Journal Applied Computing 9[23]1-9: e6923109
Article

Educational IoT platform under DevOps methodology

Plataforma Educativa IoT bajo la metodología DevOps

Montecillo-Puente, Francisco-Javier * [a] & Tapia-Muñoz, Alejandro [b]

[a] TecNM campus Salvatierra • X-2309-2018 • 0000-0001-9540-9228
[b] TecNM campus Salvatierra• LUY-0235-2024• 0009-0000-7206-6496 • 1226134

Classification:

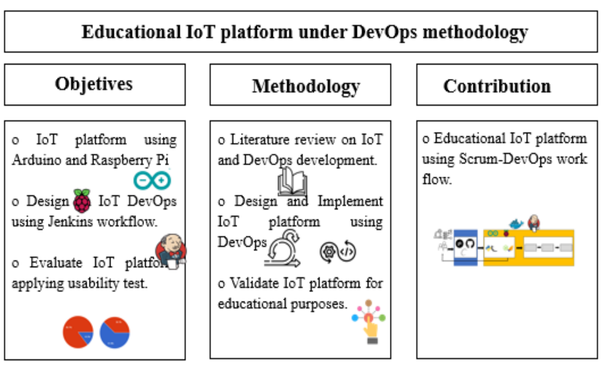
Area: Engineering
Field: Engineering
Discipline: Systems engineer
Subdiscipline: Computer Sciences

https://doi.org/10.35429/JAC.2025.9.23.6.1.9
History of the article:
Received: September 25, 2025
Accepted: December 30, 2025

* ✉ [famontecillo@itess.edu.mx]

Abstract

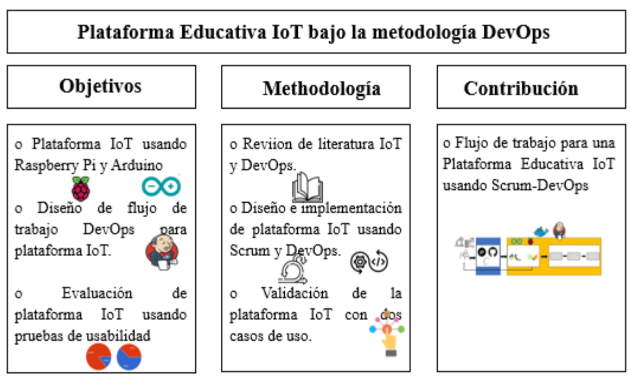
Electronics has evolved significantly through connectivity. Various devices with these capabilities exist, such as Raspberry Pi, Arduino, ESPs, and minicomputers. The Internet of Things [IoT] is the term used to encompass such devices. This technology includes sensors, actuators, communication protocols, user interfaces, data processing, security, administration and management. The industry has developed IoT cl platforms such as Arduino Cloud, AWS IoT, and Azure IoT. Implementing such platforms requires expertise in software development, communications, and electronics. This paper proposes the design and implementation general-purpose IoT platform under DevOps methodology.



Internet of things, Educational Platform, DevOps.

Resumen

Actualmente la electrónica ha evolucionado a través de la conectividad y se encuentra por todos lados [industria, vida diaria]. Existen distintos dispositivos en el mercado con estas capacidades como Raspberry Pi, Arduino, ESP o minicomputadoras. Internet de las cosas [IoT] es el término acuñado para englobar estos dispositivos. Esta tecnología incluye sensores, actuadores, protocolos de comunicación, interfaces de usuario, procesamiento de datos, seguridad, administración y gestión. La industria ha desarrollado plataformas como Arduino cloud, AWS IoT o Azure IoT. En este trabajo se propone el diseño e implementación de una plataforma IoT de propósito general bajo la metodología DevOps.



Internet de las cosas, Plataforma educativa, DevOps

Area: Development of strategic leading-edge technologies and open innovation for social transformation

Citation: Montecillo-Puente, Francisco-Javier & Tapia-Muñoz, Alejandro. [2025]. Educational IoT platform under DevOps methodology. Journal Applied Computing. 9[23]1-9: e6923109.

Introduction

The Internet of Things is a technology that enables devices to connect to each other and has found applications in manufacturing, healthcare, retail, transport, logistics, energy and agriculture [Wei, 2024]. This technology has seen a huge increase in connected devices, from 10.3 billion in 2018 to 25 billion in 2025. In terms of data volume, 86 petabytes [PB] of information were generated in 2022, and it is estimated that more than 1,100 PB will be generated in 2027. This technology is one of the main resources that feed artificial intelligence [AI] models and cloud computing [Flinders & Smalley, 2025].

From a socio-technological point of view, it encompasses different business opportunities based on this technology, including: hardware, software, networks, integrated platforms, standards and data analysis [Krotov, 2017].

The area of opportunity for IoT is so vast that platforms such as AWS IoT, Microsoft Azure IoT Hub, Google Cloud IoT, ThingWorx [PTC] IIoT, IBM Watson IoT, and Oracle IoT have been developed, to name a few. The work carried out by [Panagou et al., 2025] presents a comparative study of the most important characteristics of an IoT platform. The factors considered in their study are: 1] security, 2] scalability and performance, 3] interoperability, 4] data analysis and AI/ML integration, 5] edge computing support, 6] cost model and price-benefit ratio, 7] development tools and SDK support, 8] norms and standards, 9] OTA update capability.

With regard to IoT platform development methodologies, an analysis of platform design for the agro-industrial sector with interoperability as the main feature is described in [Muños et al., 2025]. In the same vein, a layer-based taxonomy is presented in [Arnold et al., 2022]. The taxonomy consists of:

- Infrastructure layer, which includes hardware, host, and data processing;
- Network layer, which includes physical data transfer and logical data transfer;
- Mediation layer, which incorporates data structure, data analysis type, analysis technology, external integration, and platform source code;
- Application layer, which includes APIs, implementation, and business.

Two of the most widely used platforms in academia due to their accessibility and cost are Arduino and Raspberry, [Arduino - Home, n.d., Raspberry – Home, n.d.]. These offer hardware, software, and a cloud platform and have a large academic community. The Arduino platform has strategic partners ranging from components to distribution: Adafruit Industries, SparkFun Electronics, Seeed Studio, DFRobot, Mouser Electronics, Element14, Tindie, and RobotShop. Currently, Qualcomm is one of the world's leading semiconductor and wireless technology companies. It has acquired Arduino, announcing the addition of AI, graphics and processing capabilities [Arduino – Qualcomm, n.d.].

Raspberry Pi already has networking, processing, AI, and graphics capabilities. It also has a wide range of components, from video cameras to screens. Another component that is also used in academia is the ESP32, which offers connectivity, software, hardware, and a cloud platform [Espressif Systems, n.d.].

The data ports offered by these components are:

- digital input-output,
- analogue input-output,
- I2C communication,
- serial communication,
- SPI communication,
- among others such as GPIO in the case of Raspberry Pi.

Therefore, any sensor or component with these characteristics can be connected to these devices. Having access to networks, whether via Ethernet or Wi-Fi, allows connection to the internet, which facilitates web development, REST services, or cloud services.

This work is divided into the following sections: problem statement, DevOps review [Singh, 2023], physical architecture design, logical architecture design, DevOps development and product list, DevOps sprint, case studies of platform use, and finally results and conclusions.

ISSN: 2531-2952
RENIECYT: 1702902

Problem statement

The Internet of Things is a reality both at home and in industry, although there are commercial platforms that offer IoT services. In the particular case of education, it is important to know this technology as part of job skills, but something more interesting is to study its development.

To implement an IoT platform, it is necessary to have expertise in different areas of knowledge such as software development, electronics, communications, automation, data analysis, project management and administration, to name a few. This article proposes the design and development of a general-purpose IoT platform based on Raspberry Pi and Arduino using the DevOps methodology.

The main interest of this project is to have a platform for the development of academic projects and to promote participation in student project calls. In addition, from an academic point of view, it incorporates the use of development technologies used in the workplace, such as: Scrum, DevOps, version control system [Git], networks, web services, front-end development, embedded devices, Python, HTML, CSS, and JavaScript.

DevOps Review

DevOps is a set of practices that promotes collaboration between development and operations teams in the life cycle of a software project [Singh, 2023; Lindemulder, G., & Kosinsk, 2025; Leszko, 2022; Erich et al., 2017]. DevOps adds new processes to agile methodologies, particularly CI/CD [continuous integration/continuous delivery] automation. The steps in the DevOps development lifecycle are as follows:

1. Planning: new features and functions.
2. Scheduling: the DevOps team develops the sprint according to the stories on the product list.
3. Building: new functionalities are integrated into the existing code, tests are performed, the new system is packaged and deployed.
4. Testing: tests are performed, usually automated, to verify that the appropriate standards and requirements are met.
5. Deployment: final tests are applied so that users can access the new software.
6. Implementation: the project moves into production.
7. Operation: the new project features are verified to ensure they work correctly and are available to users.
8. Monitoring: user feedback is collected and analysed, as well as lessons learned from the sprint, to improve future processes and products.

In particular, CI requires frequent code integration into a central repository and automated building and testing. This allows for rapid problem detection and feedback. CD provides functional software and enables proper management between speed of delivery and change control. CD can have two meanings: continuous delivery or continuous deployment. In this work, we will use continuous delivery.

There are many options on the market for implementing DevOps, particularly for implementing CI/CD. However, due to the academic nature of this work, it is necessary to design and implement a local infrastructure and use free tools. The tools we will use in this project are: Docker, Jenkins, GitHub, Openssl, Python, Flask, Jakarta, NodeJS, Glassfish, and Postgresql. In terms of hardware: computer, access point, Raspberry Pi, Arduino, compatible sensors and actuators.

Design of the physical architecture of IoT: Hardware

The physical components that make up the platform are: a] a computer that has the function of maintaining the system versions, configured with Docker, Jenkins, GitHub, and ssh keys; b] Raspberry Pi [there may be several], which has the function of reading sensors, controlling actuators, and communicating with the Arduino; and c] a client computer, where users use the platform to create their own projects. It uses the available software-hardware components, creates a web application to control its sensors or actuators, and creates and publishes new UI components and their physical configurations. It has a local repository for the development of its applications. The physical architecture is shown in Figure 1.

ISSN: 2531-2952
RENIECYT: 1702902

Montecillo-Puente, Francisco-Javier & Tapia-Muñoz, Alejandro. [2025]. Educational IoT platform under DevOps methodology. Journal Applied Computing. 9[23]1-9: e6923109.
https://doi.org/10.35429/JAC.2025.9.23.6.1.9

Box 1

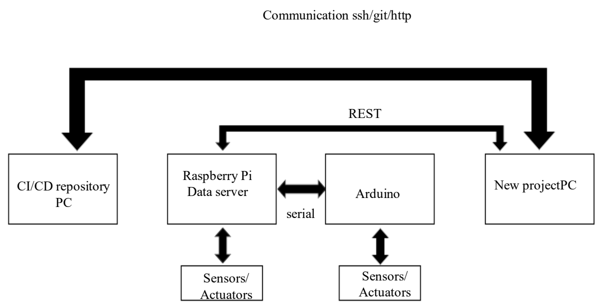


Figure 1
IoT diagram of physical components and their communication scheme: CI/CD repository, Raspberry Pi, Arduino, sensors-actuators, client.

Designing the logical architecture of IoT: Software

Code developers and testers are responsible for developing the system. Developers are generally responsible for writing and integrating code. The role of code tester has a major influence on the efficiency and quality of the CI/CD flow. This is because they have a perspective that helps the entire team avoid delays, reduce errors, and are responsible for the user experience. In addition, they define the acceptance/failure criteria. Therefore, they are responsible for promoting continuous improvement.

Within DevOps development, the CI/CD flow is important as it involves steps 2] programming, 3] building, 4] testing, and 5] deployment. Figure 2 shows how the development team programs code and submits it to the version management system, after which the server builds and performs tests. If there are errors, feedback is sent to the development team.

If there are no errors, it moves on to the CD block, where the code is put on hold to be tested against the acceptance criteria. Finally, it is published or goes into production. In our case, it is only published, as the users are the ones who will use these versions for their projects.

Box 2

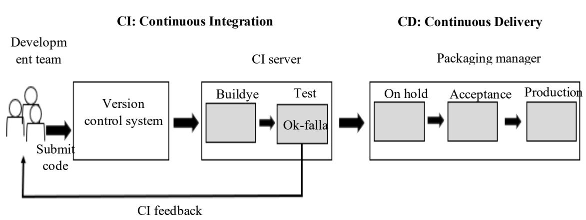


Figure 2
CI/CD DevOps flow diagram.

DevOps development team and initial product list

Specifically, the development team programmes new features based on the updated product list and then defines Sprints. The aim of this work is to develop an IoT platform using Raspberry Pi and Arduino for academic projects, incorporating sensors, actuators and a web-based UI component system. From the above, the initial product list consists of the following epics:

E0. Configuration of the working environment.

E1. Read data from temperature-humidity sensors, gyroscope, speed, GPS, fingerprint, buttons and distance using Arduino. Document connection diagrams and code.

E2. Control actuators such as motors and LEDs using Arduino. Document connection diagrams and code.

E3. Implement a serial communication model between Arduino and Raspberry Pi via messages.

E4. Read sensor data using Raspberry Pi. Document connection diagrams and code.

E5. Control actuators using Raspberry Pi. Document connection diagrams and code.

E6. Implement REST services to read sensor data or control actuators, Arduino or Raspberry Pi.

E7. Implement UI components to access REST servers for sensors and actuators.

The following tools are used for programming, construction and testing:

- Arduino Studio is used for Arduino,
- Python is used for Raspberry coding,
- Flask is used for REST services,
- Jakarta-PrimeFaces and NodeJS are used for UI components,
- GitHub-Git is used as a code management tool,
- Jenkins is used as a tool for building, testing and packaging code.

Figure 3 shows the workflow with the specific tools.

ISSN: 2531-2952
RENIECYT: 1702902

Montecillo-Puente, Francisco-Javier & Tapia-Muñoz, Alejandro. [2025]. Educational IoT platform under DevOps methodology. Journal Applied Computing. 9[23]1-9: e6923109.
https://doi.org/10.35429/JAC.2025.9.23.6.1.9

Box 3

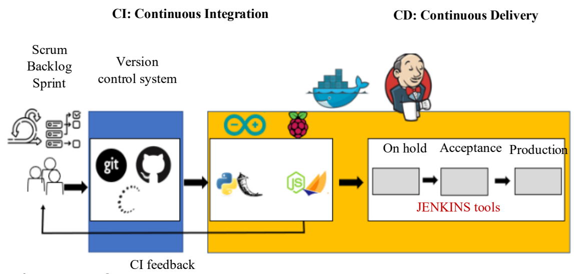


Figure 3

DevOps CI/CD flow diagram: the team performs Sprints, submits code to GitHub using a secure ssh connection. A Docker-Jenkins container configured for CI/CD. For testing, there is an Arduino-Raspberry system configured with Python, Flask, NodeJS, and Jakarta. Tools provided by Jenkins are used for the CD stage.

Own creation

For step 8] monitoring and CI feedback, Jenkins notification tools are used. Specifically in Jenkins, the CI workflow consists of [Leszko, 2022]:

- Triggers: the way in which the flow is executed. It can be caused by an external event, in our case a commit to GitHub.
- Stages: each stage is composed of steps that perform a simple operation. A stage can be, for example, building, testing, or releasing.
- Notifications: this is how Jenkins announces the status of the stages. This can be by email, chat, or in workspaces. The workspace creates boards that are visible to all team members, or it can also be a virtual board.
- In our case, we implemented the most fundamental workflow structure called ‘commit,’ which consists of:
- Verification: in this stage, the source code is downloaded from the repository.
- Compile-interpret: this stage compiles the code, if applicable. Arduino code is compiled, Python code is interpreted, NodeJS code is interpreted, and Java-Jakarta code is compiled.
- Unit testing: this stage consists of running a set of tests on the source code for validation.

DevOps for IoT platform: sprint

To keep things simple, we present the button-led development sprint. It consists of creating a circuit where a button is pressed and an LED is turned on through an Arduino circuit.

Each project carried out is accompanied by an Excel sheet containing the product list and Sprints. This can be transferred to the use of Scrum boards in tools such as Trello, Jira, Notion, Monday or any other. [Wynnedeta et al. 2024] present the development of an IoT application for sustainable environmental management based on Scrum with Raspberry Pi. In [Guerrero Ulloa et al., 2023], a review of agile tools for IoT development is conducted. Agile practices for IoT project development are presented in [Moedt van Bolhuis et. al, 2023]. These works use Scrum, and in our case, we implement DevOps.

The product list for this initial Sprint is shown in Table 1, following the aforementioned epics. For serial communication, a simple communication system was established where ‘code’ messages are sent, where the code refers to a sensor/actuator and a specific action.

Box 4

Table 1

Product list for the first sprint

History code	Description
E0-X-R	Configure the Raspberry Pi and connect to Arduino.
Configure	Connection diagram for the button on Arduino.
E1-S1-A	Program to read the button status.
Diagram	
E3-S1-A serial	Program the Arduino-Raspberry Pi communication to read the button status.
E6-S1-R rest	Program the REST service on the Raspberry Pi to read the button status.
E7-S1-C ui	Programming the UI component for the button.
E2-A1-A diagram	Diagram showing the LED connection on Arduino.
E2-A1-A program	Program to turn an LED on or off.
E3-A1-A serial	Program Arduino communication Raspberry Pi to turn the LED on or off.
E6-A1-R rest	Programming the REST service on Raspberry Pi to turn the button
E7-A1-C	on and off
ui	Programming the UI component for the LED.

The stories are written briefly and assigned a code E{N}-{S/A}{M}-{D} {task}, where E refers to Epic number N, sensor S or actuator A number M [X for another case], D refers to the device [A for Arduino, R for Raspberry Pi, C for the client], and finally the task [configure: configure the working environment, diagram: electrical diagram, programme: programming the device to read/write data, serial: serial communication, rest: REST programming on Raspberry Pi, and ui: UI programming of the component].

According to the diagram in Figure 2, the code and assets generated by the sprint are submitted to the CI/CD blocks through the version control system. That system generates an event and launches the start of the CI/CD process described in the next section.

To document the functionality of each sensor/actuator, a folder is created with the following structure:

· sensor/actuator-id,
o diagram: connection diagram figure
o codeArduino: Arduino test code
o codePython: Python code
o codeREST: REST test code, also Python
o codeUI: UI component test code

DevOps for IoT platform: CD/CI

The DevOps infrastructure for our platform has a very important feature, which is hardware integration, and as such cannot be directly added to a workflow. Since hardware integration generally requires interaction with the physical environment or human intervention, automation is not very obvious. When a new feature is added to our workflow, the following steps are followed:

1] Physical connection of components to Raspberry Pi/Arduino. A connection diagram must be available.

2] Programming in Raspberry Pi/Arduino for the component to function. There are two ways to test this: a] test the availability of the resource and its functionality in its default state; b] create multiple instances of the component in different states [available, unavailable, on, off, reading, writing, etc.]. In our case, we opt for the first approach. We also design the tests, which consist of checking the availability of the component and its default state.

3] Programming the REST service for the component using Flask. We also design the tests, which in this case consist of performing REST API queries, checking availability and default state.

4] Programming the UI component, which basically involves consuming the REST data API, checking availability and its default status.
5] Committing to the repository to trigger the CI/CD process.

The default status of the components, from point 2], refers to the component being available and configured in the Arduino or Raspberry. For example, in the case of the button, its default status is ‘not pressed’.

In this work, Jenkins is used as a tool for the CI/CD flow. It consists of the following agents: agent for Arduino components, agent for Raspberry components, agent for data server, agent for UI components. Although the UI agent contains the entire workflow, it is important to have granularity in each of the steps of the proposed development cycle.

The Arduino agent is used for compilation. The Raspberry agent uses the Python agent for the data server. Finally, for the UI agent, a Java or NodeJS agent is more than sufficient. See Figure 3.

Box 5

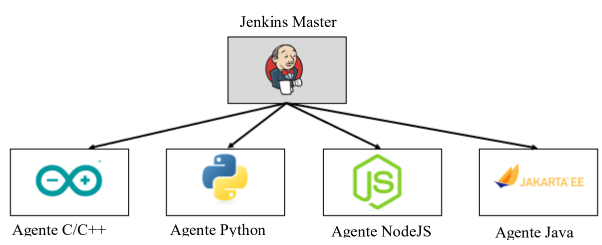


Figure 4
CI/CD agents: C/C++ for Arduino, Python for Raspberry, Java or NodeJS for UI components

Case study 1: mini weather station

In this first case study, a test group of 21 IT students was trained to use the platform. As an activity, they were asked to use the platform to create a mini weather station to measure temperature and humidity. Afterwards, the students were given a usability test for the platform. The results after the test are summarised in Table 2.

ISSN: 2531-2952
RENIECYT: 1702902

Box 6

Table 2.

Summary of usability test results. The Likert scale used in some questions:
1- Very difficult/Confusing, 2- Difficult/Confusing, 3- Neutral/Moderate, 4- Easy, 5- Very Easy/Understandable

Phase/Question	Description
Phase 1. Experience	
Academic projects completed	Average of 1.6 projects per student
Participation in academic calls for proposals	14.3% have participated, 85.7% have not participated
Phase 2. Pre-testing	
Electrical circuits-electronics	27% have done so, 73% have not done so
Use of Raspberry Pi	36.8% have used it, 63.2% have not used it
WEB programming	89.5% have programmed it, 10.5% have not programmed it
Version control	84.2% have used it, 15.8% have not used it
Command line	89.5% have used it, 10.5% have not used it
Phase 3. Testing	
Initial configuration	1:10%, 2:30%, 3:50%, 4:5%, 5: 5%
Electrical/electronic circuitry	60% Easy/Understandable, 40% Difficult/Confusing
REST communication programming	1:5%, 2:30%, 3:60%, 4:5%, 5: 0%
WEB programming, UI components	65% Easy/Understandable, 35% Difficult/Confusing
Scrum/DevOps management	1:20%, 2:20%, 3:55%, 4:5%, 5: 0%
Phase 4. Post-test	
Overall impression of the IoT platform	1:25%, 2:30%, 3:45%, 4:0%, 5:0%
Workflow	45% Easy/Understandable, 45% Difficult/Confusing
Use of the platform in future academic projects	1:15%, 2:35%, 3:50%, 4:0%, 5:0%
Participation in academic calls for proposals	50% Easy/Understandable, 50% Difficult/Confusing

The relevant results are participation and promotion of participation in events, as shown in Figure 4. It can be seen that the use of the IoT platform is relevant to students, as they show current participation in calls for proposals of 14.3% and their intention to participate of 61.9%. Furthermore, Table 4 shows that 81% of students would incorporate the platform into future academic projects [subjects].

Box 7

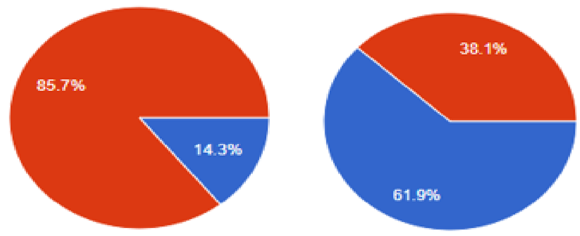


Figure 4

Left: previous participation in project calls vs Right: intention to participate after the trial

Case study 2: peregrine falcon trolley

As a case study of the platform's use, a team from the institution participated in the TecNM final in the InnoBotica category with an electric car. The participants requested support to integrate IoT tools to improve their prototype. They requested the integration of a fingerprint sensor to start the car, temperature indicators, a speed indicator, and the development of a graphical interface.

The proposal was to develop a responsive web application as a dashboard for the peregrine falcon car. A music player, voice synthesiser, and touch screen were also incorporated. In summary, the development of this project required 2HSM per member for only one month, 20 working days, 40 hours, and a team of six students. Figure 5 shows the prototype implemented in TonyKarts.

Box 8

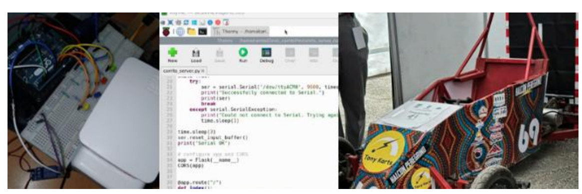


Figure 5

TonyKarts prototype participating in InnoBotica TecNM 2025

Results

This paper presented an IoT platform based on Raspberry Pi and Arduino under the DevOps methodology. To validate its functionality, two case studies were created. In the first, a test group was trained and asked to create a mini weather station. During the development of this activity, a usability test was adapted for use with the platform. The results obtained indicate that, in general, students can improve their academic projects by 81% and showed an interest in participating in calls for proposals by 61.9%.

ISSN: 2531-2952
RENIECYT: 1702902

Montecillo-Puente, Francisco-Javier & Tapia-Muñoz, Alejandro. [2025]. Educational IoT platform under DevOps methodology. Journal Applied Computing. 9[23]1-9: e6923109.
https://doi.org/10.35429/JAC.2025.9.23.6.1.9

From an academic perspective, there is no indication of creating teaching and learning strategies through the development of educational tools. In addition to using tools that job profiles demand, the global OCC in November shows 1,040 job offers using Python, 596 using REST, 150 using NodeJS, 353 for Frontend and 546 in DevOps. Git, on the other hand, is an almost mandatory tool in software development.

There is also a microSD memory image with Raspberry configuration scripts, NodeJS environment configuration scripts, a base project for Jakarta, a Docker image configured with Jenkins for DevOps, and an Excel spreadsheet for tracking your product list using Scrum.

Conclusions

Having platforms that use current tools serves three purposes: to comply with the academic content of the subjects, to promote participation in academic events, and to strengthen the graduate profile by using tools required in current job offers. Efforts should be made to cover as many subjects as possible that integrate knowledge. This platform covers subjects such as electronics, programming, web programming, computer networks, software engineering, and project management. In addition, it promotes soft skills such as teamwork, assertive communication, independent work, document writing, responsibility, and honesty.

Although these skills were not evaluated in this project, they should be considered at the end of the project in the lessons learned section. Future work includes creating academic practices that use this type of platform in the first semesters, for example, integrating object-oriented programming, data structures, databases, and electronics projects, which are subjects taught in the first three semesters.

Declarations

Conflict of interest

The authors declare no conflict of interest. They have no known competing financial interests or personal relationships that could have appeared to influence the article reported in this article.

Author contribution

Montecillo-Puente, Francisco-Javier: Contributed to the general idea of the project, development of the hardware-software platform, writing and analysis of results.

Tapia-Muñoz Alejandro contributed to hardware testing, Arduino-Raspberry programming, and document writing.

Availability of data and materials

The source code and materials can be provided upon request and agreement for academic use.

Funding

We would like to thank the Tecnológico Nacional de México campus Salvatierra, ITESS, for their support in the development of this project.

Acknowledgements

We would like to thank Jesús Antonio Carreño Sanchez and his team TonyKarts for inviting us to participate in their project at InnoBotica TecNM 2025.

Abbreviations

IoT	Internet of Things
PB	Peta Byte
AI	Artificial Intelligence
ML	Machine Learning
SDK	Software Development Kit
OTA	Over-The-Air
API	Application Programming Interface
I2C	Inter-Integrated Circuit
SPI	Serial Peripheral Interface
GPIO	General-Purpose Input / Output
REST	Representational State Transfer
CI	Continuous Integration
CD	Continuous Delivery
UI	User Interface

References

Antecedents

Wei, Y. [2024]. Business Insights into the Internet of Things: User Experiences and Organizational Strategies. International Journal of Advanced Computer Science and Applications, 15[7].

ISSN: 2531-2952
RENIECYT: 1702902

Flinders, M., & Smalley, I. [2025, November 27]. Cómo las empresas modernas están utilizando los datos de IoT para estimular la innovación. IBM.

Krotov, V. [2017]. The Internet of Things and new business opportunities. Business Horizons, Volume 60, Issue 6, Pag. 831-841, ISSN 0007-6813,

Muñoz, M., Torres, M., Gil, J.D., and Guzmán, J.L. [2025]. An Internet of Things platform for heterogeneous data integration: Methodology and application examples. Journal of Network and Computer Applications, Volume 240, 2025, ISSN 1084-8045,

Panagou, I. C., Katsoulis, S., Nannos, E., Zantalis, F., & Koulouras, G. [2025]. A Comprehensive Evaluation of IoT Cloud Platforms: A Feature-Driven Review with a Decision-Making Tool. Sensors, 25[16], 5124.

Arnold, L., Jöhnk, J., Vogt, F. & Urbach, N. [2022]. IIoT platforms' architectural features – a taxonomy and five prevalent archetypes. Electron Markets 32, 927–944

Basics

Arduino - home. [n.d.].

Raspberry –home [n.d.].

Arduino - qualcomm. [n.d.].

Espressif Systems. [n.d.].

Singh, N. [2023]. CI/CD pipeline for web applications. International Journal for Research in Applied Science and Engineering Technology, 11[5], 5218–5226.

Lindemulder, G., & Kosinski, M. [2025, November 27]. DevOps. ¿Qué es DevOps?

Leszko, R. [2022]. Continuous Delivery with Docker and Jenkins. Third Edition. Packt Publishing. 978-1-80323-748-0.

Erich, F., Amrit, C. & Daneva, M. [2017]. A Qualitative Study of DevOps Usage in Practice. Journal of Software: Evolution and Process.

Supports

Guerrero Ulloa, G., Rodríguez-Domínguez, C. & Hornos, M. [2023]. Agile Methodologies Applied to the Development of Internet of Things [IoT]-Based Systems: A Review. Sensors. 23. 790.

Differences

Wynnedeta, D. E., Suryatiningsih & Telnoni P. A. [2024]. Scrum-Based Development of CLEIN Mobile Application and IoT Smart Waste Bins: Advancing Digital Solutions for Sustainable Environmental Management. Procedia Computer Science, v. 245, 2024, p. 943-952, ISSN 1877-0509,

Discussions

Moedt van Bolhuis, W., Bernsteiner, R., Hall, M., & Fruhling, A.. [2023]. Enhancing IoT Project Success through Agile Best Practices. ACM Trans. Internet Things 4, 1, Article 5 [February 2023], 31 pages.

ISSN: 2531-2952
RENIECYT: 1702902