International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

A Spreadsheet-Based Heuristic Approach to Bulk Identity and License Lifecycle Management in Google Workspace Environments

Arora Goldy

Customer Engineer II, Google Cloud, Monroe Township, New Jersey, USA

Abstract: ***The bulk identity and license provisioning capabilities of Google Workspace spreadsheets is an interesting case study of how organizations will bridge the divide between administrative application programming interfaces and the boundaries of native graphical user interfaces. The problem space is expanding software as a service portfolios and their utilization gap, often on shadow information technology with the issues of budget inefficiency, and increased information security exposure. The Interface-as-Code concept is demonstrated through a case in which Google Sheets is used as the control plane for the Admin software development kit and Google Application Programming Interfaces for the G License Manager tool. It enables an administrator to externalize governance logic into an auditable, reproducible artifact. Using socio-technical systems (STS) theory and absorptive capacity theory, the study interprets the role of spreadsheet practices in the organization's learning, controlling and translating of policy intent into practice. The study contributes to global software engineering research by formalizing spreadsheet-based administrative work as an End-User Development (EUD) control mechanism: spreadsheet cells function as parameterized API invocations that instantiate an administrative REPL-like feedback cycle of state reading, interpretation, and change execution. This conceptualization reframes spreadsheet heuristics not as an implementation recipe but as a researchable unit, suggesting measurable constructs and testable propositions about cost, risk, and operational scalability in SaaSOps. The paper thus provides a foundation for subsequent empirical evaluation and cross-context generalization of Interface-as-Code approaches in enterprise SaaS administration.***

Keywords: Interface-as-Code, identity lifecycle management, Google Workspace, SaaS Operations, heuristic algorithms

1. Introduction

The current migration of corporate IT infrastructure from on-premise to cloud services presents new challenges to asset management practitioners. Worldwide spending on public cloud services is expected to grow from USD 805 billion in 2024 to nearly double by 2028, according to IDC. Software as a Service applications are set to comprise more than 40% of this spending. [1] Modern organizations manage, on average, between 106 and 275 different SaaS applications, resulting in a fragmented environment where IT departments lose visibility into license utilization [2].

This phenomenon, known as the utilization gap, leads to substantial financial losses. The problem is exacerbated by the growth of Shadow IT, in which employees purchase or use software without the IT department's knowledge, creating not only budget leakage but also critical vulnerabilities in the security perimeter [3]. In this context, Google Workspace, as one of the dominant productivity platforms, provides an illustrative case: the platform’s scalability often outpaces administrators’ capacity to control it effectively [4].

The fundamental problem addressed in this study is dual in nature. On the one hand, a complexity gap exists. Google Workspace offers powerful developer tools, including the Admin SDK, Directory API, and License Manager API, which enable granular and bulk operations [5]. However, effective use of these tools requires programming skills in Python, Node.js, or Google Apps Script, the system administrators may have to quickly ramp up to [6].

On the other hand, there exists a UI Limitation. Despite extensive research on SaaS governance, identity lifecycle management, and cloud administration tooling, limited attention has been paid to end-user programmable interfaces that bridge administrative APIs and non-developer operators. In particular, spreadsheet-based control mechanisms remain underexplored as a formal architectural and methodological approach rather than as ad hoc operational workarounds. The native Google Admin Console is optimized for atomic operations on individual users. It can be time-consuming for complex bulk processing [7]. The console often cannot implement the conditional logic required by modern business processes, for example: Find all users in the marketing department who have not logged in for 30 days and downgrade their license to Business Starter. This forces administrators into manual labor, which becomes economically and operationally unacceptable at the scale of thousands of users.

The purpose of this study is to conceptually and empirically substantiate spreadsheet-based heuristics as a scalable and secure approach to bulk identity and license lifecycle management in SaaS environments.

To achieve this purpose, the study addresses the following objectives:
- To analyze the architectural principles underlying spreadsheet-driven administrative control planes;
- To evaluate the economic and operational effects of spreadsheet-based automation in identity and license management;
- To assess security implications of OAuth-based execution models in end-user programmable administrative tools;
- To validate the applicability of spreadsheet heuristics for

scalable identity lifecycle management under real-world constraints.

The scientific novelty of the study lies in conceptualizing G License Manager not merely as a utility, but as a new technical framework within the End-User Development (EUD) paradigm. In an academic context, the mechanism is examined whereby spreadsheet cells function not as passive data stores, but as active parameters for API invocations to manage the lifecycle of corporate identity. The deployment scale of the tool (more than 1 million users) allows the resulting data to be considered empirical evidence of a paradigm shift in IT system management from proprietary consoles to flexible, user-programmable interfaces.

2. Materials and Methods

The study employs a combined methodology that incorporates a case study of the G License Manager tool and an algorithmic analysis of its codebase and operating logic. The selection of the object of study is justified by its broad adoption (>1 million users) and its unique architecture, which addresses industry problems. Mathematical modeling is also used to evaluate the economics of these proposals under practical conditions.

The entire system is serverless and Google Cloud based, with computational and integration logic running in the cloud without needing a dedicated server and using managed platform services.

Google Sheets serves as the interface layer, providing the user with an environment for entering data, showing the output and triggering commands, and used as a single point of access for the user to use the tool.

The logic layer is implemented in Apps Script, that is a cloud-based JavaScript platform powered by Google Drive. It implements all the business logic of the application and data validation, orchestrating the requests, linking the events happening in the spreadsheet to the calls to the outside services, and ensuring the correctness of the invoked actions.

The Google Admin SDK and its application programming interfaces (APIs), such as the Directory API for user management, the License Manager API for license management, and the Reports API for auditing network activity, provide a unified access plane to administrative activities and data (the administrative data plane).

As stated above, a key feature of the library is the Interface-as-Code model, which interprets a data structure from the spreadsheet. The script then transforms changes made (e.g., a change of status in a given cell) into JSON payloads for RESTful requests to Google APIs.

The central element of the system is the redundancy detection algorithm. It is based on cross-referencing heterogeneous data that reside in different reports within the native console.

The algorithm executes the following logic:
1) Extraction of lastLoginTime from the Directory API.
2) Extraction of assignedSku (license SKU) from the License Manager API.
3) Application of a conditional operator:

$$IF\ (T_{current} - T_{last\ login} > \Delta t) \wedge (C_{SKU} > C_{threshold}), THEN\ Action\ Required \quad (1)$$

where Δt is the idle-time threshold (e.g., 30 days), and C_{SKU} is the cost of the current license. This enables the identification of expensive users with low activity.

To overcome API constraints (Rate Limits) that protect Google infrastructure from DDoS attacks and overload, the tool implements a batch processing mechanism (Batch Requests).

The tool uses request aggregation as a baseline optimization mechanism: instead of executing N per-user requests, operations are grouped into up to 50 actions within a single batch request. This scheme reduces overhead associated with connection establishment and decreases the number of network interactions when processing large datasets.

To account for information-theoretic limitations and temporary service unavailability, exponential backoff is used. When a 503 Service Unavailable or 429 Too Many Requests error code is received, the algorithm waits for 2k seconds, where k is the number of the attempt, before the next attempt. This has proven to be more reliable on thousands.

For elevated security risks of running the scripts, the tool uses the OAuth 2.0 authorization protocol to request short-lived tokens for limited scope. This avoids permanently storing credentials. That is generally considered a security antipattern. This ensures that the script acts on behalf of the administrator only within the necessary privileges and only during the execution session.

Methodologically, the study follows a qualitative-dominant mixed design combining (1) a single-case study of a widely deployed administrative tool, (2) algorithmic analysis of its operating logic, and (3) analytical economic modeling based on observed operational scenarios. The large deployment scale of the tool enables the results to be interpreted not as isolated anecdotal evidence, but as representative of recurring administrative patterns in large SaaS environments.

3. Results and Discussion

3.1. Interpretation of Architectural Results

The architectural analysis of G License Manager confirms the effectiveness of the Interface-as-Code approach for system administration tasks. In this logic, the spreadsheet is transformed from a passive analysis tool into a control plane, i.e., a working environment in which data are not only observed but also used for controlled impact on the target system.

This transformation creates a closed Read-Eval-Print Loop (REPL) for administrators. In the Read stage, data are imported from the API into the spreadsheet. In the Eval stage, Sheets filters, sorting, and formulas are applied as computational and analytical operations that support decision-making based on the current data state. In the Print

(Execute) stage, changes are sent back to the API through the script, completing the loop and converting spreadsheet processing results into executable actions.

Such isomorphism (correspondence) between data representation and the management interface eliminates the cognitive dissonance that arises when working across multiple browser tabs in the native console [10].

The diagram below illustrates how Google Sheets serves as an intermediary, abstracting API complexity from the end user.

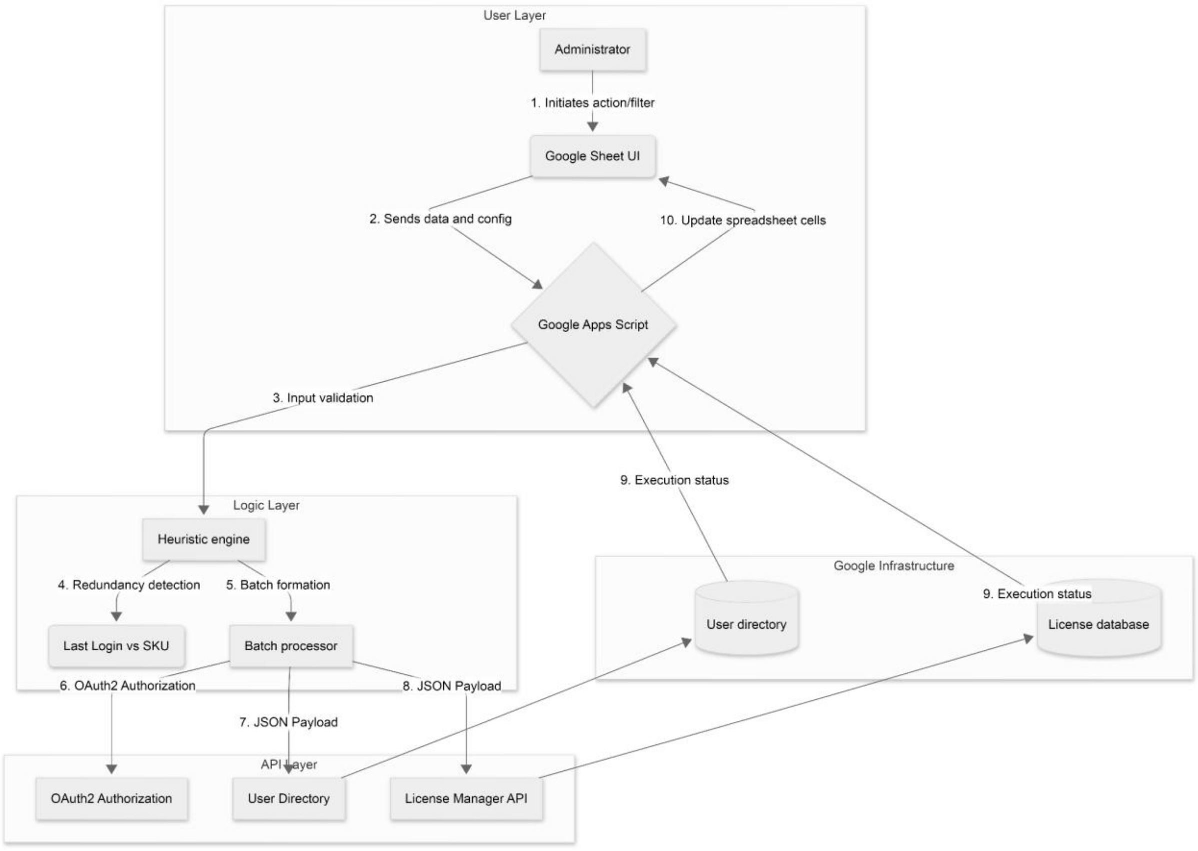

Figure 1: Architectural data flow in G License Manager

3.2. Economic Effectiveness: Model Analysis

The study validates two key economic models and compares them with industry data.

Consider the model for eliminating Zombie SaaS (i.e., inactive or under-utilized licensed accounts that still incur recurring subscription costs). Direct savings are achieved through deprovisioning or downgrading licenses for inactive users. The calculation formula is:

$$S_{direct} = \sum_{i=1}^{n} (C_{old\_i} - C_{new\_i}) \times 12, \quad (2)$$

where S_{direct} is annual direct savings, n is the number of identified inactive users, C_{old_i} is the cost of the current license (e.g., Enterprise Plus ~$30/month), and C_{new_i} is the cost of the new license (e.g., Archived User ~$5/month or $0 upon deletion).
Consider the case analysis. For an organization with 2000 users, under a conservative estimate of 5% zombie accounts (100 users) 7, identifying and replacing Enterprise licenses with Archive saves:

$$100 \times (30 - 5) \times 12 = \$30{,}000\ a\ year.$$

Given that the tool is used by more than 1 million clients, the cumulative global effect is measured in hundreds of millions of dollars.

The second model, the administrative efficiency model, estimates labor reduction (S_{labor}) through automation:

$$S_{labor} = N_{ops} \times (T_{manual} - T_{tool}) \times R_{admin}, \quad (3)$$

where N_{ops} is the number of operations (users), T_{manual} is the manual processing time per user (~2 min), T_{tool} is the automated processing time (amortized time per user in a batch ~0.005 min), and R_{admin} is the administrator's hourly rate.

Consider the case analysis of a bulk update for 1000 users, for example, when changing organizational units during restructuring. In such a scenario, labor inputs under manual and automated execution are compared, allowing the difference in the time required to process a large number of objects to be established.

In manual mode, the time estimate is 1000×2 min≈33.3 hours. In automated mode, the process takes ~5 minutes, including setup. At a rate of 60/hour, the savings are:

$$1000 \times (2 - 0.005) \times \$1 \approx \$1{,}995\ per\ operation.$$

A 99.7% time reduction supports the hypothesis that automation through a spreadsheet interface radically increases IT staff productivity, freeing time for strategic tasks. Table 1 presents a comparative analysis of the effectiveness of management methods.

Table 1: Comparative analysis of the effectiveness of management methods

Comparison parameter	*Manual mode (Admin Console)*	*Scripting (Python/API)*	*G License Manager (Sheets)*
Required skills	Low (GUI)	High (coding)	Medium (Excel/Sheets)
Speed (1,000 users)	~33 hours	~2–5 minutes	~5 minutes
Error risk	High (human factor)	Low (after debugging)	Low (visual control)
Logic flexibility	Low (rigid UI)	Maximum	High (formulas/filters)
Implementation cost	$0 (built-in)	High (development)	Low (ready-made solution)

While industry benchmarks are used to contextualize the magnitude of potential savings, the proposed economic models are intended as order-of-magnitude estimators rather than precise financial forecasts, with their primary value lying in comparative and directional analysis.

3.3. Technical Implementation of Bulk Provisioning

A critical aspect of the tool's operation is handling API quotas. Google documentation specifies strict limits on query frequency (QPS) [8]. Simple iteration over 1,000 users will result in 429 Too Many Requests or 503 Service Unavailable errors.

The tool resolves this issue through an algorithmic sequence visualized in Fig. 2.

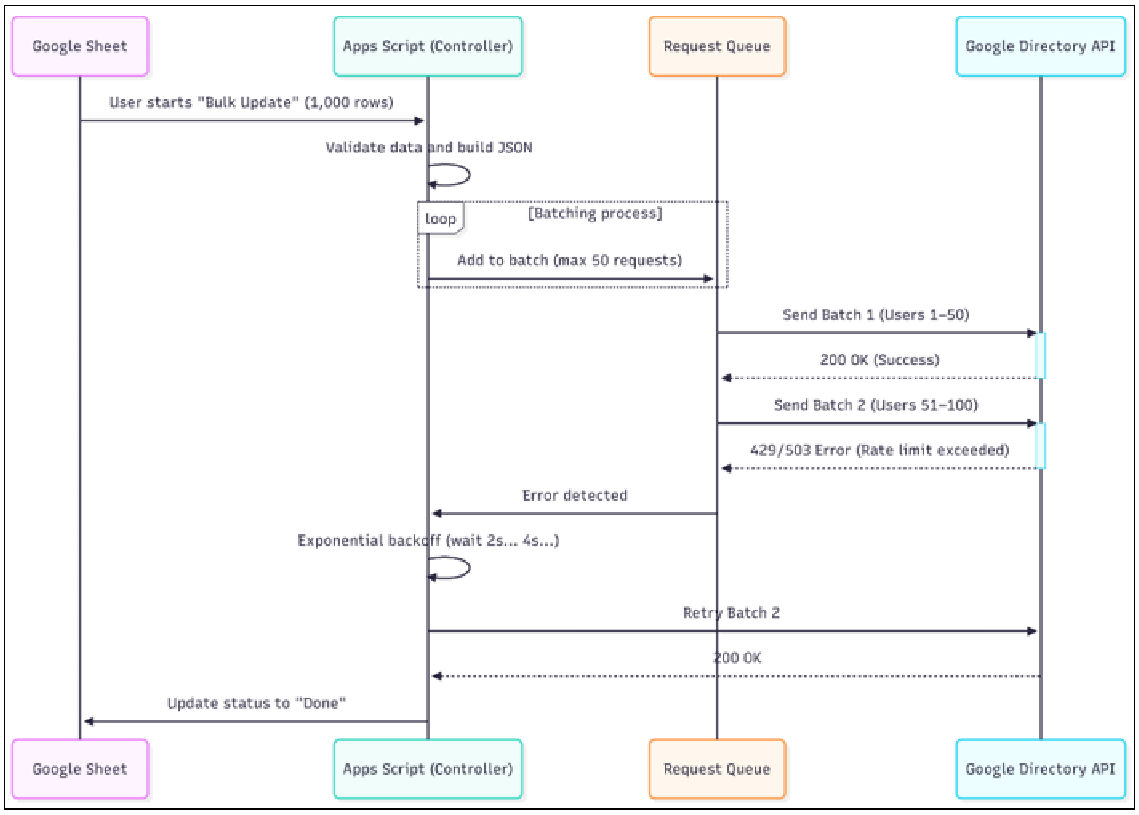

Figure 2: Sequence diagram of batch request processing with exponential backoff mechanism

Fault tolerance allows the tool to be used in situations where simple scripts would not work, making it ideal for enterprise use.

3.4 Case Analysis: Automation of Archiving and Deprovisioning

One of the most complex items in ILM is offboarding employees. This is because actors must control security, like when they disable access the moment they decide, while they must preserve useful information for compliance or legal holds. Therefore, the challenge is balancing the need to rapidly disable an account with the need to ensure information preserves itself for audit or legal purposes.

G License Manager implements a workflow that automates license swaps. First, identification is performed, i.e., locating users with the Leaver status in the spreadsheet. Next the lock is data-enabled with a process called immediate account suspension. Next is the license change by removing the expensive license and assigning the cheaper Archived User to save the cost of the expensive license. The last step is to log for auditing which action was performed. Figure 3 depicts the lifecycle of automated user archiving.

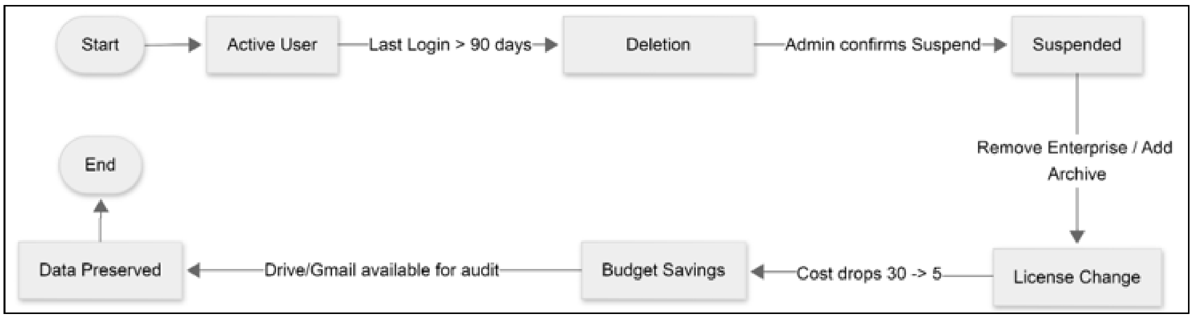

Figure 3: Life cycle of automated user archiving

This process illustrates the synergy of security and economics achieved through automation.

3.5 Barriers and Risks

However, the downsides of spreadsheet-based approaches include a risk of shadow IT, as the relative ease of creating spreadsheets encourages end users to create uncontrolled numbers of scripts, auditing of which will need a thorough recurring process. This risk is partially offset by the fact that the tool requires the Super Admin role: its use is restricted to users with elevated permissions.

A separate issue is interface scalability. Google Sheets has a hard limit of 10 million cells [9], so in extremely large organizations, such an interface may operate slowly. As a result, a practical need may arise to migrate to a database, such as BigQuery.

Finally, code security risks must be taken into consideration. Because the script code is accessible for editing (Interface-as-Code), an attacker who gains access to the spreadsheet file could potentially modify the tool's logic to execute malicious bulk actions. In this context, access control list (ACL) management for the tool file itself becomes critically important.

From a research perspective, the study is limited by its reliance on a single platform ecosystem (Google Workspace), which may constrain the generalizability of findings to other SaaS environments. However, the underlying principles of Interface-as-Code and spreadsheet-based heuristics are platform-agnostic and may be transferable to other administrative contexts supporting API-driven control.

4. Conclusion

The conducted study confirms that G License Manager constitutes a significant innovation in IT system management. The Interface-as-Code model effectively eliminates the gap in complexity between the capabilities of the Google Workspace API and an administrator's needs.

The cost savings of the Zombie SaaS elimination model are demonstrated by the economic efficiency of even finding a small percentage of inactive licenses, something that is potentially scalable to the global level. The savings estimated in this case are hundreds of millions of dollars.

Operationally, this is illustrated by the fact that cataloguing of bulk operations is automated, in some cases reducing the amount of time required to process the operation by approximately 99%, giving people additional free time to focus on policy.

The system also makes use of a batch processing system with exponential backoff to help with the problem of API quotas, and thus to achieve enterprise level execution quality.

As we've seen with a million users, this democratization of the toolchain is a worldwide phenomenon. The future of system administration tools is not in the artifice of blindingly complex consoles, but in meta-design environments that allow the user to do work in an user interface they understand, as opposed to requiring advanced knowledge of a tool. The cloud scripts have changed spreadsheets from purely analytical tools to a control console for the corporate infrastructure. A firm that is willing to invest in cost reduction and increased security in the post-COVID digital economy will benefit by using these heuristic tools.

References

[1] Shirer M. Worldwide Spending on Public Cloud Services is Forecast to Double Between 2024 and 2028, According to New IDC Spending Guide [Internet]. IDC. 2024 [cited 2025 Dec 1]. Available from: https://my.idc.com/getdoc.jsp?containerId=prUS52460024

[2] BetterCloud. Why IT uses BetterCloud platform for full control of the SaaS environment [Internet]. BetterCloud. 2025 [cited 2025 Dec 2]. Available from: https://www.bettercloud.com/monitor/why-bettercloud-saas-management-platform/

[3] Nizamuddin M. Investigating the cybersecurity risks of remote work: a systematic literature review of organizational vulnerabilities and mitigation strategies. International Journal of Information Security. 2025 Jul 27;24:187.

[4] Tang W, Yang S. Enterprise Digital Management Efficiency under Cloud Computing and Big Data. Sustainability [Internet]. 2023 Jan 1;15(17):13063. Available from: https://www.mdpi.com/2071-1050/15/17/13063

[5] Ricaurte H, Florez H. Architectural Approach for Google Services Integration. Communications in computer and information science. 2021 Jan 1; 1455: 483–96.

[6] Kalmukov Y. Research and Analysis of Employers' Opinion on the Necessary Skills that Students in the Field of Web Programming Should Possess.

Proceedings of the Third National Scientific-Practical Conference "Digital Transformation Of Education" - Problems and Solutions. 2025 Jun 4;259–64.

[7] Ntoa S. Usability and User Experience Evaluation in Intelligent Environments: A Review and Reappraisal. International Journal of Human-Computer Interaction. 2024 Sep 12;41(5):2829–58.

[8] Google for Developers. Limits and Quotas [Internet]. Google for Developers. 2025 [cited 2025 Dec 7]. Available from: https://developers.google.com/workspace/admin/reports/v1/limits

[9] Google Workspace Updates. Google Sheets doubles cell limit [Internet]. Google Workspace Updates. 2022 [cited 2025 Dec 8]. Available from: https://workspaceupdates.googleblog.com/2022/03/ten-million-cells-google-sheets.html

[10] Thees M, Kapp S, Strzys MP, Beil F, Lukowicz P, Kuhn J. Effects of augmented reality on learning and cognitive load in university physics laboratory courses. Computers in Human Behavior. 2020 Jul;108:106316.