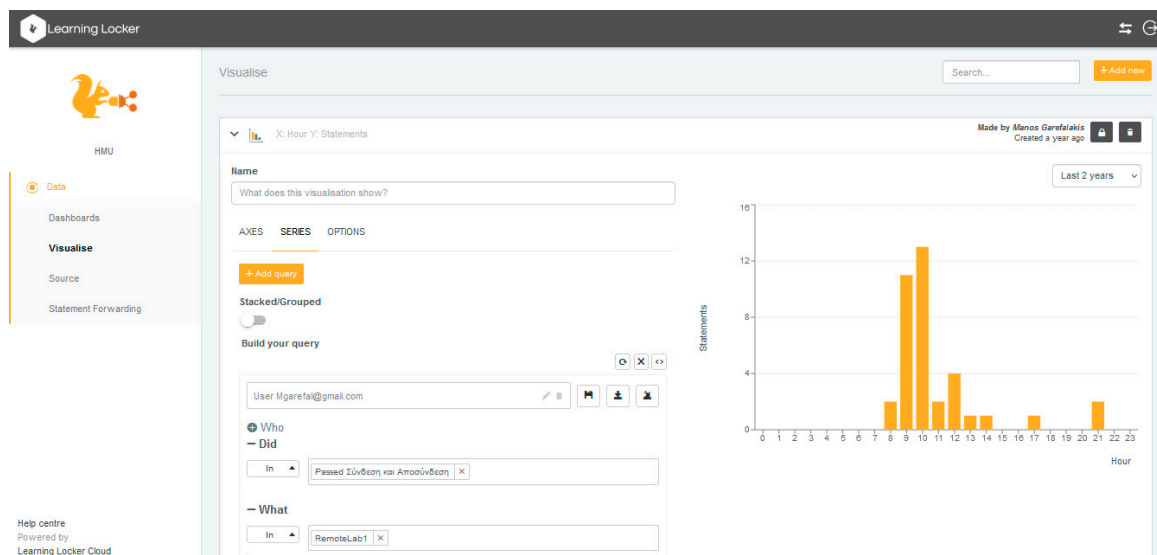


Figure 7. Learning locker visualization features.

During our research, we also found other commercial LRS reporting and visualizing tools, like Watershed LRS, Rustici LRS, etc.

Using the experience works of the HMU in the papers [40,41], we created a use case webpage (<https://garefalakis.eu/xAPI-Dashboard/examples/verbs2.html> (accessed on 28 March 2024)) where a graph of the statements stored in the LRS of the RL can be viewed (Figure 8).

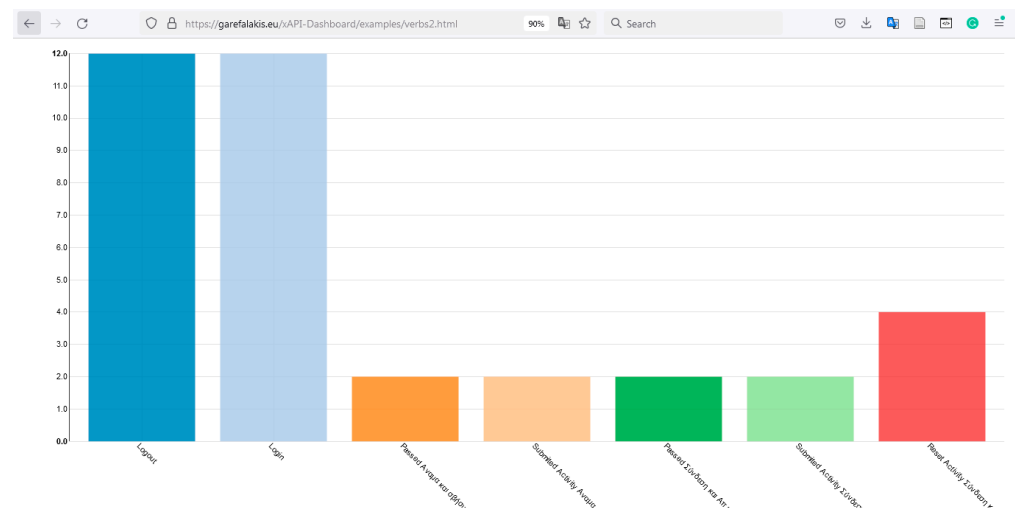


Figure 8. xAPI graph representation.

Additionally, the xAPI statement viewer can be seen in the webpage (https://garefalakis.eu/xAPI/1.0/original_prototypes/StatementViewer/ (accessed on 28 March 2024)), which can show all of the statements stored in the LRS (Figure 9). The two web pages were created for demonstration purposes and will be developed shortly on the project's site (<https://iot.hmu.gr> (accessed on 28 March 2024)).

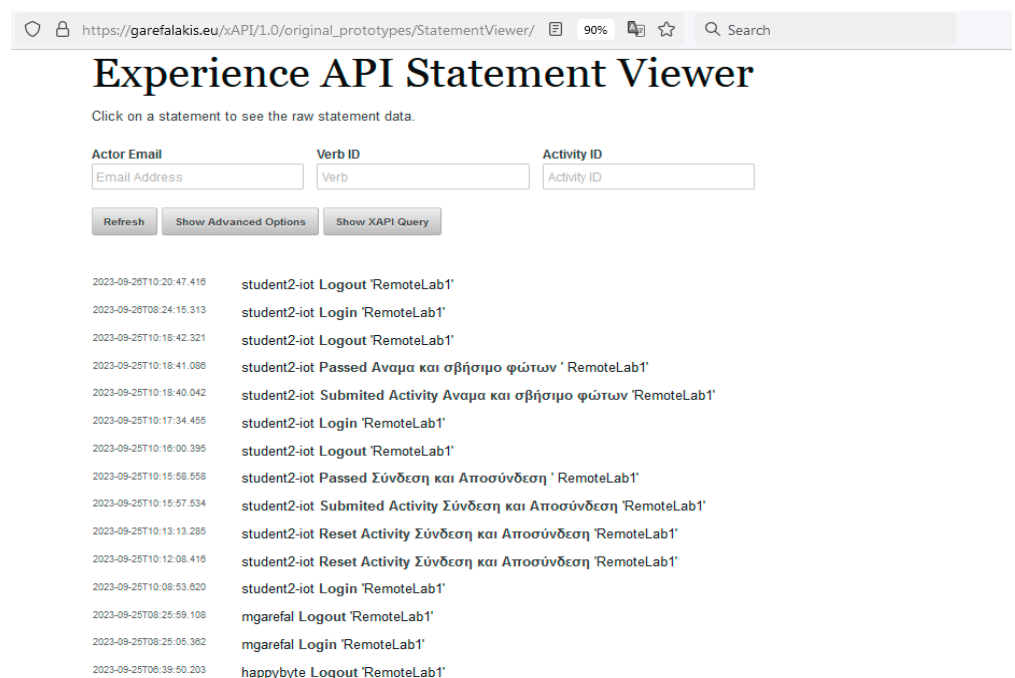


Figure 9. xAPI statement viewer.

6. Remote Lab Experiments

To give RL flexibility, the experiment controller is connected to many sensors and actuators. This gives the flexibility to create different activities and scenarios for the users' training. In Table 5, the list of electronic parts and the pins they are connected to can be seen. In Figure 10, is depicted the schematic, of all the components connected to the experiment controller, and in Figure 11, are depicted all the components connected.

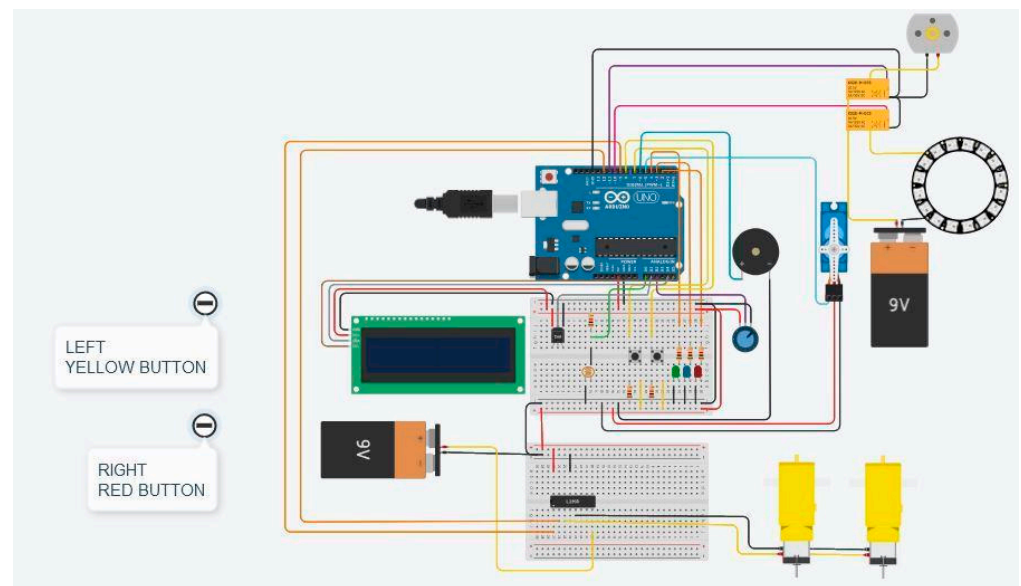


Figure 10. Tinkercad sensors and actuators connected to the experimental microcontroller.

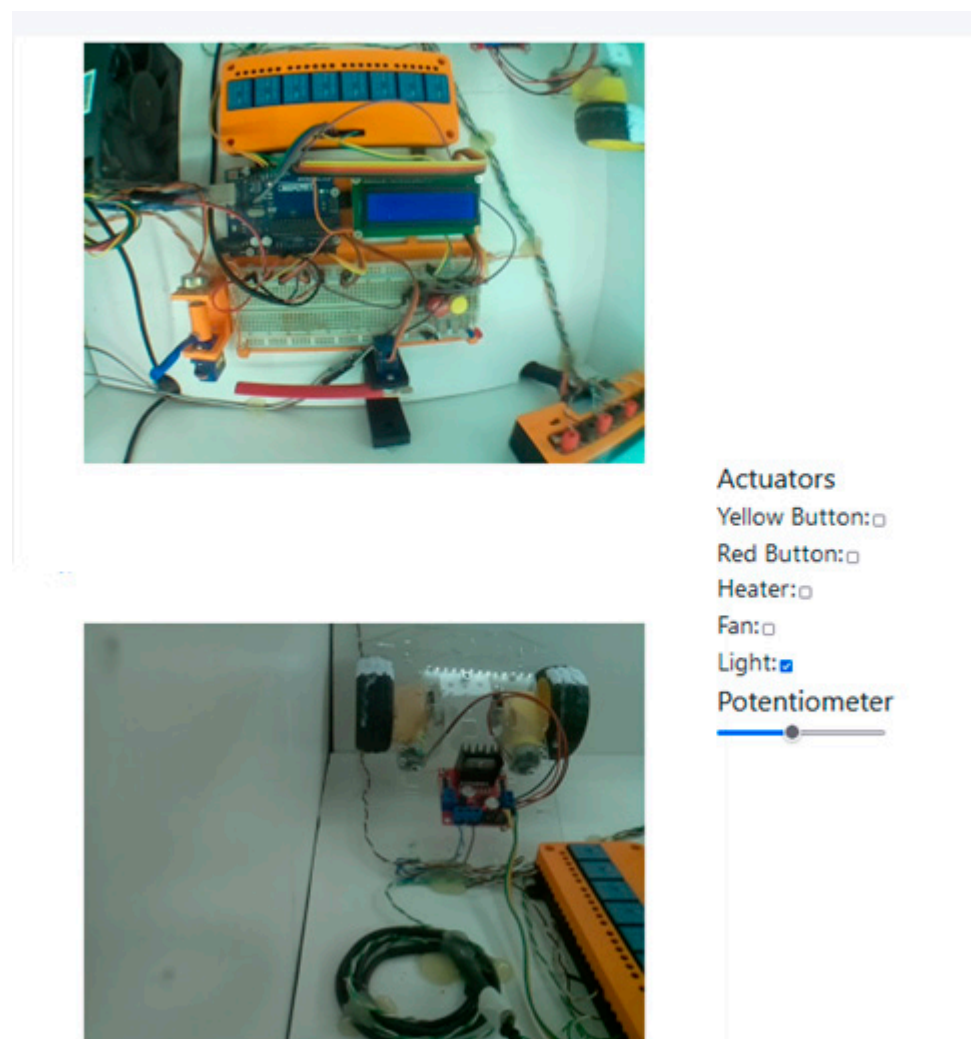


Figure 11. Experimental microcontrollers with connected sensors and actuators.

Apart from the components connected to the experimental microcontroller, other parts are connected to the RPI. These parts are activated by the user from the user interface. The user, by using these parts, changes some conditions in the RL, and the sensors sense the changes. For example, an experiment is to write a code that the microcontroller monitors when the environmental light of the RL is turned off and the microcontroller turns on the LED stripe connected to pin 10. The list of these parts is given in Table 6.

Table 6. External parts.

Relay 1–Activates RL lights
Relay 2–Pushes red button
Relay 3–Pushes yellow button
Relay 4–Activates heating resistor
Relay 5–Activates fan
Relay 6–Future use
Relay 7–Future use
Relay 8–Future use

7. Discussion and Conclusions

In this paper, we presented the related literature on the subject of remote laboratories, specifically on the RLs that aim to teach microcontroller programming of the Arduino family. The literature revealed the remote laboratory management systems (RLMSs) that exist and host a great number of such RLs, some of which are Labsland, GO-LAB, and RELLE. Also in this review, it was found that the RLP we have created in the Sensor Network LAB of the HMU shares the same architecture with a great number of RLs found in the literature. The different features of the HMU-RLP that were not found in the literature relate to the fact that, in our platform, the assessment of the experiments is not performed only by the users. In most RL cases, the user uploads his code to the microcontroller and then monitors the results on the streaming video and also interacts with the RL to verify that the code is doing what it is supposed to do.

In the HMU-RLP, we have introduced three types of user assessment: The first type monitors each action users perform over the web page offered by the RL. The second type monitors the activities of users at the hardware level. To this end, a shadow microcontroller is used that monitors the pins of the microcontroller programmed by the users. The third type automatically assesses the code uploaded by the users, checking its similarity with the prototype code uploaded by the instructors. A trained AI model is used to this end. For the assessments provided by the HMU-RLP, the experience API (xAPI) standard is exploited to store users' learning analytics (LAs). The LAs can be processed by the instructors for the students' evaluation and personalized learning. These assessment types work together toward the pervasive and supervised learning with which the HMU-RLP project intends to comply.

Future work planned for development of the HMU-RLP includes the following topics:

- We intend to further develop the features of our platform that exploit xAPI statements with users' learning analytics data to create personalized learning paths according to the adaptive and pervasive learning paradigm. We intend to follow a hybrid approach so that this process works both manually, under the supervision of the instructors with decision support offered by our system, and automatedly, so users are automatically assessed and tutored by the system.
- Further development of the AI type of assessment offered by our platform for automatically checking user coding. To this end, we plan to fine-tune a pretrained open-source large language model (e.g., Llama2) to assist the user with the coding actions that must be followed for a specific activity or provide feedback on a sketch that is not aligned with the activity scenario.

- The literature revealed that many RLs are hosted in remote laboratory management systems (RLMSs). Although the HMU-RLP can currently accommodate several RLs, it cannot be considered an RLMS. Such an option will help in RL sharing and dissemination via the HMU-RLP.
- Creation of an RL that will be used in teaching microcontroller programming for the Internet of Things. There will be two remote laboratories, one of which will have an experimental microcontroller connected with sensors and actuators, like ESP32, that will communicate with an MQTT server, and one software RL that will host a Node-RED server where the user will develop an application that will interact with the MQTT server and will display in the user interface controls and charts.
- Thorough evaluation and testing of the HMU-RLP into real training and learning environments so we can scale its readiness level from an experimental proof-of-concept platform to a production-ready toolkit. The HMU-RLP has already been presented to teachers of Greek secondary education and we will soon have their opinions and evaluation. Next, it is planned for the teachers to use the HMU-RLP in their classes for teaching Arduino and IoT programming, and students will be able to further evaluate the HMU-RLP.
- The implementation of the H/W configuration of the RL by the user remotely, as is described in the paper [20]. The user will be able to use more components connected to the Arduino board by switching and enabling different connections to new circuits remotely, using relay matrixes.
- Implementation of augmented and virtual reality applications for the user to see the experiments working and interact with them using AR and VR technology.

Author Contributions: Conceptualization, M.G., Z.K. and S.P.; methodology, M.G. and Z.K.; validation, M.G. and Z.K.; formal analysis, M.G. and Z.K.; investigation, M.G. and Z.K.; resources, M.G. and Z.K.; data curation, M.G. and Z.K.; writing—original draft preparation, M.G.; writing—review and editing, M.G. and Z.K.; visualization, M.G. and Z.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. 1876–2019—IEEE Standard for Networked Smart Learning Objects for Online Laboratories | IEEE Standard | IEEE Xplore. Available online: <https://ieeexplore.ieee.org/document/8723446> (accessed on 30 August 2023).
2. Zutin, D.G.; Auer, M.E.; Maier, C.; Niederstätter, M. Lab2go—A repository to locate educational online laboratories. In Proceedings of the IEEE EDUCON 2010 Conference, Madrid, Spain, 14–16 April 2010; pp. 1741–1746. [CrossRef]
3. Tinkercad—From Mind to Design in Minutes. Tinkercad. Available online: <https://www.tinkercad.com/> (accessed on 13 March 2024).
4. Panagiotakis, S.; Karampidis, K.; Garefalakis, M.; Tsironi-Lamari, A.; Rallis, I.; Kamarianakis, Z.; Papadourakis, G. Remote Arduino Labs for Teaching Microcontrollers and Internet of Things Programming. In Proceedings of the 2022 31st Annual Conference of the European Association for Education in Electrical and Information Engineering (EAEEIE), Coimbra, Portugal, 29 June–1 July 2022.
5. Garefalakis, M.; Panagiotakis, S. Integration of a Remote Lab with a Learning System for training on Microcontrollers’ programming. In Proceedings of the 27th PanHellenic Conference on Progress in Computing and Informatics—PCI 2023, Lamia, Greece, 24–26 November 2023.
6. Villar-Martinez, A.; Ortiz-de-Zarate, L.; Rodriguez-Gil, L.; Hernandez-Jayo, U.; Garcia-Zubia, J.; Angulo, I.; Terkowsky, C.; Ortelt, T.R.; Wilkesmann, U.; Nowak, R.; et al. LabsLand Electronics Laboratory: Distributed, Scalable and Reliable Remote Laboratory for Teaching Electronics. In *Open Science in Engineering*; Auer, M.E., Langmann, R., Tsiatsos, T., Eds.; Lecture Notes in Networks and Systems; Springer Nature: Cham, Switzerland, 2023; Volume 763, pp. 261–272. [CrossRef]

7. de Zarate, L.O.; Angulo, I.; Villar-Martínez, A.; Rodríguez-Gil, L.; García-Zubía, J. Remote Laboratory for the Development of Customized Low-Power Computing and IoT Systems. *Lect. Notes Netw. Syst.* **2023**, *763*, 249–260. [CrossRef]
8. Villar-Martínez, A.; Rodríguez-Gil, L.; Ortiz-de-Zarate, L.; Hussein, R.; Orduña, P. ARM Distributed and Scalable Remote Laboratory for Texas Instruments Launchpad Boards. *Lect. Notes Netw. Syst.* **2023**, *763*, 177–186. [CrossRef]
9. Buitrago, P.A.; Camacho, R.; Pérez, H.E.; Jaramillo, O.; Villar-Martínez, A.; Rodríguez-Gil, L.; Orduna, P. Mobile Arduino Robot Programming Using a Remote Laboratory in UNAD: Pedagogic and Technical Aspects: Experience Using a Remote Mobile Robotics Laboratory at UNAD. *Adv. Intell. Syst. Comput.* **2021**, *1231*, 171–183. [CrossRef]
10. Del Villar, I.; Rodríguez-Gil, L.; Orduña, P. Learning CAN Bus Communication with a Remote Laboratory. 2022. Available online: <https://ieeexplore.ieee.org/abstract/document/9766633/> (accessed on 28 March 2024).
11. Sapeha, A.; Zlatkova, A.; Poposka, M.; Donchevski, F.; Karpov, K.B.; Todorov, Z.; Efnusheva, D.; Kokolanski, Z.; Sarjas, A.; Gleich, D.; et al. Learning Management Systems as a Platform for Deployment of Remote and Virtual Laboratory Environments. 2022. Available online: <https://repo.bibliothek.uni-halle.de/handle/1981185920/78898> (accessed on 28 March 2024).
12. da Silva, R.C.; de Magalhães Netto, J.F.; Lopes, A.M.M.; de Menezes, M.F.; Menezes, R.A. ERPLab: Remote Laboratory for Teaching Robotics and Programming. 2023. Available online: <https://ieeexplore.ieee.org/abstract/document/10343379/> (accessed on 28 March 2024).
13. Oballe-Peinado, Ó.; Castellanos-Ramos, J.; Sánchez-Durán, J.A.; Navas-González, R.; Daza-Márquez, A.; Botín-Córdoba, J.A. Fpga-Based Remote Laboratory for Digital Electronics. 2020. Available online: <https://ieeexplore.ieee.org/abstract/document/9163676/> (accessed on 28 March 2024).
14. Navas-González, R.; Oballe-Peinado, Ó.; Castellanos-Ramos, J.; Rosas-Cervantes, D.; Sánchez-Durán, J.A. Practice Projects for an FPGA-Based Remote Laboratory to Teach and Learn Digital Electronics. *Information* **2023**, *14*, 558. [CrossRef]
15. Bukovac, A.; Pleše, E.; Maravić, U.; Petrović, P.; Jaguš, T. Teaching Programming and Microcontrollers with an Arduino Remote Laboratory Application. 2023. Available online: <https://ieeexplore.ieee.org/abstract/document/10159730/> (accessed on 28 March 2024).
16. Martin, S.; Fernandez-Pacheco, A.; Ruipérez-Valiente, J.A.; Carro, G.; Castro, M. Remote experimentation through Arduino-based Remote Laboratories. *IEEE Rev. Iberoam. De Tecnol. Del Aprendiz.* **2021**, *16*, 180–186. [CrossRef]
17. Terauds, M.; Smolaninovs, V. Remote Laboratory for Microcontroller Programming Course. 2022. Available online: <https://ieeexplore.ieee.org/abstract/document/9978868/> (accessed on 28 March 2024).
18. Seničić, Đ.; Matijević, M.; Tanasković, M.; De La Torre, L. An Implementation of a Web Laboratory Converting Off-Line Experiments into Remotely Accessible Experiments. In Proceedings of the Sinteza 2022—International Scientific Conference on Information Technology and Data Related Research, Belgrade, Serbia, 16 April 2022. [CrossRef]
19. Domski, W. Remote Laboratory Offered as Hardware-as-a-Service Infrastructure. *Electronics* **2022**, *11*, 1568. [CrossRef]
20. Scaffidi, C.; Distefano, S. A Remotely Configurable Hardware/Software Architecture for a Distance IoT Lab. 2021. Available online: <https://ieeexplore.ieee.org/abstract/document/9556236/> (accessed on 28 March 2024).
21. Tokarz, K.; Czekalski, P.; Drabik, G.; Paduch, J.; Distefano, S.; Di Pietro, R.; Merlino, G.; Scaffidi, C.; Sell, R.; Kuaban, G.S. Internet of Things Network Infrastructure for the Educational Purpose. 2020. Available online: <https://ieeexplore.ieee.org/abstract/document/9274040/> (accessed on 28 March 2024).
22. Admin. IOT-OPEN.EU: Introduction to the IoT Practical Projects in English—IOT-Open. Available online: <https://iot-open.eu/download/iot-open-eu-introduction-to-the-iot-practical-projects-in-english/> (accessed on 23 January 2024).
23. Costa, R.; Pérola, F.; Felgueiras, C. µLAB A Remote Laboratory to Teach and Learn the ATmega328p µC. 2020. Available online: <https://ieeexplore.ieee.org/abstract/document/9125336/> (accessed on 28 March 2024).
24. Da Silva, J.B.; De Oliveira, G.; Da Silva, I.N.; Mafra, P.M.; Meister, S.; Bilessimo, S. Block. Ino: Remote lab for programming teaching and learning. *Int. J. Adv. Eng. Res. Sci.* **2020**, *7*, 41–47. [CrossRef]
25. Platunov, A.; Kluchev, A.; Pinkevich, V.; Kluchev, V.; Kolchurin, M. Training Laboratories with Online Access on the ITMO. cLAB Platform. 2020. Available online: http://ceur-ws.org/Vol-2893/paper_12.pdf (accessed on 28 March 2024).
26. Jo, H.S.; Jo, R.S. Design and development of remote laboratory system to facilitate online learning in hardware programming subjects. In Proceedings of the 2020 13th International UNIMAS Engineering Conference (EnCon) 2020, Kota Samarahan, Malaysia, 27–28 October 2020. [CrossRef]
27. Fernández-Pacheco, A.; Martin, S.; Castro, M. Implementation of an Arduino remote laboratory with raspberry Pi. In Proceedings of the 2019 IEEE Global Engineering Education Conference (EDUCON), Dubai, United Arab Emirates, 8–11 April 2019; pp. 1415–1418. Available online: <https://ieeexplore.ieee.org/abstract/document/8725030/> (accessed on 23 January 2024).
28. Villar-Martínez, A.; Rodríguez-Gil, L.; Angulo, I.; Orduña, P.; García-Zubía, J.; López-De-Ipiña, D. Improving the scalability and replicability of embedded systems remote laboratories through a cost-effective architecture. *IEEE Access* **2019**, *7*, 164164–164185. [CrossRef]
29. HMU-RLP Hellenic Mediterranean University Remote Laboratory Platform. Available online: <http://rlp.hmu.gr:5000/> (accessed on 11 March 2024).
30. LabsLand—Home. Available online: <https://labsland.com/en> (accessed on 28 February 2024).
31. The REMOCLEC Project. Available online: <https://remoclec.eu/> (accessed on 13 March 2024).
32. The Remote Hub Lab. Remote Hub Lab. Available online: <https://rhlab.ece.uw.edu/> (accessed on 14 March 2024).
33. Home | Golabz. Available online: <https://www.golabz.eu/> (accessed on 28 February 2024).

34. Inicio | Laboratorio Remoto de Electrónica Digital. Available online: <https://fpga-lab.uma.es/> (accessed on 15 March 2024).
35. Home. IOT-Open. Available online: <https://iot-open.eu/> (accessed on 16 March 2024).
36. de Lima, J.P.C.; Carlos, L.M.; Simão, J.P.S.; Pereira, J.; Mafra, P.M.; da Silva, J.B. Design and implementation of a remote lab for teaching programming and robotics. *IFAC-PapersOnLine* **2016**, *49*, 86–91. [CrossRef]
37. Labs | RELLE—Remote Labs Learning Environment. Available online: <http://relle.ufsc.br/> (accessed on 28 February 2024).
38. Abd El-Haleem, A.M.; Eid, M.M.; Elmesalawy, M.M.; Hosny, H.A.H. A Generic AI-Based Technique for Assessing Student Performance in Conducting Online Virtual and Remote Controlled Laboratories. Available online: <https://ieeexplore.ieee.org/abstract/document/9973300/> (accessed on 28 March 2024).
39. “Home”, Learning Locker. Available online: <https://www.learninglocker.co.uk/> (accessed on 28 March 2024).
40. Arvaniti, D. Tracking learning with Experience API. March 2023. Available online: <https://apothesis.lib.hmu.gr/handle/20.500.12688/10503> (accessed on 7 February 2024).
41. Papadokostaki, K. Ubiquitous learning with Experience API. December 2017. Available online: <https://apothesis.lib.hmu.gr/handle/20.500.12688/8505> (accessed on 7 February 2024).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.