Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
https://sol.sbc.org.br/journals/index.php/rbie

Submission: 03/05/2024; Camera ready: 24/01/2025; 1st round notif.: 15/10/2024; Edition review: 06/03/2025; New version: 08/11/2024; Available online: 06/03/2025; 2nd round notif.: 28/11/2024; Published: 06/03/2025;

Technical and tactical factors of an educational platform aiming to deal with a large number of students and inequalities when teaching programming for STEM students

Gabriel Xará
Universidade Federal do Estado do Rio de Janeiro
ORCID: 0009-0007-4231-8980
gabriel.xara@edu.unirio.br

Carla A. D. M. Delgado
Universidade Federal do Rio de Janeiro
ORCID: 0000-0003-3570-4465
carla@ic.ufrj.br

Claudio M. de Farias
Universidade Federal do Rio de Janeiro
ORCID: 0000-0002-1927-7398
cmicelifarias@cos.ufrj.br

Hugo Folloni Guarilha
Universidade Federal do Rio de Janeiro
ORCID: 0009-0005-1145-8222
hugofg@ic.ufrj.br

Laura O. Moraes
Universidade Federal do Estado do Rio de Janeiro
ORCID: 0000-0003-0965-6703
laura@uniriotec.br

João Pedro Freire
Universidade Federal do Rio de Janeiro
ORCID: 0009-0002-4913-2795
joaofreire@dme.ufrj.br

Eldânae Nogueira Teixeira
Universidade Federal do Rio de Janeiro
ORCID: 0000-0002-9857-8710
eldany@ic.ufrj.br

Abstract

Considering the characteristics found in the post-pandemic scenario of public higher education in Brazil, we must address issues related to equity in access to educational resources. In STEM (Science, Technology, Engineering and Mathematics) courses, these differences become striking regarding the conditions and resources for extra-class study that the student can count on. Machine Teaching is a web-based system used in introductory programming classes to support students and instructors and has been used since 2018 in Universidade Federal do Rio de Janeiro. In this work, we present the challenges faced to adapt Machine Teaching to the post-pandemic scenario. Issues of equity in access were addressed in three ways: changes to the system architecture, optimization of page loading times, and actions to improve user satisfaction. Our novel implementation allows students to undertake exercises from any device with internet connectivity, enabling access from low computational power machines. Also, our optimization results decreased up to 80% the web system loading time, allowing slower connections to still access the system. This work contributes to the discussion on the risks and benefits of using digital technologies in education, placing our specific scenario in the big picture of inequality of conditions in connectivity, AI and data analysis in education.

Keywords: *nformatics in education; educational data mining; higher education; digital technologies; programming education; introductory programming; digital equity; computer science education*

1 Introduction

According to Lima Neto, 2012, a systemic and articulated vision of policies for social development must contemplate quality primary and higher education for large portions of the population, combined with heavy investments in science, technology, and innovation. A brief description of three forms of higher education was provided by Trow, 2007: (1) elite-shaping the mind and character of a ruling class, a preparation for elite roles; (2) mass-transmission of skills and preparation for a broader range of technical and economic elite roles; and (3) universal - "whole population" adaptation to rapid social and technological change.

The expansion of higher education gained momentum on the Brazilian agenda around 2000. The 21st century's first two decades were a significant encouragement period from the Brazilian Federal Government to expand access to higher education. During these years, the Federal Government offered more than 9 million higher education student vacancies, directly or indirectly, in all of Brazil's federate units[1].

The attendees' scenario in Brazilian universities changed due to policies to democratize access to higher education, which include opening new institutions in all regions of the country, reserving places for students from public schools (quotas policy - Lima et al., 2014), and offering students financial support. Consequently, the educational work of higher education also changes. A critical point is the increase in the number of students per classroom, as mentioned by Ake-Little et al., 2020; Mitchell et al., 1990. Although instructors and students may prefer classes with a small number of students (Baker et al., 1974; Gomes, 2005), this option has lost economic viability in Brazilian public universities compared with other priorities for resource investments.

To deal with a large number of students per instructor at a public university located in a region characterized by great socioeconomic disparities like Rio de Janeiro, we must address issues related to equity in access to educational resources. In courses in the STEM area, these differences become striking regarding the conditions and resources for extra-class study that the student can count on; the knowledge and study habits that the student brings from their school life; and the type of mentoring and need for adaptation that will be necessary for them to succeed in university education.

In a country like Brazil, a significant portion of university students must be assisted by financial support policies to make their studies viable. Their personal access conditions to resources such as the internet and computers are precarious. In regions of social vulnerability, such as *favelas*, broadband internet access services are rarely available (Cunha et al., 2023). Conditions worsened during the COVID-19 pandemic when remote teaching became the only possible practice (Castioni et al., 2021). Also, several students' families lost their main sources of income. In 2023, as the WHO declared the end of the pandemic state, Brazilian universities fully returned to face-to-face activities, with a deficient budget and highly deteriorated infrastructure and computational equipment (Hipólito et al., 2022).

The lack of personalized attention from the instructor is another issue in classes with many students, which leads to an impersonal relationship between these two fundamental actors in

[1]Data obtained from the 2017 report "Censo da Educação Superior: Sinopse Estatística – 2017" produced by "Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP)". Available at https://www.gov.br/inep/pt-br/acesso-a-informacao/dados-abertos/sinopses-estatisticas

teaching and learning. The "just another one in the crowd" effect negatively influences students' impressions of their own relevance and academic ambitions. Lack of motivation and lack of self-perception about the whole are consequences of impersonality and strongly contribute to dropout rates, particularly among students without rewarding and encouraging school experiences (Rodrigues and Chechia, 2017). To a large extent, this group coincides with students who live in disadvantageous socioeconomic scenarios (Paulilo, 2017), and who would undergo an effective social transformation if successful in higher education.

In previous scenarios, the student's perception of their own learning was constructed with great influence from the attention given by the instructor, tutor, or lecturer to each student, both through comments on their activities and evaluations, and through personalized, direct interactions. In addition to fewer opportunities for personalized interactions, the current scenario makes it difficult for instructors to understand each student's learning process (Ake-Little et al., 2020). Designing alternatives to deal with this difficulty is one of the most important factors (if not the most important) in mitigating the loss of teaching quality as the number of students per class increases.

In the quest to maintain introductory programming courses running at Universidade Federal do Rio de Janeiro (UFRJ), the main pedagogical support digital tool used, Machine Teaching (Moraes et al., 2022), was modified in favor of students' equal access and class size increase. Machine Teaching is an online learning environment to support introductory programming classes. Its features include an in-browser integrated development environment (IDE) for students to code and submit their assignments; dashboards where instructors can analyze students' difficulties; and alerts for instructors signaling students who are at risk of low performance or dropout. Machine Teaching's main functionalities and achievements have been presented in previous papers (Moraes et al., 2021, 2022).

The system is useful for students as it provides didactic support for programming practice through automatic testing and feedback mechanisms, as well as operational support through the student activities' organization and dashboards with information on expected resolution time and the activities' difficulty. For instructors, the tool is useful for dealing with a large number of students by supporting programming task feedback, providing dashboards to visualize class difficulties and students' performance over time, and alerting about students at risk of dropping out.

In this article, we present the points in which the educational platform Machine Teaching was improved with the aim of better dealing with some of the challenges currently encountered in Brazilian public universities - high number of students per instructor, deterioration of the institutional computer labs, and increase in the number of students in unfavorable socioeconomic conditions. Our main contribution is to address issues related to the adaptation of digital educational resources and services to the scenario of inequalities, which we perceive to have worsened post-pandemic. This paper addresses challenges related to the United Nations (UN) sustainable global goals number 4, "quality education", and 10, "reduced inequalities".

We present recent updates in the Machine Teaching platform designed to address computer and internet access difficulties common among Brazilian university students. The architecture allows access by smartphones and machines with low computational power and deals with weak and intermittent internet connections. The tool supports instructors in managing a large number of students, even in non-ideal conditions of internet and computer access (Xará et al., 2023). Now, we extend this discussion with instruments to mitigate the decrease in personalized attention that an instructor can dedicate to each student in a large class size scenario and technology adoption resistance. Furthermore, we situate our work in the scenario of AI and data analysis in education.

Although our research has UFRJ as its scenario, the issues studied are relevant to Brazilian institutions in any region. It is important to invest in adaptations to accommodate the changes Brazilian public universities have undergone in the last decades, as their effects will be felt for many years. There is no quick prospect of recovery from the deterioration of infrastructure caused by the emptying of facilities during social isolation and the meager injection of resources during these times. The increase in the number of students per instructor could be mitigated by reducing the number of students –an option with several negative impacts– or a significant increase in the number of teachers – which is not likely to happen in the short or medium term. In addition to class size, the socioeconomic disadvantage that affects a large proportion of students also restricts the potential for transformation that higher education can bring to their lives.

The issues pointed are complex. Actions on different fronts will be necessary to tackle these problems. Among the options available in this scenario of scarcity to implement the necessary changes, we discuss the use and adaptation of educational technologies. In a crucial area like education, it is essential to make sensible use of technologies such as AI, cloud computing, and data analytics. The use of AI and data analysis in the area of education is currently the subject of great controversy (Bates et al., 2020; Borenstein and Howard, 2021; Özer, 2024). Zawacki-Richter et al., 2019 provides an overview of research on AI applications in higher education through a systematic review. Their conclusions highlight the lack of critical reflection on the challenges and risks of the use of AI in education, the weak connection to theoretical pedagogical perspectives, and the need for further exploration of ethical and educational approaches in the application of AI technology in higher education. According to Borenstein and Howard, 2021, educators and stakeholders must reflect on how AI might impact people's lives and embrace their responsibilities to enhance its benefits while mitigating its potential harms.

All the work carried out by the Machine Teaching team involves students, educators, and course managers. The agenda of the "Machine Teaching team" has three main themes: (1) programming learning; (2) the use of data analytics in education; and (3) the operationalization of programming teaching considering equity and inclusion in the Brazilian public higher education scenario. These three aspects have gone hand in hand since the launch of the platform in 2018, already coupled with a teaching plan for introduction to programming and teaching material produced in-house (Delgado et al., 2016; Freire et al., 2024; Moraes and Pedreira, 2020; Moraes et al., 2021, 2022; Sasse et al., 2024; Xará et al., 2023). It is through the articulation of these three themes that a tactical plan and a technical specification are created for the continuous evolution of programming courses at UFRJ and the Machine Teaching platform. From this larger plan, we address the issues of access difficulties, class size, and resistance to technological adoption.

Section 2 presents works related to ours – other online environments that support programming teaching and learning tasks, and also briefly describes the system Machine Teaching, highlighting the functionalities to support instructors in dealing with a massive number of students. Section 3 details how Machine Teaching's architecture was modified to support students accessing from any device with internet connectivity. Section 4 presents the improvements regarding response time optimization. Technology adoption issues are discussed in section 5. Our conclusions and future work are stated in section 6.

2 Literature review

Online environments that automatically correct source code are commonly used as pedagogical support tools in programming courses, mainly due to their speed in providing student feedback. Table 1 gathers automatic code correction environments for Python language taken from systematic literature reviews conducted by Ihantola et al., 2015 and Luxton-Reilly et al., 2018: CloudCoder (Hovemeyer and Spacco, 2013; Papancea et al., 2013), CodeWorkout (Panamalai Murali, 2016), PCRS (Zingaro et al., 2013), UUhistle (Sorva and Sirkiä, 2010), Pythy (Edwards et al., 2014), Web-CAT (Edwards and Perez-Quinones, 2008), PEEF (Araujo et al., 2021), as well as two environments developed by Brazilian universities that were not included in the conducted review: CodeBench (Galvão et al., 2016) and Beecrowd (Bez et al., 2013; Tonin et al., 2012), a popular commercial web-system (Repl.it), and the Machine Teaching environment mentioned in this paper. These environments' functionalities were analyzed and compared in a previous paper (Moraes et al., 2022). In this paper, we investigate whether these tools are suitable for use in classes with a large number of students, severe budget restrictions, and students with deficient access to computers and the internet. These tools are analyzed and compared based on five criteria[2]:

1. Is the online learning environment still active (recent developments)? When was the last update?
2. Can the environment be accessed using a smartphone?
3. Is the environment distributed as open source?
4. Does the environment provide a dedicated area for instructors with statistics and graphs so that they can have both an overview of the class and specific students? What is available?
5. Was the environment evaluated? By which method?

The second question assesses if the tool can be used in a smartphone. According to the Brazilian Institute of Geography and Statistics (IBGE), 99.6% of the individuals with internet access, do it using a smartphone in contrast with 42.7% that do it using a computer (an individual can use both). It means that only half of the people with internet access also have access to a computer with internet. But almost everyone with internet access has access to a smartphone with the internet as well[3]. Nowadays, most web systems are built upon javascript libraries and HTML5 that provide them with a responsive design and allow them to function and automatically adjust to different screen sizes and devices. Therefore, all the tools that were developed as web systems

[2]This review is limited in comparing the tool's availability and features. The analyzed papers do not present their architecture, so they could not be compared.

[3]Data obtained from the 2021 Continuous National Household Sample Survey. Available at https://sidra.ibge.gov.br/tabela/7311

Table 1: Online environments to autocorrect Python code. Dashboard features: G=grades; S=submissions; ST=statistics; D=dropout prediction.

Environment	**1. Active / last update**	**2. Smart-phone**	**3. Open Source**	**4. Dashboards**	**5. Evaluation**
CloudCoder	No / Aug 2019	N/A	Yes	G	Not evaluated
CodeWorkout	Yes / Nov 2024	Yes	Yes	G S	Questionnaire
PCRS	Yes / Aug 2024	Yes	Yes	G S	Not evaluated
UUhistle	No / N/A	No	No	No dashboard	Questionnaire and control group
Pythy	No / Feb 2020	Yes	Yes	No dashboard	Questionnaire
Web-CAT	Yes / Jun 2024	Yes	Yes	G S	Control group
PEEF	Yes / N/A	No	No	G S ST	Not evaluated
CodeBench	Yes / N/A	Yes	No	G S ST	Questionnaire and control group
Beecrowd	Yes / N/A	Yes	No	G S ST	Questionnaire
Repl.it	Yes / N/A	Yes	No	G S	N/A
Machine Teaching	Yes / Nov 2024	Yes	Yes	G S ST D	Questionnaire

ran perfectly on smartphones. The tools that did not work were developed as desktop applications or were not available online.

To consider budget restrictions, we analyzed if the tools are distributed as open source and if they still have active developments (questions three and one, respectively). A tool distributed as open source is guaranteed to be always accessible (institutions could always manage the system locally) and to never have a fee that could derail the tool's adoption. Also, recent developments indicate that the system development is active and it still has support for bugs and improved functionalities.

Finally, as analyzed by the fourth question, the investigated tools support the use by instructors and students. For students, the tools generally function as an integrated development environment (IDE), where students code and receive real-time feedback. However, the features for instructors vary significantly. In UUhistle, for example, instructors use the system for simulations and examples. Most tools allow instructors to see whether or not a student has passed a question. Additionally, Beecrowd and the Brazilians CodeBench and Machine Teaching provide student submissions with in-depth analysis through dashboards with graphs or statistics about students and classes. To efficiently deal with a large number of students, the tool must provide consolidated statistics about the questions, classes, and students, instead of only providing a final grade for a question. The statistics allow instructors and course administrators to pinpoint students' difficulties throughout the semester before their final exams and grades.

Machine Teaching differs from these systems because, in addition to supporting the regular use by students, it focuses on data collection for decision support. The dashboards presented in the system provide information so that instructors can reflect on the syllabus' order, the content difficulty, and activities' deadlines, encouraging reflection and supporting the continuous course update. It also provides a dropout prediction feature to alert instructors about students at risk of low performance or dropout.

2.1 Machine Teaching Web System Review

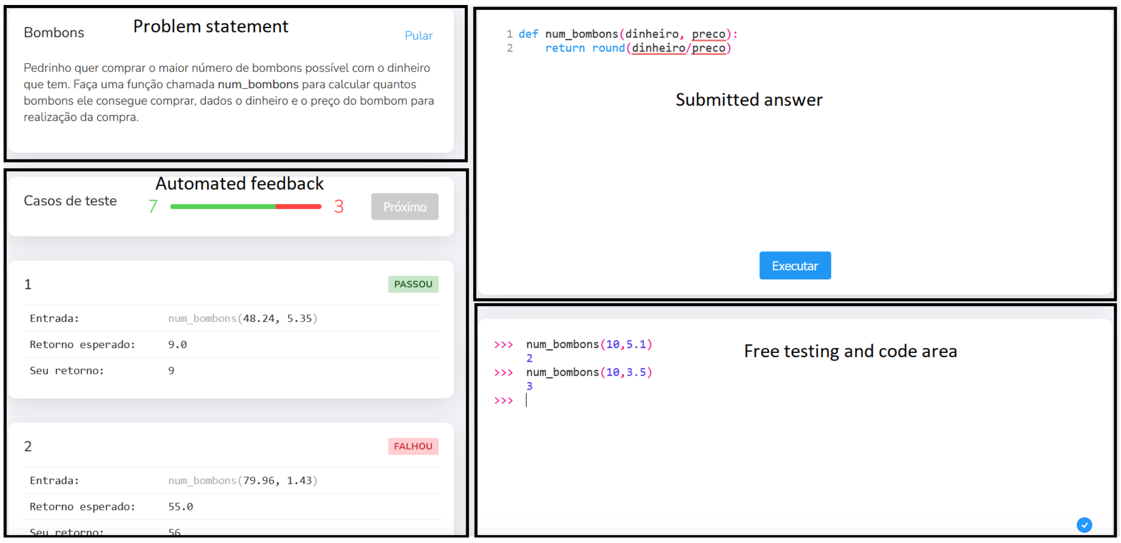
This section presents the developed online learning environment to support instructors and students in programming classes. The focus of this paper is on the new requirements and functionalities concerning the support given to instructors in dealing with a great number of students and the differences in access across university students in the Brazilian scenario. However, we felt it was important to highlight our already existing most important functionalities for the presented context and describe the system's limitations and motivation for improvement. We present the following functionalities: an auto-corrected integrated development environment (IDE) for students to solve exercises online without the need to install anything on the computer or mobile, dashboards for students' and instructors' awareness, and a dropout prediction model. These features and the system's usability and effectiveness were evaluated in previous papers (Moraes et al., 2022, Freire et al., 2024).

2.1.1 Integrated Development Environment

After logging in the system, the students have access to an IDE, where they are presented with a problem, and they should write the expected answer in a free-text coding format. For each exercise, a test case function generator was defined to correct the results. The students receive feedback every time they submit an answer, and they can see whether they passed or failed a unit test case. If they get all of them correct, the task is considered done, and the student may move on to another problem. The system saves a state every time a student submits an answer. The set of submissions is used to generate aggregated statistics regarding problems, content, students, instructors, course, among others. The IDE interface is shown in Figure 1.

2.1.2 Dashboards for awareness

The system offers three interfaces for instructors to analyze student submissions at the individual level or aggregated by class. Figures 2 and 3 present the interface with detailed information for each student, including their codes, outcomes, and the timestamp of each submission. In particular, the interface in Figure 3 was requested by the instructors to be able to compare individual student submissions. This interface displays all student submissions side by side, along with the respective submission timestamp, and the percentage of successful test cases for a single problem. In this way, the instructor can identify common errors among students.



Bombons

Pedrinho quer comprar o maior número de bombons possível com o dinheiro que tem. Faça uma função chamada **num_bombons** para calcular quantos bombons ele consegue comprar, dados o dinheiro e o preço do bombom para realização da compra.

Casos de teste 7 3

1 PASSOU

```
Entrada:            num_bombons(48.24, 5.35)
Retorno esperado:   9.0
Seu retorno:        9
```

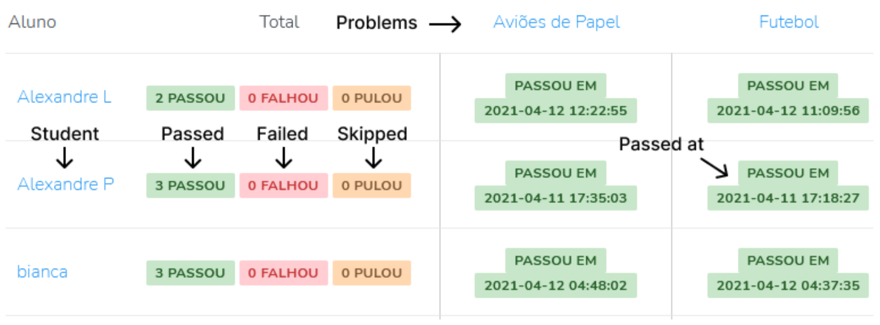
2 FALHOU

```
Entrada:            num_bombons(79.96, 1.43)
Retorno esperado:   55.0
Seu retorno:        56
```

```python
def num_bombons(dinheiro, preco):
    return round(dinheiro/preco)
```

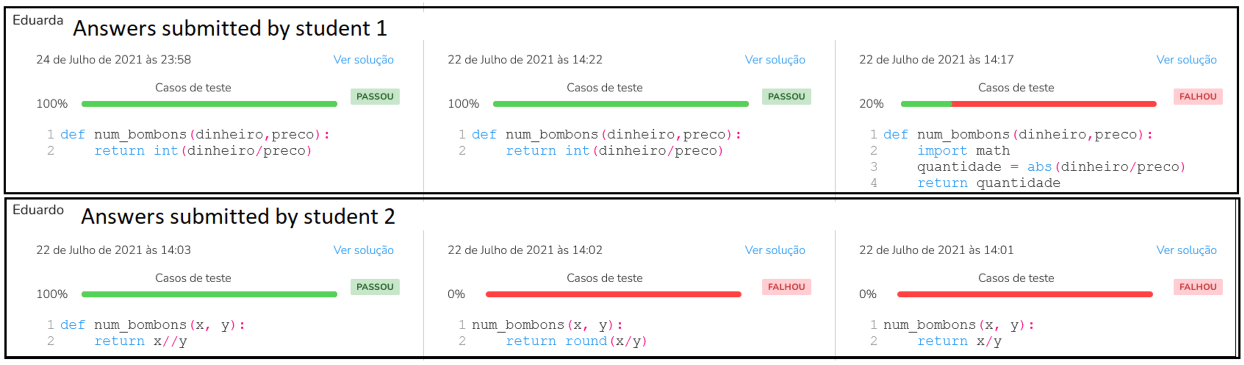
```
>>> num_bombons(10,5.1)
2
>>> num_bombons(10,3.5)
3
>>> |
```

Figure 1: Integrated development environment.



Aluno	Total			Aviões de Papel	Futebol
Alexandre L	2 PASSOU	0 FALHOU	0 PULOU	PASSOU EM 2021-04-12 12:22:55	PASSOU EM 2021-04-12 11:09:56
Alexandre P	3 PASSOU	0 FALHOU	0 PULOU	PASSOU EM 2021-04-11 17:35:03	PASSOU EM 2021-04-11 17:18:27
bianca	3 PASSOU	0 FALHOU	0 PULOU	PASSOU EM 2021-04-12 04:48:02	PASSOU EM 2021-04-12 04:37:35

Figure 2: Interface with each student's outcome for a specific chapter.


Answers submitted by student 1
Answers submitted by student 2

Eduarda

24 de Julho de 2021 às 23:58 Ver solução
Casos de teste 100% PASSOU

```python
def num_bombons(dinheiro,preco):
    return int(dinheiro/preco)
```

22 de Julho de 2021 às 14:22 Ver solução
Casos de teste 100% PASSOU

```python
def num_bombons(dinheiro,preco):
    return int(dinheiro/preco)
```

22 de Julho de 2021 às 14:17 Ver solução
Casos de teste 20% FALHOU

```python
def num_bombons(dinheiro,preco):
    import math
    quantidade = abs(dinheiro/preco)
    return quantidade
```

Eduardo

22 de Julho de 2021 às 14:03 Ver solução
Casos de teste 100% PASSOU

```python
def num_bombons(x, y):
    return x//y
```

22 de Julho de 2021 às 14:02 Ver solução
Casos de teste 0% FALHOU

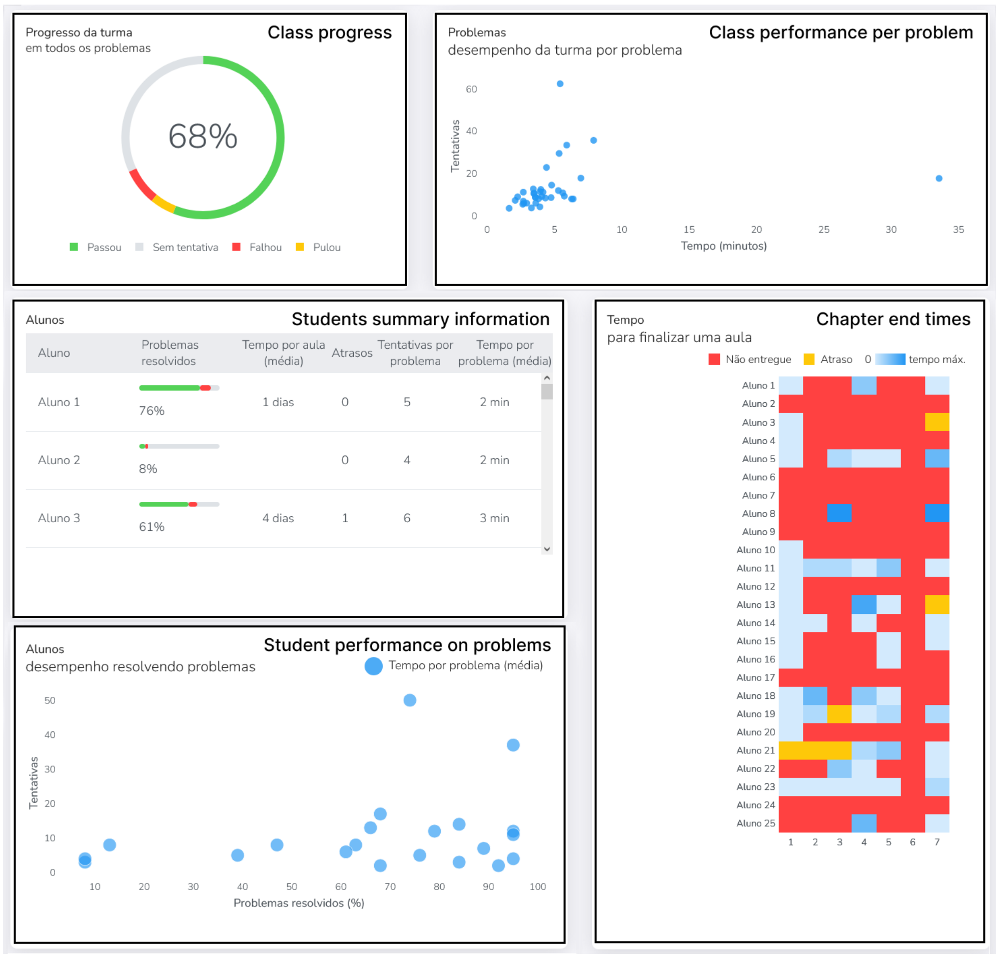
```python
num_bombons(x, y):
    return round(x/y)
```

22 de Julho de 2021 às 14:01 Ver solução
Casos de teste 0% FALHOU

```python
num_bombons(x, y):
    return x/y
```

Figure 3: Interface with the detailed solutions for a specific problem for each student.

To support analysis by class, the dashboard shown in Figure 4 was created. In this dashboard, instructors can verify the content and problems that took the longest for students to solve or required more attempts, which may indicate the need for adjustments in the teaching activity. Consolidated information is also available about individual students. These dashboards are de-



Alunos

Students summary information

Aluno	Problemas resolvidos	Tempo por aula (média)	Atrasos	Tentativas por problema	Tempo por problema (média)
Aluno 1	76%	1 dias	0	5	2 min
Aluno 2	8%		0	4	2 min
Aluno 3	61%	4 dias	1	6	3 min

Figure 4: Dashboard for a class. Contains overall class progress, indications of problems and content that required more time or more attempts, and per-student completion statistics.

signed to make the instructor's work more efficient and to reduce the time needed to access a class overview and each student's evolution.

2.1.3 Dropout prediction

The research already presented in Freire et al., 2024 focused on predicting student dropout risk in introductory programming courses and identifying key variables that could help in their early detection. To achieve this, we integrated statistical inference with machine learning techniques, ensuring both interpretability and high performance in our models. The results achieved an Area Under the Curve (AUC) greater than 0.8 in weekly assessments from the fourth week of classes onward. This performance allows instructors to receive timely warnings about potentially struggling students. Interestingly, our findings indicated that a student's consistency in solving exercises is a more critical factor in assessing dropout risk than the amount of time they spend on developing solutions. This insight can be instrumental for educators aiming to provide targeted support to at-risk students.

2.1.4 Limitations

In the original implementation of Machine Teaching, the code processing, as submitted by the student, occurred within the student's browser. This client-side processing demanded that the students' hardware satisfied minimum requirements. As we experienced during the pandemic period, when classes became remote, many students did not have access to devices which were adequate to successfully use the platform. Also, as the classes size increased, the dashboards become slow to load, specially for students who live in areas not served by fast internet connection. As the classes were remote and big, the dashboards were the main resource for students to have a learning context overview and reflect upon their learning, and instructors to reflect upon the learning of their students.

In order to address issues related to equity in access and adapt the platform to this scenario of inequalities, we redesigned several aspects of the platform architecture. The first and big one: decentralize the code processing, eliminating the need for a robust client hardware. The second adaptation was to redo the dashboards, identifying the points that were slow to calculate and modifying them. Both solutions used the strategy to increase the load on the server (which is in a cloud environment), reducing the workload on the client side.

Another lesson learned during the emergency remote teaching during the pandemic years was that, when out of class, students accessed the learning web platforms the least they could to deliver their assignments. It became necessary to instruct them to access the side functionalities of Machine Teaching so that they could find their ways through the learning dashboards and the free (not guided) study interface. This signed to us that improvements should be done regarding promoting these functionalities to students.

When the university doors opened again to host teaching activities, the need for an online platform to support learning diminished. Instructors who were already using Machine Teaching continued to use it, but we noticed that new instructors tended not to invest in starting to use the platform in their classes. In fact, the course coordinator noticed that new instructors tended to replicate the introductory programming teaching as they were taught when they were novices (despite the course guidelines, reference material, or even the course syllabus). It is worth mentioning that the key to this observation was precisely the fact that the new instructors did not use the Machine Teaching platform and the answers they gave when the course coordinator asked them why they did not use it. The intent of this "why" question was primarily to understand which aspects of the platform were obstacles to its adoption, but the conclusions were that novice instructors needed assistance in better understanding not only the use of the platform and its advantages, but also to better understand the course objectives, study plan and all the "internal" knowledge already acquired about taking these courses at our institution. An action front was then planned to offer this support.

3 Architecture redesign

In this section, we present the new system architecture designed to allow an increased number of simultaneous access and access from low computational power machines is presented.

We propose a refined version of Machine Teaching to bypass the restrictive element of the client-side architecture inherent in the prior version. This novel implementation resolves the issue through the establishment of a remote environment dedicated to code execution, encompassing an auto-scalable server cluster. This adaptation allows students to undertake exercises from any device with internet connectivity. Furthermore, the processing time will no longer be a function of the student's hardware, as a dedicated server cluster will facilitate the processing of exercises submitted through Machine Teaching.

3.1 Cluster Management

The proposed solution is delineated in the diagram presented in Figure 5. The platform adaptation was facilitated by employing the Kubernetes container orchestration system. Each container in the cluster operates as an isolated environment for executing the students' code. The key factors underscoring the decision to utilize Kubernetes encompass:

- **Load Balancing**: Kubernetes exhibits the ability to balance the load and manage network traffic distribution. Consequently, student submissions will be directed towards the available container.
- **CPU and Memory Bounds**: The system permits the definition of CPU and memory (RAM) limitations for each cluster container. This prevents a surge in workload, whether due to malicious code or non-performant code, from adversely affecting the execution environment of other students.
- **Self-healing**: Kubernetes autonomously manages the restart of containers that encounter failure, replacing containers when necessary, and discarding containers that do not respond to health checks.

Beyond alterations in the student's code execution environment, business rules were instituted to ensure scalability and continued security. Evaluation of the implemented system was performed, taking into consideration two principal aspects: scalability and security. The security analysis was predicated on the three pillars of information security, as per Beal, 2005: confidentiality, integrity, and availability. Conversely, the scalability analysis was conducted with the intent to estimate the maximum number of simultaneous users that the system could accommodate without compromising its operational speed.

3.2 Security Evaluation

To safeguard against server compromise, a source code scanning stage was incorporated to identify potentially malicious code or vulnerabilities. This procedure, known as SAST (Static Application Security Testing), leverages several extensively utilized solutions, both open-source and proprietary.

1. **Confidentiality:** Confidentiality is infringed upon when data, expected to be accessed exclusively by a pre-defined set of users, are inadvertently exposed to unauthorized users. To

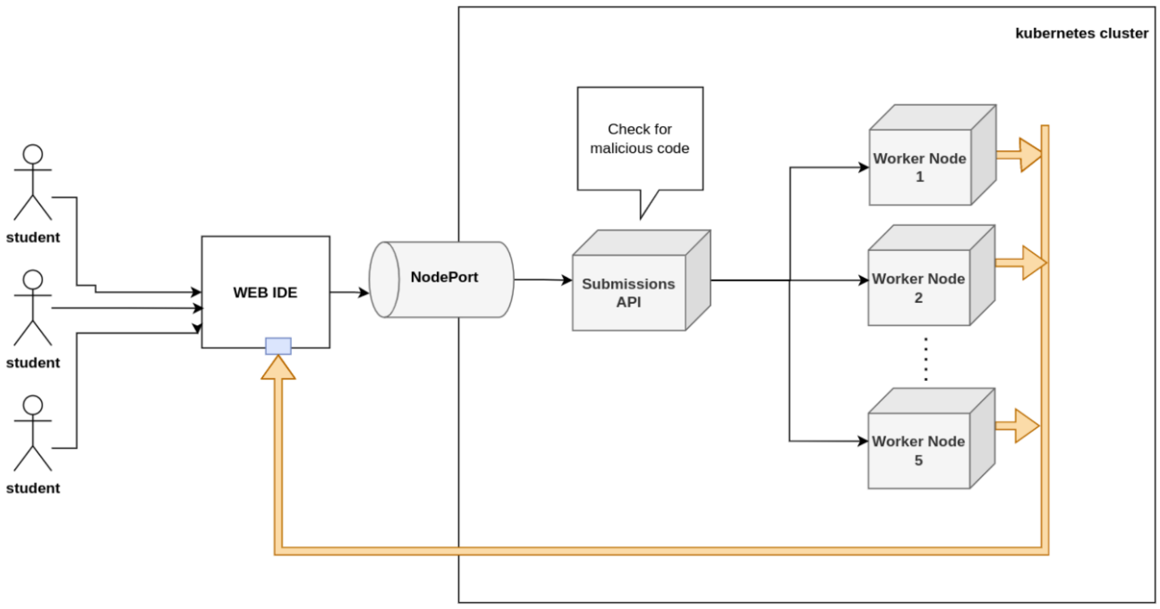


Figure 5: Architecture using a container orchestration system and source code scanning stage..

guarantee data confidentiality, the Machine Teaching system is required to prevent system users from accessing data generated during another user's session. Through a series of tests, it was ascertained that file manipulation and read operations are identified as potential threats and subsequently obstructed. Thus, a malicious user would be unable to access the code submitted by another user or the files generated by them.

2. **Integrity:** The system's integrity would be at risk if a malicious user or agent managed to improperly manipulate any system file. As corroborated by the confidentiality test, breaching the system's integrity proved impossible, as it effectively detects and precludes any form of file manipulation.

3. **Availability:** Lastly, to assess the system's availability, the possibility of compromising it was considered in the event of a code submission rendering one of the servers unavailable, thereby impacting the processing of other users' submissions. Currently, the system lacks any interruption mechanism for submissions that excessively consume CPU and memory resources. Hence, it has been empirically demonstrated that compromising the system's availability is relatively straightforward.

3.3 Scalability Evaluation

The evaluation of the system's scalability and resilience includes studying its performance under a load of multiple users and identifying any constraints that may manifest as the number of simultaneous requests increases. A load test was conducted to evaluate the maximum number of users the system could accommodate without the response time exceeding one second. The metric of RPS (requests per second) was employed, assuming each user would generate approximately 0.3 requests per second. The result can be seen in Table 2.

If there is only one available container (all student submissions are sent to the same container in a queue), the system can handle only five requests per second, resulting in 17 simultaneous students submitting exercises. By increasing the number of containers to five, the system can handle up to 18 requests per second, which results in 60 simultaneous users submitting their code. The number of containers can increase or decrease depending on the expected number of simultaneous users and requests.

It is crucial to highlight that the system did not employ threads during this evaluation. Consequently, the system processed the code synchronously. However, it would be misleading to assume that each container managed only one request at a time. In reality, each container provided the submissions API via a WSGI (Web Server Gateway Interface), which utilized a prefork worker to spawn and manage multiple application instances. For this evaluation, the WSGI was configured to spawn 5 instances of the submissions API for each container.

Table 2: Scalability results achieved varying the number of available containers.

No. containers	Requests per second (RPS)	Maximum No. simultaneous users
1	5	17
3	15	50
5	18	60

Figures 6 to 8 present the detailed evaluation using 1, 3, and 5 containers, respectively. In Figure 6 one can notice that for up to 17 simultaneous users (third plot), the system response time (second plot) remains low, just a few milliseconds. However, when the number of users is increased beyond 17, the system response starts to queue, increasing the response time. Therefore, the maximum number of requests per second achieved before the response time increased was five (first plot). In Figure 7, when three containers were used, the response time (second plot) only increased after achieving 50 users (third plot), corresponding to 15 requests per second (first plot). Finally, in Figure 8, the response time remains stable (second plot) up to 80 users (third plot), when it increases dramatically. Even though the maximum number of requests per second was 18, the average remained at 15, the same as the case with three containers, revealing a noteworthy pattern. As the number of containers increased from one to three, there was a predictable increase in requests per second (RPS). However, the transition from three to five containers did not achieve the expected linear scaling; the server reached only 18 RPS instead of the projected 25 RPS. The underlying cause of this performance deviation remains uncertain, but preliminary investigations suggest that it could be linked to the load-balancing strategy. Within the Kubernetes cluster, the kube-proxy component is responsible for routing requests to an operational pod. A potential limitation here is that kube-proxy lacks insight into a container's internal status. This could result in requests being directed to containers that are already managing their full quota of five requests, as dictated by the number of WSGI forks. Further analysis is required to confirm this hypothesis. However, these empirical results are enough for us to decide how to operate and choose the number of working containers in this application.

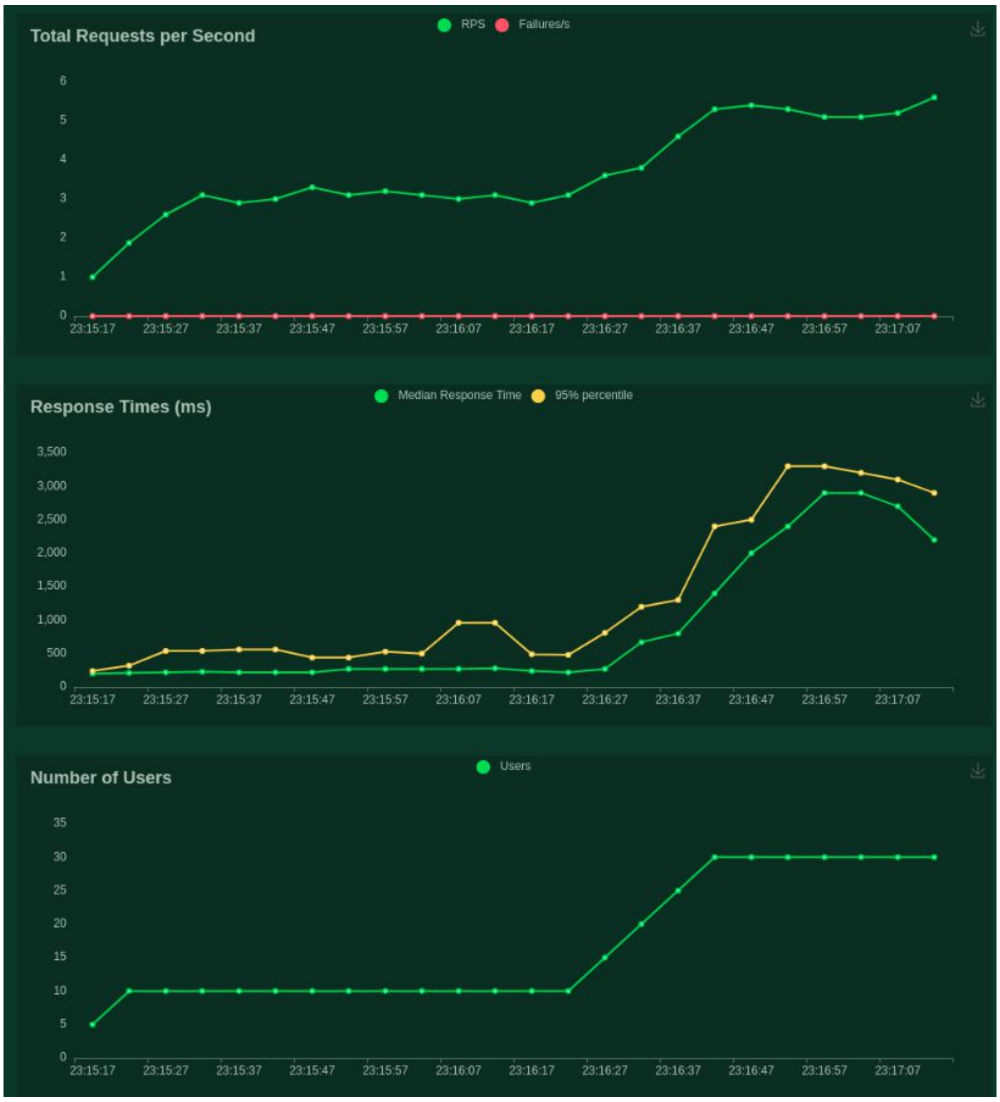


Figure 6: Metrics obtained using a single container.

4 System response time optimization

In addition to evaluating and increasing the number of containers, another strategy to decrease the response time is to optimize the web pages, making them lighter to load. The load of the dashboards, presented in Section 2.1.2, was slow because they require the calculation of many variables to best consolidate students' information. This delay often discouraged instructors and students from understanding their performance, making these features less impactful. To optimize the web page load, we employed the DMAIC problem-solving technique (de Mast and Lokkerbol, 2012), presented in Figure 9, to identify and optimize the found bottlenecks.

In step 1 (Define), we determined that although important to user experience, the dashboards' page performance did not meet the standard for an acceptable user experience. In step 2 (Measure), we conducted a simplified web workload test on the pages. We used the Lighthouse

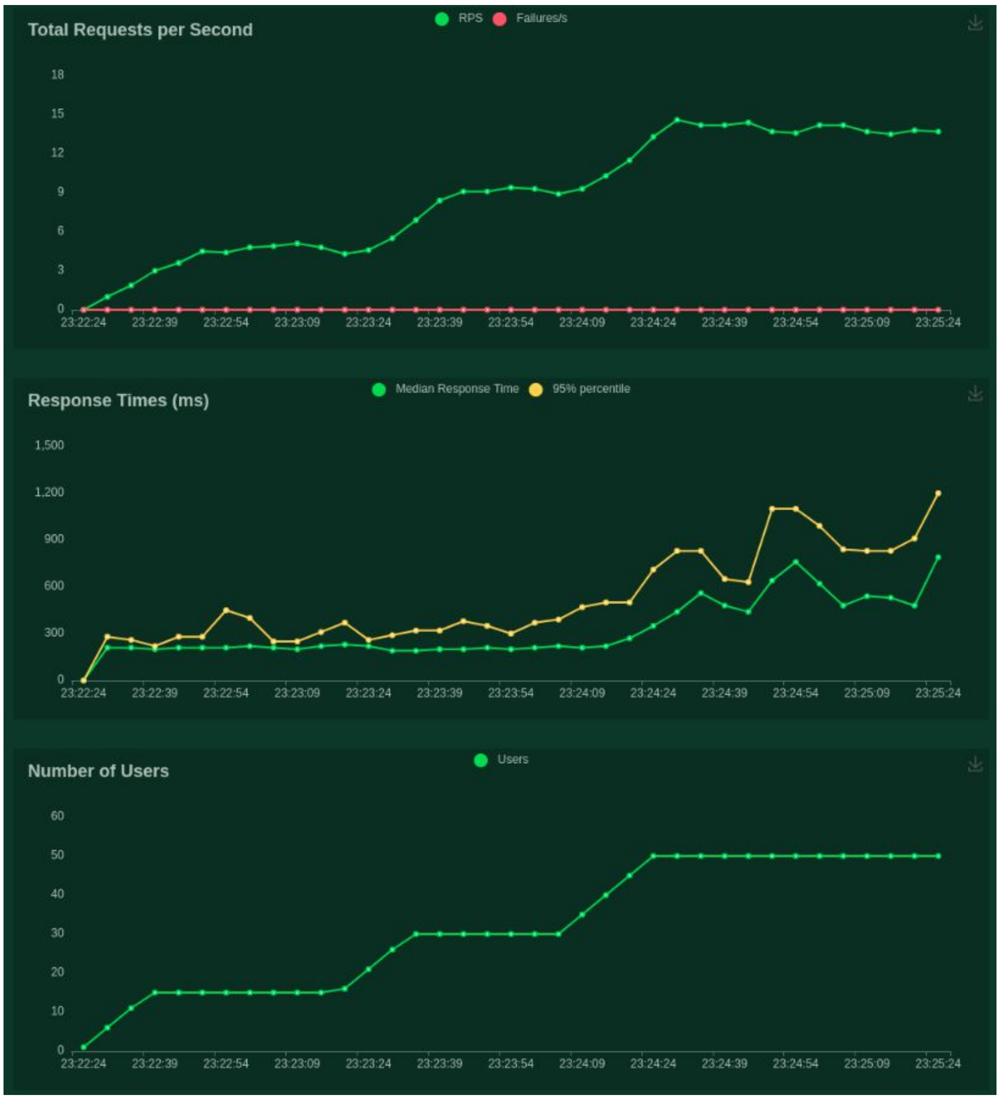


Figure 7: Metrics obtained using three containers.

tool[4] to evaluate the performance aspects. Lighthouse is an open-source tool to audit webpages about performance, accessibility, best practices and search engine optimization (SEO). This tool evaluates the following criteria to create a single performance score (Siahaan and Vianto, 2022):

- First Contentful Paint (FCP): it measures how long it takes for the first image or text to appear.
- Largest Contentful Paint (LCP): it measures how long it takes for the largest image or text to appear.
- Time to Interactive (TTI): it measures how long takes a page to become fully interactive (Heriǎko et al., 2021).
- Total Blocking Time (TBT): it measures the time between FCP and TTI.

[4]https://developer.chrome.com/docs/lighthouse/overview

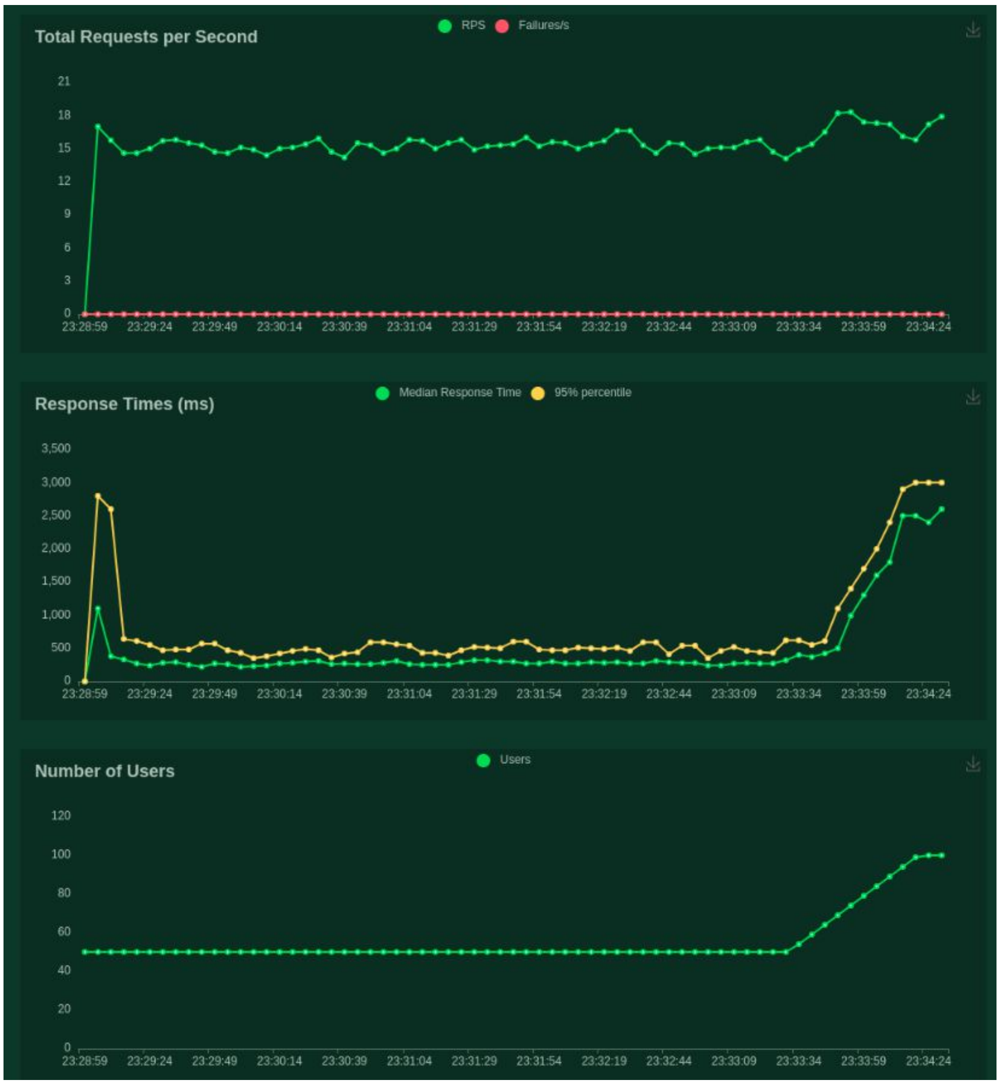


Figure 8: Metrics obtained using five containers.

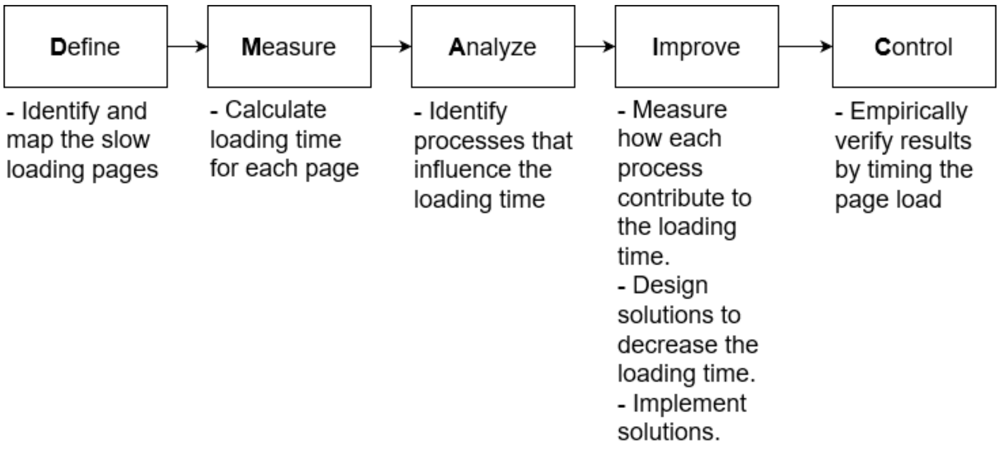

Define
Measure
Analyze
Improve
Control
- Identify and map the slow loading pages
- Calculate loading time for each page
- Identify processes that influence the loading time
- Measure how each process contribute to the loading time.
- Design solutions to decrease the loading time.
- Implement solutions.
- Empirically verify results by timing the page load

Figure 9: Steps to optimize the system response time.

- Speed Index (SI): a score to how quickly the contents of a page are visibly populated.

In step 3 (Analyze), we identified three primary issues:

1. **The development framework:** The Machine Teaching Web System is developed using the Django framework and the Python programming language. Django provides an Object-Relational Mapper (ORM), allowing developers to interact with the database as if it were a Python object. However, this feature builds generic queries, making it slow when querying large amounts of data.
2. **Code inefficiency:** The code was filled with nested and multiple iterations. When the system worked with less data, this was not a relevant factor because the user could not notice the loading time. However, when the system grew larger, the continuous usage of $O(n^3)$ processes slowed the requests.
3. **Chart generation:** The dashboard charts were assembled on the server and then sent to the user interface, which made the initiative slow due to the volume of information exchanged.

As a solution to problems 1 and 2, we decided to transfer the entire workload performed by Django to direct database requests, bypassing Django's default ORM. Using the SQL language, we were able to perform direct queries in PostgreSQL. This allowed us to create cached views and efficient queries, thus avoiding expensive steps. To solve problem 3, we decided to decrease the amount of information exchanged by exporting only numerical data from the server and generating charts in the HTML page using the Chart.js library. As a result, each chart data was mapped into a database query and only this result was sent to the client, decreasing the amount of information exchanged between the server and client and the time needed for calculation.

Table 3 presents the results in four different classes. To analyze the results, we selected the four classes with most registered students. In classes 3 and 4, the previous dashboard did not load, reaching timeout in the Lighthouse analysis every round. The new dashboard version for both classes loaded successfully, providing satisfactory load times and performance score over 50 points (in a scale of 100). For classes 1 and 2, all metrics improved, decreasing the load time between 63% and 80% across the presented metrics. In both cases the score doubled, achieving more than 55 points in both classes.

5 User satisfaction and technology acceptance improvements

Some studies were initiated to amplify the adoption of the platform in more introductory programming classes at UFRJ context and in other diverse scenarios, as future possible applications in further institutions.

One way to achieve this goal is to improve user satisfaction. Many models have been developed to measure user satisfaction. This study considered as the basis a model (Ainin et al., 2012) which incorporates four constructs: system quality, information quality, service quality, and perceived usefulness. The first three constructs were introduced by DeLone and McLean (Delone and

Table 3: Dashboard performance metrics comparison. We present the mean and the standard deviation for each dashboard version for four different classes over three rounds. NS = Number of students in class. FCP = First contentful paint (in seconds). LCP = Largest contentful paint (in seconds). TTI = Time to interactive (in seconds). TBT = Total blocking time (in seconds). SI = Speed index (in seconds). Final score = combined score, varies between 0 and 100.

ID	NS	Version	FCP	LCP	TTI	TBT	SI	Final Score
1	43	Old	17.7 ± 0.1	18.4 ± 0.2	20.3 ± 0.6	1.5 ± 0.5	19.3 ± 1.2	25 ±1
		New	3.7 ± 0.0	4.1 ± 0.0	6.4 ± 0.1	0.1 ± 0.0	4.0 ± 0.5	59 ± 1
2	48	Old	17.8 ± 0.2	19.3 ± 0.1	19.3 ± 0.1	0.8 ± 0.1	19.5 ± 0.1	28 ± 2
		New	4.6 ± 1.5	6.0 ± 1.4	6.5 ± 2.0	0.1 ± 0.0	5.1 ± 1.9	57 ± 3
3	49	Old	Timeout	Timeout	Timeout	Timeout	Timeout	0
		New	5.6 ± 1.6	5.7 ± 1.4	8.1 ± 1.7	0.1 ± 0.0	6.9 ± 0.7	55 ± 3
4	55	Old	Timeout	Timeout	Timeout	Timeout	Timeout	0
		New	3.8 ± 0.0	4.1 ± 0.0	6.4 ± 0.2	0.2 ± 0.0	8.7 ± 0.8	52 ± 2

McLean, 2003) and the last construct was first introduced in the Technology Acceptance Model – TAM (Davis, 1989).

Lack of training and support, and the student's inability to understand the tools of the communication channel can contribute to the non-acceptance of information systems by users (Scarpin et al., 2018). Documentation, manuals and user guides, tutorials, active community, clear communication channels, and training are important elements for effective user support.

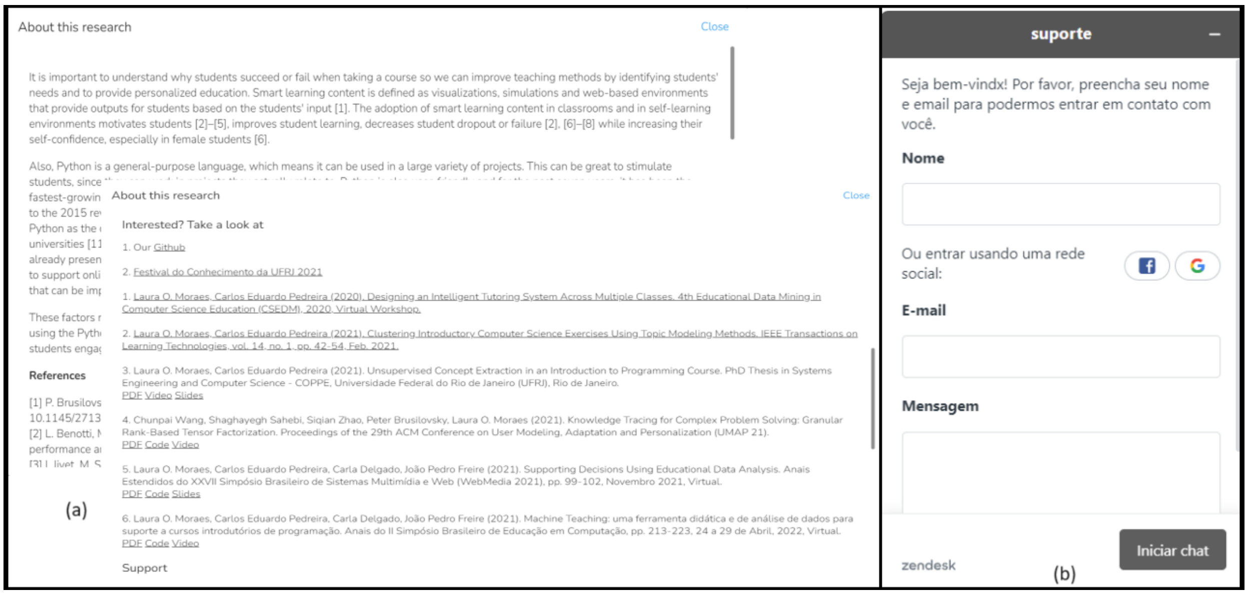
Previous surveys were performed using students and instructors to obtain a preliminary understanding of system usability and decision-making process support (Moraes et al., 2021). It was observed that students' overall experience using the system was positive and that they considered the system easy to use. Also, instructors perceived the system as helpful to gain awareness about student performance and difficulties.

Despite the perceived benefits, the full potential of the platform was not achieved throughout the courses. Although the perception of the platform as helpful for learning, "the student survey showed that they did not use it to optimize their studying strategies" (Moraes et al., 2021), many of the students are having their first contact with programming and can present some difficulties using the environment and reading the data provided by it. Some functionalities as operational support through the student activities' organization and dashboards with information on expected resolution time and the activities' difficulty were not fully used. As stated in Section 4, part of the problem was the slow loading of these pages in big-size classes. However, we would also like to understand if these functionalities were useful for instructors and students.

Furthermore, it was noticed that the adoption of the platform can be expanded for more introductory programming classes at UFRJ. Therefore, from the instructor's view, it was perceived the need for more visibility of the platform and some of its benefits, such as its alignment with the teaching university methodology developed for introductory programming courses (Delgado et al., 2016).

To achieve both groups and amplify platform adoption, a training program was initially proposed considering two distinct profiles: student and instructor. This initiative is complementary to other user support already provided by the platform, such as user support as tutorial research material and contact with the team by messages directly from the platform, as shown in Figure 10.

Video tutorials and other videos specifying the system's functionalities details are also available in an educational channel [5] (Figure 11).


About this research

Close

It is important to understand why students succeed or fail when taking a course so we can improve teaching methods by identifying students' needs and to provide personalized education. Smart learning content is defined as visualizations, simulations and web-based environments that provide outputs for students based on the students' input [1]. The adoption of smart learning content in classrooms and in self-learning environments motivates students [2]–[5], improves student learning, decreases student dropout or failure [2], [6]–[8] while increasing their self-confidence, especially in female students [6].

Also, Python is a general-purpose language, which means it can be used in a large variety of projects. This can be great to stimulate students, since

References

About this research

Close

Interested? Take a look at

1. Our Github

2. Festival do Conhecimento da UFRJ 2021

1. Laura O. Moraes, Carlos Eduardo Pedreira (2020). Designing an Intelligent Tutoring System Across Multiple Classes. 4th Educational Data Mining in Computer Science Education (CSEDM), 2020, Virtual Workshop.

2. Laura O. Moraes, Carlos Eduardo Pedreira (2021). Clustering Introductory Computer Science Exercises Using Topic Modeling Methods. IEEE Transactions on Learning Technologies, vol. 14, no. 1, pp. 42-54, Feb. 2021.

3. Laura O. Moraes, Carlos Eduardo Pedreira (2021). Unsupervised Concept Extraction in an Introduction to Programming Course. PhD Thesis in Systems Engineering and Computer Science - COPPE, Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro.
PDF Video Slides

4. Chunpai Wang, Shaghayegh Sahebi, Siqian Zhao, Peter Brusilovsky, Laura O. Moraes (2021). Knowledge Tracing for Complex Problem Solving: Granular Rank-Based Tensor Factorization. Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization (UMAP 21).
PDF Code Video

5. Laura O. Moraes, Carlos Eduardo Pedreira, Carla Delgado, João Pedro Freire (2021). Supporting Decisions Using Educational Data Analysis. Anais Estendidos do XXVII Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia 2021), pp. 99-102, Novembro 2021, Virtual.
PDF Code Slides

6. Laura O. Moraes, Carlos Eduardo Pedreira, Carla Delgado, João Pedro Freire (2021). Machine Teaching: uma ferramenta didática e de análise de dados para suporte a cursos introdutórios de programação. Anais do II Simpósio Brasileiro de Educação em Computação, pp. 213-223, 24 a 29 de Abril, 2022, Virtual.
PDF Code Video

Support

(a)

suporte

Seja bem-vindx! Por favor, preencha seu nome e email para podermos entrar em contato com você.

Nome

Ou entrar usando uma rede social:

E-mail

Mensagem

Iniciar chat

zendesk

(b)

Figure 10: Machine Teaching Support.

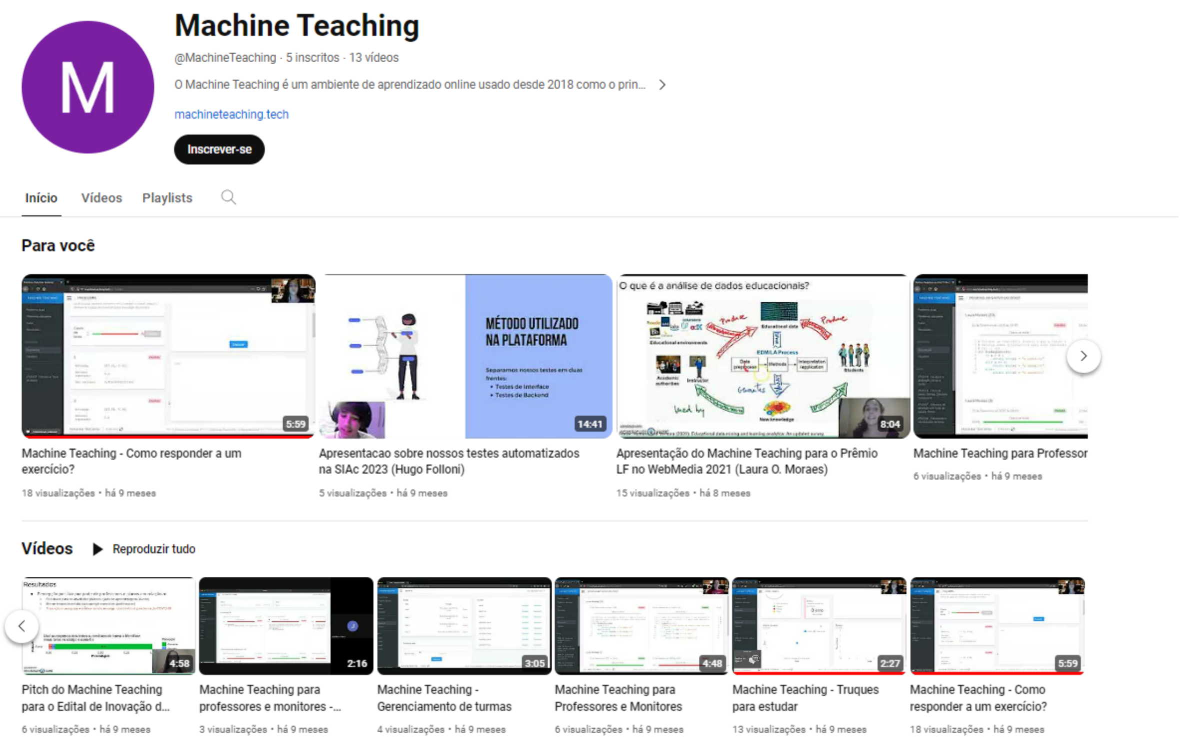
The first training framework developed was to meet the students' perspective. It was performed as a pilot study concentrating on resolving students' main doubts. The first step was to identify these main doubts using a questionnaire administered in three introductory programming classes. The questions covered aspects of the platform's usability and utility, focusing on its support during programming practice, automatic testing, and feedback mechanisms. Based on the collected results and those from previous studies (Moraes et al., 2021), an in-person training session was developed and applied in a class. Training can be viewed as a dialogical space to exchange tool information, answer questions, and clarify doubts. Also, it was used as an opportunity to collect suggestions and observations that are already being applied as platform improvements to better suit students' needs.

After this first round, arrangements were made in the training session to align the content to the students' more frequent doubts and to calibrate the complexity of information to perform the training during the initial use of the platform during the course beginning.

To disseminate the platform among instructors, an online training session was performed with a general view of the main functionalities and some specific observations of the decision-making support as ways of knowing how much time each student spent in each exercise and which ones they had more difficulty in getting the correct answer. The potential of the platform and its educational data mining (EDM) to make educational processes more efficient and effective was also discussed to encourage instructors to adopt the platform in their classes.

Communication channels were also presented to provide more individual support. It is an important mechanism to suggest new questions to increase the number of exercise lists and the

[5]Available at https://www.youtube.com/@MachineTeaching


Machine Teaching
@MachineTeaching · 5 inscritos · 13 vídeos
O Machine Teaching é um ambiente de aprendizado online usado desde 2018 como o prin...
machineteaching.tech
Inscrever-se
Início Vídeos Playlists

Para você

Machine Teaching - Como responder a um exercício? 5:59
18 visualizações • há 9 meses

Apresentacao sobre nossos testes automatizados na SIAc 2023 (Hugo Folloni) 14:41
5 visualizações • há 9 meses

Apresentação do Machine Teaching para o Prêmio LF no WebMedia 2021 (Laura O. Moraes) 8:04
15 visualizações • há 8 meses

Machine Teaching para Professor
6 visualizações • há 9 meses

Vídeos ▶ Reproduzir tudo

Pitch do Machine Teaching para o Edital de Inovação d... 4:58
6 visualizações • há 9 meses

Machine Teaching para professores e monitores -... 2:16
3 visualizações • há 9 meses

Machine Teaching - Gerenciamento de turmas 3:05
4 visualizações • há 9 meses

Machine Teaching para Professores e Monitores 4:48
6 visualizações • há 9 meses

Machine Teaching - Truques para estudar 2:27
13 visualizações • há 9 meses

Machine Teaching - Como responder a um exercício? 5:59
18 visualizações • há 9 meses

Figure 11: Machine Teaching Educational Channel.

diversity of platform content, which requires the Machine Teaching team validation and quality assurance.

The next steps of this study include the development of user guides that consider the two profiles, more frequent training sessions, video tutorials, and the execution of more systematic empirical studies with survey questionnaires to analyze specific aspects of perceived usefulness before and after training sessions.

These instruments' development intends to improve information dissemination of platform use and user support as effective ways to amplify its adoption. These can be considered important elements for plans to expand the Machine Teaching use by other institutions besides UFRJ and to promote computational thinking in schools. In this last case, some system adaptations may also be necessary.

6 Conclusion and future work

In pursuit of fulfilling the UN's sustainable goal of "quality education", policies to expand access to public universities were implemented in Brazil. This changed the scenario in classrooms and brought many challenges, including classes with a large number of students, students in an unfavorable economic situation, and students whose previous relationship with formal education was unsatisfactory.

This article presented technological choices underlying the improvement of the "Machine Teaching" platform for problems that did not arise from the technological field but from the pedagogical, socioeconomic, or political fields. The perception of these problems and the discussion of how they could be mitigated have passed through several institutional bodies. It is important to emphasize that educational technology alone will not provide a solution to this class of problems; however, it can be a powerful factor in the construction of a mitigation strategy.

We have reached a point in time where resources such as cloud computing, AI, and data analytics become economically advantageous, considering the trade-off cost-benefits –on the one hand, production and application costs; on the other, financial gains or cost savings generated by their use. The current challenge is to balance the economic value of using these resources against other values involved, such as quality of education and equal access conditions. Studies that bring this type of discussion are still rare (Bates et al., 2020).

The effort to make the university a place of welcome, representation, and social transformation requires the recognition of inequalities in access to what the university can offer that will make the most difference in students' lives: a quality higher education that allows them to be protagonists of dignified and fulfilling life stories and contribute to a better society. From the perspective of educators and course managers, this investment translates into the mapping of issues relating to equal access to courses and specific conditions of academic failure, as well as the design of strategies and instruments to mitigate them.

This work presented the actions taken to enable a web learning environment system to support instructors in programming classes under the conditions described. We considered the instructor's difficulties in managing such a large class and presented functionalities to mitigate these problems, such as automated student feedback in real-time, dashboards with consolidated statistics per class and student, and alerts indicating disengaged students who are likely to drop out.

It is important to mention that an experienced teacher is a professional with super-specialized skills whose performance as an educator cannot be surpassed or even compared to what technology can achieve. The problems of class size and lack of sufficient qualified personnel are the prices to be paid for the increase in the supply of places in higher education, and this is the scenario currently faced in several places in Brazil and the world.

The presented system had an initial architecture that did not consider students' low computational power machines by automatically correcting the source code on the client. Therefore, we also presented the system's new architecture, which establishes a safe remote environment for code execution, releasing the workload from the client machine. It is based on an auto-scalable server cluster and incorporates a source code scanning stage to increase server security. We ran a security evaluation on this architecture and it passed two out of three criteria. The system's confidentiality and integrity were not compromised, but its availability was. Future work will implement an interruption mechanism if the executed code takes longer than expected. The new architecture scalability was also tested. As the number of containers increases, the maximum number of simultaneous users also increases. In our tests, we achieved 60 simultaneous users with a response time of less than one second by deploying just five workers.

Another approach to optimize the system's response time was presented. By employing the DMAIC problem-solving technique, we identified and addressed critical bottlenecks, including issues with the development framework, code inefficiencies, and chart generation. Implementing direct database queries instead of relying on Django's ORM, along with minimizing data exchange and adopting client-side chart generation, yielded good results. Performance enhancements were quantified through both system clock time measurements and client-side timing, thereby reducing the loading times for both dashboards by at least 50% and enhancing the user experience and usability of the application. These optimizations not only addressed immediate concerns but also laid a foundation for ongoing improvements in system efficiency and responsiveness.

In future work, we plan to continue studies to amplify the adoption of the platform in more introductory programming classes at UFRJ and other institutions. Besides the work done with instructors, our current perception is that it is important to build a direct dialog between the Machine Teaching team and the students. Currently, the instructor is the one mediating students' learning and Machine Teaching's usage. However, she/he is a figure of authority and has, among others, the role of a grader. It is interesting that students can see the platform as a tool that is available for their learning and not as a mere instrument at the service of instructors to supervise them. This would be another step in the pedagogical goal of making students the protagonists of their education life.

Machine Teaching has already proven to be a very effective system in supporting programming teaching at UFRJ. During the Covid-19 pandemic, it was an essential resource to enable practical classes to be conducted in the form of remote emergency teaching, during the period of social isolation. Positive feedback from instructors and students regarding its functionalities to support programming teaching-learning, combined with the possibility of maintaining a long-term follow-up of the evolution of programming teaching at UFRJ, justify investments in keeping the system up-to-date, robust, and accessible to all students. Given the results obtained so far, we believe that Machine Teaching is an ally in the challenges of mass and inclusive teaching.

Online Resources

1. Machine Teaching: www.machineteaching.tech
2. Github: https://github.com/MachineTeachingEdu/
3. Youtube: https://www.youtube.com/@MachineTeaching

Acknowledgements

This work was carried out with the support of CNPq (PIBIC UFRJ), PPInst-2023 UNIRIO and the Instituto Reditus.

Award-winning Paper Extended

This publication is an extended version of award-winning article at the Workshop de Informática na Escola (WIE 2023), entitled "Dealing with a large number of students and inequality when teaching programming in higher education", DOI: https://doi.org/10.5753/wie.2023.235057.

References

Ainin, S., Bahri, S., & Ahmad, A. (2012). Evaluating portal performance: A study of the National Higher Education Fund Corporation (PTPTN) portal [GS Search]. *Telematics and Informatics*, *29*(3), 314–323. https://doi.org/10.1016/j.tele.2011.11.004

Ake-Little, E., von der Embse, N., & Dawson, D. (2020). Does class size matter in the university setting? [GS Search]. *Educational Researcher*, *49*(8), 595–605. https://doi.org/10.3102/0013189X20933836

Araujo, L. G., Bittencourt, R., & Chavez, C. (2021). Python enhanced error feedback: Uma IDE online de apoio ao processo de ensino-aprendizagem em programação [GS Search]. *Anais do Simpósio Brasileiro de Educação em Computação*, 326–333. https://doi.org/10.5753/educomp.2021.14500

Baker, P. J., Bakshis, R., & Tolone, W. (1974). Diversifying learning opportunities: A response to the problems of mass education [GS Search]. *Research in Higher Education*, *2*, 251–263. https://doi.org/10.1007/BF00991169

Bates, T., Cobo, C., Mariño, O., & Wheeler, S. (2020). Can artificial intelligence transform higher education? [GS Search]. *International Journal of Educational Technology in Higher Education*, *17*(42). https://doi.org/10.1186/s41239-020-00218-x

Beal, A. (2005). *Segurança da informação: Princípios e melhores práticas para a proteção dos ativos de informação nas organizações* (1st ed.) [GS Search]. Atlas.

Bez, J. L., Ferreira, C. E., & Tonin, N. (2013). URI Online Judge Academic: A Tool for Professors [GS Search]. *Proceedings of the 2013 International Conference on Advanced ICT and Education*, 744–747. https://doi.org/10.2991/icaicte.2013.153

Borenstein, J., & Howard, A. (2021). Emerging challenges in AI and the need for AI ethics education [GS Search]. *AI and Ethics*, *1*, 61–65. https://doi.org/10.1007/s43681-020-00002-7

Castioni, R., Melo, A. A. S., Nascimento, P. M., & Ramos, D. L. (2021). Universidades federais na pandemia da covid-19: Acesso discente à internet e ensino remoto emergencial [GS Search]. *Ensaio: Avaliação e Políticas Públicas em Educação*, *29*(111), 399–419. https://doi.org/10.1590/S0104-40362021002903108

Cunha, J. K. A., Oliveira, B. R., & Fernandes, N. R. (2023). Assistência estudantil na educação superior: A trajetória do programa nacional de assistência estudantil na Universidade Federal de Ouro Preto [GS Search]. *Revista Tempos e Espaços em Educação*, *16*(35), e18808. https://doi.org/10.20952/revtee.v16i35.18808

Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology [GS Search]. *MIS Quarterly*, *13*(3), 319–340. https://doi.org/10.2307/249008 de Mast, J., & Lokkerbol, J. (2012). An analysis of the Six Sigma DMAIC method from the perspective of problem solving [GS Search]. *International Journal of Production Economics*, *139*(2), 604–614. https://doi.org/10.1016/j.ijpe.2012.05.035

Delgado, C., Silva, J., Mascarenhas, F., & Duboc, A. (2016). The teaching of functions as the first step to learn imperative programming [GS Search]. *Anais do XXIV Workshop sobre Educação em Computação*, 2393–2402. https://doi.org/10.5753/wei.2016.9683

Delone, W. H., & McLean, E. R. (2003). The DeLone and McLean Model of Information Systems Success: A Ten-Year Update [GS Search]. *Journal of Management Information Systems*, *19*(4), 9–30. https://doi.org/10.1080/07421222.2003.11045748

Edwards, S. H., & Perez-Quinones, M. A. (2008). Web-CAT: Automatically grading programming assignments [GS Search]. *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education*, 328. https://doi.org/10.1145/1384271.1384371

Edwards, S. H., Tilden, D. S., & Allevato, A. (2014). Pythy: Improving the introductory python programming experience [GS Search]. *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, 641–646. https://doi.org/10.1145/2538862.2538977

Freire, J. P., Landim, F. M. P. F., Moraes, L. O., Delgado, C. A. D. M., & Pedreira, C. E. (2024). Modelo para previsão precoce de abandono de uma disciplina de introdução à programação [GS Search]. *Anais do XXXII Workshop sobre Educação em Computação*, 635–645. https://doi.org/10.5753/wei.2024.2526

Galvão, L., Fernandes, D., & Gadelha, B. (2016). Juiz online como ferramenta de apoio a uma metodologia de ensino híbrido em programação [GS Search]. *Anais do XXVII Simpósio Brasileiro de Informática na Educação (SBIE 2016)*, 140–149.

Gomes, C. A. (2005). A escola de qualidade para todos: Abrindo as camadas da cebola [GS Search]. *Ensaio: Avaliação e Políticas Públicas em Educação*, *13*(48), 281–306. https://doi.org/10.1590/S0104-40362005000300002

Heriăko, T., Čučko, Š., Šumak, B., & Brdnik, S. (2021). Web performance tuning of wordpress-based websites through automatic image optimization [GS Search][Link]. *Proceedings of the Central European Conference on Information and Intelligent Systems*, 343–350.

Hipólito, J., Shirai, L. T., Diele-Viegas, L. M., Halinski, R., Pires, C. S. S., & Fontes, E. M. G. (2022). Brazilian budget cuts further threaten gender equality in research [GS Search]. *Nature Ecology & Evolution*, *6*, 234. https://doi.org/10.1038/s41559-021-01640-8

Hovemeyer, D., & Spacco, J. (2013). Cloudcoder: A web-based programming exercise system [GS Search]. *Journal of Computing Sciences in Colleges*, *28*(3), 30. https://doi.org/10.1145/2493394.2493401

Ihantola, P., Vihavainen, A., Ahadi, A., Butler, M., Börstler, J., Edwards, S. H., Isohanni, E., Korhonen, A., Petersen, A., Rivers, K., Rubio, M. Á., Sheard, J., Skupas, B., Spacco, J., Szabo, C., & Toll, D. (2015). Educational Data Mining and Learning Analytics in Programming: Literature Review and Case Studies [GS Search]. *Proceedings of the 2015 ITiCSE on Working Group Reports*, 41–63. https://doi.org/10.1145/2858796.2858798

Lima, M. E. O., Neves, P. S. C., & e Silva, P. B. (2014). A implantação de cotas na universidade: paternalismo e ameaça à posição dos grupos dominantes [GS Search][Link]. *Revista Brasileira de Educação*, *19*(56), 141–163.

Lima Neto, N. (2012). Desafios da educação superior brasileira para a próxima década [GS Search][Link]. In UNESCO Office, Brasilia; National Education Council, Brazil. Ministry of Education, Brasil (Ed.), *Desafios e perspectivas da educação superior brasileira para a próxima década, 2011-2020* (pp. 37–44). UNESCO Office Brasilia/CNE/MEC.

Luxton-Reilly, A., Simon, Albluwi, I., Becker, B. A., Giannakos, M., Kumar, A. N., Ott, L., Paterson, J., Scott, M. J., Sheard, J., & Szabo, C. (2018). Introductory programming: A systematic literature review [GS Search]. *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, 55–106. https://doi.org/10.1145/3293881.3295779

Mitchell, D., Carson, C., & Badarak, G. (1990, May). *How Changing Class Size Affects Classrooms and Students* (tech. rep.) ([GS Search][Link]). California Educational Research Cooperative, Riverside.

Moraes, L. O., Delgado, C. A. D. M., Freire, J. P., & Pedreira, C. E. (2022). Machine teaching: Uma ferramenta didática e de análise de dados para suporte a cursos introdutórios de programação [GS Search]. *Anais do II Simpósio Brasileiro de Educação em Computação*, 213–223. https://doi.org/10.5753/educomp.2022.19216

Moraes, L. O., & Pedreira, C. E. (2020). Designing an intelligent tutoring system across multiple classes [GS Search][Link]. *4th Educational Data Mining in Computer Science Education Workshop*.

Moraes, L. O., Pedreira, C. E., Delgado, C., & Freire, J. P. (2021). Supporting decisions using educational data analysis [GS Search]. *Anais Estendidos do XXVII Simpósio Brasileiro de Sistemas Multimídia e Web*, 99–102. https://doi.org/10.5753/webmedia_estendido.2021.17622

Özer, M. (2024). Potential Benefits and Risks of Artificial Intelligence in Education [GS Search]. *Bartın University Journal of Faculty of Education*, *13*(2), 232–244. https://doi.org/10.14686/buefad.1416087

Panamalai Murali, K. (2016, June). *CodeWorkout: Design and implementation of an online drill-and-practice system for introductory programming* [Thesis]. Virginia Tech [GS Search][Link].

Papancea, A., Spacco, J., & Hovemeyer, D. (2013). An open platform for managing short programming exercises [GS Search]. *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, 47–52. https://doi.org/10.1145/2493394.2493401

Paulilo, A. L. (2017). A compreensão histórica do fracasso escolar no brasil [GS Search]. *Cadernos de Pesquisa*, *47*(166), 1252–1267. https://doi.org/10.1590/198053144445

Rodrigues, A., & Chechia, V. A. (2017). O fracasso escolar e suas implicações no processo de ensino e de aprendizagem [GS Search][Link]. *Psicologia – Saberes & Práticas*, *1*(1), 29–36.

Sasse, A. M., Moraes, L. O., Delgado, C. A. D. M., & Pedreira, C. E. (2024). Analisando a efetividade da formação de clusters na avaliação de exercícios de programação [GS Search]. *Anais do IV Simpósio Brasileiro de Educação em Computação*, 325–335. https://doi.org/10.5753/educomp.2024.237380

Scarpin, J. E., Mondini, V. E. D., & Scarpin, M. R. S. (2018). Technology acceptance factors and student retention in online courses [GS Search][Link]. *E-Journal of Business Education and Scholarship of Teaching*, *12*(3), 44–68.

Siahaan, M., & Vianto, V. O. (2022). Comparative Analysis Study of Front-End JavaScript Frameworks Performance Using Lighthouse Tool [GS Search][Link]. *Jurnal Mantik*, *6*(3), 2462–2468.

Sorva, J., & Sirkiä, T. (2010). UUhistle: A software tool for visual program simulation [GS Search]. *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*, 49–54. https://doi.org/10.1145/1930464.1930471

Tonin, N., Zanin, F., & Bez, J. L. (2012). Enhancing traditional Algorithms classes using URI Online Judge [GS Search]. *2012 International Conference on e-Learning and e-Technologies in Education (ICEEE)*, 110–113. https://doi.org/10.1109/ICeLeTE.2012.6333402

Trow, M. (2007). Reflections on the Transition from Elite to Mass to Universal Access: Forms and Phases of Higher Education in Modern Societies since WWII [GS Search]. In J. J. F. Forest & P. G. Altbach (Eds.), *International handbook of higher education* (pp. 243–280). Springer Netherlands. https://doi.org/10.1007/978-1-4020-4012-2_13

Xará, G., Moraes, L. O., Delgado, C. A. D. M., Freire, J. P., & Farias, C. M. (2023). Dealing with a large number of students and inequality when teaching programming in higher education [GS Search]. *Anais do XXIX Workshop de Informática na Escola*, 1230–1242. https://doi.org/10.5753/wie.2023.235057

Zawacki-Richter, O., Marín, V. I., Bond, M., & Gouverneur, F. (2019). Systematic review of research on artificial intelligence applications in higher education – where are the educators? [GS Search]. *International Journal of Educational Technology in Higher Education*, *16*(39). https://doi.org/10.1186/s41239-019-0171-0

Zingaro, D., Cherenkova, Y., Karpova, O., & Petersen, A. (2013). Facilitating code-writing in PI classes [GS Search]. *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, 585–590. https://doi.org/10.1145/2445196.2445369