Entramado
ISSN: 1900-3803, e-ISSN: 2539-0279
Volumen 22, número 2, julio-diciembre 2026

Editorial Universidad Libre

Developing a Management System for Course Enrollment Amendments: Implementing Google Apps for an Efficient Workflow

https://doi.org/10.18041/1900-3803/entramado.2.13429

Carlos Gaviria-Mendoza
Universidad Militar Nueva Granada, Cajicá, Colombia

John F. Aguilar-Sánchez
Universidad Militar Nueva Granada, Cajicá, Colombia

Cyndi Ospina-Garcia
Universidad Militar Nueva Granada, Cajicá, Colombia

Abstract

Efficient course enrollment amendments require scalable tools to manage administrative complexities. This study develops and evaluates a Google Apps Script (GAS)-based smart office system implemented in a university to modernize online asynchronous workflows and analyze staff adoption of the system. The descriptive research encompassed 44 academic units managing 9,914 requests from 5,432 students across 226 interaction routes. The results demonstrated a 66% improvement in response time efficiency compared to the institutional benchmarks. A critical efficiency threshold was identified, where staff adoption reached 100%, indicating that low-code adoption is driven by administrative survival during peak workloads. The system successfully managed high-complexity structures by simulating relational data in a free-tier cloud environment. This implementation establishes a robust knowledge management framework, proving the cost-effective potential of low-code solutions for digital transformation in higher education. The findings confirm a non-linear correlation between workload and technology adoption, providing a verified background for institutional scalability.

Keywords

Management system; data system; cloud service; Apps Script; Google Suite; course enrollment; amendment; knowledge management; efficiency threshold.

Registration

Research article
Received: 20/02/2026
Accepted: 27/04/2026
Published:28/05/2026

Desarrollo de un sistema de gestión para modificación de matrícula de asignaturas: implementación en Google Apps

Resumen

Gestionar de forma eficaz las modificaciones en la matrícula de asignaturas requiere herramientas escalables que permitan hacer frente a la complejidad administrativa de este proceso. Este estudio desarrolla y evalúa un sistema de oficina inteligente basado en Google Apps Script (GAS) implementado en una universidad con el fin de modernizar los flujos de trabajo asincrónicos en línea y analizar su adopción por parte del personal. La investigación descriptiva abarcó 44 unidades académicas que gestionaron 9,914 solicitudes de 5,432 estudiantes a través de 226 vías de interacción. Los resultados demostraron una mejora del 66 % en la eficiencia del tiempo de respuesta en comparación con los puntos de referencia institucionales. Se identificó un umbral crítico de eficiencia en el que la adopción por parte del personal alcanzó el 100%, lo que indica que la adopción de soluciones bajo código viene impulsada por la supervivencia administrativa durante los picos de carga de trabajo. Los hallazgos confirman una correlación no lineal entre la carga de trabajo y la adopción de la tecnología, lo que proporciona un marco verificado para la escalabilidad institucional.

Palabras clave

Sistema de gestión; sistema de datos; servicios en la nube; Apps Script; Google Suite; matrícula de asignaturas; gestión del conocimiento; umbral de eficiencia.

How to cite this article / Cómo citar este artículo

GAVIRIA-MENDOZA, Carlos; AGUILAR-SÁNCHEZ, John F.; OSPINA-GARCIA, Cyndi. Developing a Management System for Course Enrollment Amendments: Implementing Google Apps for an Efficient Workflow. En: Entramado. Julio - diciembre, 2026. vol. 22, no. 2. p. 1-22. e-13429 https://doi.org/10.18041/1900-3803/entramado.2.13429

1. Introduction

The modern organizational landscape is increasingly defined by the integration of Software as a Service (SaaS) and cloud computing, which empower organizations to maximize their information resources while simplifying infrastructure management (Shivakumar, 2019). This technological shift has facilitated the global rise of Low-Code Development Platforms (LCDPs), which significantly enhance software development efficiency by allowing for rapid, customized solutions developed even by non-technical users (Lutsenko and Hrytsenko, 2025). Together, these trends provide a necessary framework for addressing complex administrative challenges through scalable and cost-effective digital transformation. Current cloud services make it possible to allocate resources and simplify infrastructure management (Pacios et al., 2025).

Within this technological ecosystem, Google Apps Script (GAS) serves as a prominent example of an LCDP that integrates seamlessly with the Google Workspace environment (Google, 2023). As a tool capable of generating and manipulating data objects with full access to a variety of Google Cloud Services, GAS bridges the gap between standard office tools and sophisticated automation (Bakrania and Johnson, 2015). Its versatility is demonstrated in diverse applications, ranging from monitoring autonomous unmanned aerial vehicles (UAV) on Google Maps (Thakkar, Rajput, Dubey and Parekh, 2019) to facilitating complex multi-user configurations and medical workforce rostering (Thomas, 2017). Consequently, the GAS represents a strategic choice for implementing smart office systems that are both functional and highly efficient.

The urgency of this transformation became evident following the COVID-19 pandemic -2020-, which drastically increased the demand for digital transformation in traditional in-person academic environments. The primary challenge is migrating from manual request handling to online asynchronous support and integrated response systems. Although electronic rostering and management platforms can lessen this administrative burden, they are often expensive, offer little customization (Thomas, 2017), or operate under rigid software and data structures that necessitate continuous maintenance costs (Debell et al., 2019).

This study aims to modernize the course enrollment amendment process at Universidad Militar Nueva Granada (UMNG) by developing and implementing a GAS-based smart office system. The central purpose of this work is to evaluate the system's impact on organizational efficiency and staff adoption. The proposed system is designed according to the three fundamental aspects for efficient management identified by Torres-Velandia, Barona-Ríos and García-Ponce-De-León (2010): (1) driving information to stakeholders, (2) improving communication, and (3) contemplating a user-friendly interface.

A detailed description of the applications of GAS is presented for the development and implementation of our smart office system. Next, the challenges in managing student requests for course enrollment or non-enrollment amendments are exposed. In the same section, an automation approach using Google Apps integration (i.e., Google Forms, Google Spreadsheets, and Gmail) is discussed. Then, the system development is presented, and insights into implementing this system are provided. Subsequently, the results and discussion of the implementation at the university are presented, emphasizing the system's impact on organizational efficiency and staff adoption. Finally, the conclusions derived from this study are depicted.

2. Theoretical framework: GAS in organizational automation

The existing literature confirms that Google Apps Script has moved beyond simple automation to become a core component of modern knowledge management (KM) frameworks (Shivakumar, 2019). Several applications of GAS have been reported, demonstrating its capacity to increase efficiency, productivity, and flexibility across diverse organizational scenarios (Logeshwaran, Madanika, Manuprabha, Venkateshkumar and Suyampulingam, 2024).

Google tools can simplify the management of academic information for departments, plans, subjects, and other entities (White and Allen, 2014; Revuelta-Arribas, 2020), and GAS-based development has promoted

the modernization of teaching management (Siiman and Mäeots, 2019). The integration of GAS has simplified academic management, from student tracking to project assignments. Bakrania and Johnson (2015) replaced manual sorting with multi-level criteria algorithms, and Lira, Ogosi and Vera (2025) automated pre-registration to eliminate waiting time. Bazán-Cruz (2017) and Cuiping (2014) emphasized the role of such systems as quality indicators in information management. In the area of advanced analytics, Laxmi, Jacob, G-S, Mohandas and Gatty (2024) fused data visualization in GAS with machine learning techniques to offer a visual analytics tool for student performance data in programming courses. Furthermore, Ward (2021) utilized GAS for student engagement assessment in both campus and remote connected modes, creating a dashboard to identify where individual and group interventions were required. Although these studies demonstrate technical success, they often focus on isolated departments or single-user processes. There is a lack of understanding of how GAS handles high-density organizational networks and a clear gap in understanding the socio-technical factors of adoption when staff are given the choice between automated and manual methods.

The GAS serves as an interface for database management and decision support. Velásquez-Guevara, Pedraza and Chavarriaga (2018) used GAS to resolve conflicts in multi-user configurations, while Homocianu and Homocianu (2019) enabled cloud-based visualization of large data sets. In complex clinical settings, Nishioka et al., (2024) automated tasks using worksheets and GAS to facilitate multiperson risk analysis (in radiotherapy), promoting collaborative safety work and reducing workloads. Technical studies by Charão, Hoffmann, Steffenel, Kirsch-Pinheiro and De-Oliveira-Stein (2017), and Ekanayake, Ihalage and Abyesundara (2021), evaluated the performance of cloud relational database management systems (RDBMS) and spreadsheets, finding that while Google Sheets are highly reliable for concurrent users, they face execution time limits. Most performance evaluations are conducted in controlled and technical environments (Charão et al., 2017). There is a vacuum regarding the long-term operational stability of simulated relational structures within free-tier constraints (simulated tables via sheets) during high-volume administrative cycles.

Recent trends show the integration of GAS with the Internet of Things (IoT) and behavioral monitoring. Alcaide, Finez and Magwili (2024) used it for fall detection among cyclists, and Jhan, Sreekumar and Kuriakose (2025) used it for automated news delivery in libraries. Diaz-Arrunategui (2015) demonstrated the integration of external services like JotForms, Cyfe, and Insightly via GAS to centralize business intelligence and facilitate internal data flow. In infrastructure, Kaprielian et al., (2019) employed GAS as a web service to record water flow events and reduce the risk of freezing in winter cabins. Smart manufacturing models by Kok-Hoe and Yotsuyanagi (2024) used GAS for wireless communication between production lines and dashboards, whereas Kishan Hameed, Charan and Srikar (2024) introduced elevator access control using GAS to transfer data to the cloud.

Thomas (2017) demonstrated that medical workforce rostering automation using GAS, generated high user satisfaction, and Barankov, Barankova, Mikhailova and Kalugina (2018) created cloud services for accounting sales on installments. Furthermore, Preechadech and Yongsiriwit (2024) developed a smart office system utilizing a GAS-based Webhook API to enable seamless interactions and automated responses. Finally, Rajput and Parekh (2020) developed a GAS for automated report generation and mailing in vehicular onboard diagnostic (OBD) emission checks. Existing smart office literature (e.g., helpdesk ticketing by Wibisono, Wardhana and Pamungkas, 2022) proves cost-effectiveness but fails to correlate workload metrics with employee engagement. Literature lacks a defined efficiency threshold, i.e., a quantitative point at which the perceived value of automation triggers full institutional adoption.

This study contributes to the field by moving from technical deployment to organizational impact analyses. It addresses the scaling of implementation to a complex network of interaction routes, identifies the efficiency threshold, and provides a theoretical contribution to the organizational adoption of LCDPs. Moreover, this research verifies efficiency improvements against real institutional benchmarks, ensuring data integrity when simulating relational data structures in free-tier cloud environments.

3. Materials and methods

This study adopted a descriptive and exploratory research design. The methodological approach is justified by the need for a cost-effective digital transformation in academic administrative processes, moving from in-person handling to online asynchronous management. The choice of LCDPs and SaaS was based on their ability to maximize information resources while simplifying infrastructure management, which is essential for scalable university-wide implementation. The framework explaining the need for GAS, and the source and traditional workflow for course enrollment amendments are presented in the following sections.

When a student sends an email request, it is often unclear what ticket number they are supposed to be helping with, and the only thing that can be automatically sent is confirmation of receipt of the email. However, the content of an email can be generated automatically using programming on the GAS interface, for example, by event-driven triggers. Because GAS offers an easily understandable interface, the low-code approach allows developers and non-technical users to quickly understand its functionality and develop tools (Lutsenko and Hrytsenko, 2025). Moreover, GAS developments for managing client communication, automating replies, and tracking project progress have been extended in recent years, as presented in the theoretical framework. Furthermore, the restrictions of the Google platform have not affected Google Apps Script-based development; for example, GAS has been successfully used to examine the functionality of cloud-based relational database services (Charão et al., 2017).

The course enrollment at the UMNG is an online process that students do themselves. The website system proposes a list of courses automatically based on the academic curriculum, students' curriculum revision, and conditions established in the student policies and regulations documents. Students should choose the courses they wish to take. UNIVEX is the university's management information system for academic records. This system is designed for use in higher education institutions. It manages the entire administrative and academic cycle of each student, from admission application to graduation. However, students may be unable to enroll in their desired courses on the UNIVEX platform because they do not meet certain conditions. In this situation, only administrative staff can address these issues. As a result, course enrollment or non-enrollment amendments were attended in person by academic units.

To migrate the support for the course enrollment amendment from in-person to online, the system had to allow students to send their requests to the corresponding academic unit within the organizational structure of UMNG. Figure 1 displays the network diagram of students' requests, where each node represents a unit. The data used were collected during the implementation of the proposed system. The figure shows all 44 academic units, i.e., 32 undergraduate programs (blue nodes), two legal practical training centers (orange nodes), four technical diploma programs (green nodes), and six departments (red nodes). In this network, the size of each node and thickness of each link represents the total number of requests reported in the unit and the total number of interactions between units, respectively. The names of the programs and departments were anonymized (i.e., AUXX). The lines connecting the nodes in Figure 1 represent all possible interactions between units during an enrollment amendment request (i.e., the academic unit of the undergraduate program) and the departments offering the course. Figure 1 illustrates that this dynamic procedure was developed by addressing 226 potential recipients. The current avenues for information and response to requests, including emails from each academic unit and the contact center, among others, were extremely overloaded, given that this procedure covered approximately 19,000 undergraduate students (UMNG, 2022).

This study was done within the next scope:

- Spatial scope: The system was implemented in all undergraduate programs at the UMNG.
- Temporal scope: This project was conducted during the second semester of 2022.
- Universe: Students from all academic years participated in the study.
- Sample: All students who submitted the 9,914 requests via the Forms (i.e., 5,432 students).

- Instruments: Include Forms, data collected in worksheets from the Forms, and data visualized and consolidated on the dashboard.
- Technique: Data analysis.
- Variables: (1) staff engagement (system acceptance): measured by the ratio between the number of student requests received and the number of responses processed by administrative officials via the implemented system; (2) response time (workflow efficiency): defined as the elapsed time (in hours) from the initial request submission to the final notification of the decision; (3) system usage validation: measured by the number of courses per request to verify if the tool is used for specific amendments rather than full enrollment; and (4) service load: defined by the volume of requests and interactions across the 44 academic units.
- The evaluation criteria: (1) operability: the ability of the system to handle the complexity of 226 potential interaction routes among 44 diverse academic units; (2) efficiency: achieving a response time below the institutional expectation of 72 hours; (3) functionality: successful execution of automated triggers (event-driven triggers) for ticket generation and real-time status updates without data loss; and (4) adoption rate: achieving a response rate of approximately 80% or higher in units with high request volumes.
- The analysis followed a quantitative descriptive approach, incorporating network analysis, descriptive statistics (e.g., frequency distributions and trend analyses), time-series analysis, and comparative analysis. Four phases were developed for the implementation of the automated task system: (1) designing and implementing tailored Google Forms, (2) creating a dashboard format in Google Spreadsheets, (3) developing GAS code for automatically generating ticket numbers and mailings, and (4) deploying it across all university undergraduate programs.

Figure 1. Network diagram of units involved in student requests, including all possible course enrollment/non-enrollment amendments.
Note: Author's own elaboration.

4. Results

This section outlines the development, functionality, and implementation of the system designed to manage course enrollment amendment requests.

4.1 Integrations between Google Apps

Focusing on improving the course enrollment or non-enrollment requirements process through an automated workflow, a network of Google Forms was designed to allow students to submit information to the program or department offering the required course. The published Forms collected and stored the data using Spreadsheets. As a complement, self-contained and adaptable web-based data collection Google Sheet was created, establishing a dashboard for real-time monitoring and management of the process. Several features of this spreadsheet were implemented (i.e., conditional formatting of cells, protection of cells and ranges, data input validation, formulas within cells, and formulas in the sheet's advanced options). Consequently, the development environment of the proposed system was completed on Google Cloud.

When a Google Sheets file is edited, it provides access to the GAS scripting interface, which offers the capability of developing functions for automatically generating emails from the collected data (e.g., Fransen, Kocher and Kempf, 2011). Scripts were created to enable automatic email delivery to both the student and the academic unit officials as soon as a Form was submitted with the information provided by the student or official. This enables smoother interactions and the automatic formatting and pre-filling of basic fields in the response body.

The diagram in Figure 2 illustrates the interactions between Google apps and automation through GAS scripting. These components were distributed via Google Cloud Run, ensuring scalability, while remaining entirely within free-tier usage constraints. Figure 2 shows that the subjects involved in the system (students and academic officials) are only able to provide information through the Forms, allowing simple and moderate data input by users. Before the students enter their personal information, the Form for students request submission (i.e., the Home page in Figure 2) requires a single-select question. This question allows students to choose whether they will use the current Form (i.e., a request directed to their academic unit) or follow one of the links that redirects them to the Form of a specific department. In this way, the information is channeled to the respective academic unit, where the request is ultimately processed.

Once students submit their requirements (Home page), the data are stored in their associated individual database (database 1), and an email is automatically generated by the script (top-right App Script icon in Figure 2) using the information provided by the student in their Form submission. In particular, a ticket number is assigned to each incoming request, which is automatically sent back to both the student and the corresponding academic unit. The dashboard (multi-spreadsheet database) consolidates students request information and response data in a single location, enabling administrative officials to track the status of each request. This incorporates the items required for generating control statistics for a department or academic unit through a multi-tab spreadsheet visualization scheme.

Furthermore, once a request is submitted, the dashboard sheet generates a link next to the student request status log that automatically creates a pre-filled Form populated with the submitted student data (pre-filled Form in Figure 2). This reduced the amount of data input required from administrative staff when they are notified about the request status. As a result, the academic official only needs to enter the decision regarding the student's request. When the official submits the pre-filled Form, an email containing the response is automatically sent to the student via the developed App Script (bottom-right App Script icon in Figure 2).

Figure 3 depicts the fundamental data elements compiled by both Forms templates (i.e., Home page and pre-filled Forms) implemented within the system. These tailored Google Forms and their integration with other Google Suite tools were planned through several discussions and a literature review. To that end, a focus group engaging the Office of Academic Affairs, the Information and Communications Technology Advisory Office, and researchers was organized to discuss the findings. The decision to add or remove labels and fields from the Forms was carefully evaluated, weighing the trade-offs among practical utility, comprehensiveness, and simplicity. As previously mentioned, the first step in the process requires students to select the respective academic unit where the request is ultimately processed, i.e., a specific Google Form was created for each academic unit, enabling students to communicate with all academic departments as needed.

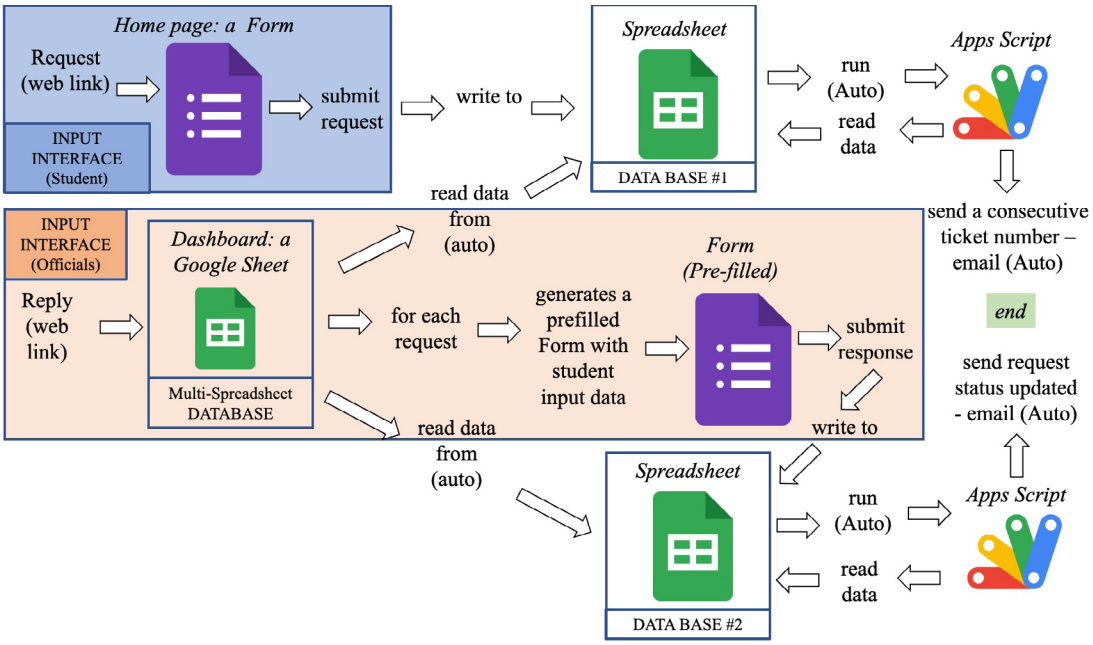
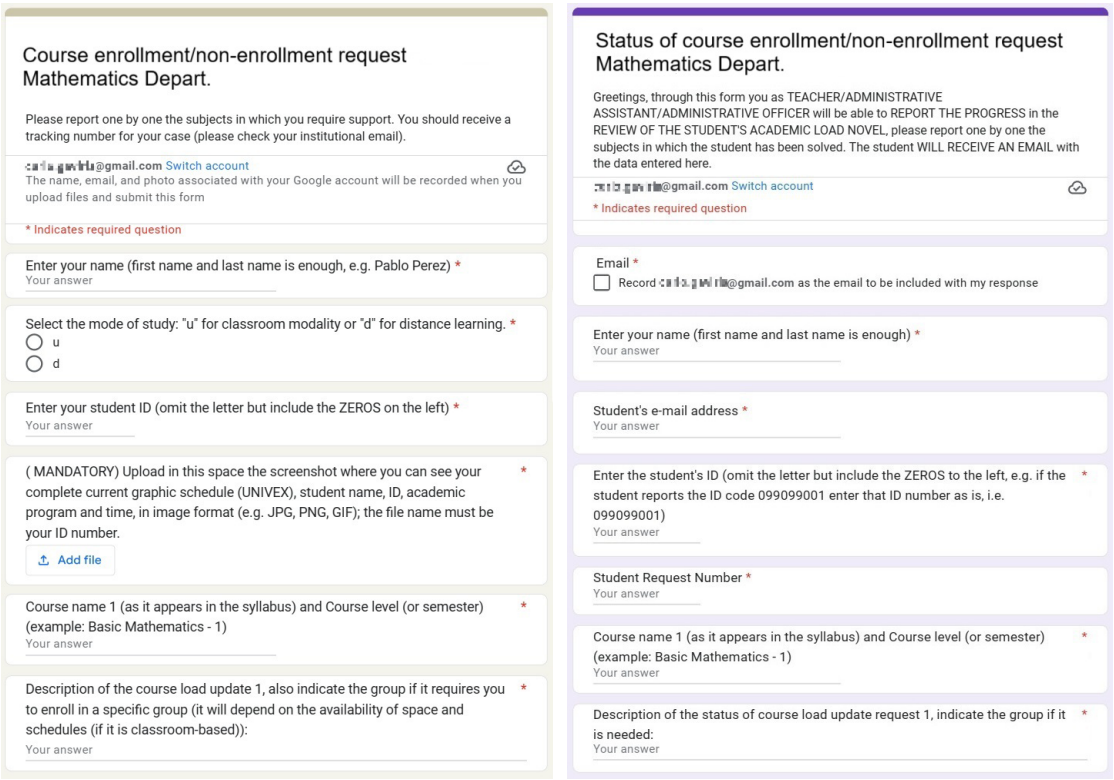


Figure 2. Interaction between Google Apps and developed GAS scripts.
Note: Author's own elaboration.

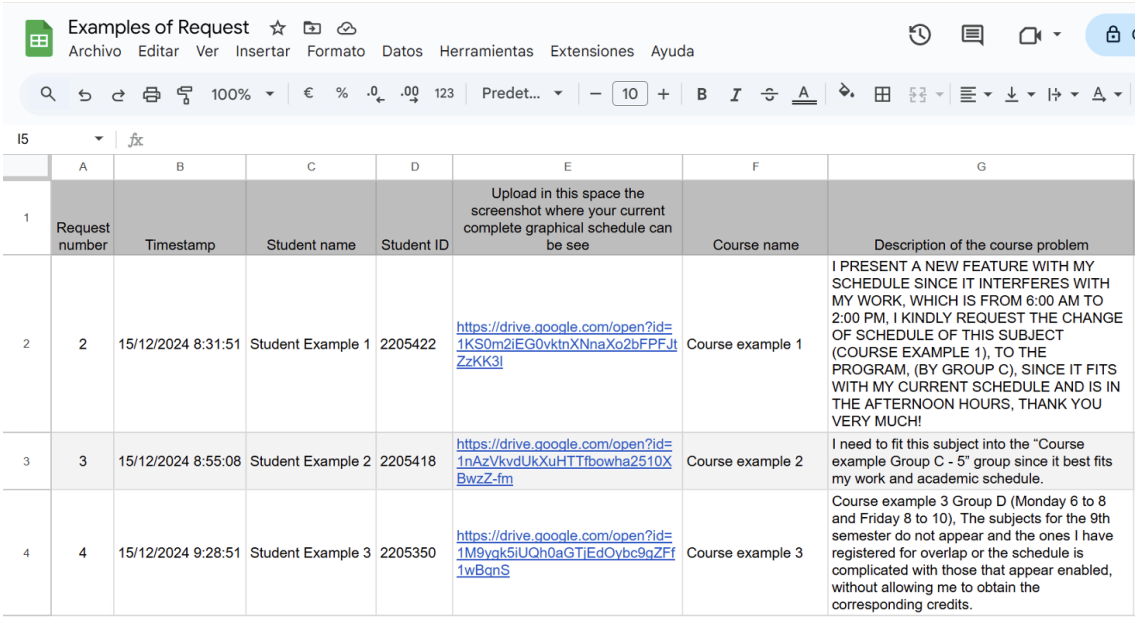


(a) (b)

Figure 3. Implemented Forms for (a) student request and (b) notification of request status.
Note: Author's own elaboration.

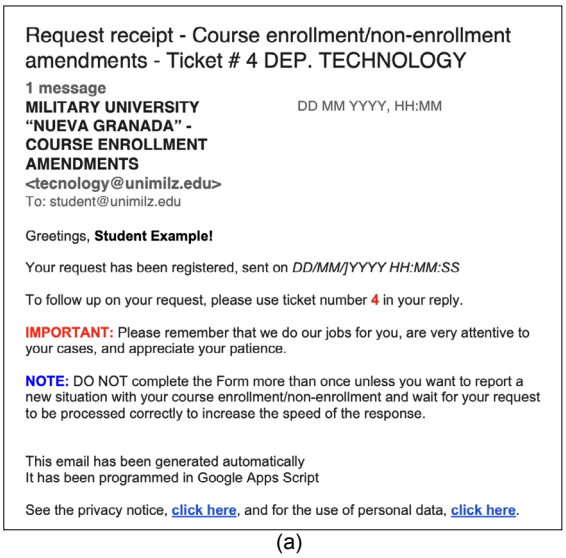
In general, the student requests an exception to course enrollment or non-enrollment for one of the following reasons: (1) enrolling in a course that has reached full capacity, (2) enrolling in more courses than those established in the curriculum (i.e., extra-crediting), or (3) enrolling in courses or requesting a place in courses within a specific schedule. Accordingly, a field in the Form was used to describe the request motivating the course enrollment or non-enrollment amendment (left Google Form in Figure 3). Furthermore, several fields in the Forms and guideline labels (help fields) were discussed and established throughout this project. The Forms were used for real-time data collection, and their integration with Google Sheets served as a lightweight data management system.

Figure 4 presents an example of data stored in the spreadsheet, while Figure 5 illustrates the layout of automatic emails generated through GAS scripting.

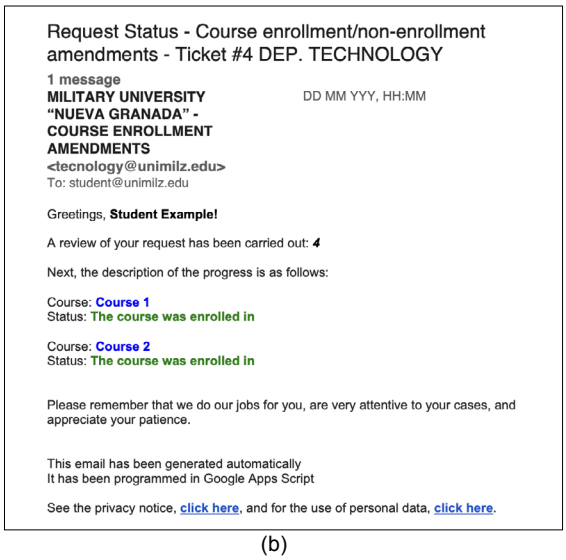


	A	B	C	D	E	F	G
1	Request number	Timestamp	Student name	Student ID	Upload in this space the screenshot where your current complete graphical schedule can be see	Course name	Description of the course problem
2	2	15/12/2024 8:31:51	Student Example 1	2205422	https://drive.google.com/open?id=1KS0m2iEG0vktnXNnaXo2bFPFJtZzKK3I	Course example 1	I PRESENT A NEW FEATURE WITH MY SCHEDULE SINCE IT INTERFERES WITH MY WORK, WHICH IS FROM 6:00 AM TO 2:00 PM, I KINDLY REQUEST THE CHANGE OF SCHEDULE OF THIS SUBJECT (COURSE EXAMPLE 1), TO THE PROGRAM, (BY GROUP C), SINCE IT FITS WITH MY CURRENT SCHEDULE AND IS IN THE AFTERNOON HOURS, THANK YOU VERY MUCH!
3	3	15/12/2024 8:55:08	Student Example 2	2205418	https://drive.google.com/open?id=1nAzVkvdUkXuHTTfbowha2510XBwzZ-fm	Course example 2	I need to fit this subject into the "Course example Group C - 5" group since it best fits my work and academic schedule.
4	4	15/12/2024 9:28:51	Student Example 3	2205350	https://drive.google.com/open?id=1M9yqk5iUQh0aGTjEdOybc9qZFf1wBqnS	Course example 3	Course example 3 Group D (Monday 6 to 8 and Friday 8 to 10), The subjects for the 9th semester do not appear and the ones I have registered for overlap or the schedule is complicated with those that appear enabled, without allowing me to obtain the corresponding credits.

Figure 4. Example of collected data of student requests.
Note: Author's own elaboration.



(a)



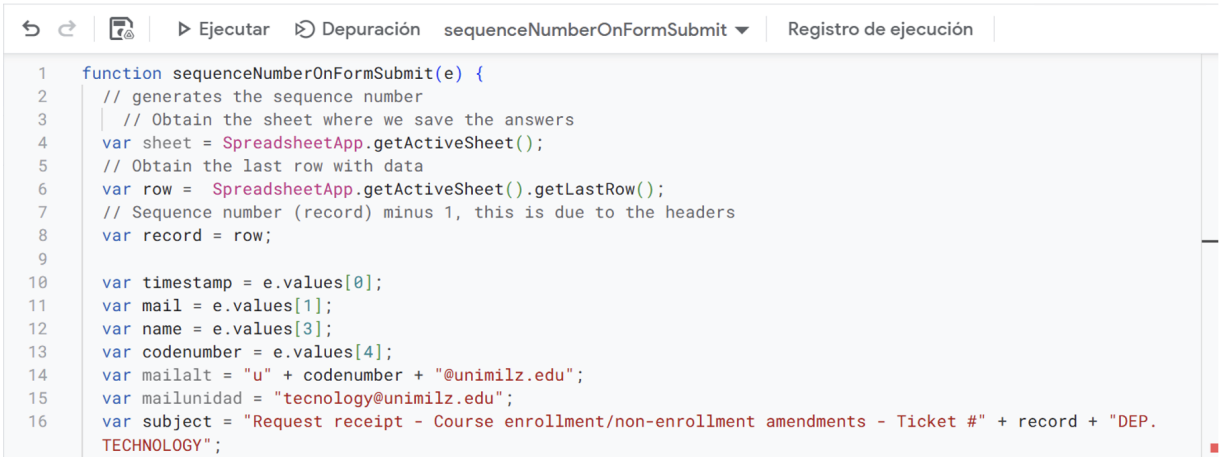
(b)

Figure 5. Example of automatic emails when (a) students send a request or (b) administrative officials send a request status notification.
Note: Author's own elaboration.

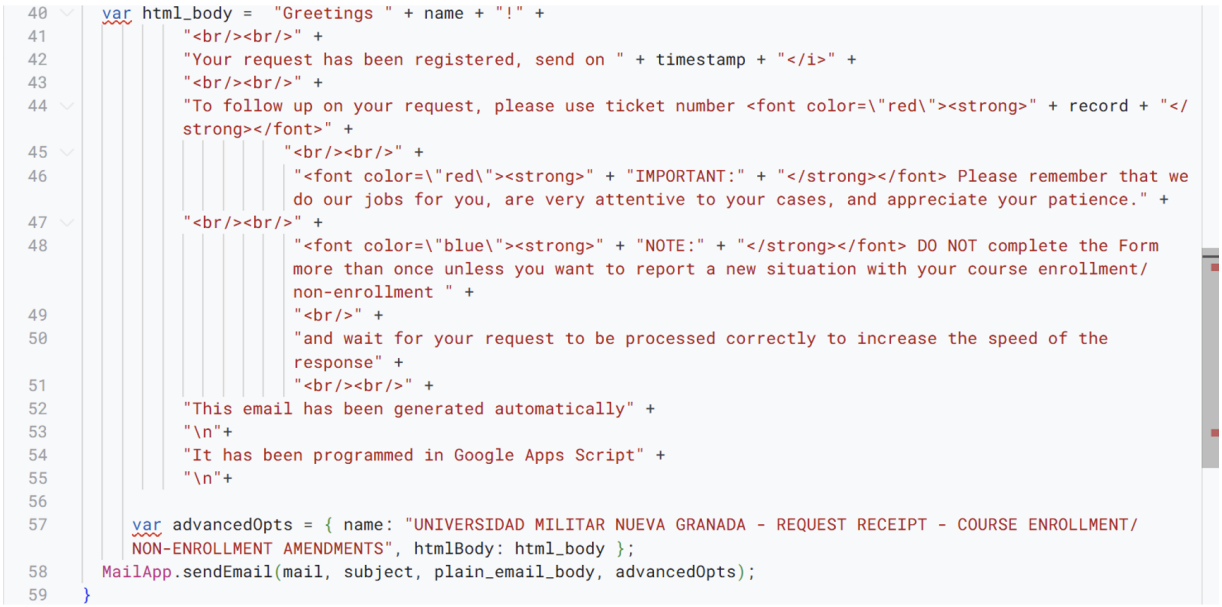
4.2 Automatic mailings programming

One of the key features of GAS is the ability to program customized messages to be sent automatically in response to a submitted request (e.g., Ward, 2021). In this case, the trigger is activated when students or officials submit a Form. The algorithm implemented for generating a receipt email upon a student's request submission is presented in Figure 6. The email was generated in three steps as follows:

1. The basic data (email, date, and time of the request), the consecutive number is read, and the subject of the email is defined (i.e., "var" command, lines 4 – 16 in Figure 6a).
2. The body of the email is generated from the read data (i.e., defining the variables "html_body", and "advancedOpts, lines 40 – 57 in Figure 6b).
3. An email is sent to the student who made the request (i.e., "MailApp" command with the option "sendEmail," line 58 in Figure 6b).



(a)

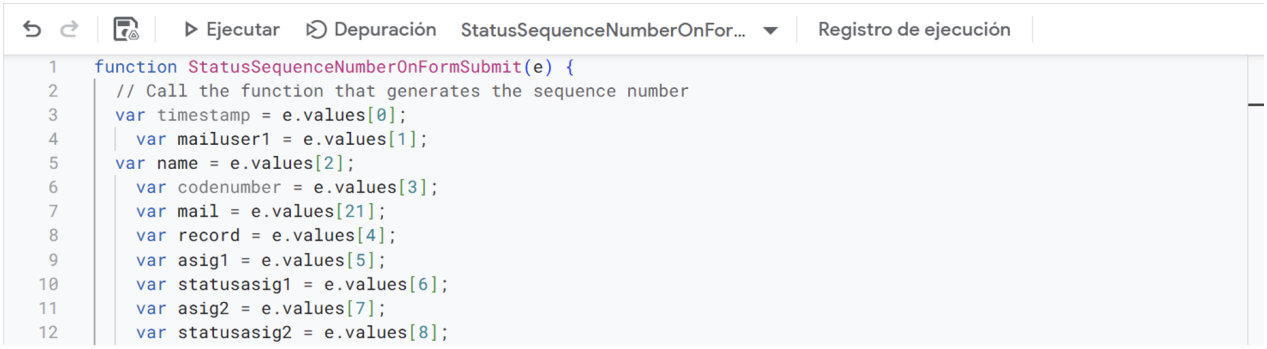


(b)

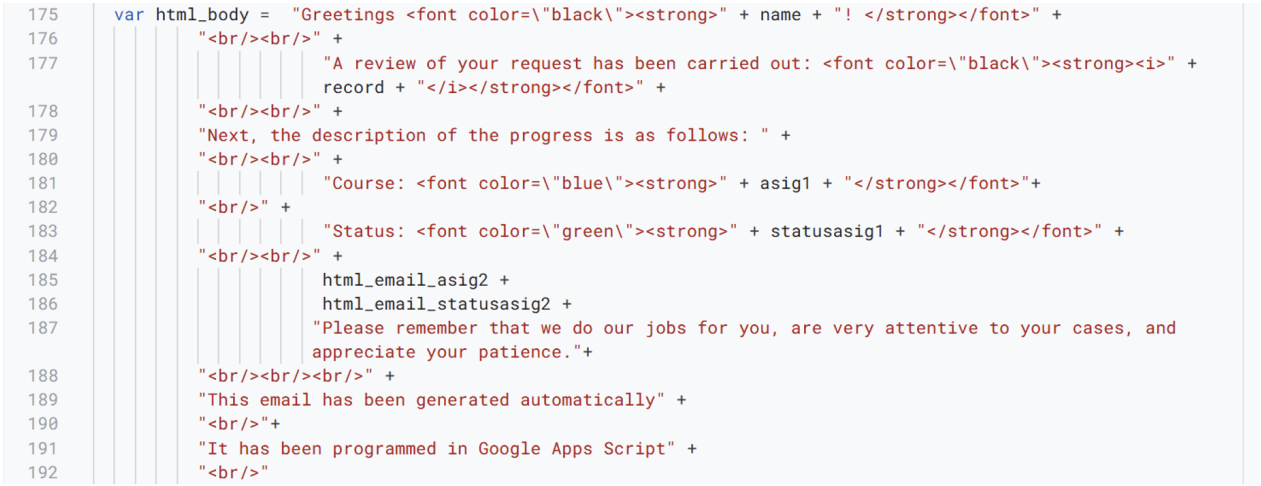
Figure 6. GAS scripting for automatic ticket number and email generation for student requests, script lines for: a) reading basic data, b) generating the body and sending the email.
Note: Author's own elaboration.

The algorithm implemented for automatically notifying students via email when administrative staff submits a status update or decision regarding a request through the Form is presented in Figure 7. The email is generated by integrating two additional elements:

1. The basic course data for the enrollment or non-enrollment amendment, along with the decision and/or solution by the officials, are also retrieved (i.e., "var" command, lines 9 – 12 in Figure 7a).
2. As a complement, this information is incorporated into the body of the generated email (i.e., additional variables and text in the "html_body," lines 177 – 186 in Figure 7b).



(a)



(b)

Figure 7. GAS scripting for automatic email generation of request status, script lines for: a) retrieving additional data, and b) generating the body of the email.
Note: Author's own elaboration.

Upon completion of the scripting phase, training materials and reference documents were created to support the implementation process.

4.3 Results of implementation

The developed smart office system was implemented during the second semester of 2022 in 44 academic units. The academic staff were trained, and several performance tests were conducted. Unlike traditional emails, where ticket numbers were often absent, the GAS implementation provided instant notification with unique identifiers, resolving previous real-time communication issues. The network diagram in Figure 1 was used to model the interactions among students, programs, and departments. Six of these units (referred to as departments, i.e., red nodes in Figure 1) correspond to units that do not offer undergraduate programs

and provide academic services exclusively across all or at least one of the units. During implementation, 9,914 requests from 5,432 students were collected by the developed system, indicating an average of 1.8 requests per student. Of the 9,914 requests, 8,848 responses were recorded, reflecting an overall staff engagement rate of 89.2%. The system was able to handle high-density nodes (departments) and peripheral nodes (undergraduate programs) without data loss or processing delays.

Descriptive statistics (frequency distributions and trend analyses) were used to characterize the workload across units. Figure 8 shows the distribution of requests. This figure reveals that the distribution of workload follows an exponential function, where a minority of units (primarily departments providing basic/cross-disciplinary courses, such as AU17 and AU16) handle the bulk of the administrative load.

For the comparative analysis, adoption was evaluated by comparing input (requests) versus output (responses) metrics per academic unit, as shown in Figure 8. This comparison serves to measure the acceptance level of the corresponding academic officials within each unit. The figure reveals zero values in the replies of a few units, indicating that the academic staff of these units did not use the proposed system to send responses to students regarding the status or final decision of their requests. These units received a lower rate of enrollment or non-enrollment amendments, i.e., the number of requests received by these units was 1, 2, 3, 4, 7, and 21 for AU42, AU12, AU19, AU41, AU44, and AU5, respectively. Comparing the number of requests and responses within the same unit reveals that the proportions differ across academic units. Furthermore, the two undergraduate programs reporting the highest proportion of both responses and requests through the developed system were AU35 and AU2, as they served larger student populations than the remaining 30 academic units. In some academic units (e.g., AU17), the number of responses exceeded the number of requests for the same unit because the system tracked each communication with the student individually, even when it was temporary or partial (e.g., enrolling a student in one course while awaiting the addition of a place in another course for the same student).

Figure 8 also reveals a correlation between workload and system usage. In units with more than 300 requests (threshold value), the adoption of the GAS-based response system reached 100%. Conversely, in units with fewer than 25 requests, officials occasionally reverted to traditional methods, suggesting that the perceived value of automation increases exponentially with service load. Approximately 80% of the academic units achieved a 1:1 ratio between requests and responses, demonstrating a robust transition from traditional face-to-face handling to the digital smart office model. The findings indicate that the proposed smart office system meets the operational needs of both students and administrative staff members. Additionally, although the database of responses to requests is not shown due to space constraints, the organized log of responses to requests further demonstrates that the transition from a traditional face-to-face process to technological innovation was successful, confirming that the proposed system was satisfactorily adopted by officials.

According to the curriculum of the university's undergraduate programs, students may enroll in a maximum of eight courses per semester. To ensure the system's validity as a tool for processing "amendments" rather than "full enrollment", the number of courses per request was analyzed. Figure 9 illustrates the relationship between the number of requests and the number of courses reported in these course enrollment or non-enrollment amendments. The analysis found that 61% of requests involved only one course, validating that the system effectively targeted specific exceptions and adjustments.

For practical purposes, a sample of 5% of total requests was subjected to a detailed temporal analysis (i.e., time-series analysis) to determine average response times per academic unit. This average response time spans from the reception of the request to the delivery of the final decision by an academic unit (see workflow in Figure 2). Figure 10 presents the results for the 38 academic units in which administrative officials used the proposed system to send responses (the blue, orange, green, and red bars identify the undergraduate programs, legal practical training centers, technical diploma programs, and departments, respectively). It was observed that the implemented system enabled most units (i.e., 68.4% or 26 units) to process requests in under 24 hours, while the institutional expectation for administrative responses is

72 hours. This can be attributed to the fact that the automated dashboard facilitated real-time monitoring and pre-filled responses, which significantly reduced data entry demands on staff. This represents a 66% improvement in efficiency relative to the maximum allowable response time.

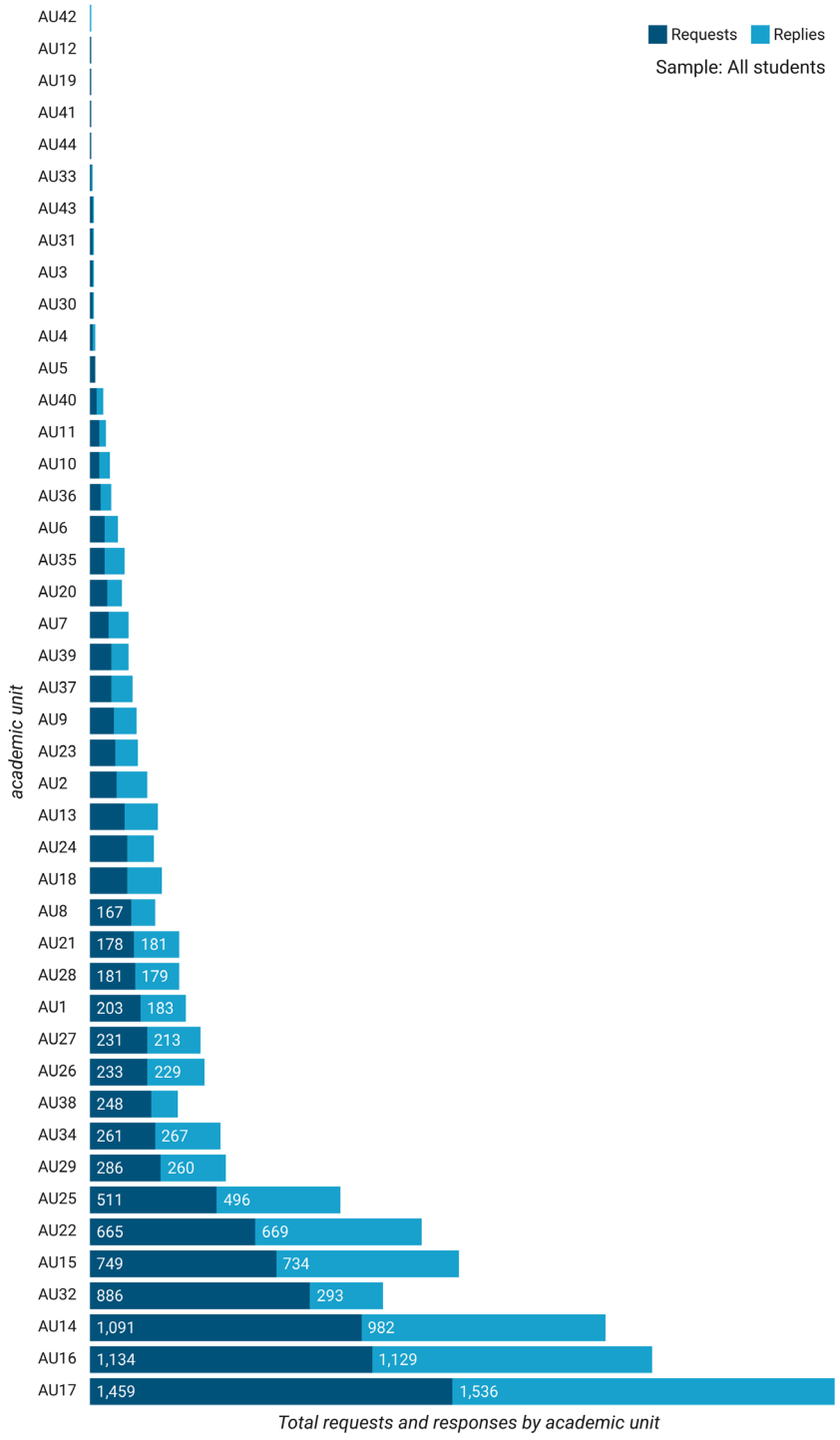


Figure 8. The number of requests and responses handled by each academic unit.
Note: Author's own elaboration.

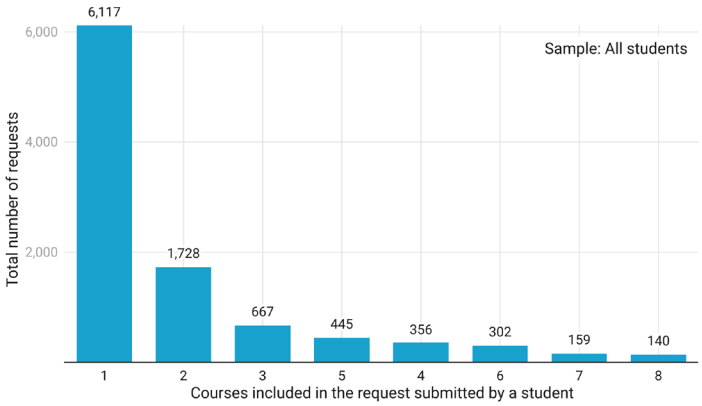


Figure 9. Total number of requests compared to the number of courses included in each request.
Note: Author's own elaboration.

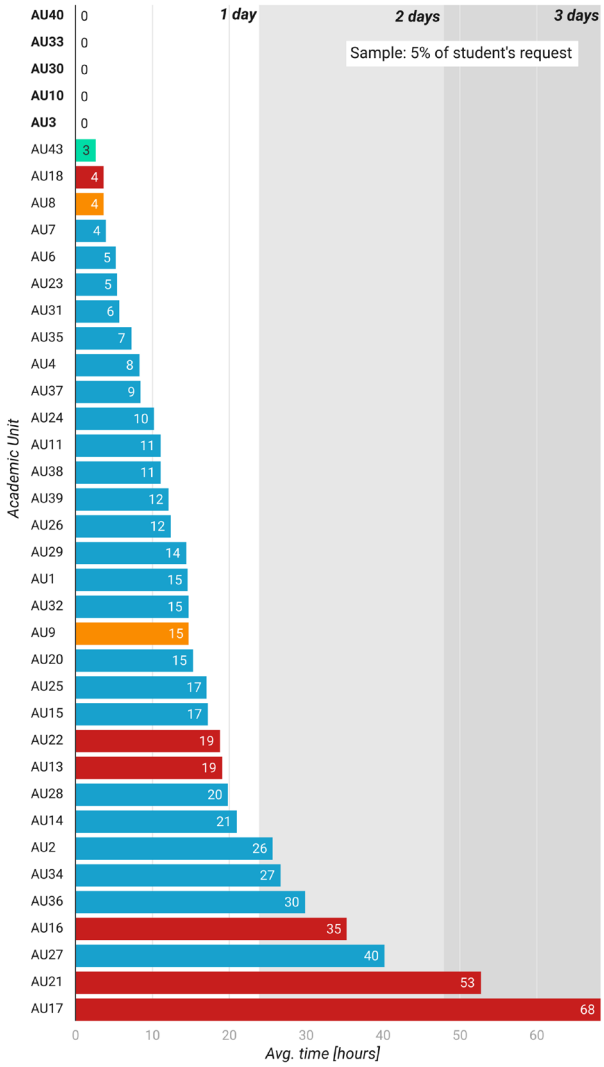


Figure 10. Average time to handle requests in each academic unit.
Note: Author's own elaboration.

The zero values in Figure 10 indicate that the responses were sent in under 30 minutes. The units with zero response times correspond to those that received a lower rate of enrollment or non-enrollment amendments (i.e., fewer than 40 requests, see Figure 8). The number of requests received by these units was 7, 10, 12, 29, and 39 for AU33, AU3, AU30, AU40, and AU10 (Figure 8), respectively. Comparing the results in Figures 8 and 10, as expected, the highest response time was reported by the unit AU17, as this unit recorded the highest volume of enrollment or non-enrollment amendments (the longest bar in Figure 8). Even in high-volume units such as AU17, the average response time remained within functional limits, confirming the scalability of the GAS-based architecture.

The results demonstrate that the transition to a SaaS and LCDP model offers more than just a technical fix, i.e., it establishes a knowledge management framework. The high acceptance rates (8,848 responses) and the significant reduction in response times (mostly under 24 hours) provide robust evidence that the system is a viable alternative to proprietary and expensive software. Furthermore, the system's ability to facilitate asynchronous online management has modernized the administrative cycle from application to decision, making the course enrollment amendment process smoother and more transparent for the entire university community.

5. Discussion

After the COVID-19 pandemic, the demand for digital transformation in in-person academic subjects has increased significantly. The challenge involved transitioning from face-to-face administrative tasks to a remote assistance model. As part of these tasks, the requirements for amending course enrollment demanded an online, asynchronous support and response integrated system.

While some academic specialties have developed systems for course enrollment, these are often restricted to basic activities, such as pre-registration and student quota limits. Information related to individual requests can be retrieved, organized, and displayed using conventional automated systems. Lira-Camargo, Ogosi-Auqui and Vera-Tito (2025) focused on automating pre-registration forms to manage course saturation. In contrast, our proposed system addresses a higher degree of administrative complexity by managing amendments (adds/drops) across a network of 44 academic units with 226 potential interaction paths. In addition, the developed smart office system extends this functionality by not only collecting student requests but also consolidating tracking data that reflects the progression of each request as reported by officials. The dashboard includes the components required to perform control statistics using a spreadsheet visualization scheme with multiple tabs. This ensures that the collected databases can be analyzed and provide real-time information for decision-based support of course enrollment amendments. The system is smart, as defined by Torres-Velandia et al., (2010), because it was aimed at (1) task automation, (2) integration of Google Apps, (3) real-time collaboration, and (4) a user-friendly interface.

This study also contributes to the growing body of smart office literature, which includes helpdesk ticketing (Wibisono et al., 2022), event certification (Ramadhani, 2020), and automated attendance (Devaprakash, Gowtham, Murali, Muralidharan and Vijayalakshmi, 2020; Bhuvaneshwari, Prasadh, Sathish, Anuja and Mythily, 2023). Compared to the news alert framework by Jhan et al., (2025) or cycling safety alerts by Alcaide et al., (2024), our system is uniquely administratively oriented, requiring active validation by staff to close decision cycles. Furthermore, while Thomas (2017) demonstrated the success of GAS in medical workforce rostering with 100% satisfaction, the study noted that complex tasks were limited by the lack of a true relational database. Our research overcomes this limitation by simulating relational structures through interconnected sheets and scripts, successfully handling nearly 10,000 requests.

The selection of SaaS and Platform as a Service (PaaS) models is aligned with the global trends reported by Lutsenko and Hrytsenko (2025). They identified LCDPs as essential for adapting to industrial and business processes. The decision to use Google Sheets as a lightweight database was supported by Ekanayake et al., (2021). They found that spreadsheets can show higher reliability in writing data with concurrent users

than some external RDBMS connections. This is critical for our implementation, in which multiple students and officials interact with the dashboard simultaneously. Moreover, the versatility of GAS found in this study mirrors its application in diverse fields, such as the multi-cloud satellite detection models described by Pacios et al., (2025) and the autonomous vehicle tracking systems described by Thakkar et al., (2019). These comparisons reinforce the validity of the GAS as a robust tool for handling critical workloads in a smart office environment.

A critical finding of this research is the direct correlation between service load and system adoption. The results showed that academic units with more than 300 requests had a 100% adoption rate, whereas units with fewer than 25 requests occasionally reverted to manual methods of processing requests. This suggests an efficiency threshold, where the perceived value of automation is only fully realized by the staff when the manual workload becomes unmanageable. This phenomenon is related to the reduction in perceived time documented by Airinei and Homocianu (2017). Moreover, the 66% improvement in response time efficiency (processing most requests in less than 24 hours against a 72 hours institutional expectation) provides a quantitative benchmark for the success of the LCDP approach. By automating ticket generation and pre-filling response bodies, the system effectively mitigated the communication gaps that are inherent in traditional email-based workflows.

For the first time at the university, a real-time database was created that allowed for a multidisciplinary KM approach. As suggested by Shivakumar (2019), the effective utilization of such analytical tools is vital for achieving organizational objectives and enhancing decision-making. Our findings prove that officials with moderate technical knowledge can build sophisticated automation frameworks that rival expensive and proprietary software. The developed system helped with the biannual Office of Academic Affairs meeting, and the course enrollment discussion was smoother and completed more quickly.

One significant limitation is the absolute reliance on the provider's policies (Google), which could impact long-term stability if quotas or terms of service change. Future work should explore the integration of lightweight Natural Language Processing (NLP) and serverless free tiers to further optimize these event-driven workflows. In conclusion, the implemented system not only modernizes a critical administrative cycle but also establishes a scalable, cost-effective model for digital transformation in higher education.

6. Conclusions and future work

This research successfully achieved the modernization of the course enrollment amendment process at Universidad Militar Nueva Granada through the design and implementation of a scalable GAS-based smart office system. Beyond technical deployment, the findings provide a verified framework for digital transformation in higher education. The following conclusions were drawn:

• This study identifies a non-linear correlation between administrative workload and technology adoption. The evidence reveals an efficiency threshold (approximately 300 requests) where the perceived value of automation triggers a 100% adoption rate among staff. This finding contributes to organizational theory by suggesting that the adoption of LCDP is driven by administrative survival when manual processing becomes unfeasible.

• The system's internal consistency is supported by a 66% improvement in response time efficiency, reducing processing times to less than 24 hours compared to the 72-hour institutional benchmark. The high overall adoption rate of 89.2% (8,848 responses) validates the reliability and functional stability of the simulation of relational data structures within a non-relational, free-tier cloud environment.

• The implementation represents a significant advance from a mere technical tool to a robust knowledge management framework. For the first time, the institution has a centralized, real-time database that allows for proactive decision-making and multidisciplinary analysis of student interaction nodes, which was previously impossible under a fragmented, email-based model.

• The results demonstrate that LCDP and SaaS solutions can effectively manage high-complexity organizational structures, involving 226 interaction routes, without the need for expensive proprietary software. The system's validity as a specific amendment tool was confirmed by the fact that 61% of the requests targeted single-course adjustments.

Despite the success of the model, total reliance on a single provider (Google) remains a strategic limitation. To ensure long-term sustainability and further enhance the smart office concept, the following future works are proposed: 1) implementing targeted training for academic units below the identified efficiency threshold to prevent reversion to traditional, non-traceable methods; 2) investigating the use of lightweight Natural Language Processing (NLP) to automate the classification of open-text request justifications, further reducing the administrative burden; and 3) exploring the articulation of serverless free tiers to mitigate provider-specific quotas and enhance the system's event-driven architecture.

About the authors

Carlos Gaviria-Mendoza

Distance Learning Faculty, Universidad Militar Nueva Granada, Cajicá, Colombia, Professor
carlos.gaviria@unimilitar.edu.co https://orcid.org/0000-0002-3741-4325

John F. Aguilar-Sánchez

Basic and Applied Sciences Faculty, Universidad Militar Nueva Granada, Cajicá, Colombia, Professor,
john.aguilar@unimilitar.edu.co https://orcid.org/0000-0002-4752-2784

Cyndi Ospina-Garcia

Basic and Applied Sciences Faculty, Universidad Militar Nueva Granada, Cajicá, Colombia, Professor,
cjog1985@gmail.com https://orcid.org/0000-0003-3375-475X

Data availability statement

The authors declare that the article contains all the data necessary and sufficient for understanding the research.

Disclosure statement

The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Funding

This research did not receive specific funding from any entity in the private, public, commercial, or nonprofit sectors.

Acknowledgments

The authors express their gratitude to the Universidad Militar Nueva Granada and state that this article is the result of their academic exercise as professors at the Universidad Militar Nueva Granada.

Co-authorship

Carlos Gaviria-Mendoza: Conceptualization, Investigation, Software, Writing-review-editing.

John F. Aguilar-Sánchez: Investigation, Formal analysis, Methodology, Writing-review-editing.

Cyndi Ospina-Garcia: Conceptualization, Investigation, Visualization, Methodology, Writing-original draft.

References

1. AIRINEI, Dinu; HOMOCIANU, Daniel. Cloud Computing Based Web Applications. Examples and Considerations on Google Apps Script. In: Proceedings of the IE 2017 International Conference, pp.64-69. 2017. http://doi.org/10.2139/ssrn.2964756

2. ALCAIDE, Paulino III; FINEZ, Kaizley D; MAGWILI, Glenn V. Real-time Monitoring of Bicycle Behavior in IoT Fall Detection System. In: 2024 8th International Conference on Communication and Information Systems. 2024. p. 191-196. http://doi.org/10.1109/iccis63642.2024.10779382

3. BAKRANIA, Smitesh; JOHNSON, Brad. A Cloud-based Tool for Assigning Students to Projects. In: 2015 ASEE Annual Conference & Exposition. 2015. p. 20-26. http://doi.org/10.18260/p.23361

4. BARANKOV, Viktor; BARANKOVA, Irina; MIKHAILOVA, Ulyana; KALUGINA, Olga. Experience of Developing Cloud Service for accounting Sales in installments. In: Journal Of Physics Conference Series. 2018. vol. 1015, no. 4. http://doi.org/10.1088/1742-6596/1015/4/042004

5. BAZÁN-CRUZ, Eduardo Daniel. Implementación de un sistema de seguimiento de egresados de la EAP de Ingeniería Agroindustrial basado en un modelo de gestión. Tesis de Ingeniería Agroindustrial. Trujillo-Perú: Universidad Nacional de Trujillo. 2017. https://hdl.handle.net/20.500.14414/10953

6. BHUVANESHWARI, Balan; PRASADH, Harish; SATHISH, P; ANUJA, Vishnu; MYTHILY, A. Automated Contactless Attendance System. In: Lecture Notes on Data Engineering and Communications Technologies. 2023. vol 166. p. 523-532. http://doi.org/10.1007/978-981-99-0835-6_37

7. CHARÃO, Andrea; HOFFMANN, Guilherme; STEFFENEL, Luiz; KIRSCH-PINHEIRO, Manuele; DE-OLIVEIRA-STEIN, Benhur. Performance Evaluation of Cloud-based RDBMS through a Cloud Scripting Language. In: Proceedings of the 19th International Conference on Enterprise Information Systems. 2017. vol 1. p. 332-337. http://doi.org/10.5220/0006350803320337

8. CUIPING, Han. Design and implementation of Student Management System of Educational Management System. In: 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA). 2014. p. 199-202. http://doi.org/10.1109/wartia.2014.6976231

9. DEBELL, Thomas; GOERTZEN, Luke; LARSON, Lars; SELBIE, William; SELKER, John; UDELL, Chet. OPEnS Hub: Real-Time Data Logging, Connecting Field Sensors to Google Sheets. In: Frontiers In Earth Science. 2019. vol. 7. http://doi.org/10.3389/feart.2019.00137

10. DEVAPRAKASH; GOWTHAM; MURALI; MURALIDHARAN; VIJAYALAKSHMI, V. J. Centralized attendance monitoring system. In: 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS). 2020. p. 1288-1291. http://doi.org/10.1109/ICACCS48705.2020.9074162

11. DIAZ-ARRUNATEGUI, Karina Lucia. External services integration for internal usage of an organization, using Google Apps Script. Treball Final de GrauEstudis de primer/segon cicle. Enginyeria Tècnica de Telecomunicació, Especialitat en Telemàtica (Pla 2000). Universitat Politècnica de Catalunya, 2015. https://upcommons.upc.edu/entities/publication/c92c0e2b-a633-4dbc-b108-8fb894a0d9cb

12. EKANAYAKE, Lahiru; IHALAGE, Deepthi; ABYESUNDARA, Sachith. Performance Evaluation of Google Spreadsheet over RDBMS through Cloud Scripting Algorithms. In: 2021 International Conference on Computer Communication and Informatics (ICCCI). 2021. p. 1-7. http://doi.org/10.1109/ICCCI50826.2021.9402432

13. FRANSEN, Janet; KOCHER, Megan; KEMPF, Jody. Google forms for staff self-assessment: Creating customization. In: College and Research Libraries. News (CRL). 2011. vol. 72, no. 10. p. 587-591. https://doi.org/10.5860/crln.72.10.8653

14. GOOGLE, Apps Script - Google Apps Script. 2023. https://developers.google.com/apps-script?hl=es-419

15. HOMOCIANU, Daniel; HOMOCIANU, Mihaela. GiPlot: An interactive cloud-based tool for visualizing and interpreting large spectral data sets. In: Spectrochimica Acta Part A Molecular and Biomolecular Spectroscopy. 2019. vol. 209, p. 234-240. http://doi.org/10.1016/j.saa.2018.10.046

16. JHAN, Prasobj; SREEKUMAR, M; KURIAKOSE, Rosemary. Enhancing library services with artificial intelligence: A framework for an automated news delivery system. In: IFLA Journal. 2025. vol. 51, no. 3. p. 836-848. http://doi.org/10.1177/03400352251358064

17. KAPRIELIAN, Paul; PHAM, Thach; YOUNG, Alex; DOW, Douglas; GRENQUIST, Scott; DECKER, Bruce. Winter Cabin Water Tap Monitor and Release System to Reduce Risk of Freezing. In: 2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). 2019. p. 606-609. http://doi.org/10.1109/uemcon47517.2019.8993098

18. KISHAN, Surapaneni; HAMEED, Syed; CHARAN, Jampani; SRIKAR, Yaragudipati. RFID-Based Elevator Access Control System. In: 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT). 2024. p. 1-5. http://doi.org/10.1109/icccnt61001.2024.10724057

19. KOK-HOE, Ho; YOTSUYANAGI, Tamon. Development of Smart Mini Manufacturing System Model for Industry 4.0 Application. In: Journal of Advanced Research in Applied Mechanics. 2024. vol. 114, no. 1. p. 94-108. https://semarakilmu.com.my/journals/index.php/appl_mech/article/view/5275

20. LAXMI, Geetha; JACOB, Alwyn; G-S, Jeevitha; MOHANDAS, Mahesh; GATTY, Mithali. Web-based University Results Analytics Tool. In: Grenze International Journal of Engineering and Technology. 2024. Vol. 10, no. 2. p. 5157-5162. https://thegrenze.com/index.php?display=page&view=journalabstract&absid=3340&id=8

21. LIRA-CAMARGO, Jorge; OGOSI-AUQUI, José Antonio; VERA-TITO, Francisca Sonia. Implementation of Google App script for automatic generation of pre-registration form. In: DYNA. 2025. vol. 92, no. 238. p. 35-38. https://doi.org/10.15446/dyna.v92n238.117750

22. LOGESHWARAN, E; MADANIKA, M; MANUPRABHA, R; VENKATESHKUMAR, M; SUYAMPULINGAM, A. Smart Agriculture: Real-Time Environmental Monitoring and Temperature Forecasting Using Machine Learning. In: 2024 2nd International Conference on Cyber Physical Systems, Power Electronics and Electric Vehicles (ICPEEV). 2024. p. 1-6. http://doi.org/10.1109/icpeev63032.2024.10932004

23. LUTSENKO, Galyna; HRYTSENKO, Valerii. Using Low-Code Development in Teaching Project Work Technology. In: Futureproofing Engineering Education for Global Responsibility. 2025. vol. 1280. p. 501-508. http://doi.org/10.1007/978-3-031-83523-0_46

24. NISHIOKA, Shie; OKAMOTO, Hiroyuki; CHIBA, Takahito; KITO, Satoshi; ISHIHARA, Yoshitomo; ISONO, Masaru; ONO, Tomohiro; MIZOGUCHI, Asumi; MIZUNO, Norifumi; TOHYAMA, Naoki; KUROOKA, Masahiko; OTA, Seiichi; SHIMIZU, Daisuke. Technical note: A universal worksheet for failure mode and effects analysis-A project of the Japanese College of Medical Physics. In: Medical Physics. 2024. vol. 51, no. 5. p. 3658-3664. http://doi.org/10.1002/mp.17033

25. PACIOS, David; IGNACIO-CERRATO, Sara; VAZQUEZ-POLETTI, José Luis; MORENO-VOZMEDIANO, Rafael; SCHETAKIS, Nikolaos; STAVRAKAKIS, Konstantinos; DI-IORIO, Alessio; GOMEZ-SANZ, Jorge J; VAZQUEZ, Luis. Amazon Web Service–Google Cross-Cloud Platform for Machine Learning-Based Satellite Image Detection. In: Information. 2025. vol. 16, no. 5. p. 381. http://doi.org/10.3390/info16050381

26. PREECHADECH, Jirayus; YONGSIRIWIT, Karn. Development of a Smart Office System through LINE Official Account: A Case Study of the Public Sector Development Group, Department of Medical Services. In: 2024 8th International Conference on Information Technology (InCIT). 2024. p. 411-416. https://doi.org:10.1109/InCIT63192.2024.10810552

27. RAJPUT, Pruthvish; PAREKH, Rutu. On-Board Diagnostics based remote emission test for Light Motor Vehicles. In: 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT). 2020. p. 1-6. http://doi.org/10.1109/CONECCT50063.2020.9198374

28. RAMADHANI, Shumaya. The utilization of G-suite features combination on developing small size of android application. In: 2020 3rd International Conference on Applied Science and Technology. 2020. p. 230-235. http://doi.org/10.1109/iCAST51016.2020.9557668

29. REVUELTA-ARRIBAS, Alberto. Creación de un complemento para Google Apps Script para la gestión de datos UPM. Trabajo Fin de Grado. Grado en Ingeniería Informática. Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, España. 2020. https://oa.upm.es/58093/

30. SHIVAKUMAR, Ramasami. Implementation and Effective Utilization of Analytical Tools and Techniques in Knowledge Management. In: International Journal of Recent Technology and Engineering (IJRTE). 2019. vol. 8, no. 2. p. 6469-6482. http://doi.org/10.35940/ijrte.a9488.078219

31. SIIMAN, Leo; MÄEOTS, Mario. Assessing Students' Digital Literacy with Interactive Computer-Based Tasks Created in Google Apps Script In: EDULEARN19 Proceedings. 11th International Conference on Education and New Learning Technologies. 2019. p. 10265-10271. http://doi.org/10.21125/edulearn.2019.2578

32. THAKKAR, Devang; RAJPUT, Pruthvish; DUBEY, Rahul; PAREKH, Rutu. Design and Implementation of Autonomous UAV Tracking System Using GPS and GPRS. In: Progress in Advanced Computing and Intelligent Engineering – Proceedings of ICACIE 2017. 2019. vol. 714. p. 433-439. http://doi.org/10.1007/978-981-13-0224-4_39

33. THOMAS, Peter. Bespoke automation of medical workforce rostering using Google's free cloud applications. In: Journal Of Innovation in Health Informatics. 2017. vol. 24, no. 4. p. 334-338. http://doi.org/10.14236/jhi.v24i4.885

34. TORRES-VELANDIA, Serafín Ángel; BARONA-RÍOS, César; GARCÍA-PONCE-DE-LEÓN, Omar. Infraestructura tecnológica y apropiación de las TIC en la Universidad Autónoma del Estado de Morelos. Estudio de caso. En: Perfiles Educativos. 2010. vol. 32, no. 127. p. 125-127. http://doi.org/10.22201/iisue.24486167e.2010.127.18881

35. UMNG, Anuario Estadístico Institucional UMNG 2022, Bogotá, 2022. Universidad Militar Nueva Granada. https://umng.edu.co/gestion-estadistica/

36. VELÁSQUEZ-GUEVARA, Sebastian; PEDRAZA, Gilberto; CHAVARRIAGA, Jaime. Multi-SPLOT: Supporting Multi-user Configurations with Constraint Programming. In: Applied Informatics. ICAI 2018. Communications in Computer and Information Science. 2018. Vol. 942. p. 364-378. http://doi.org/10.1007/978-3-030-01535-0_27

37. WARD, Anthony. An example of the use of Google Suite Apps to monitor student engagement in a complex delivery situation. In: 2021 30th Annual Conference of The European Association for Education in Electrical and Information Engineering (EAEEIE). 2021. p. 1-5. http://doi.org/10.1109/eaeeie50507.2021.9530855

38. WHITE, Laurie; ALLEN, Robert. Using Google apps script for classroom management and more. In: Journal Of Computing Sciences in Colleges. 2014. vol. 30, no. 2. p. 171-173. https://dl.acm.org/doi/abs/10.5555/2667432.2667456

39. WIBISONO, Yusuf., WARDHANA, Mada; PAMUNGKAS, Wisnu. Design of Google Apps-based helpdesk ticketing system at University using extreme programming approach. In: The 2nd International Conference of Science and Information Technology in Smart Administration. 2022. vol. 2658, no. 1. https://doi.org/10.1063/5.0106944