Saint Petersburg State University Graduate School of Management

Master of Business Analytics and Big Data

**COMPARISON AND OPTIMIZATION OF IT CODING PLATFORMS IN EDUCATIONAL CONTEXT**

Master Thesis

Master’s Thesis by the 2nd year students

Master in Business Analytics and Big Data

M. Bano

J. Lambert

Research Advisor:

T. S. Stanko

Senior Lecturer, Department of Operations Management

Saint Petersburg 2023

# DECLARATION OF INDEPENDENT WORK

We, Madiya Bano and Jerome Lambert, second year master students, program "Business Analytics", state that our master thesis on the topic "Comparison and Optimization of IT Coding Platforms in Educational Context", which is presented to the Master Office to be submitted to the Official Defence Committee for the public defence, does not contain any element of plagiarism.

All direct borrowings from printed and electronic sources, as well as from master theses, PhD and doctorate theses, which were defended earlier, have appropriate references.

We are aware that according to paragraph 9.7.1. of Guidelines for instruction in major curriculum programs of higher and secondary professional education at St. Petersburg University "A master thesis must be completed by each of the degree candidates individually under the supervision of his or her advisor", and according to paragraph 51 of charter of the Federal State Institution of Higher Education Saint-Petersburg State University "a student can be expelled from St. Petersburg University for submitting of the course or graduation qualification work developed by other person(s)"

Date: 01/06/2023

Authors’ signatures:

A close-up of a name

Description automatically generated with low confidence A close-up of a text

Description automatically generated with low confidence

# ABSTRACT

A few words about the authors. Madiya Bano is a second-year student in the MiBA program at SPbU GSOM with a background in economics and programming. Her research areas include prediction models using Machine Learning (ML) algorithms. She is also interested in applications of (Internet of Things) IoTs in Analytics and Natural Language Processing (NLP). Jerome Lambert is also a second-year student in the MiBA program at SPbU GSOM with a background in management and strategy.

A few words about the scientific advisor: Dr Tatyana Stanko is an expert in engineering education and university organizational changes. She has worked in academic leadership since 2012. Her experience includes launching two new green-field universities in Russia. She is an active researcher in engineering education and university management, having published over 30 research papers. Her main research areas are global trends in engineering education, talent education, gender diversity and faculty development.

Since the covid-19 pandemic, cloud computing has become very widespread in teaching programming in universities, as is now the case in GSOM. The university switched from computer classes to Jupyter Hub in the last few years and aims to understand how to improve its cloud computing use. This paper tries to understand how to optimise this cloud computing use in 3 ways. First, in understanding if Jupyter Hub is the best alternative, using the specific case of GSOM, with a method that may be extended to other similar institutions. Even though it appeared that Jupyter Hub is one of the best choices, Google Colab appeared to complement it and should be implemented along with it. Second, fewer bugs would improve UX, and bugs are legion. Most of the bugs occur due to too many user requests at once. They could be tempered by adding more power to the cloud when demand is predicted to be high. It has been found that user activity is highly significantly correlated – and very likely enhanced by – the starting time of lectures. Therefore, when many lectures start simultaneously – standard in GSOM – it is recommended to add cloud power. Lastly, a survey was conducted to understand students’ opinions. They broadly appreciate the 24/7 access to the platform as well as its usability. The majority does not seem to want changes on Jupyter Hub the way it is, but a significant part would want more languages and a better interface.

*Keywords: IT platform, platform optimization, cloud computing, programming platform*

TABLE OF CONTENTS

[DECLARATION OF INDEPENDENT WORK 2](#_Toc136543701)

[ABSTRACT 3](#_Toc136543702)

[INTRODUCTION 5](#_Toc136543703)

[Problem Statement 5](#_Toc136543704)

[Theoretical and practical significance 7](#_Toc136543705)

[Work division 8](#_Toc136543706)

[Article’s structure 8](#_Toc136543707)

[CHAPTER 1. LITERATURE REVIEW ON PLATFORM COMPARISON 9](#_Toc136543708)

[1.1 Goals and Structure of literature review 9](#_Toc136543709)

[1.2 Search methodology 10](#_Toc136543710)

[1.3 Platform comparison 11](#_Toc136543711)

[1.4 Platform optimisation 40](#_Toc136543712)

[Summary and Limitations of Chapter 1 47](#_Toc136543713)

[CHAPTER 2. METHODOLOGY OF RESEARCH 48](#_Toc136543714)

[CHAPTER 3. DATA ANALYSIS 49](#_Toc136543715)

[3.1 Formal Platform Comparison 49](#_Toc136543716)

[3.2 Logs analysis 56](#_Toc136543717)

[3.3 Survey analysis 63](#_Toc136543718)

[CHAPTER 4: CONCLUSIONS 71](#_Toc136543719)

[4.1 Summary 71](#_Toc136543720)

[4.2 Future research possibilities 71](#_Toc136543721)

[REFERENCES 72](#_Toc136543722)

[ANNEXES 77](#_Toc136543723)

[Annexe 1 77](#_Toc136543724)

[Survey composition 77](#_Toc136543725)

[Annexe 2 79](#_Toc136543726)

[Survey results 79](#_Toc136543727)

[Annexe 3 83](#_Toc136543728)

[Log codes 83](#_Toc136543729)

[Annexe 4 84](#_Toc136543730)

[Code link 84](#_Toc136543731)

# INTRODUCTION

## Problem Statement

Cloud-based programming platforms have become increasingly popular in recent years, offering a range of benefits for universities seeking to optimise their programming workflows. These platforms provide access to shared resources and infrastructure, reducing the need for local installation and management of software and hardware while enabling collaboration between students and researchers across multiple locations. Additionally, cloud-based platforms can offer scalability and flexibility, allowing universities to adjust their computing resources to match their needs quickly.

Jupyter Hub is an open-source multi-user server that allows users to access Jupyter Notebook environments through a single server. It enables administrators to provide users with a shared computing environment that supports multiple languages and kernels. Jupyter Notebook, on the other hand, is a web-based interactive development environment that allows users to create and share documents that contain live code, visualisations, and narrative text.

Jupyter Hub is built to support large-scale deployments of Jupyter Notebooks. It can oversee multiple users simultaneously and provides an authentication mechanism allowing administrators to control server access. The authentication mechanism can be integrated with various systems, including OAuth, Google, and GitHub, to allow users to log in using their existing credentials.

The platform is highly configurable and can be customised to meet the needs of different organisations. It allows administrators to define resource limits for individual users or groups, including CPU and memory usage. These characteristics are beneficial when computing resources are limited or expensive. Administrators can also set up different environments for separate groups of users, allowing them to work with specific libraries or packages.

Jupyter Notebook, as mentioned earlier, is a web-based interactive development environment that allows users to create and share documents that contain live code, visualisations, and narrative text. It supports multiple programming languages, including Python, R, and Julia, and has various libraries and packages commonly used for data analysis and machine learning.

In Jupyter Notebook, users can create a notebook containing a cell series. Each cell can contain code, text, or visualisations. Users can execute the code interactively and see the output of the code in real time. This possibility makes it easy to explore data and experiment with different algorithms.

Jupyter Notebook also supports the creation of interactive widgets that allow users to manipulate data and parameters in real time. This ability is advantageous when working with complex data sets or models that require parameter tuning.

Jupyter Hub and Jupyter Notebook are powerful tools that allow users to create and share interactive documents that combine code, data, and narrative text. They are widely used in scientific and data analysis communities and are popular tools for exploring and sharing data. The combination of Jupyter Hub and Jupyter Notebook provides an efficient and scalable solution for organisations that require a shared computing environment.

GSOM is used to teach programming and coding projects in computer classes, or on students’ computers with their configuration and environment, instead of using a shared, standardised cloud. This situation led to several issues. First, the classrooms were unavailable 24/7, had to be booked, and were not adapted to students’ setup (one of the authors of this paper uses AZERTY and loses efficiency when using QWERTY, which is the only keyboard available on campus). Second, when using own computer and environment, there are issues with per reproducibility of the code. A code that runs well in a Jupyter Notebook might not work on Anaconda because the user forgot to install a module not in Anaconda by default (for instance, the basic NumPy module for Python). This issue undoubtedly leads to headaches but might get worse in some situations, for instance, coding on one platform, submitting, and having the code run by the teacher who uses another setup, who will see a bug and lower the student’s grade for making a code that is not fully functional.

Nowadays, GSOM uses Jupyter Hub for its students to learn to program and practice using Python and various sub-languages.

This platform is versatile, user-friendly, and one of the most optimal choices that could be made for GSOM's use. Nevertheless, it has a few flaws.

First, it uses many resources, creating bugs like the platform running out of memory. On top of this, it often has issues starting when many students are on it simultaneously. Indeed, when trying to load Jupyter Hub while many users work on it, it might take up to 20 minutes, according to several students, if the platform loads. This issue might be problematic for simultaneous use of this platform in class, as the teacher and an entire classroom (at least twenty students) are expected to connect in a fleeting time frame, leading to delays at the start of the class or some students being left aside at the start of it. This imperfection also creates issues when a user wishes to use the platform as soon as possible to tackle a task urgently. For instance, wanting to change a typo in the graph legend generated thanks to Jupyter before presenting slides that include this graph.

On top of this, it is poorly understood which tasks GSOM’s users fulfil on the platform. Understanding this last point would allow better customisation for users’ needs, resulting in more general efficiency. Without further investigation, it would be a mistake to assume that the current usage is what users would like to make of that platform. Indeed, while this might not be the case, it is desirable to check whether users would like to undertake some tasks on Jupyter that are not currently possible because of a lack of computing power, bugs, or simply impossible to run on Jupyter Hub while this might be possible on alternative platforms.

This research aims to optimise costs incurred by GSOM for its cloud platforms and UX and scale up user usage of said platforms, as per a request from GSOM’s IT department to have extended information about this, which is the reason behind the writing of this paper.

The first question that arises from this is whether Jupyter Hub is, to start with, the most optimal choice for GSOM users’ use of it or if other platforms might constitute a better choice in terms of costs, errors per hundred use, user-friendliness, et cetera.

The second one is how to optimise the use of the platform, which is the most optimal to limit GSOM costs, increase user involvement, decrease users’ pain, and, if possible, increase users’ benefits in using the platform.

This paper's third and last research question is about better understanding the students’ opinions on this issue.

The methodology to answer the first question will summarise the information about Jupyter Hub and alternative platforms in a table to better visualise the results and draw conclusions.

The second part will be analysed using advanced coding to treat data from users’ logs and university timetables and test the significance of found results.

The last research question will be approached by a survey sent to students to check their opinion on issues the company would like to hear of and some additional ones to understand what is to be improved on the platform besides decreasing its bugs.

## Theoretical and practical significance

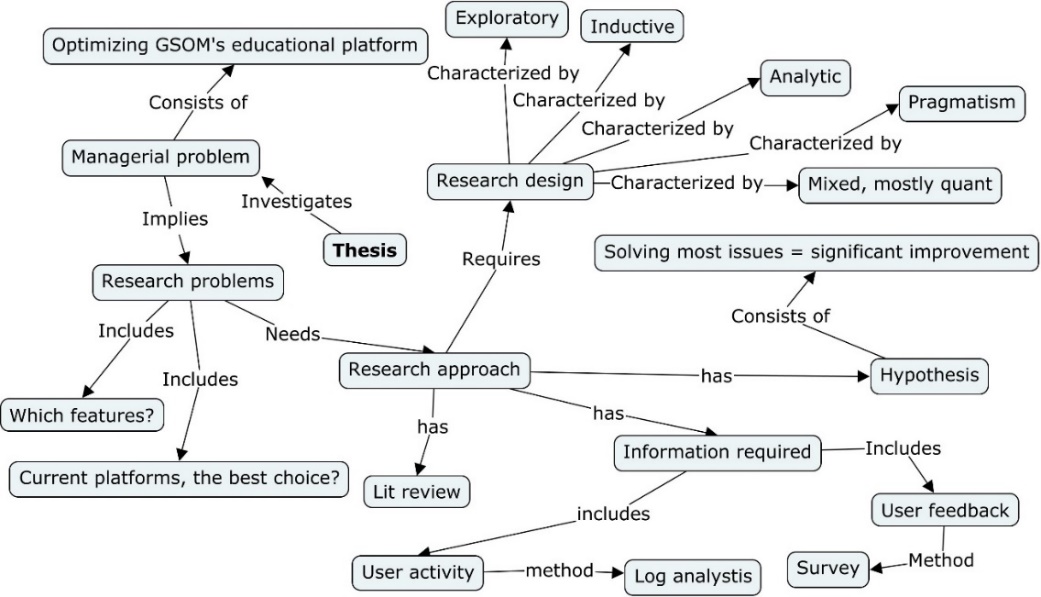
This research aims to understand how to make IT platforms more optimal in the academic context and compare different alternative platforms to select the most optimal ones for academic purposes.

The scope of the study is to suggest concrete solutions as per the improvement for GSOM on its Jupyter Hub’s use but also to draw more general conclusions for other platforms and other members of the academic spheres, as well as finding a comparison between platforms that might serve as a framework for similar cases, which is expectedly for other academic institutions.

## Work division

The tasks have been divided nearly perfectly evenly between the two authors.

## Article’s structure

**

*Figure 1: Research structure*

Figure 1 denotes the article’s structure and the information required to follow this structure.

*A picture containing text, screenshot, diagram, font

Description automatically generated*

*Figure 2: Research plan*

Figure 2 describes the concrete research plan that is based on the detailed research structure represented in Figure 1

The following chapter will make an extensive analysis of the currently available literature, followed by an analysis of the data collected and cleaned, and the paper will then conclude and state the research’s limitations.

The literature review will provide a background for this research and indicate previous findings, thus giving a more explicit orientation for the methodology. Papers about comparison methods for cloud computing platforms and platform optimisation will be presented and summed up in a continuous text below.

# CHAPTER 1. LITERATURE REVIEW ON PLATFORM COMPARISON

## 1.1 Goals and Structure of literature review

This literature review examines the current state of cloud-based programming platforms for universities, focusing on their comparison and optimisation for teaching and research purposes. Cloud computing has revolutionised how universities approach computing infrastructure, offering access to shared resources, reducing the need for local installation and management of software and hardware, and facilitating collaboration between students and researchers. However, with a growing number of platforms available, it is essential to identify the most effective options and strategies for utilising them.

The first part of this review concerns the methodology used for making it, as to make the process transparent and reproducible for anyone interested in this paper.

The second part of this review will explore strategies for comparing and evaluating different cloud-based programming platforms, including Jupyter Hub, Google Colab, Azure Notebooks, R Studio, Binder and IBM Watson. We will identify the various criteria that can be used to evaluate these platforms, such as ease of use, scalability, performance, security, and cost, and provide recommendations for selecting the most effective platform for a university’s needs.

In the third part, the focus will be on discovering the characteristics of the platforms mentioned above. It is essential to ensure that a strict and objective comparison is carried out. Nevertheless, finding platforms is not sufficient. After getting an idea of which ones are most suitable, it is necessary to understand how to optimise them for GSOM’s use, which will be done by researching how visual improvements of the platforms as well as identifying significant issues that are worth considering in the case at hand, along with solutions to them.

Finally, a summary of the current chapter will be drawn, including significant findings and limitations.

## 1.2 Search methodology

The primary method for finding papers was a targeted keyword search on Semantic Scholar[[1]](#footnote-1) and Google Scholar[[2]](#footnote-2). The keywords used in the search included “cloud computing,” “educational platform,” “platform comparison,” and other related terms. These searches were designed to identify papers that were relevant to the topic of cloud-based programming platforms for universities.

Semantic Scholar was a handy tool for this study due to its advanced search algorithms and filtering capabilities. The platform uses machine learning and natural language processing techniques to analyse and understand academic papers, which allows it to provide highly relevant search results. By using Semantic Scholar, a sizeable number of relevant papers were identified and collected.

One of the most valuable features of Semantic Scholar is its “similar papers” tab. This feature allows researchers to find articles like the ones considered relevant during the initial keyword search. This feature identified and collected additional papers that may have been missed through the initial keyword search.

In addition to using Semantic Scholar and Google Scholar, some papers were found by looking into the literature review of articles identified through the two methods above. This strategy, commonly known as the snowball method, proved an effective way to uncover additional papers not initially identified through the keyword search. This practice was beneficial for identifying papers not necessarily focused on cloud-based programming platforms for universities but included relevant information or data.

Several criteria were used for inclusion to ensure that the collected papers were relevant and high-quality. These criteria included the publication date, the source’s credibility, the study’s scope, and the methodology used. Only papers that met these criteria were included in the literature review.

Once the papers were collected, they were organised and analysed according to relevance and quality. This method involved reading each paper in detail, extracting critical information, and identifying common themes or trends across the literature. This process was time-consuming but necessary to ensure the literature review was comprehensive and accurate.

The methodology used to collect papers for this literature review was comprehensive and practical. Many relevant and high-quality papers were identified and collected by combining targeted keyword searches, advanced filtering and searching capabilities, the snowball method, and strict inclusion criteria. These papers will form the basis of the literature review, providing valuable insights into comparing and optimising cloud-based programming platforms for universities.

## 1.3 Platform comparison

### 1.3.1 Platforms and criterion definition

Cloud computing has become increasingly popular due to its many benefits, such as cost-effectiveness, scalability, and flexibility. In their paper “Cloud Computing and Comparison based on Service and Performance between Amazon AWS, Microsoft Azure, and Google Cloud,” Prakarsh Kaushik & al. provide an overview of cloud computing and compare the cloud services offered by Amazon Web Services (AWS), Microsoft Azure, and Google Cloud based on their features and performance. The authors analyse factors such as cost, availability, scalability, security, and support and perform a performance analysis of the platforms by running several tests and benchmarks. The paper concludes that all three platforms have their strengths and weaknesses and that the choice of platform depends on the specific needs and requirements of the user (Kaushik & al., 2021).

Nidhi Rajak and Ranjit Rajak discussed the computation of priority and scheduling of tasks in a cloud server’s virtual machines. The priority of tasks is decided by their rank value, computed using AET, DCT, and PTR. The tasks are then sorted for scheduling using minimum EST and EFT attributes. Four heuristic algorithms are compared for scheduling length, efficiency, speedup, SLR, and resource utilisation using two directed graphs, DAG1 and DAG2, with 10 and 15 tasks, respectively. The study shows that HEFT, CPOP, ALAP, and PETS algorithms can be used as standard algorithms for comparison purposes.

The article focuses on task scheduling algorithms in cloud computing based on performance metrics such as scheduling length, speedup, efficiency, resource utilisation, and cost. The article compares four well-known heuristics of task scheduling algorithms, namely HEFT, CPOP, ALAP, and PETS, and uses an example of DAG1 to explain their computation. The article explains the computational task attributes for scheduling and discusses the details of seven attributes. The article also discusses the objectives of cloud computing platforms and the importance of better resource utilisation (Rajak & Rajak, 2021).

In their article, T. Kluyver & al. discuss the benefits for researchers to make their computer code public, improving reproducibility for scientific research. It presents Jupyter notebooks as a way for publishing code, results, and, most importantly, explanations in a readable and executable form. Notebooks include many parts, allowing for a complete computational narrative. Jupyter is open-source and can work with code in many programming languages. Various tools are available for sharing and publishing notebooks, including № convert for converting notebooks to various file formats and № viewer for hosting notebooks on the web. The binder enables sharing live notebooks, including a computational environment where users can execute the code. The article also mentions the usefulness of notebooks in supporting materials for papers and books and the future possibility of authoring academic papers as notebooks. Real-time collaboration in notebooks is also being developed (Kluyver & al., 2016).

J. W. Johnson discussed the benefits, technical considerations, and best practices of using Jupyter notebooks as a teaching tool in various fields, including information technology. It highlights the importance of integrating exercises and interactive examples and suggests postponing the introduction of magic commands until students are familiar with the notebook. Additionally, the article outlines the advantages and disadvantages of installing the notebook locally or via a server and suggests that instructors require students to work linearly, from top to bottom. The authors propose a set of best practices for instructors and identify potential pitfalls, such as unexpected behaviour, difficulty reproducing results, security issues, and poor coding practices (Johnson, 2020).

Shubham Chandel, Colin B. Clement, Guillermo Serrato and Neel Sundaresan describe the training and evaluation of their transformer model, JuPyT5, designed to generate code in response to Data Science Problems (DSPs). The model is trained on publicly available Jupyter Notebook GitHub repositories and achieves a pass rate of over 77% on the DSP benchmark. The article introduces a new evaluation metric, DSP, and discusses the content and quality of this curated set of pedagogical Python notebooks. JuPyT5 is also evaluated on two other benchmarks, Humane Val and Mostly Basic Programming Problems (MBPP) and outperforms a larger Codex model at MBPP (Chandel, 2022).

S. Pulukuri and B. Abrams’s article discusses using Edpuzzle to monitor student progress and engagement through video assignments in STEM classrooms, particularly in teaching chemistry. Edpuzzle allows instructors to track metrics such as time spent on a video and grades, add interactive questions, chemical equations, and structures, and leave personalised comments on student answers. It also provides tips for using Edpuzzle effectively, including modifying existing videos, using the Live Classroom feature, and addressing the problem of students skipping videos. Additionally, the platform offers editing features, integrated questions for active learning, open-ended questions for in-depth responses, and notes for extra information. The article concludes that Edpuzzle is well-suited for designing and implementing meaningful video lessons in chemistry classrooms, with students finding it engaging and helpful for their learning (Pulukuri & Abrams, 2020).

A non-exhaustive yet representative list of Jupyter Hub competitors which could be reasonably considered to be alternatives for GSOM can be drawn as follows:

1. Google Colaboratory: a free platform for interactive coding in Python, based on Jupyter Notebooks, which runs on Google’s cloud servers.

2. Microsoft Azure Notebooks: a free, cloud-based Jupyter Notebook environment that allows users to run and share Jupyter Notebooks in a web browser.

3. RStudio Server: an open-source web-based interface for R programming, which allows users to run R code and work with RStudio in a web browser.

4. Binder is a free, open-source platform allowing users to share interactive Jupyter notebooks and other computational environments online.

5. IBM Watson Studio: a cloud-based data science and machine learning platform that provides Jupyter notebooks, as well as other tools for data preparation, analysis, and deployment.

The platforms to be compared and comparison points can be summed up in Table 1. The paper's first author related to the criterion, or the platform, is denoted once in full and then using subscripts. The symbol \* indicates that a particular platform has been chosen for GSOM’s need, not based on literature.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Jupyter Hub (Kluyver kl, Johnson j) | Google Colaboratory (Kaushik k) | Microsoft Azure Notebooks k | RStudio Server\* | Binder\* | IBM Watson Studio\* |
| Cost k |  |  |  |  |  |  |
| Availability k |  |  |  |  |  |  |
| Scalability k, r |  |  |  |  |  |  |
| Security k |  |  |  |  |  |  |
| Speed (Rajak, Chandel) r, ch |  |  |  |  |  |  |
| User functionalities (Pulukuri p) kl |  |  |  |  |  |  |
| IT functionalities p, kl |  |  |  |  |  |  |

*Table 1: Visualisation of comparison structure*

In conclusion, this literature review section covers various computing-related topics, including cloud computing, task scheduling algorithms, and Jupyter notebooks. The first article highlights the benefits of cloud computing and compares the services offered by three major cloud platforms, while the second article compares four well-known heuristics of task scheduling algorithms. The third and fourth articles focus on using Jupyter Notebooks for publishing and teaching purposes. The take-aways for the research at hand are that cloud computing is very promising, especially for purposes that GSOM pursues, and that several platforms could, in theory, do the job but are, at the end of the day, not significantly different in terms of functionalities from Jupyter Hub, which has the core advantages that GSOM already rolls it out. It is thus expected at this point that Jupyter Hub will not be found to be significantly worse than other platforms and should likely not be replaced. The platforms indicated above will be examined in further chapters more profoundly, considering GSOM’s usage, important aspects noted in this subchapter, and suggested methods found in this subsection to make a solid conclusion as to which platform is the most optimal for GSOM.

### 1.3.2 Platforms Comparison in Literature

This subsection extensively reviews the existing ways alternatives and complements to Jupyter Hub, which, as indicated earlier, is the current tool used by GSOM. The other platforms include Google Colab, Microsoft Azure, Binder, and IBM Watson.

#### 1.3.2.1 Jupyter

A. Rule & al. present “Ten simple rules for writing and sharing computational analyses in Jupyter Notebooks,” which covers aspects such as organising and annotating code, making notebooks readable and exploratory, and ensuring reproducibility and robustness in collaborative computational research. The authors also offer a Git repository with annotated example notebooks and encourage others to contribute and share their own experiences. The rules address challenges and opportunities in using Jupyter Notebooks for research and stress the importance of straightforward narrative and over-explaining, as well as managing dependencies and version control (Rule & al., 2019).

Jacek Rapiński’s, Michał Bednarczyk’s, and Daniel Sienkiewicz’s article is about Jupy TEP IDE, an online environment for programming and processing Earth Observation (EO) data using open-source software. The software combines Jupyter and Docker components to provide a highly configured and optimised platform for EO cloud infrastructure. Users can fully configure their environment, and the software offers ready-to-use algorithms, direct access to data and information access services. Jupy TEP IDE offers immense opportunities for advanced users to work with, process, and modify data. The European Space Agency funded the platform, and it is available for testing at https://jupyteo.com.

Jupy TEP IDE is an extension of the Jupyter Notebook web application that allows users to manage Jupyter extensions, run multiple notebooks simultaneously, manage clusters, and perform various operations on the operating system. The software provides a predefined environment with the most popular EO data processing tools pre-installed, real-time visualisation of processing results, and easy customisation of libraries and programming languages. The software integrates the most common open-source EO-based geospatial tools, libraries, and toolboxes for EO data, vector and raster data processing, visualisation, and presentation. Jupy TEP IDE does have limitations, including large resource requirements and infrastructure costs. However, it migrates to different DIAS-es and plans to enhance services and performance.

Jupy TEP IDE is built with the reuse paradigm, using core components such as Python environment, Jupyter-related tools, Container platform Docker, EO-based tools, and deep learning tools. The architecture of the system and its use cases are discussed, including targeting remote sensing experts and scientists. The platform is designed for multi-tenancy and an optimised EO cloud infrastructure software environment.

Jupy TEP IDE is an online tool for processing EO data, designed to operate as a web application with cloud-based infrastructure using Docker stacks. The platform aims to provide simplified access to EO data for developers and includes a set of predefined functions called cell magic. The system also supports parallel processing and is built as a multiservice platform with components in the form of running Docker containers in swarm mode. From the user perspective, the most crucial part is the Client part, which enables interaction with Jupy TEP IDE through a web browser (Rapiński & al., 2019).

Jeff Brown introduces Jupyter notebooks, used for scientific computing and data analysis, and explains how Jupyter hub, a multi-user server for Jupyter notebooks, can be set up on a CentOS server seven. Following a problem-based learning approach, it discusses the advantages of using Jupyter notebooks in data science classes. Step-by-step instructions for installing Anaconda, Node.js, and Jupyter Hub on a Linux machine and configuring Jupyter Hub are provided.

SE Linux must be set up correctly for Jupyter Hub to function properly. The article explains how to find and allow denied activities associated with the Jupyter hub command in the SE Linux audit log file. Moreover, the article highlights the usefulness of a Jupyter hub for a classroom setting and provides tips on conserving server memory (Brown, 2018).

F. M. Sallabi & S. Lazarova-Molnar’s article discusses using Jupyter Notebooks in teaching a graduate-level modelling, simulation, and performance evaluation course. The course covers assorted topics such as verification, calibration, and validation of simulation models, performance measurement in simulation-based environments, and Monte Carlo simulation. The main objective is to describe the methods used in the verification and validation process and increase the model's credibility.

The authors highlight how Jupyter Notebook is a valuable tool for working on the verification and validation processes. The notebook provides powerful functions to plot input and output data and make informative analyses and decisions. The authors also emphasize that using Jupyter Notebook requires competency in programming, LaTeX, Markdown, HTML, et cetera.

The authors provide several textbook examples and code these in Python. They emphasize the importance of validation and how to confirm that the previous model is correct before moving to the next module. As the model produces different outputs, the authors create cells to discuss the input parameters and output results and validate the model.

The authors also discuss using Monte Carlo simulation and how it solves specific stochastic and deterministic problems. Three analytical examples and one in-class exercise were given in this lecture.

Furthermore, the authors discuss using Spark Programming Model, which will exploit the processor's multi-cores and run the program faster. They executed one of the examples without Spark and with Spark and noticed the difference in execution time.

Additionally, the authors provide some discussions on the literature on using Jupyter Notebook, which is growing, and most are recent. The authors acknowledge the limitations and pitfalls of Jupyter Notebooks.

Overall, this scientific article presents a successful experience of teaching a modelling, simulation, and performance evaluation course online during the COVID-19 pandemic using Jupyter Notebook. The authors provide valuable insights into how Jupyter Notebook can be used in teaching and learning and highlight the advantages of this tool for developing and sharing educational materials. They demonstrate how to maximize the benefits of Jupyter Notebook using add-ins and tools useful for teaching and provide examples of other fields and courses where Jupyter Notebook has been used. The authors also provide a course structure and assessment tools to help other instructors develop and teach similar courses online. The article presents a comprehensive overview of the course, including using Jupyter Notebooks in teaching simulation and modelling (Sallabi & Lazarova-Molnar, 2022).

The article by L. Zhao & al. discusses using the My Geo Hub platform for cyber training and workforce development, which has successfully taught both graduate and undergraduate students about FAIR-compliant data practices. The platform has also been utilised in various training events and workshops to teach users about sustainability analysis and global gridded modelling. Additionally, the platform features general-purpose tools such as Multi Spec Online, which allows for image analysis and data visualisation and has been used for undergraduate research and engaging K-12 students in geospatial data analysis. The article demonstrates how My Geo Hub can be used for cyber training and workforce development in different fields.

This scientific article introduces the concept of science gateways and their role in research, education, and online collaboration. It highlights the use of science gateways to disseminate training materials and online modules but notes that the contents of these online learning modules are primarily limited to inert materials, which lack support for dynamic and interactive learning. The paper describes the newly augmented HUB zero platform developed under the NSF-funded Geo EDF project, which includes developing general-purpose online tools for education and integrating data and tool functions in the HUB zero course modules. The article showcases several projects using these capabilities on My Geo Hub to support a broad spectrum of learning and training activities, including cyber training modules, global sustainability workshops, and outreach programs for middle school students. The paper discusses how the HUB zero platform has been enhanced to provide seamless interactive online learning, including using Jupyter Notebook and RStudio environments (Zhao & al., 2020).

M. Chen discussed problems using PowerPoint (PPT) courseware as a teaching tool for computer language and programming courses. Teachers usually employ PPT courseware to teach grammar and related content, and screenshots of code segments and running results are given in PPT. It is believed that the disadvantage of this method is that students cannot see the actual program execution process and the program design process, so it is not easy for students to understand the teaching content and review it after class. Students cannot understand how to write programs from scratch.

The article proposes using a web-based application known as Jupyter Notebook as a teaching tool. The application allows for integrating text, pictures, and interactive code execution. The article highlights Jupyter Notebook's advantages at different learning stages and its suitability for teaching programming courses like Python.

The article notes traditional PPT teaching in computer language programming courses uses sample code, usually scattered as unrelated code blocks. Directly showing the source code file and passing it to students with PPT can result in students making mistakes or encountering errors when directly executing the code. Different grammar knowledge and essential techniques are mixed in the source code, and the output results are mixed, which is inconvenient for student observation and understanding. Many students will not bother to open the files individually and give up the experiment. Even initiative-taking students may just run the code file once and think the learning is complete after the results are obtained successfully.

The authors propose Jupyter Notebook as a teaching tool to overcome these problems. By using Jupyter Notebook, the teaching content (including text and pictures, et cetera) and code segments that can be interactively executed in real-time are integrated into one document, which is clear and intuitive, convenient for experiments, reduces the difficulty of understanding, and helps to enhance students’ interest in learning and positivity.

According to the article, Jupyter Notebook integrates all required teaching elements (titles, texts, diagrams, code blocks that can be interactively executed in real-time, and running results) on the same page, which is clear and intuitive. The text, pictures, the execution context, and the execution result are shown on the page as an executable code block or markdown. Students can follow the courseware to see the entire thinking evolution process and the program design process from scratch. They can also experiment directly, understand the program’s step-by-step construction process from simple to complex, and observe how it evolves from prototype to perfection, from error to correction, and the whole programming iterative process.

Jupyter Notebook allows for dynamic exploration in time and recording in class, teachers can initiate new explorations conveniently, and the teaching contents can be better reviewed after class. For essential knowledge points, teachers can directly enter the code snippets in Jupyter Notebook and execute them in the classroom. On the one hand, students have personal experience and see the design process and the execution process of the code segment. On the other hand, this dynamic adaptation slows the teaching and learning process, introduces a more flexible classroom rhythm, and leaves enough time for students to think, observe, experience, and remember. Additionally, when multiple senses can be mobilized simultaneously, it will be more conducive to learning and memory.

When using PPT, students only look at the courseware without firsthand practice. The Jupyter Notebook, an interactive execution software, is adopted to interactively execute corresponding code segments on the same webpage in real-time while watching text and pictures to understand the teaching content. Observations, experiences, and experiments are conveniently made.

Compared with PPT teaching, Jupyter Notebook integrates text, pictures, code, and running results with flexible and powerful notes. Students can get a complete story. The text, picture requirements, and interactive code are organized on the same page, and the code can be executed interactively in real time, which is clear and intuitive. The text, pictures, each intermediate step code, and the execution result appear on the page as a markdown note block or executable code block.

Jupyter Notebook makes programming more accessible. It allows people to develop programs according to their logical thinking. Jupyter Notebook dramatically lowers the threshold of exploratory learning and significantly improves the learning effect. Students can ask questions and implement their ideas at any time according to their thinking logic. They can start interactive experiments with small code blocks and essential functions, get the results, and visualize them. They can gradually improve the design, build more extensive and complex code blocks and debug. All of this can be done on the same page.

During this teaching reform, the results show that the introduction of Jupyter Notebook in teaching instead of traditional PowerPoint courseware has lowered the threshold for students to practice. Programming and experiments are more convenient than a regular presentation. The students have higher enthusiasm (Chen, 2021).

G. Zhang & al.’s article introduces an Online Education Big Data Platform that integrates educational data, including master data, user behaviour data, and system log data, to improve the quality of online education. By employing big data technologies such as GFS, MapReduce, Hadoop ecosystem, and Spark, the platform collects, cleans, stores, and analyses massive amounts of data generated during online learning. These technologies facilitate course building and teaching quality and provide a bridge between students, teachers, and job providers for better career opportunities.

The platform consists of four essential parts: primary education data, data collection/cleaning, ample educational data storage, and big data analytics for different users. The article presents an overview of the architecture, functional modules, and use cases for various users, such as students, teachers, and employers. It concludes with a summary of existing research, acknowledgements, funding sources, and a discussion of future work focusing on streaming computing for real-time online education analysis (Zhang & al., 2016).

#### 1.3.2.2 Google Colab

In his article, L. Baptista explores using Google Colab and Python scripts in the online teaching of physical chemistry. The discussion revolves around the advantages and difficulties associated with employing these tools. A survey conducted by the author indicates that most students found Google Colab to be user-friendly and easily accessible. To enhance the learning experience for small classes, the authors created seven Python scripts available to students through Google Colab.

Python, a user-friendly computational language, offers various scientific libraries that facilitate simulations, regression analysis, solving ordinary differential equations, and other advanced calculations. Google Colab, a free web platform, leverages Google's infrastructure to enable users to run Python and Tensor Flow calculations without any setup requirements. It eliminates the need for new users to download, install, and configure Python and its libraries on their personal computers, granting them free access to computational resources.

All the scripts developed by the authors were in the form of Jupyter Notebooks using the iPython format. These notebooks were shared with students via the Colab platform, with the link distributed through the Google Classroom used for the discipline. Multiple users can simultaneously access and utilize the notebooks through Colab.

Furthermore, the platform offers instructors features that aid in tracking student progress and activities. With its capability for scientific calculations, Google Colab proves to be a suitable tool, as it allows users to perform such computations without downloading, installing, or configuring any computational packages.

In conclusion, the article elucidates the implementation of Google Colab and Python scripts for online instruction in physical chemistry. It delves into the benefits and challenges associated with utilizing the platform, drawing insights from the authors' experiences with the scripts and platform. (Baptista, 2021).

W. Vallejo & al.’s article describes using Google Colab as an e-learning resource to teach thermodynamics through coding and simulation activities. The authors highlight that programming thinking can be an effective tool for learning thermodynamics, and interactive simulations that support multi-representational fluency are necessary for students to understand complex concepts. The Colab notebooks were used to teach thermodynamics in a physical chemistry class, with six notebooks divided into two sections covering introductory and specific topics in coding.

The authors provide step-by-step instructions for coding activities focused on physical chemistry properties such as ideal gases, van der Waals, Redlich-Kwong equations, and the heat capacity of chemical substances. The notebooks also contain exercises, solutions, and virtual lab simulation links. The article explains that using Colab notebooks and virtual lab platforms as e-learning resources can help overcome difficulties in learning thermodynamics, and interactive simulations and real-world datasets can engage students in active and constructive learning activities.

Moreover, the article highlights the importance of programming concepts and using Colab notebooks to teach chemistry students digital skills necessary in scientific and industrial sectors. Although introducing coding to students can be challenging, once the basic concepts are learned, students can use computer codes to solve complex problems in chemistry. The Colab notebooks are free on GitHub, and the estimated time to cover all content is 17 hours.

The article also mentions open-access platforms such as the Merlot system, the ChemCollective, and Harvard University's free online platform for science education that offer simulations. However, not all are free or easily accessible to many students in developing countries. The article emphasizes the importance of virtual labs and online simulations in overcoming the challenges posed by the COVID-19 pandemic and providing access to high-quality educational resources.

The authors conclude that Google Colab notebooks offer an excellent opportunity to enhance the teaching of physical chemistry and other sciences, providing a valuable resource for students and teachers. The article provides insights into using Google Colab notebooks in the chemistry teaching process, highlighting the versatility and significant scope of programming tools in all areas of knowledge (Vallejo & al., 2022).

M. Kuroki discussed how Google Colab could be a teaching tool for economics education. Google Colab is an online Python code editor for which the author provides a step-by-step guide on using the software, from creating a new notebook to executing code. The author covers simple examples in microeconomics for the first week.

Next, the article explores optimization problems in economics, including utility maximization and cost minimization. The article also covers pricing and output decisions in competitive and monopolistic markets. The second part of the article discusses how Python can be integrated to teach microeconomic theory to undergraduates, giving several examples of economic models that can be illustrated with Python.

The author argues that Python is a valuable tool for teaching microeconomic theory and recommends using Google Colab as it is web-based and does not require installation. Furthermore, using Python can help students prepare for tech and data analysis jobs, making it a valuable skill set.

Overall, the article emphasizes the importance of instructors learning Python and incorporating it into their teaching methods to stay up-to-date with modern technology and provide students with valuable skills in programming and data analytics (Kuroki, 2021).

#### 1.3.2.3 Microsoft Azure

Kolte & Pradhan’s article aims to evaluate the best ways to use various cloud computing services to achieve cost-effective, high-performance load balancing. In both education and business environments, load balancing plays a significant role in optimizing performance and response time. The article compares the disk performance of Azure cloud services, as disks play a vital role in data replication and load balancing.

Cloud computing provides a pay-as-you-go platform that offers various services, such as database storage, computing power, and applications. Data storage is the primary cloud service, becoming an essential technology in recent years. Various IT companies offer different platforms, like Amazon and Google app engines, making it easier to use cloud resources. Cloud storage consists of devices that use various software applications to provide data storage services.

Load balancing distributes workload on different resources of a node, shifting them to other nodes in a distributed network without disturbing the task. The load balancer uses the IP address of the web application, so all web data communication goes through the load balancer. There are different load balancers, such as software, hardware, and virtual load balancers, each with advantages and disadvantages.

Data replication is copying data across the storage or from one instance of the cloud to another. If a node fails, the data is still available from replicas stored on the server, which means data replication is crucial for data availability. For effective functioning, data replication and load-balanced solutions require real-time data replication on storage and compute nodes. Disk performance plays a vital role in data replication and query performance, making it essential to choose the proper disk when migrating from on-premises to the cloud.

Azure offers two types of storage disks, managed and unmanaged, for virtual machines. Unmanaged disks create a storage account in resources to hold the disk, while managed disks have no storage account limits and are easier to manage. Azure has several disk options, such as standard HDD, standard SSD, and premium SSD, each designed for different use cases.

The article compares the performance of different storage disks in Azure, finding that premium SSDs provide better read-and-write performance, making them suitable for high-performance applications. The study is significant for achieving efficient load balancing and data replication in Azure cloud systems.

Several articles in this collection also focus on data storage and cloud computing topics. They discuss the effects of power outages, errors and mitigation strategies for flash-memory-based drives, load balancing, and replication strategies.

Overall, the articles demonstrate the importance of data storage and cloud computing and provide valuable insights into current challenges and potential solutions. From improving the reliability of solid-state drives to optimizing load balancing, these studies significantly contribute to data storage and cloud computing (Kolte & Pradhan, 2022).

Danihelka & Kencl’s article explores the potential of cloud platforms, focusing on Microsoft Windows Azure, for creating rapidly scalable 3D environments suitable for multi-player online games, interactive 3D education, and e-commerce. However, synchronizing multiple users in a single world poses significant challenges due to network and cloud operation delays. The paper introduces a project to build 3D services in the cloud to create a platform allowing various devices to access 3D virtual environments stored in the cloud.

To facilitate the easy creation of 3D virtual shops and games, users could upload their 3D models and customize the environment's appearance. The article investigates how to provide scalable 3D world management in the cloud, requiring dynamic sharing of 3D virtual-world content while managing network and cloud latency and concurrency. Practical, measurements-based analysis shows the limits of the current cloud-based environment.

Despite these challenges, the article suggests that cloud computing is suitable for introducing more sophisticated services beyond the current web client-server paradigm. Cloud platforms can rapidly scale with fluctuating workloads due to fluctuating numbers of users and provide any high-performance computing power needed to operate the back end of 3D applications and environments.

The article includes a prototype approach for building interactive multi-user 3D services in the Windows Azure cloud environment, utilizing Silverlight 5 technology. However, creating a library to encapsulate the differences between the platforms supported proves challenging.

The article discusses a variety of experiments, including measuring the latency of responses from Windows Azure and the scalability, reliability, and response latency under a shared interaction scenario. Ultimately, the study finds that more measurements and perhaps alternative approaches may be necessary to determine the best approach to support proper multi-user interaction in 3D environments in the cloud.

Finally, the article explores three topics related to technology and computing: 3D talking-head interfaces for mobile phones, a reduction method for animated models for embedded devices, and cloud computing's potential to support online game developers. These topics are all related to using technology to enhance the user experience and improve the efficiency of computing systems (Danihelka & Kencl, 2013).

The shift to online teaching due to the COVID-19 pandemic has made online testing the primary method of evaluating student performance. However, online testing has its drawbacks, including factors that can influence test results. Ionescu & al.’s paper aims to analyse these factors related to test design quality, such as validity and reliability. Examples of these factors include the length and complexity of questions, question order, and the inability to go back to previous questions. The paper also presents an open-source Azure cloud-based web application designed to test the influence of these factors.

The study notes that external factors such as fatigue, medication, nutrition, health problems, motivation, anxiety, depression, psychological trauma, and others can also influence test results. However, these factors are harder to consider in an online test environment.

When designing tests, factors related to reliability and validity must be considered, such as test length, question length, complexity, order, and clarity of the test instructions. Adequately designed tests better indicate the student's knowledge of a subject, and the content must be relevant to the target audience. In most cases, a good test is moderate in difficulty and creates a positive learning experience by being clear and concise.

Environmental factors such as ambient temperature and the student's experience with ITC tools can also influence results. The paper highlights that online testing requires more fantastic planning and strategy; therefore, test design quality is crucial to obtaining relevant results.

The paper concludes by stating that the teacher's ability to construct tests is essential and requires proper training. The design of an online test must consider various factors and ensure that it is calibrated correctly. Additionally, more research is needed regarding testing in an online environment, but the development of this application provides a valuable tool for conducting further research.

The article reviews the implementation of an online testing platform that focuses on factors related to test quality. The platform was developed using .NET Core Identity API and is accessible on GitHub. However, the implementation had some limitations and challenges, including issues with the text overlapping with buttons in the Edge browser and no spending limit that can be set for the service.

Despite the challenges, the paper reviews factors influencing online tests and provides an open-source web application that addresses user experience, security, and performance issues. The authors plan to analyse the test result data and draw conclusions regarding the effect of poorly designed online tests on students' performance (Ionescu & al., 2021).

Virk & Maini’s scientific article introduces and provides insights into developing and implementing Windows Azure, Microsoft's cloud computing offering. Cloud computing provides an abstraction between the computing resource and its underlying technical architecture, allowing for lower upfront capital investments, improved business agility, and a pay-per-use model that aligns IT investment to operational expenditure. The article categorizes cloud computing into three forms: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

Windows Azure is an operating system as a service that offers a comprehensively managed hosted platform for running applications and services. It provides internet-scale cloud and developer services that can be used individually or together. The Azure Fabric controller governs the provisioning and management of nodes and additional technical infrastructure resources on the cloud.

The Windows Azure platform provides an internet-based cloud computing environment for running applications and storing data in Microsoft data centres worldwide. Its fabric provides two main areas of functionality: compute and storage, as well as Service Bus and Access Control capabilities. The article also describes the Azure Development Lifecycle, which follows two stages: application development and deployment & release.

Microsoft has invested heavily in building data centres worldwide and making them run efficiently, providing sustainable technologies to its customers. Windows Azure offers a range of options addressing various needs, including computing and storage, relational database management, cloud-based infrastructure, and online marketplaces for finding and purchasing datasets and cloud applications.

In conclusion, the article provides a comprehensive overview of Windows Azure and cloud computing, highlighting their benefits and potential applications in enterprise IT landscapes (Virk & Maini, 2012).

#### 1.3.2.4 R Studio

Understanding and applying statistical concepts have become critical for health professionals in the modern era of evidence-based medicine. However, students often find statistics complex and daunting, leading to anxiety and negative attitudes towards the subject. As a result, teaching basic statistical concepts can be challenging. To address this issue, da Silva & Moura conducted a study to analyse the effect of an introductory biostatistics course using RStudio on students' attitudes towards statistics and to assess the acceptance of the course among medical students.

The study included 43 first-year medical students, who were assessed for their attitudes towards statistics before and after the course using the Survey of Attitudes Toward Statistics (SATS-28) scale. Additionally, their acceptance of RStudio was assessed using a Technology Acceptance Model (TAM) scale at the end of the course. The study found statistically significant gains in the scores of three SATS dimensions: affect, cognitive competence, and difficulty. The authors also found that acceptance of RStudio was moderate to high in 93% of the participants, without any discernible differences between genders.

Many authors suggest using computers and statistics packages as teaching tools to engage students in the learning process. Statistical software enables students to visualize and interact with datasets, which can significantly improve their comprehension and performance. However, adding software to introductory statistical classes must consider several factors, such as availability, ease of use, and documentation.

One popular program for data analysis is the R language, which the authors introduced to medical students using RStudio. They found that introducing RStudio to students in a friendly manner was essential, as ease of use significantly shapes students' attitudes towards statistics. They also found that using RStudio's Notebook to work with previously prepared scripts decreased the difficulty of using the R language.

The study noted that the use of RStudio in health sciences education has not been adequately evaluated. Most students are familiar with user-friendly apps with functions easily accessed with one click, so the need to type in statistical commands may cause some discomfort initially. The authors emphasize that introducing RStudio in a friendly and supportive manner can help overcome this initial hurdle.

In conclusion, the study found that using RStudio to teach introductory statistics courses improved students' attitudes towards statistics, specifically in cognitive competence, affect, and difficulty. The study also found that women initially demonstrated lower levels of cognitive competence and difficulty towards statistics than men, but this difference was no longer apparent by the end of the course. The authors believe that introducing RStudio in a friendly and supportive manner can overcome the hesitance to use the R language effectively. The study suggests that RStudio is a valuable tool in teaching statistics to medical students, with positive associations with student attitudes and acceptance (da Silva & Moura, 2020).

Engels & al.’s article examines the effectiveness of active pedagogical methodologies in teaching data science for a biology curriculum at the University of Mons in Belgium. The course utilized blended learning and flipped classroom approaches, emphasising project-based biological data analysis. The student population was heterogeneous, and the learning objectives focused on developing high-level cognitive skills.

The authors employed a cyclic approach that involved stating goals, building pedagogical material, and analysing collected data to improve teaching practices progressively. They identified suboptimal learning strategies through learning analytics and discriminated between student profiles. The impact of the COVID-19 lockdown on students' production and interactions with the teaching staff was also evaluated.

The study found that interactive materials and ongoing assessment were effective in teaching data science and promoting independent and peer learning. The authors attempted to measure cognitive workload, but the objective quantitative measurement was challenging to obtain, and a proxy was used.

In another study, the authors describe a course that combined machine learning, time series analysis, and visualization of geo-referenced data for the biology section of the UMONS master’s program. The coursework was centralized on a WordPress site, and students had to log in with their GitHub account, enabling the collection of educational data from the UMONS Moodle server. The primary focus was on analysing actual data through projects proposed by students with exercises built according to Bloom's taxonomy of cognitive difficulty.

The authors collected data on students’ cognitive workload and learning profiles through the NASA LTX questionnaire and a non-supervised classification technique called a Self-Organizing Map. They also compared data from the pre-pandemic face-to-face learning sessions with data during lockdown-imposed remote learning. The study found that ongoing assessment promoted independent and peer learning and enhanced student self-confidence.

The article also discusses using R learnr tutorials in a data science course, focusing on optimising students' cognitive workload. The tutorials were online interactive documents that provided step-by-step guidance to students, with exercises and quizzes at each step. The authors measured cognitive workload using the average number of trials required for each student to find the correct answer in the tutorial exercises.

This article discusses several aspects of teaching data science to students with limited experience in mathematics and statistics. The authors describe their approach, including breaking down concepts into subunits, using blended and flipped classrooms, and emphasizing proactive exchanges between students and teachers. The study was conducted over three years and consisted of five terms, with ongoing assessment to measure student behaviour and learning profiles.

The article examines the cognitive workload in R learnr tutorials, using the number of trials needed to find the correct answer as a proxy. The authors combine this with the perceived cognitive workload to optimize the tutorials, emphasizing the need for reflective and systematic approaches to blended learning.

The authors highlight the importance of engagement and the transition from passive to proactive learning. Auto-evaluation exercises guided individual projects, and progression in difficulty was used to aid in this transition.

The study also presents the measured and perceived cognitive workload in R learnr tutorials. The tutorial was identified as a critical activity in the learning process, prompting the authors to focus on it. They use the number of trials needed to find the correct answer as a proxy for the cognitive workload and combine it with the perceived cognitive workload to gauge possible optimizations of the tutorials. The authors conclude that measuring the perceived and measured cognitive workload is essential to maintain reflective and systematic approaches in developing and evaluating a blended approach.

Group projects are essential to the course's methodology, but the authors observe that groups sometimes do not work well, and one student does most of the work. The study suggests that identifying different student profiles early in the course would enable the creation of better groups with a blend of complementary profiles to enrich the experience of all learners.

The study examines the transition from face-to-face to distance learning imposed by the COVID-19 lockdown. The authors note a marked decrease in contributions at the onset of the pandemic, followed by a significant compensatory activity. They suggest that diversification of activities is beneficial to guarantee student engagement, and an alternation between asynchronous work at home and synchronous work in the computer lab was more beneficial to student interactions.

In conclusion, the study highlights the challenges of teaching data science to students with limited experience in mathematics and statistics. The authors present their approach to addressing these challenges, emphasizing the importance of engagement, measurement of cognitive workload, ongoing assessment, and identification of student behaviour profiles. The authors conclude that the radical changes required to transition to distance learning during the COVID-19 lockdown show that students can adapt but also highlight the need for diversification of activities to ensure student engagement (Engels & al., 2022).

Gunawan & al.’s article discusses a new postgraduate course, "Applied Statistical Analysis with R". The course is designed to teach students the basic concepts of statistics, the knowledge of applying statistical theory in analysing accurate data, and the skill of developing statistical applications with R programming language. The course also adopts a Project-Based Learning (PBL) approach to encourage students to apply the knowledge gained to solve real-world problems and answer complex questions to generate high-quality results.

The article discusses the course structure and learning outcomes in detail, starting with the first half of each lesson dedicated to statistical concepts and the second half to practical implementation using the RStudio console. The PBL approach encourages students to use technology to find resources and information, create products, and collaborate more effectively. Various exciting projects are included in the course to demonstrate how statistical knowledge has been applied to real problems.

The assessment framework adopted is designed to monitor students' performance and identify their learning needs continuously. The framework includes two individual tests, in-class participation, one group project, and a final examination. The PBL approach is an effective teaching methodology as it enables students to gain hands-on experience using RStudio, motivates them to study and learn the relevant information for their projects, and the assessment framework enables continuous monitoring of student performance and identification of learning needs.

The article highlights the successful implementation of PBL in the course and provides valuable insights into how PBL strategies can be implemented to enhance statistical learning. However, local issues students face regarding finding authentic problems and time management require support and guidance for successful implementation.

The study on using project-based learning (PBL) in a statistics course shows that it effectively developed skills and improved interest in applying statistics. The PBL approach has successfully fostered students' motivation and creativity, built problem-solving skills, and provided opportunities for real-world applications. The survey showed that most students spent between 4 and 7 hours per week outside of class on the course and felt that the course was challenging (Gunawan & al., 2018).

McNamara’s article examines different approaches to teaching introductory statistics using the R programming language. Specifically, the study compares tidyverse and formula syntax and aims to provide instructors with empirical evidence to aid their decision-making process. The study involved two sections of an introductory statistics lab, with one section taught through formula syntax and the other through tidyverse syntax. The study focused on students' experiences in terms of performance, data collected from pre and post-surveys, observations on lab materials, and incidental data from YouTube and RStudio cloud.

The article highlights that educators often integrate technology into their instruction, and programming languages can be a valuable tool for teaching novices. However, the choice of syntax can impact user experiences, and there are divergent views on whether to use tidyverse or formula syntax. The study provides evidence that instructors should consider their pedagogical objectives and preferences when selecting syntaxes for teaching.

The study found slight differences between the two syntaxes, such as tidyverse covering more data manipulation and visualization options. However, there was no significant difference in performance between the two sections. Overall, the study suggests that both syntaxes could be taught in an introductory statistics course, with tidyverse providing students with additional skills and knowledge.

In addition to comparing syntaxes, the study also examined student evaluations of the instructor and course and their programming experiences. While there were some improvements in areas such as programming confidence, there was no overall trend.

Overall, this article provides valuable insights to instructors looking to teach introductory statistics using programming languages. The study suggests that various approaches may lead to better student outcomes and that the choice of syntax should align with learning objectives and student familiarity with programming languages.

The article delves into teaching introductory statistics using formula or tidyverse syntax in R programming. The study compared student engagement, learning outcomes, and materials used for the two syntaxes. The research encompassed various incidental data sources, including YouTube analytics, pre-and post-course surveys, and RStudio Cloud usage logs.

The pre-survey questions inquired about students' anxieties and aspirations, while the post-survey asked about their preferences between the two syntaxes. Students overwhelmingly preferred the syntax they were taught, even without exposure to the alternative. The study found no significant differences in the two syntaxes' teaching effectiveness, underscoring the importance of choosing concise and appropriate survey questions. The study also highlighted the potential usefulness of incidental data sources such as YouTube analytics to gauge student engagement.

In another study, the authors compared the formulas and functions used in the two syntaxes. They found that although students in the tidyverse section used quite nested functions, their reuse rates were higher than in other cases. The study also revealed how instructional materials for the tidyverse section had several additional functions that authors never showed in class. They noted that instructors must align instructional materials with the functions used in class, avoiding overloading students with unfamiliar concepts.

The study recommended that instructors reduce the number of functions and choose the right syntax to improve classroom instruction, as both syntaxes contain their advantages and disadvantages. The tidyverse used more functions and extended codes, while formulas had shorter codes but relied on more complicated syntax. The article emphasizes revisiting the function count after teaching a course, reducing the number of functions used, and repeating standard functions to enhance cognitive retention.

Overall, the teaching experience in either syntax was relatively similar and highlights the importance of analysing student engagement and materials used when teaching programming languages. Moreover, the study encourages reducing unfamiliar concepts and repeating common concepts while selecting appropriate syntax for classroom instruction (McNamara, 2023).

Stander & Valle’s article discusses a module implemented at the University of Plymouth that taught students how to manipulate, analyse and visualize big data from social networks using R programming language. The module had three learning goals (LGs), including manipulating and visualizing large datasets, extracting information from social media sources and reporting code and results reproducibly using RMarkdown.

The module introduced students to different topics such as data visualization, data manipulation using dplyr and tidyr, statistical inference, social media sentiment analysis, reproducible research using RMarkdown, dimension reduction, clustering, and parallel R.

The authors created lesson outlines to help students understand these topics and provided materials such as booklets and cheat sheets.

The module was successful in teaching data science skills to undergraduate students. The authors encountered some challenges while delivering the module, addressed through support, one-to-one help, and an open-door provision. Students were assessed through two exercises, and their reports and presentations were assessed on the appropriateness, insightfulness, and quality of analyses, visualizations, code, and reports.

The module was successful, and student performance and feedback indicated positive results. The authors recommended improving the module, including using RStudio Server, introducing physical presentations, and providing additional opportunities for students who fall behind. The module has shown that project-based learning methods that challenge students to apply their knowledge and skills to real-world problems are practical. The authors suggest that future modules aimed at teaching extensive data analysis should adopt similar strategies to ensure that students gain practical skills in extensive data analysis needed to address the challenges of the natural world (Stander & Valle, 2017).

#### 1.3.2.5 Binder

Citossi’s article discusses using simulations as an effective tool for teaching physics concepts to students. Specifically, the authors focus on a simulation code that models how gamma rays behave in a medium. The code uses Jupyter, AppMode, and MyBinder tools to create a user-friendly web interface accessible from multiple devices.

The authors provide several examples of how the simulation code can be used to teach physics concepts related to gamma rays. For instance, they demonstrate how the code enables students to examine energy distribution in a 500-photon beam and to understand the impact of Compton collisions on gamma-ray behaviour.

The article identifies the importance of understanding how gamma rays interact with matter and outlines different types of interactions, such as the photoelectric effect, Compton scattering, and pair production. The authors point out that simulations provide a valuable means of exploring such interactions interactively and engagingly.

The study also notes the relevance of simulations as a crucial teaching tool. They argue that simulations provide a way to learn about the behaviour of gamma rays in a medium that is otherwise impossible in a classroom setting and that simulations can help students build impactful concepts.

The article concludes that simulations are an innovative approach to enhancing physics learning experiences. The authors suggest that innovations like Jupyter, AppMode, and MyBinder tools are essential in creating an accessible and interactive platform for students to learn about complex physics concepts, especially during the pandemic (Citossi, 2020).

Dubach’s article delves into teachers' role in students' learning process. The project aims to understand the effectiveness of different teaching techniques and how these techniques impact students' ability to retain information. The results would provide critical insights to help beginning or less effective teachers by narrowing down the most effective teaching techniques.

To achieve this goal, the researcher employed various tools such as a binder notebook, digital camera, lined paper, computer, flash drive, and Microsoft Word program to record findings from three middle schools - Heritage, Quail Valley, and Pinion Mesa. The data collected focused on the teacher's techniques and how students reacted to them.

The findings indicated that eye contact with students was one of the most effective ways to keep their attention in the classroom. Additionally, the students reacted positively to voice pitch changes, significantly improving their concentration levels. Humour was another technique that proved helpful in creating a relaxed classroom environment, keeping students attentive and engaged. Contrarily, the students seemed to "zone out" when the teacher's facial expressions changed, indicating a lack of impact on this technique.

Interestingly, the teacher's posture also influenced how students responded in class. Teachers who stood straight or carried themselves with poise garnered better outcomes than their slouchy counterparts.

Based on the data collected, the project successfully proves the hypothesis that the effectiveness of teaching techniques goes beyond what is taught. How the information is taught significantly affects students' retention and overall learning outcomes. As such, teachers must improve their teaching techniques to achieve better results continuously.

The results of this study provide insightful information to help beginning teachers orient themselves and become better at their craft. By adopting some of the teaching techniques students prefer, teachers can significantly improve their effectiveness in the classroom.

The researcher enjoyed studying teaching techniques and observing how they affected students. They noted that each teacher had a unique approach that worked best for them, and tweaking some old techniques could also lead to better student outcomes. Here is hope that the results of this study can create a new generation of enjoyable teachers in our society (Dubach, 2020).

#### 1.3.2.6 IBM Watson

Müller & al.’s article explores the application of IBM Watson, an AI-based cognitive computing system, in education, specifically focusing on its use as a virtual tutor to answer students' questions. The authors conduct a study to evaluate the effectiveness of Watson's conversation tool as an educational tool, but they identify certain limitations like the pre-written answers. The limitations are caused by the lite version, which is incomplete compared to the Standard and Premium versions, where the NLC and the knowledge studio can define or create more precise and up-to-date answers in real-time according to the specific scenario.

While it is advantageous to use AI-based systems like Watson as a virtual tutor in specific didactical settings and software limitations, the authors suggest further research is required to explore the potential for using AI in computer science tutorials for undergraduate students, which may deliver more valuable insights, if executed more extensively. The authors note that exploring other AI education companies could help overcome the limitations of Watson while adding speech-to-text and voice output features could enhance the experience and make a robot teacher more realistic. Overall, the study highlights the potential of AI in education, but some limitations need to be addressed to make an AI-based virtual tutor effective and a reliable replacement for human tutors (Müller & al., 2018).

The importance of data analytics has increased significantly in businesses, healthcare, insurance, and education. IBM, a leading company, is developing cognitive learning systems to deliver personalized learning and improve students' results using natural language processing and machine learning. This integration allows for communication between humans and machines.

In the current education system, most students face difficulties without specific goals. Cognitive learning systems, which utilize AI, are being implemented to improve the quality and policy of education. IBM's Watson supercomputer assists all professionals in their relevant fields and educational institutions by helping educators develop instructional practices more effective than current ones.

Cognitive learning systems also help educators collect attendance, mark details and analyse individual student interests based on their results. Through predictive analytics, Watson can determine students' interests in finding better careers, considering their behaviour patterns. AI provides solutions to improve education quality and policy by assessing individual student outcomes. Studying how this technology works together helps prepare a report on admission, the number of attendances, student dropouts, their result analysis, and prospects. This personalized education approach will analyse the success rate of students using cognitive learning skills.

The education system follows the same structure for all students, regardless of their potential or learning ability. Each student is a standard part of a class or school, with a single teacher and a set curriculum. However, each student has a unique socio-economic background, quality, and ability to understand, and the current system is not personalized. Even student scorecards do not continually assess their talent or skill. Due to unforeseen factors after admission, new students may also succeed or struggle, which statistical models may not consider.

Learners need motivation, courage to face unforeseen challenges, hard work, and a good curriculum in all learning directions. Therefore, teachers must collect data to help them plan their syllabus and teaching policies and help build successful student futures. IBM Watson is also used in various fields like healthcare, agriculture, finance, and social media. Initially, it was designed to answer natural language questions posed during the Jeopardy quiz in 2011, and it has been consistently upgraded since then.

Cognitive learning systems build on neuroscience-based approaches that utilize brain-based computing and artificial network algorithms. While Watson was designed to handle text, it needs more medical field areas such as surgical videos, dermatological photographs and other non-text-based information. The AI-equipped machine generates user knowledge by processing nearly 80% of structured or unstructured data in any field or business.

Muraleedhar & Karani’s article highlights the significance of cognitive learning systems in education, emphasizing personalized or adaptive learning. It discusses the cognitive system prototype IBM developed through natural language processing, machine learning, and other technologies to improve human knowledge and expertise. This technology includes automated cognitive learning content, tutors, and assistants. These systems are designed to understand the needs of students, offer personalized education, and track progress.

Furthermore, the Watson Teacher Advisor is a web-based instructional planning tool that helps educators develop effective curricula based on their ability to reach a student's capability. This tool augments and extends the role of teachers through a recursive machine and human cognition. IBM Watson's two products, Watson Element and Watson Enlight, provide educators with data on socioeconomic status and academic strengths and weaknesses to help create a profile for each student unique to their needs.

The future of education is digital, interactive, and personalized. The article concludes that technology tools like mobile phones, applications, cognitive assistants, and connected objects will create a personalized learning environment where students can quickly learn and interact with teachers in a virtual space. Massive Open Online Courses (MOOCs) have already garnered student preference. Therefore, the future classroom should be interactive and digitized, with remote access to enhance learning (Muraleedhar & Karani, 2020).

Kollia & Siolas’ article showcases the successful application of IBM's Watson in educational and research contexts, specifically in developing question-answering systems. Three different applications were developed by teams of graduate students from the National Technical University of Athens, focusing on healthcare and tourism domains. Using deep learning and question-answering methodologies, the teams followed a specific development and testing process.

The first application was centred on breast cancer, and the team collected a corpus of relevant sources to develop questions and train Watson to answer them accurately. The system achieved an accuracy rate of 70% for blind questions, with the potential for further improvement.

The second application was focused on tourism in New York and relied on online sources to develop a set of questions and answers. The team reviewed question-and-answer pairs for phrasing and completeness and achieved an overall accuracy rate of 66%.

The third application focused on autism, and the team split into subgroups to gather relevant information and develop questions on different aspects of the disorder. Watson was trained with variations on question phrasing and achieved an accuracy rate between 65-80% for blind questions.

Overall, the article emphasizes the need for a diverse and representative corpus for training the system and utilizing blind questions for testing accuracy. The teams demonstrated successful implementation of Watson, and the article highlights the potential for further refinement and extension for real-life applications (Kollia & Siolas, 2016).

In their scientific article, Oliveira & al. discuss the use of chatbots in the educational context of student care in higher education is explored. The article begins by discussing the evolution of technologies in both educational and professional contexts and the emergence of new teaching and learning methods. Students now have access to face-to-face, blended, and distance learning environments, making efficient communication between students and educational contexts necessary. A chatbot interacting with students through text messages regarding doubts about the Higher Education Institution course is presented.

The work aims to create a chatbot that uses artificial intelligence and natural language processing to simulate human conversations effectively. The chatbot is designed to provide a more natural interaction with students, thereby improving autonomous assistance in the learning context. The study emphasizes that using bots capable of automatic student attendance is nothing new. Its efficiency in simulating conversations on specific topics from pre-established questions and answers allows for automating routine and repetitive tasks.

The article discusses the various chatbots available, including rule-based, expression-based, and generator bots. The author notes that the student-robot interaction process begins with a question posed by the student, with the chatbot using natural language processing and AI to ensure that the interaction resembles a human-to-human conversation.

The article discusses the need to choose a suitable platform for chatbot development and the evaluation process involved in selecting IBM’s Watson platform as the most suitable solution for the project.

The article then considers the architectural requirements for deploying the chatbot and the various modules involved in its operation. The chatbot is tested with fundamental doubts about the AVA Moodle system, with the interactions classified according to three criteria. The preliminary tests demonstrate that the chatbot can provide correct answers a significant percentage of the time.

Finally, the article outlines the steps in creating the chatbot via IBM’s Watson platform, from training the bot to recognising a student’s intention to testing the bot's accuracy.

In conclusion, the study discusses the development of a chatbot using the Watson cognitive platform to assist students in using an AVA Moodle platform. The chatbot is integrated into Facebook Messenger and is tested with students. Most students rated the chatbot excellent (Oliveira & al., 2019).

The impact of technology on medical education has rapidly evolved over the past two decades. Many physicians use electronic health records, transmit e-prescriptions, access smartphone clinical references, and complete continuing medical education courses online. Contemporary medical students are digital natives, unique from prior generations, and textbooks have now become electronic, while lectures are increasingly online. However, the ultimate objective of medical training remains the same: to prepare future doctors to be effective communicators, diagnosticians, and healers, as embodied by William Osler. While there is much enthusiasm for using technology in medical education, some fear that pupils may lose sight of humanism in medicine by uncritically embracing the latest digital gadgets.

Some innovative advancements are taking place outside of academia, with businessman and innovator Vinod Khosla suggesting that 80% of what physicians do could be replaced by computers. Khosla urges medical students to learn computing and data analytics to enhance their diagnostic and management skills. Medical education apps have been launched on Apple’s App Store to supplement the traditional curriculum, some of which were developed by medical students. IBM has even entered medical education by utilizing an iterative learning process to train the Watson supercomputer to become a powerful diagnostician.

The authors of this article, Colbert & Chokshi, believe that technology is not a panacea, and some fear that medical students who use iPhone heart monitors and access online learning resources may lose sight of humanism in medicine. While human physicians cannot compete against computers in memory or analytical skill, clinical reasoning and empathy remain an art as much as a science. Electronic health records (EHRs) are expected to improve patient outcomes, but some studies have demonstrated that medical trainees spend more time typing on a computer than engaging in conversation with patients. Overcoming this issue is hindered by the 'hidden curriculum', which can reward students who efficiently navigate the EHR and rapidly look up information on smartphones during ward rounds over those who demonstrate compassion through lengthier patient interactions.

Even so, it is vital to accept that technology in medicine is here to stay, but as an educational tool, technology must be utilized to develop medical students into effective communicators, collaborators, problem-solvers, and team players. Moreover, clinician-educators must model practical techniques to strengthen the patient and doctor relationship upon which medical practice is founded. To this end, the authors propose a few components that they believe are necessary for successful technological innovation in medical education.

Firstly, any technological innovation must be integrated into changes in pedagogy. For example, the Duke-NUS medical school in Singapore condenses preclinical science into one year, using team-based learning exercises in which students and faculty interact in class and clinical labs.

Furthermore, the first-year curriculum at Duke-NUS uses online lectures and interactive modules, enabling time during the school day for collaborative learning sessions. Secondly, technology enables learners to embrace an era of personalized medical education, differentiated according to the pace and mode of learning. For instance, the New York University (NYU) School of Medicine is developing a three-year, individualized curriculum grounded in population health, care coordination, and quality improvement. Students maintain virtual patient panels using de-identified NYU Langone Medical Center physician practice Data to assess competencies.

Thirdly, today’s learners are wired differently from their predecessors, consuming up to 11 hours of media daily, with only 6% of all media exposure representing print media. As such, medical students are distinguished from prior generations and demand more than static textbooks and non-interactive lecture formats for learning. Technology can be employed to tremendous advantage via interactive online coursework and in-person clinical labs. Additionally, social media is an excellent opportunity for students and educators to share ideas and work collaboratively.

Finally, the medical education sector needs rigorous assessment of technological advancements. Large-scale collaborations among schools that allow for joint experimentation, innovation, and outcome measurement are necessary. Ultimately, didactic resources that have proven to be the most successful could be pooled, benefiting the practice of medicine.

In summary, the authors propose that technology must help expose and immerse learners in the rapid changes in health care delivery. Medical education must not happen in an ivory tower, and technology should not be used to distance trainees from frontline healthcare delivery further. Rather than moving to digitized education, thoughtful use of technology is necessary to enable future physicians to utilize Watson completely to enhance medical practice (Colbert & Chokshi, 2014).

#### 1.3.2.7 Summary

The main takeaways from this section are that there is no one-fit solution for every use that could be made with online educational platforms. Different solutions will be the best in different cases depending on the context and the requirements. It should be noted, however, that it was often underlined that Jupyter Hub gives a highly satisfying adaptability, making it fit in many contexts. These contexts include industry, scientifical research, geology, and of course, the most relevant topic in this study, education. Indeed, frequently in the literature, it is mentioned that Jupyter Hub is a handy tool in the classroom, fostering student satisfaction. Google Colab, on the other hand, also has satisfactory results, but they do not allow to supervise the students or give them a familiar environment to work in, which remains a vital aspect of Jupyter Hub.

Table 2 summarises the information found in section 1.4.1, sorted according to the structure that concluded section 1.3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Jupyter Hub (Kluyver kl, Johnson j) | Google Colaboratory (Kaushik k) | Microsoft Azure Notebooks k | RStudio Server\* | Binder\* | IBM Watson Studio\* |
| Cost k | Rapiński ra | Baptista ba | Ionescu i, Virk v | - | - | - |
| Availability k | Brown b | Kuroki k, ba | i, v | da Silva da, Engels en, McNamara mc | - | Müller mü, Muraleedhar mu |
| Scalability k, r | Rule ru, ra | Vallejo v, k | Kolte ko, Danihelka d | en | - | Kollia ko |
| Security k | b | - | - | - | - | - |
| Speed (Rajak, Chandel) r, ch | Sallabi s | k | ko | - | - | - |
| User functionalities (Pulukuri p) kl | Chen c, Citossi ci, ru, ra | k, ba, v | i, d | Gunawan g, Stander s, da, en, mc | Dubach du, ci | Oliveira O, mü, mu, ko |
| IT functionalities p, kl | Zhao z, Zhang zh, c, ra | k, ba | v, ko, d | da, mc, s | du | Colbert co, mü, mu |

*Table 2: Visualisations of findings*

## 1.4 Platform optimisation

This section aims to find out existing methods for optimising educational platforms in several aspects. Those are visual and algorithm aspects of the educational platforms. Indeed, those appear to be the two main ways to improve them, as the visual aspect influences how students interact with the platform, and the challenges posed by using online platforms will also be covered in this section.

### 1.4.1 Visual perspective

This subsection aims to see what solutions have been found in the past for improving students’ involvement on educational platforms by improving their appearance. This definition might include the appeal that the platform has, as well as its general design.

In their study, Wang et al. delve into designing an educational information platform to visualise big educational data. The article outlines the four critical stages of the platform's development process: platform design, implementation, testing, and analysis. Extensive analysis is conducted to understand the characteristics and scope of big educational data. The authors present current design concepts for visual educational information platforms catering to Chinese elementary school students, teachers, and school administrators. Additionally, they explore critical technologies for big educational data visualization, architectural design processes, platform integration challenges, and relevant technology introductions. The conclusion is that China's big educational data visualization information platform is still in its early stages, requiring further theoretical exploration and addressing technical application challenges.

The article underscores the significance of visualizing big data in education and its impact on comprehending complex educational patterns. The researcher identifies a need for enhancing current research on the visualization of education big data, particularly in designing education informatization platforms centred around data visualization. The article provides insights into the framework design process, platform selection, introductions to pertinent technologies, platform implementation, and functional analysis of the educational information platform powered by big data visualization. Various technologies are used to develop the platform, including the LAMP architecture, B/S structure, and technologies like WEB3.0, PHP, Python, and MySQL. Finally, the article elucidates the implementation process of the educational information platform based on the visualization of education big data. (Wang & al., 2022).

The article “Optimisation of Online Course Platform for Piano Preschool Education Based on Internet Cloud Computing System” discusses developing, implementing, and optimising an online course platform for learning piano at the preschool educational level. It analyses the simulation and platform demand for various users, such as students, parents, teachers, administrators, and principals. The platform’s function design includes the square module, job module, user management module, and database design.

The article explains the system architecture and its components like backend, WeChat, client, and web page, and their integration, and highlights the use of internet cloud computing systems and voice sensor technology to enhance effectiveness. The results of extensive performance and functional tests showcase the system’s stability, reliability, and positive consumer response.

It also delves into the feasibility and benefits of combining designer classroom teaching and offline network teaching using the Moodle online teaching platform. The framework is considered financially and technically feasible with solid operability. Designing appropriate teaching content, employing constructivist teaching system design, and fostering independent learning strategies are emphasised to stimulate students’ initiative and enthusiasm.

The study, supported by the Research and Practice of Piano Teaching Reform in Preschool Education Majors from the Perspective of “Internet +,” proposes a highly interactive piano preschool education online course platform to cater to different users (Hu, 2022).

Y. Bai discussed using learning data from a platform to analyse and compare the effectiveness of various teaching designs in military vocational education courses. The low learning motivation among learners is addressed by focusing on classifying resources, planning the learning paths, providing support, and using incentive methods to encourage continued learning. Furthermore, the article proposes sharing learning record links with units and institutions to identify credits and calculate workload to strengthen learners’ initial motivation. The importance of activating resources and engaging all parties to achieve the best course design is emphasised. Finally, the study suggests platform optimisation strategies such as providing course knowledge guidance, following the memory curve, fostering a learning atmosphere through communication and discussion, and promoting habitual motivation through external incentives (Bai, 2020).

J. Tang discussed the design of a platform’s submodules to achieve efficiency by reducing unnecessary searching for users through low coupling and high cohesion. The study analyses the design method of each submodule and emphasises the importance of collaboration between different modules. The authors declare no competing financial interests or personal relationships.

The article proposes a blended learning model for optimising online English learning platforms using a collaborative filtering algorithm. The personalised recommendation algorithm is emphasised to provide learners with accurate and relevant learning resources. An improved version of the Jaccard coefficient addresses the issue of sparse data. The article concludes with the proposal’s significant theoretical and practical value in improving cybersecurity personnel training.

A hybrid learning model combining online and offline teaching is discussed in the article, which includes student evaluation of teachers for optimised content and personalised learning paths. The platform uses deep learning to recommend personalised test questions based on student’s ability portraits constructed from learning logs. The article emphasises the need for teachers to update their teaching methods to keep up with the fast-changing technological landscape.

The article proposes using modified cosine similarity and Jaccard coefficient measures for similarity calculations based on user ratings. The optimised modified cosine similarity combines the advantages of both measures and is considered more widely applicable.

The article proposes using a graph convolutional neural network for convolutional operations on a graph in various applications, such as data mining and neural network model training.

The article discusses the design of a self-directed learning platform with personalised learning recommendation services and four main functional modules. The article concludes with a focus on improving students’ vocabulary and grammar through pre-learning before class in blended learning (Tang, 2021).

Jiang’s article discusses the design of an innovative experiment platform for the “Modern Educational Technology” online course using virtual reality technology. The proposed platform is based on a time series of online courses linked to the same time point and uses virtual reality technology to optimise the online courses. The experimental results show that the proposed method can adapt to environmental changes quickly and with good portability. The article also compares the proposed design method with NSGA-II and DSS design methods using statistical analysis. The results show that the proposed method achieved better convergence speed and convergence degree with an excellent ability to maintain the diversity of “Modern Educational Technology” online courses. The Education Scientific Research of Inner Mongolia funded this article, “13th five-year plan” Project (Jiang, 2018).

What may be noted as a conclusion to this subsection is that platforms should generally be integrated with relevant tools correctly. For instance, the data from a particular platform may be retrieved on the platform under focus. In the case at hand, this would, for instance, mean linking Jupyter Hub and Docker so that the user does not have to parameter anything. Another fascinating insight underlined in the existing literature is that cloud-based programming platforms may be used in hybrid mode, offline and online. Indeed, nothing prevents students from using the platform during a course to experiment with a code presented to them or during an online class. Some students informally apply this method in GSOM, as Python courses can be watched live with a window opened on Jupyter Hub and another on Teams. Lastly, several times, it has been asserted that structure is a crucial element. By structure, it is meant the way the content is presented to the user, be it the order, the division and grouping between topics, or the link with a different structure. As per several review articles, much effort should be paid to improve the platform's ergonomics to improve UX and user involvement.

### 1.4.2 Challenges

This subsection investigates the issues of using online educational platforms and their potential or approved solutions. One idea underlying this research is to identify potential issues that are genuinely relevant to GSOM’s case and find inspiration for ideas to solve them.

The article authored by Talal H. Noora, Sherali Zeadally, Abdullah Alfazic, and Quan Z. Sheng provides an in-depth exploration of mobile cloud computing architecture, which consists of three layers: the mobile user layer, the mobile network layer, and the cloud services provider layer. Evaluating mobile cloud computing architectures along dimensions like connectivity, energy efficiency, and dynamics is a focal point of the article. Several research architectures in mobile cloud computing are introduced and compared, including a context-aware navigation system for the visually impaired, a virtual cloud computing provider for mobile devices, and Clone Cloud, which enhances mobile applications using cloud resources to optimize energy consumption and processing speed.

The challenges faced by mobile cloud computing environments encompass security, privacy, trust, bandwidth, data transfer, data management, synchronization, and energy efficiency. The article discusses future research directions in these areas. It emphasizes the need for robust, efficient, and scalable techniques to ensure security, preserve privacy, and effectively manage data in mobile cloud computing environments. Energy-efficient cloud models are also highlighted as vital to overcome hardware limitations in mobile devices.

The article compares and summarizes various research surveys on mobile cloud computing, covering aspects such as application processing, heterogeneity, security, privacy, energy efficiency, and multimedia. It explains cloud computing services' characteristics and defines two types: software as a service (SaaS) and platform as a service (PaaS).

While mobile devices can enhance energy efficiency by offloading compute-intensive tasks to cloud data centres, wireless networks consume more energy than data centres, leading to hidden costs for mobile users. Research challenges in mobile cloud computing include security, privacy, bandwidth, data management, energy efficiency, and heterogeneity. A comprehensive review of recent mobile cloud computing architectures is presented, comparing them to traditional cloud computing. The article concludes by calling for more efficient techniques to optimize energy usage and enable mobile users to benefit from mobile cloud computing fully.

In summary, mobile cloud computing offers advantages such as extended battery life and scalability. However, security, privacy, data management, synchronization, and energy efficiency challenges still require resolution. The article provides an overview of mobile cloud computing, distinguishing it from traditional cloud computing, and evaluates thirty proposed mobile cloud computing research architectures using specific assessment criteria. Future research challenges related to data delivery, task division, and improved service provisioning are also discussed. Comparative analysis and benchmarking are conducted to determine research challenges while highlighting implications for future investigations. (Noora & al., 2018).

The article by Soukaina Sraidi, El Miloud Smaili, Salma Azzouzi, and My El Hassan Charaf presents a sentiment analysis-based approach to address the problem of high dropout rates in MOOCs. The approach involves collecting and pre-processing data from social media platforms and MOOC forums, classifying the data based on polarity (positive, negative, or neutral) using machine learning algorithms, and grouping the negative interactions into five clusters to extract the root causes of failure. The causes are then presented using the Pareto quality control method to highlight the important ones. The aim is to provide decision-makers with the necessary support to define corrective and preventive actions to eliminate the causes generating a high dropout rate. The approach is limited to data in the Latin language, and the authors plan to propose new quality methods with different strategies to evaluate the learning process within MOOCs in the future.

In the article, the authors propose a quality-based and machine-learning approach to identify the leading causes of dropout and improve the learners’ experience. They suggest using sentiment analysis to extract data from MOOC forums and social media groups to understand learners’ difficulties and motivations. The article reviews previous works on MOOC dropout prediction and proposes a new system to analyse learner engagement and prevent dropout. The model is implemented in five steps, including data collection from MOOC forum data and social networks (Sraidi & al., 2022).

The article by S. Zheng and S. Yang discusses a simulation experiment on an SDN network that assesses network traffic and predicts congestion using a prediction model based on speech recognition technology. This model was compared to other models and could predict traffic situations with a relative error of 15%. It also utilised a pet control algorithm for queue management to balance traffic flow, showing better results than ECMP. This research could provide technical support for optimising educational information platforms.

While the study achieved some results, there is room for improvement, and future research should focus on finding a unified algorithm solution to enhance packet delay performance and maximise network resource utilisation. The article also explores the development, applications, and various fields of speech recognition technology, such as education and network traffic prediction.

Constructing an SDN network traffic prediction model can improve network programmability and flexibility. Machine learning algorithms were used to compare actual and predicted data streams in the SDN network. The article highlights the significance of network traffic control engineering in enhancing network performance and the limitations of traditional network management methods.

The innovative IRS mechanism proposed in the study is based on the advantages of SDN’s centralised control and the different performance requirements of data streams. The model analyses numerous factors influencing SDN network equipment, communication links, and network traffic. It employs a queue management algorithm for flow control during data plane implementation. The paper reviews previous research on speech recognition technology and network traffic control models, demonstrating the potential for better resource sharing among educational users using modern information technology (Zheng & Yang, 2022).

G. Ou discussed the limitations of current wireless resource management platforms and the need for more advanced algorithms to improve 5G simulation accuracy in high-density network scenarios. It focuses on packet processing based on matrix operations to enhance execution efficiency and reduce memory consumption for ultra-large-scale terminal simulations. The platform will use wavelet and artificial intelligence techniques to integrate resource and interference management algorithms.

The article also highlights the high demand for a 5G English education information platform, as it would enhance students’ development and meet society’s requirements. Current information and technology integration in education is gradually replacing traditional teaching methods. The article proposes establishing a 5G English education information platform following reliability, economy, advancement level, practicality, scalability, maintainability, and openness principles to meet user needs and improve learning efficiency.

The research, supported by the Construction and Research of POA Mode in Higher Vocational Colleges, analyses users’ support and expectation rate and evaluates the network’s performance and protocol optimisation. Results indicate that 5G is stable and feasible, but there is low satisfaction with the current education platform, mainly due to slow and inefficient network data transmission. The article suggests building a 5G platform to increase the effectiveness and reliability of network transmission and ensure a satisfactory student learning experience (Ou, 2020).

Papamitsiou’s article addresses the issue of constructing individual educational trajectories in massive open online courses (MOOCs) and the lack of a differentiated approach for each student. It proposes a hybrid solution combining intelligent data analysis methods and heuristic algorithms for effectively managing individual trajectories. The initial data and an algorithm for controlling the formation process using a neural network are presented. The Kohonen neural network is used for clustering and classifying objects, as it is a two-dimensional matrix of neurons divided into two sets for identifying assigned course elements and analysing the load state. The neural network is trained by the method of successive approximations. Optimising educational trajectories within the online course is achieved by forming maps of the optimal location of elements for a specific learner and allowing possible changes to the course structure. This proposed algorithm enhances the effectiveness and quality of knowledge gained throughout the online course (Papamitsiou, 2019).

G. Chen and al. discussed an information security sharing mechanism for online education platforms (OEP) based on blockchain technology, focusing on the challenges posed by data sharing and security during the COVID-19 pandemic. The proposed mechanism involves platform hash verification and node public key verification to reduce information tampering. A DPOS algorithm is used for distributed bookkeeping and data exchange, with the authors proposing a dual authentication mode of node + platform to protect data from hacking. The article provides specific code for creating and calling intelligent contracts using Ethereum as an example and emphasises the potential of blockchain to optimise information security in online education platforms. Future studies are suggested to integrate artificial intelligence technology and 5G communication for industry upgrading. This study was supported by grants from the Social Science Innovation and Development Research Project of Anhui Province and the Key Project of Party School (School of Administration) System of Anhui Province in 2020 (Chen and al., 2021).

The takeaway of this subsection is that several challenges remain when using online educational platforms. Those include students’ connection possibilities – there exists a gap between those with a stable and robust internet connection and those with more limited access to it, if there is any access. This issue must be a vital consideration for any university or educational organisation whose students might live in places with little to no internet connectivity. If students are not online, solutions exist, such as university-free Wi-Fi and computers. On top of this, security and privacy must be monitored to prevent the risk of data theft. Student dropout was also mentioned as an issue frequently occurring in online courses. In this case, the solution appears to be monitoring the students closely and grouping them to improvise this supervision.

## Summary and Limitations of Chapter 1

This last section is a sum up of the general idea of the literature review. The first part introduces the methods used to find the appropriate literature: keywords and the snowball method. The second part introduces a structure as a table to compare six platforms according to seven criteria, either backed by literature or based on GSOM’s suggestion, as is the case of 3 platforms. Secondly, it finds out the mentions of platforms and the according criterion in the literature review; all summed up in Table 2. The third part introduces the two ways to optimize a platform: the visual aspect heavily influencing the UX and several miscellaneous challenges that need to be overcome, such as connectivity issues. It also covers existing solutions to these challenges.

# CHAPTER 2. METHODOLOGY OF RESEARCH

This section’s purpose is to describe the methods used in the data analysis part of this paper. Three methods will be used.

The first, for the platform comparison part, is to be conducted as a table gathering the characteristics of platforms that matter to GSOM and for which data was possible to gather and compare. For this part, there is little sense and no practicality in making a quantitative analysis, as a summary of all characteristics, user gains, and user pains is mainly sufficient to understand which platforms suit GSOM better than the rest. The method used is relatively standard, as well as platform characteristics, and they may be used for further research even for a different institution. Indeed, the table for comparison itself is based on criteria suggested in previous literature, and not only in an academic context – for more detailed information, please refer to Chapter 1. The conclusions drawn, however, are done so using the particular case of GSOM, and it is unlikely that they will be suitable as such for making a base for further research, especially outside of the academic context. The methodology for finding papers has already been explicated in the previous chapter, and as a reminder is a blend of keyword searches on specialised AI-powered online libraries along with the snowball method.

The second part concerns platform optimisation from a more concrete perspective. There, the quantitative method will prevail as the goal is to find more information about which bugs can be found on the platform and in which intensity, as well as linking those bugs' occurrences with mainly one other aspect of the institution: lectures. Both users’ logs and lectures have timestamps, which is unique in the case of logs, as these are punctual events, and periods of 1,5 hours for the lectures, as they are events that last a specific time. To better look at the correlation between the lectures and the logs generation, it is more logical to match the start of these lectures with logs than the end. Indeed, problems generally occur when students want to log in, not when they want to log out. Even if bugs occur when users log out, they tend to be of minor importance as the user may close the tab and let the server figure out the issue. In the case of log in it might be touchier as users might log in to do work or could want to follow a programming lecture by reproducing the code in real-time. In those cases, a bug or delay is much more likely to be a real user pain. Therefore, it was chosen to match the timestamps for one hour. More would not make sense as there is no apparent link between a user connecting several hours before and during this lecture.

Meanwhile, it makes sense to keep some time before the lecture starts and after – as students might be logging in during this frame. On top of this, users need to be filtered. Indeed, only students need to be considered to test the hypothesis that most bugs happen around the time a lecture starts. The methods used to understand which events occur most often, which users are most active and which periods are most intensive in terms of lectures happening are mostly graphical – pie charts, heatmaps and bar plots – that are primarily sufficient to grasp the complete information. For the correlation, however, a Chi-Square test will be run to check for the significance of the correlation – and, as explained later – causation between lectures and bugs. Finally, the lecture peak occurrences are illustrated for a given period to illustrate how to determine which moments to add more server power to temper bug occurrence.

The third and last part of this research consists of a relatively simple yet very indicative survey that has been conducted to provide information to the company about general student satisfaction and wishes, as well as to understand the fit with the literature and to give recommendations for the platform besides which ones are most optimal and how to eliminate bugs. This survey has nearly enough respondents to be considered significant in results – 47 –, but given the context, this number is relatively satisfactory, as the population among GSOM that can give their opinion on Jupyter Hub is not so high. Results are thus used to give the company recommendations, but perhaps less so to serve as a basis for further research.

# CHAPTER 3. DATA ANALYSIS

The data used in this study consists of the attributes of platforms that GSOM’s administration considered as alternatives for Jupyter Hub, internal logs possessed by Saint-Peterburg State University, Graduate School of Management, from now on referred to as SPBU and GSOM, and data collected by a survey among GSOM students.

Considering the amount of information gathered, it is reasonable to say that the results presented later in this study can not be fully generalised. Indeed, slightly less than 50 respondents were found for the survey, and logs are not as numerous as they would be with a massive organisation, i.e., more extensive than GSOM, but the principal purpose of this research is to find a solution for GSOM, which then could be used in some similar cases.

## 3.1 Formal Platform Comparison

The platforms considered in this research were those mentioned during the interview with GSOM’s staff responsible for the cloud. As mentioned in the literature review, where they were summarised, they consist of the following list:

* Jupyter Hub
* Google Colaboratory
* Microsoft Azure Notebooks
* RStudio Server
* Binder
* IBM Watson Studio

They shall be compared per the criteria of interest to GSOM’s administration: the pricing, the ease of use, the availability in the Russian Federation, the presence of necessary functionalities and the need to roll out a platform from scratch. Before this, a summary of their overall structure and other relevant information will be given below.

At a high level, Jupyter Hub's architecture is composed of four main components:

1. Proxy: The proxy is responsible for routing incoming requests to the appropriate notebook server instance based on the user's session. It inspects the request headers for a Jupyter Hub cookie containing the user's session ID. Based on this ID, the proxy forwards the request to the appropriate notebook server instance, which can be running on the same machine as Jupyter Hub or on a remote machine.
2. Hub: The hub is the central component of Jupyter Hub and is responsible for managing user authentication and spawning notebook server instances. When a user logs in, the hub authenticates their credentials using the configured authenticator and then uses the spawner to launch a new notebook server instance for the user. The hub also monitors the status of notebook servers and can restart or terminate them if they become unresponsive.
3. Authenticator: The authenticator validates user credentials and returns a user object to the hub. Jupyter Hub supports a range of authentication methods, including OAuth, LDAP, and local authentication. When a user attempts to log in, the authenticator validates their credentials and returns a user object to the hub, which contains information about the user, such as their username, groups, and permissions.
4. Spawner: The spawner is responsible for launching and managing notebook server instances for each user. When the hub requests a new notebook server instance for a user, the spawner launches a new process or container and then communicates with the hub using REST APIs to report the status of the server instance. The spawner can also set up environment variables, mount directories, and configure other settings for each user's notebook server instance.

Jupyter Hub's architecture is designed to be scalable and flexible, allowing it to support a range of deployment scenarios and use cases. Its modular design makes it easy to customize and extend, and its REST APIs make it easy to integrate with other systems and tools.

Google Colab is a cloud-based Jupyter Notebook environment that allows users to run and share code online using Google's servers. It provides a free computing resource that can be used for research, education, and development purposes.

Google Colab is built on top of the Google Cloud Platform (GCP) and provides access to various computing resources, including CPU, GPU, and TPU instances. It also provides a range of pre-installed libraries and tools, including TensorFlow, PyTorch, and Scikit-learn, making it easy for users to start with machine learning and data analysis.

Google Colab allows users to create and share Jupyter Notebook documents that contain live code, equations, visualizations, and narrative text. Notebooks can be saved to Google Drive, shared with others, and edited collaboratively in real time.

Google Colab provides a range of features and benefits, including:

1. Free computing resources: Google Colab provides free access to cloud-based computing resources, making it easy for users to experiment with machine learning and data analysis without purchasing expensive hardware.
2. Pre-installed libraries and tools: Google Colab has a range of pre-installed libraries and tools, including popular machine learning frameworks like TensorFlow and PyTorch and data analysis libraries like Pandas and NumPy.
3. Collaboration features: Google Colab allows users to share notebooks with others and collaborate in real-time, making it easy to work together on projects and share knowledge.
4. Customizable environment: Google Colab allows users to install custom libraries and tools and provides a range of customization options for the computing environment, including CPU, GPU, and TPU instances.

Microsoft Azure Notebooks is a cloud-based service that provides a free, hosted Jupyter Notebook environment. Users can create and run Jupyter Notebooks on Microsoft's servers, eliminating the need to install or maintain any software on their local machines. The platform provides access to various pre-installed libraries and tools for data science, machine learning, and artificial intelligence.

Here are some key features of Microsoft Azure Notebooks:

1. Collaboration: Users can share their Jupyter Notebooks with others and collaborate on projects in real time.
2. Integration: Microsoft Azure Notebooks integrates with other Microsoft services, such as Azure Machine Learning, to provide users access to additional tools and services.
3. Security: Microsoft Azure Notebooks provides secure access to notebooks through Azure Active Directory authentication and supports multi-factor authentication for added security.
4. Customization: Users can customize their Jupyter Notebook environments by installing additional libraries or software or using pre-built environments provided by Microsoft.
5. Deployment: Microsoft Azure Notebooks provides tools for deploying Jupyter Notebooks to other platforms, such as Azure Virtual Machines or Azure Kubernetes Service, for scalable and reliable deployment.

Microsoft Azure Notebooks is a free service, but users can upgrade to paid plans for additional features and resources. The platform is released under the MIT License, a permissive open-source license allowing users to freely use, modify, and distribute the software for any purpose, including commercial purposes if they include the original copyright notice and disclaimer in any modified versions.

RStudio Server is a web-based integrated development environment (IDE) for the R programming language. It allows users to access RStudio's interface and functionality from a web browser, which makes it easier to collaborate on R projects and use R on remote servers. RStudio Server includes code editing, project management, version control integration, and debugging tools.

Here are some key features of the RStudio Server:

1. Web-based Interface: RStudio Server's interface is accessed through a web browser, meaning users can work on R projects from anywhere with an internet connection.
2. Collaboration: Multiple users can work on the same project simultaneously, facilitating collaboration and sharing of code and data.
3. Version Control Integration: RStudio Server integrates with popular version control systems such as Git and SVN, making it easier to manage changes to R code and collaborate with others.
4. Package Management: RStudio Server makes it easy to manage R packages, which are collections of code that extend R's functionality.
5. Security: RStudio Server provides secure authentication and encryption options to protect user data.
6. Customization: Users can customize the RStudio Server environment by installing additional R packages or pre-built configurations.

RStudio Server is available in both free and paid versions. The free version includes most of the features listed above but has some limitations, such as limited support and fewer options for customization. The paid version includes additional features and support options, such as priority support and access to specialized R packages. It is essential to add that R Studio is not much adapted for using Python, which is the primary language of interest for GSOM.

Binder is an open-source tool that allows users to create and share interactive, web-based computing environments called "Binder repositories". A Binder repository typically contains a collection of code, data, and configuration files that can be used to reproduce a specific computational environment, such as a Jupyter Notebook or RStudio project.

Here are some key features of Binder:

1. Reproducibility: Binder helps ensure computational reproducibility by allowing users to easily create and share computational environments that include all the necessary dependencies and configurations to run a specific project.
2. Interactivity: Binder repositories are web-based and interactive, allowing users to explore and run code in real time without installing software on their local machine.
3. Collaboration: Binder repositories can be easily shared, facilitating collaboration and making it easier for others to reproduce and build upon one’s work.
4. Customization: Binder repositories can be customized with different versions of software packages, libraries, and other dependencies, which allows users to create and share tailored computing environments for specific tasks.
5. Scalability: Binder repositories can be deployed on cloud platforms such as Google Cloud or Amazon Web Services, which allows them to scale up to handle larger datasets and more computationally intensive tasks.

One of the most common use cases for Binder is to create interactive Jupyter Notebooks that can be shared with others. Users can create a Binder repository containing a Jupyter Notebook and any necessary dependencies, then share the repository with others by providing a link. When someone clicks on the link, Binder will launch a web-based Jupyter Notebook server with the specific environment needed to run the notebook.

Binder is free and open-source and can be used to create and share computational environments for various languages and tools, including Python, R, Julia, and more.

IBM Watson Studio is a web-based integrated development environment (IDE) for data scientists and machine learning engineers. It provides a collaborative environment for building and deploying machine learning models, data pipelines, and data visualizations and offers a range of tools and services to support the end-to-end data science workflow.

Here are some key features of IBM Watson Studio:

1. Collaboration: IBM Watson Studio allows users to collaborate on data science projects with team members and stakeholders, making it easy to share code, data, and results.
2. Data preparation: IBM Watson Studio offers a range of tools and services to help users prepare and clean data, including data wrangling tools, data visualization tools, and integration with data catalogue services.
3. Model building: IBM Watson Studio provides various tools and services to help users build and train machine learning models, including support for popular open-source frameworks such as TensorFlow, PyTorch, and Scikit-learn.
4. Model deployment: IBM Watson Studio makes it easy to deploy machine learning models to production environments, with support for popular deployment options such as Kubernetes, Docker, and IBM Cloud.
5. Automation: IBM Watson Studio includes tools and services to help automate the data science workflow, such as model retraining, pipeline automation, and machine learning monitoring.
6. Integration: IBM Watson Studio integrates with a range of other IBM Cloud services, such as Watson Machine Learning, Watson Assistant, and Watson Discovery, as well as popular third-party tools and services.

IBM Watson Studio offers a range of pricing options, including a free plan with limited functionality and a range of paid plans with additional features and capabilities. Users can pay per user or hour of usage, depending on their needs.

The formal platform comparison will be shown in Table 3, a recall of Table 2, situated at the end of part 1.3, with a simplified sum of presence in literature and additional technical data added for the informed reader.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Jupyter Hub | Google Colaboratory | Microsoft Azure Notebooks | RStudio Server | Binder | IBM Watson Studio |
| Cost | Positively noted | Positively noted | Positively noted | Not noted | Not noted | Not noted |
| Availability | Positively noted | Positively noted | Positively noted | Not noted | Not noted | Not noted |
| Scalability | Positively noted | Positively noted | Positively noted | Positively noted | Not noted | Positively noted |
| User functionalities | Very positively noted | Positively noted | Positively noted | Very positively noted | Positively noted | Very positively noted |
| IT functionalities | Very positively noted | Positively noted | Positively noted | Positively noted | Positively noted | Very positively noted |
| Speed | Positively noted | Positively noted | Positively noted | Not noted | Not noted | Not noted |
| Security | Positively noted | Not noted | Not noted | Not noted | Not noted | Not noted |
| Works in Russia | Yes | Yes | Yes | Yes | Yes | Leaving |
| New roll-out | No | No | Yes | Yes | Yes | Yes |
| Python friendly | Yes | Yes | Yes | No | Yes | Yes |
| License | BSD-3 | Apache License 2.0 | MIT License | AGPL Version 3 | BSD-3 | Apache License 2.0 |

*Table 3: Formal platform comparison*

Several things may be noted in Table 3. First, only the three first platforms have been positively referred to in terms of cost. It is clear that free software, ceteris paribus, has advantages over paid one. Regarding availability, the first three platforms have an absolute advantage over the rest, as per available literature. Regarding users and IT functionalities, Jupyter Hub and IBM Watson are very well ranked in most papers that cover them; user functionalities are also a pillar advantage of R Studio. Again, speed is significantly positive only for the three first software. Lastly, only Jupyter Hub was mentioned for security for platform users.

In the present case, it is important to note three additional aspects that were unlikely to be present in literature – and indeed were not but are highly relevant to the practical case. If the software is not present in Russia, as is soon to be the case for IBM Watson, it can not be considered a severe alternative for GSOM, as GSOM can only pay for software in roubles and is in the Russian Federation. On top of this, it is impossible not to consider that Jupyter Hub is already up and running. Since it does not require tremendous new investments to configure, it shall be granted a bonus over other alternatives and Google Colab, a ready-made software. Finally, R Studio does not appear to be very Python-friendly, as it focuses on R language. This peculiarity is a minus as GSOM aims to teach mainly Python to its students, while R is of little concern.

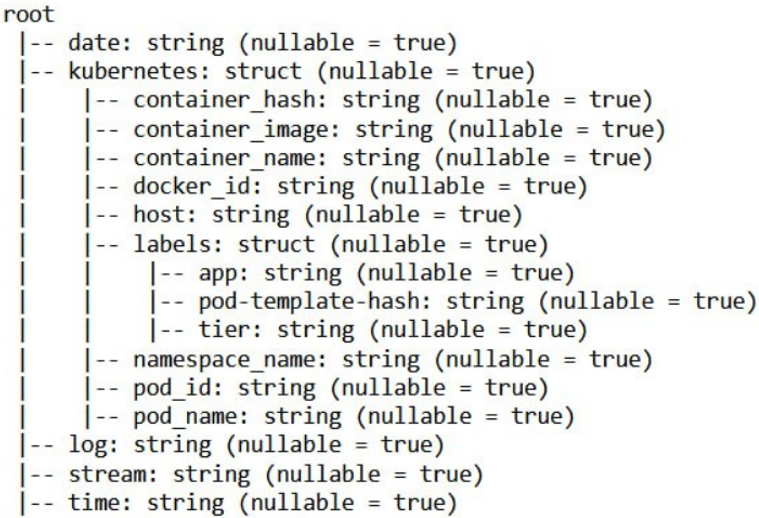
### Summary of part 3.1

In a nutshell, Jupyter Hub seems to be, per se, a good alternative, and considering practical factors, it emerges as the one with the most positive aspects and minor pains for users, be they students or IT. Nevertheless, while being a free and robust system, Google Colab possesses interesting functionalities that Jupyter Hub does not have. Therefore, this research indicates that the best solution, taking all factors into account, reducing costs and giving the best functionalities, is to use at the same time Jupyter Hub and Google Colab. Jupyter Hub gives the possibility to track students’ activity and work in a standardized environment, while Google Colab allows for much flexibility for individual student work. Whereas Jupyter Hub is on the entire university’s scale, Google Colab is much more suited for a single class’s scale.

## 3.2 Logs analysis

### 3.2.1 Data description

This section is dedicated to the in-depth analysis of the logs issues from Jupyter Hub’s exploitation by GSOM’s students and staff. Student use is deemed most important, as staff use does not generally happen during the peak time of usage that is predicted to be during lectures. Therefore, the logs will be filtered to account for st-users to test the primary hypothesis that most bugs are happening during the start of lectures.

**

*Figure 3: Logs structure*

As seen in Figure 3, the reports issued from Jupyter Hub’s use; logs are composed of the following information: a date and time (or, taken together, a timestamp, from year to second), Kubernetes (miscellaneous information about the log), a log which contains a log message, that is mostly the reference of errors. Stream is of little relevance in the case at hand. The timestamp and logs are the most important source of information, as they enable knowing which events occurred at which moment, allowing researching correlations with a set of other events, such as courses, as will be done later in this chapter.

*A picture containing screenshot, design

Description automatically generated*

*Figure 4: Bugs by recurrence*

As seen in Figure 4, the bug that is happening the most frequently, and by far, has code 1344. The correspondence between the code of errors and their entire denomination is presented in Annexe 3. In the case of 1344, the name is “failing suspected API request to not running server”. Other bugs represent miscellaneous categories, some of which are considered malfunctioning by the system but do not impact the user much, while 689 indicates an “unhandled error starting with timeout”, which is also quite problematic and forces the users to wait. Some others are bugs due to logging out; these affect fewer students as they may close the window and let the server solve it without human intervention. These logs are, therefore, of variable importance compared to 1344 but, at worst, match it. On top of this, looking at the utterly disproportionate division between errors, with error 1344 representing over 60% of all bugs, and the nature of such a bug, which is caused by a server overload, it is rather sensible to imagine that forecasting the demand and allocating more power to GSOM’s server when peaks are planned would tremendously reduce the number of errors occurring at all. This assumption will be investigated later in this subchapter, with more visuals showing evidence and explanations of why students often connect simultaneously.

*A picture containing circle, colorfulness, text, compact disk

Description automatically generated*

*Figure 5: Logs per user*

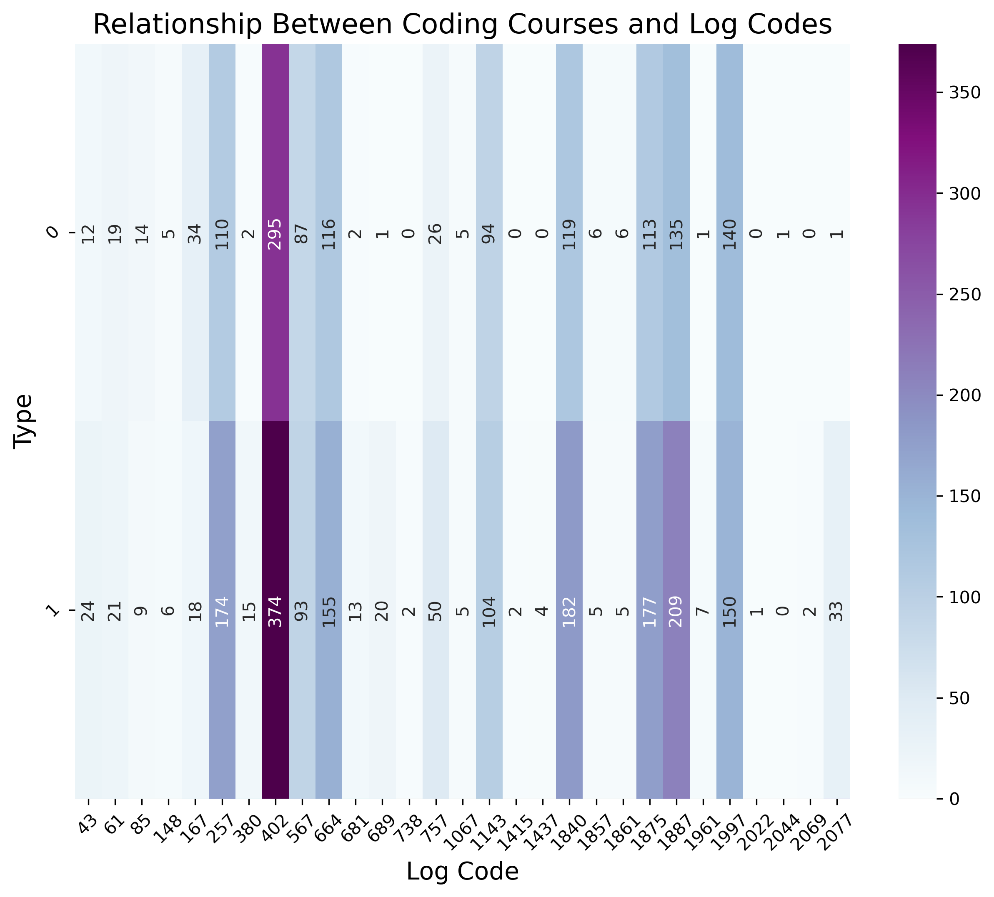
Figure 5 suggests that the 80/20 Pareto rule is partly illustrated here, with very few users (about eight) representing most of the use (55-60%). Nevertheless, this does not infirm the idea that many users try to connect at once when certain events occur, such as the start of a lecture, while others may use Jupyter Hub more comprehensively outside of any predictable event for personal or professional use outside of lecture hours.

This figure also indicates the need to filter on “st” users, which indicates students. Indeed, the main goal is to optimize the platform for users, not for GSOM’s IT staff. On top of this, the staff is likely not to start using a server at the start of a lecture since they do not attend those lectures, which would bias the correlation between course start and log generation, as the log they generate are conceptually not going to be linked to the event we investigate. Finally, in the largest log generators, one may spot two non-st users, whose massive log generation represents more than 10% of the total and reinforces the idea that their logs would badly influence the results that the research would find.

*A screenshot of a computer

Description automatically generated with medium confidence*

*Figure 6.1: Log code count per event*

**

*Figure 6.2: Log code count per event without code 1344*

Figures 6.1 and 6.2 show the link between the number of logs generated at the start of a course, denoted as 1 in Type and no course occurring, denoted as 0 in Type.

It should be noted that the logs and the courses’ start have been matched based on the exact start time of each course and the exact generation time of each log with a time delta of one hour. This relatively little frame was used as more significant approximations was considered not representative enough. Indeed, if a log was generated too long before a course started, it is implausible to be due to that course. On the other hand, if it has been generated not too long after the start of the lecture, while it is still well ongoing, it is likely enough to be linked to it. It is also necessary to consider that the logs generated due to many students logging in to Jupyter Hub will have been generated around the course start, not exactly on it. The choice was made, as indicated earlier, to take a time delta of one hour to match the timestamps to each other to mitigate the effects of these trade-offs.

This matching type enables finding exciting results. There is a visible correlation between lectures occurring and logs being generated. This correlation goes, as seen in Figure 6.1, for the logs belonging to code 1344, which are the most prominent, but for all other types of bugs, as seen in Figure 6.2. Sometimes the effect is more mitigated; this is the case for codes 1143 and 567. This fact indicates the need to check this correlation's significance between the start of GSOM lectures and log generation.

*A picture containing text, font, screenshot, number

Description automatically generated*

*Figure 7: Chi-Square test*

A simple yet very effective Chi-Square test correlation has been run to check for the significance of this correlation, as seen in Figure 7. The results, as denoted in the figure, are that not only is there a correlation, which corresponds to the figures shown earlier in the paper, but that the p-value of the test is very close to 0, which means that the null hypothesis, namely that there are no significant correlations between the start of lectures and log generation should be rejected. This paper thus concludes that there is a link between lectures and logs – meaning bugs, delays when loading Jupyter Hub and the like – but more than this, there are serious grounds for concluding that this relation is causation, not merely correlation. Indeed, it is conceptually logical that at the start of the course, many students rush to class, are asked to start Jupyter Hub to work on it, and all do so in a minimal time frame, especially since classes are all scheduled to start at given times (all at 9:00 AM, 10:45 AM and so on for master, 10 AM, ... for bachelors), indicating that there is a high demand not due to one class at once, but several at once.

Using the Chi-Square test requires checking a few assumptions. First, the sample size is considerable, containing more than 120.000 logs, which is much above what is required to consider the assumption to be met. Secondly, Chi-Square must be used on categorical data, which is very much the case here. Thirdly, the sample must be chosen randomly. In this case, it was chosen to focus on December 2022 for no reason other than to make the data amount convenient. There is, therefore, no bias in choosing the sample. Lastly, the observations need to be independent. This assumption is the only one that can not be crossed out perfectly, as observations might be influencing each other. Indeed, one user logging in will cause this very same user to log out. There is, however, no substantial influence of one on the other. Indeed, it is not possible to predict in advance when the user will log out, if the user will perform manipulations, of which nature those manipulations are, et cetera. Therefore, the assumption may be considered to be met, at the very least, in the context of this study. It is worthwhile to remind the reader that the data distribution is no concern for the Chi-Square test.

*A screenshot of a computer

Description automatically generated with medium confidence*

*Figure 8: Class activity per day and time*

Figure 8 shows the concentration of classes at a given time and days of the week. It might be noted that there are times in the week, especially Thursday around 10, 11, 14 and 16, as well as Tuesday at 10, 11, 14, and 16 when many classes start or resume simultaneously. This finding means many students will attempt to log in to Jupyter Hub simultaneously. Accepting that courses cause more activity, as seen via log creation and that too much activity for too little server power results in bugs, allocating more power in moments indicated in purple (in particular dark purple) should temper most bugs. It should be noted that this example is for December 2022 and should be made for up-to-date, fore coming lectures to allocate server power in time. Figure 9 does not, in any case, show a permanent course repartition.

### Summary of part 3.2

Most bugs occurring belong to one category – 1344, caused by too high demand on Jupyter Hub’s servers. Most other bugs potentially emerge from this too-high sudden demand. Such high demand is significantly correlated to, and presumably significantly caused by, the beginning of lectures. A reasonable way to significantly reduce the occurrence of this bug, along with most others, is to allocate more power to Jupyter Hub when more demand is forecasted; that is when many courses happen simultaneously, as presented in the previous subpart.

An idea for future research once allocating more power during high-demand zones has been done would be to check the effects of such a measure. Indeed, since this research is based on past data, it is not possible to test the efficiency of the suggested solution in this research.

## 3.3 Survey analysis

### 3.3.1 Data description

The survey structure may be found in Annex 1. It was sent to all GSOM students from all years, programs, and levels of study. In total, forty-one answers have been received, which is reasonable considering the size of the faculty and the fact that answer rates are usually not exceedingly high for surveys in any given configuration. It will therefore be assumed that this level is sufficient for the analysis, which will be carried forward.

Survey results may be found in Annex 2

Several imperfections might be noted in the data. Demographics are not balanced. First, the gender division is 66% female and 34% male respondents, as seen in Figure 9.1. It is unclear why such consequent disbalance is present, but considering the research question, the perceived imperfections of Jupyter Hub as it is, and suggested improvements, the gender gap should not be biasing the results. Missing functionalities would not appear to be detected in separate ways by male or female students. Age, as represented in Figure 9.2, seems to be distributed in an instead spread way, besides for age 21. Ages 17 and 26 are outliers academically and were expected to be low. However, bachelor students and first and second-course students are more represented, as presented in Figure 9.3. This issue might be explained sheerly by the presence of more students in bachelor, while there are very few in PhD studies. Also, a master’s last for two years, which means that the first and second courses from both bachelor, master’s and PhDs are adding up, inflating the numbers. It also appears that the management program is the most represented, with 40% of the respondents having indicated this option, while 21% indicated coming from the international management program, both being bachelor programs. Next come MiBA (19%), MiM (7%), and Smart Cities (7%), which are all master programs. The bachelor program in public management and PhD contain only one respondent each. Corporate finance has no respondent, which may be easily explained by the fact that this program has no coding course. Again, this should not significantly influence the conclusions drawn later in this paper.

*Figure 9.1: Gender distribution*

*Figure 9.2: Age distribution*

*Figure 9.3: Educational distribution*

The survey results, as per Figure 10, show that the primary usage students of Jupyter Hub are to use it for GSOM coding courses. Indeed, about 55% of the respondents ticked this box. GSOM gives several programming introduction courses for which the most indicated platform to follow everything up with a standard configuration, and where the teacher can post code, is Jupyter Hub. The two main other uses are for projects, about 25% and by themselves, 20%. Those are for completing (among others, GSOM) programming projects that are much less supervised, i.e., different for most students and can be easily tackled using tools other than Jupyter Hub. The “by myself” variant indicates a more personal use of the respondent, such as practising skills not required by GSOM, experimenting, or working on it for benefits. One respondent indicated in their answers that the usage in their specific case is for teaching.

*Figure 10: Jupyter Hub usage*

In the positive aspects of Jupyter Hub noted by the respondents, indicated in Figure 11, the 24/7 access (32%), the usability (30%), the interface (17%), and the many environments currently available (17%) were noted. Three options, “your opinion in the comments,” were selected, but none was receivable.

*Figure 11: Positive aspects*

The most recurrent answer from respondents to the question “Which changes would be desirable?” is “None,” with 36,4%. It should be noted that no question was asked as per bug correction. Indeed, it is assumed that any reasonable person would prefer having fewer bugs and loading time on the platform, and this should not be investigated. This part of the survey focuses on whether respondents want to see more coding languages, environments, and interface changes, propose other solutions, or do not need such changes. As seen in Figure 12, most do not see any need for change. Sharply 20% would like more languages on the platform, 18% would want a better interface and 15% more environments for coding. Among the acceptable comments, one suggested more comments on the platform to help beginners, another noted that there are often problems when many students connect at once and suggested making the cloud more powerful, and two others suggested improving upload speed and storage space.

*Figure 12: Changes desired*

On the self-estimation of maths, stats and coding skills, students indicated on average that they had 2,38/5 for coding, 3,31/5 for math and 2,97 for stats prior to Jupyter Hub use, graphed in Figures 13.1, 13.2, and 13.3. This result indicates low programming skills, relatively poor stats, and average math skills. It is thus safe that concluding students have much to learn when starting to use Jupyter Hub.

*Figure 13.1: Anterior coding level*

*Figure 13.2: Anterior math level*

*Figure 13.3: Anterior stats level*

Nearly half of the users have used the platform for a few months, 30% less than one month and 20% over a year, as per Figure 14. This result makes sense compared to the demographics above, considering that many users are in first class, indicating that they started using Jupyter for programming courses a month before the survey.

*Figure 14: Platform usage*

Regarding the question related to which other platforms they use, plotted in Figure 15, most respondents (28%) answered none. PyCharm is the second option, with 26%, Google Colab and Kaggle ticked by 14% of the respondents, and one chose Leetcode. R Studio was mentioned three times in the comments, SQL once, and Visual Studio Code twice.

*Figure 15: Other platforms used*

Two-thirds of the respondents indicated that their laptop, and thus their configuration, is their preferred option for programming, almost one-third indicated cloud computing as their favourite, and only one respondent chose computer class as the best option for their needs, as seen in Figure 16. There might exist some confusion in the answers to this question, as Jupyter Hub is cloud-based, but users interact with it on their personal computers. Since respondents do not take long to analyse survey questions, many likely answer “own laptop”, meaning “own laptop to connect to Jupyter Hub”. Therefore, no conclusion should be drawn on this data as a precaution.

*Figure 16: Preferred tool*

### Summary of part 3.3

The survey results show that students are mostly satisfied with the current platform and how it is parameters are made currently. The most positive aspects noted are the 24/7 access, which was not possible before the introduction of Jupyter Hub in GSOM, which mainly relied on computer classrooms. Only 2% of the respondents prefer that option. The useability is also praised. The number of environments, especially the interface, does not create the most extensive enthusiasm.

Regarding things students would like to change, “nothing” is the most prominent choice, confirming general satisfaction with the system. Although, a prominent part of the students would like more languages and changes in the interface.

Confirming what was seen in the literature review, the connectivity and accessibility issues (chapter 1.4.2) and the need for a visually engaging and ergonomic platform (chapter 1.4.1) are core concerns per students’ opinions. Likely, students with strong programming skills do not pay much attention to the platform problems that the more novice students might face. Indeed, it is logical that people who have never coded would suffer the lack of intuitiveness on Jupyter Hub. This student category is significant, as might be noted in the previous math, stats and coding level. While students arrive with high math and stats levels, their coding literacy is low-average. This statement, however, makes assumptions that may not be confirmed or infirmed scientifically in this paper and indicate potential future research paths.

# CHAPTER 4: CONCLUSIONS

## 4.1 Summary

From the research, Jupyter Hub appears to be the best choice for GSOM, and the authors see no reason to discontinue its use for the profit of other platforms. Nevertheless, introducing the use of Google Colab might prove helpful as its functionalities are complementary to those of Jupyter Hub. A suggestion to implement the use of Google Collab on top of keeping Jupyter Hub is thus made to the company.

Additionally, bugs occur in their large majority due to server overload. This fact is significantly explained by many lectures starting at once. The proposal from this research is to forecast demand by summarizing the schedule of upcoming lectures and adding more cloud computing power to the server when many lectures are starting at once. This solution is easy to implement and should prove effective, even though this needs to be checked in future research.

Lastly, this paper confirms most of what was noticed in the literature review, namely that platforms should be easy to use, user-friendly and ergonomic and that connectivity issues are a significant concern for cloud computing. Indeed, large broadband and sufficient server capacities are a must for cloud computing to be a helpful asset and not a creator of inequalities or inefficiencies. Students are primarily satisfied with it, but a significant part would like more languages and an improved interface. These results suggest that the company improve these aspects of Jupyter Hub if possible.

## 4.2 Future research possibilities

Future research based on this paper may be from mainly two natures: checking the solution for server overload and conducting in-depth interviews to understand better students’ needs that might not have been foresighted in this research. While the first is likely a simple yet very efficient solution, it must be bullet-proofed to make conclusions even more solid. Also, it is not impossible that this study has only partially understood students’ needs, and it would be desirable to have this checked to ensure that the conclusions made in this paper correctly represent students’ wishes for cloud computing at GSOM.

# REFERENCES

Bai, Y. (2020). Optimization Design of Military Vocational Education Platform Based on Learning Motivation. 2020 International Symposium on Advances in Informatics, Electronics and Education (ISAIEE). <https://doi.org/10.1109/isaiee51769.2020.00071>

Baptista, L. (2021). Using Python and Google Colab to Teach Physical Chemistry During Pandemic. ChemRxiv. <https://doi.org/10.26434/chemrxiv.13656665.v1>

Brown, J. L. (2018). Using Jupyter Hub in the Classroom: Setup and Lessons Learned. International Journal of Software Engineering & Applications, 9(2), 01–08. <https://doi.org/10.5121/ijsea.2018.9201>

Chen, G., Miao, G., & Xue, Y. (2021). Optimization of Information Security Sharing Mechanism of Online Education Platform based on Block Chain Technology. 2021 5th International Conference on Digital Technology in Education. <https://doi.org/10.1145/3488466.3488474>

Chen, M. (2021). Application of Jupyter Notebook in the Teaching of Programming Courses (Python). Advances in Higher Education, 5(2), 7–9. <https://doi.org/10.18686/ahe.v5i2.3309>

Citossi, M. (2020). IMPLEMENTATION OF GAMMA-RAY SIMULATION FOR EDUCATIONAL PURPOSES WITH AN INNOVATIVE WEB APPROACH. International E-Conference on Engineering, Technology and Management - ICETM 2020. <https://doi.org/10.15224/978-1-63248-188-7-10>

Colbert, J., & Chokshi, D. A. (2014). Technology in Medical Education—Osler Meets Watson. Journal of General Internal Medicine, 29(12), 1584–1585. <https://doi.org/10.1007/s11606-014-2975-x>

Da Silva Oliveira, J., Espindola, D., Barwaldt, R., Ribeiro, L., & Pias, M. (2019). IBM Watson Application as FAQ Assistant about Moodle. In Frontiers in Education Conference. <https://doi.org/10.1109/fie43999.2019.9028667>

Da Silva, H. S., & Moura, A. S. (2020). Teaching Introductory Statistical Classes in Medical Schools Using RStudio and R Statistical Language: Evaluating Technology Acceptance and Change in Attitude Toward Statistics. Journal of Statistics Education, 28(2), 212–219. <https://doi.org/10.1080/10691898.2020.1773354>

Danihelka, J., & Kencl, L. (2013). Collaborative 3D Environments over Windows Azure. Service Oriented Software Engineering. <https://doi.org/10.1109/sose.2013.41>

Dubach, A. L. (2020). Teaching Techniques. In Encyclopedia of the UN sustainable development goals (p. 860). Springer International Publishing. <https://doi.org/10.1007/978-3-319-95870-5_300262>

Engels, G., Grosjean, P., & Artus, F. (2022). Teaching data science to students in biology using R, RStudio and Learnr: Analysis of three years data. Foundations of Data Science, 0(0), 0. <https://doi.org/10.3934/fods.2022022>

Gunawan, A., Cheong, M. L. F., & Poh, J. (2018). An Essential Applied Statistical Analysis Course using RStudio with Project-Based Learning for Data Science. IEEE International Conference on Teaching, Assessment, and Learning for Engineering. <https://doi.org/10.1109/tale.2018.8615145>

Hu, Q. (2022). Optimization of Online Course Platform for Piano Preschool Education Based on Internet Cloud Computing System. Computational Intelligence and Neuroscience, 2022, 1–10. <https://doi.org/10.1155/2022/6525866>

Ionescu, V. M., Enescu, F. M., & Tudor, A. D. (2021). Azure Web Application for Testing the Factors Influencing Online Assessment Quality. International Conference on Electronics, Computers and Artificial Intelligence. <https://doi.org/10.1109/ecai52376.2021.9515099>

Jiang, H. (2018). Design of Innovative Experiment Platform for Network Courses of Modern Educational Technology Based on Virtual Reality. International Conference on Intelligent Computation Technology and Automation. <https://doi.org/10.1109/icicta.2018.00030>

Johnson, J. A. (2020). Benefits and Pitfalls of Jupyter Notebooks in the Classroom. Conference on Information Technology Education. <https://doi.org/10.1145/3368308.3415397>

Kaushik, P., Rao, A., Singh, D. P., Vashisht, S., & Gupta, S. (2021). Cloud Computing and Comparison based on Service and Performance between Amazon AWS, Microsoft Azure, and Google Cloud. 2021 International Conference on Technological Advancements and Innovations (ICTAI). <https://doi.org/10.1109/ictai53825.2021.9673425>

Kluyver, T., Ragan-Kelley, B., Pérez, F. L., Granger, B. E., Bussonnier, M., Frederic, J., Kelley, K. P., Hamrick, J. B., Grout, J., Corlay, S., Ivanov, P., Avila, D. E. G., Abdalla, S., & Willing, C. (2016). Jupyter Notebooks – a publishing format for reproducible computational workflows. International Conference on Electronic Publishing, 87–90. <https://doi.org/10.3233/978-1-61499-649-1-87>

Kollia, I., & Siolas, G. (2016). Using the IBM Watson cognitive system in educational contexts. In IEEE Symposium Series on Computational Intelligence. <https://doi.org/10.1109/ssci.2016.7849999>

Kolte, S. S., & Pradhan, M. A. (2022). Disk Comparison to Achieve a Load-balanced Solution in Azure Cloud Systems. SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology, 14(Spl-2 issue), 249–251. <https://doi.org/10.18090/samriddhi.v14spli02.09>

Kuroki, M. (2021). Using Python and Google Colab to teach undergraduate microeconomic theory. International Review of Economics Education, 38, 100225. <https://doi.org/10.1016/j.iree.2021.100225>

McNamara, A. (2022). Teaching modelling in introductory statistics: A comparison of formula and tidyverse syntaxes. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.2201.12960>

Müller, S., Bergande, B., & Brune, P. (2018). Robot Tutoring. In Proceedings of the 3rd European Conference of Software Engineering Education. <https://doi.org/10.1145/3209087.3209093>

Muraleedhar, R., & Karani, K. P. (2020). IBM Watson Industry Cognitive Education Methods. International Journal of Case Studies in Business, IT, and Education, 38–50. <https://doi.org/10.47992/ijcsbe.2581.6942.0066>

Noor, T. H., Zeadally, S., Alfazi, A., & Sheng, Q. Z. (2018). Mobile cloud computing: Challenges and future research directions. Journal of Network and Computer Applications, 115, 70–85. <https://doi.org/10.1016/j.jnca.2018.04.018>

Ou, G. (2020). Design Method of English Education Informationization Platform Based on 5G. IOP Conference Series, 750(1), 012052. <https://doi.org/10.1088/1757-899x/750/1/012052>

Pulukuri, S., & Abrams, B. (2020). Incorporating an Online Interactive Video Platform to Optimize Active Learning and Improve Student Accountability through Educational Videos. Journal of Chemical Education, 97(12), 4505–4514. <https://doi.org/10.1021/acs.jchemed.0c00855>

Rajak, N., & Rajak, R. (2021). Performance Metrics for Comparison of Heuristics Task Scheduling Algorithms in Cloud Computing Platform. Machine Learning Approach for Cloud Data Analytics in IoT, 195–226. <https://doi.org/10.1002/9781119785873.ch9>

Rapiński, J., Bednarczyk, M., & Zinkiewicz, D. (2019). Jupy TEP IDE as an Online Tool for Earth Observation Data Processing. Remote Sensing, 11(17), 1973. <https://doi.org/10.3390/rs11171973>

Rule, A., Birmingham, A., Zuñiga, C., Altintas, I., Huang, S., Knight, R., Moshiri, N., Nguyen, M. T. T., Rosenthal, S. B., Pérez, F. L., & Rose, P. S. (2019). Ten simple rules for writing and sharing computational analyses in Jupyter Notebooks. PLOS Computational Biology, 15(7), e1007007. <https://doi.org/10.1371/journal.pcbi.1007007>

Sallabi, F. M., & Lazarova-Molnar, S. (2022). Teaching Modelling, Simulation, and Performance Evaluation Course Online with Jupyter Notebook: Course Development and Lessons Learned. 2022 IEEE Frontiers in Education Conference (FIE). <https://doi.org/10.1109/fie56618.2022.9962690>

Soukaina, S., Miloud, S. E., Azzouzi, S., & Charaf, M. E. H. (2021). A Sentiment Analysis-Based Approach to Fight MOOCs’ Drop Out. Smart Innovation, Systems and Technologies, 509–520. <https://doi.org/10.1007/978-981-16-3637-0_36>

Stander, J., & Valle, L. D. (2017). On Enthusing Students About Big Data and Social Media Visualization and Analysis Using R, RStudio, and RMarkdown. Journal of Statistics Education, 25(2), 60–67. <https://doi.org/10.1080/10691898.2017.1322474>

Tang, J. (2021). Optimization of English Learning Platform Based on a Collaborative Filtering Algorithm. Complexity, 2021, 1–14. <https://doi.org/10.1155/2021/6624012>

Vallejo, W., Diaz-Uribe, C., & Fajardo, C. (2022). Google Colab and Virtual Simulations: Practical e-Learning Tools to Support the Teaching of Thermodynamics and to Introduce Coding to Students. ACS Omega, 7(8), 7421–7429. <https://doi.org/10.1021/acsomega.2c00362>

Virk, I. S., & Maini, R. (2012). CLOUD COMPUTING: WINDOWS AZURE PLATFORM. Journal of Global Research in Computer Sciences, 3(1). <http://www.rroij.com/open-access/cloud-computing-windows-azure-platform-74-76.pdf>

Wang, P., Zhao, P., & Li, Y. (2022). Design of Education Information Platform on Education Big Data Visualization. Wireless Communications and Mobile Computing, 2022, 1–13. <https://doi.org/10.1155/2022/6779105>

Zhang, G., Yang, Y., Zhai, X., Yao, Q., & Wang, J. (2016). Online education big data platform. International Conference on Computer Science and Education. <https://doi.org/10.1109/iccse.2016.7581555>

Zhao, L., Song, C., Biehl, L., Merwade, V., Huber, M., Liu, J., Baldos, U. L. C., & Shunko, I. (2020). Building a Gateway Infrastructure for Interactive Cyber Training and Workforce Development \*. Practice and Experience in Advanced Research Computing. <https://doi.org/10.1145/3311790.3396639>

Zhen, S., & Yang, S. (2022). Application of SDN Network Traffic Prediction Based on Speech Recognition in Educational Information Optimization Platform. Computational Intelligence and Neuroscience, 2022, 1–11. <https://doi.org/10.1155/2022/5716698>

# ANNEXES

## Annexe 1

## Survey composition

1. Gender:

* M
* F

1. Age:

* Whole number

1. Educational level:

* Bachelor
* Master
* PhD

1. Course:

* 1
* 2
* 3
* 4

1. Program:

* International Management
* Management (Bachelor)
* Public Administration
* Management (Master)
* MiBA
* Smart Cities
* Corporate Finance
* PhD

1. What do you use Jupyter Hub for?

* Programming course at GSOM
* Project
* By myself
* Your option in the comments

1. What do you like about the platform as such?

* Usability
* Interface
* Online access 24/7
* Many environments available
* Your option in the comments

1. What improvements would you like to see?

* Add more languages.
* Change interface.
* More environments
* None
* Your option in the comments

1. What was your experience in coding before using Jupyter Hub?

1 - knew nothing

5 - knew it all beforehand

* Programming
* Math
* Stats

1. How long did you use the platform?

* Less than a month
* Few months
* Over a year

1. Which other platforms or tools do you use for programming/data analysis?

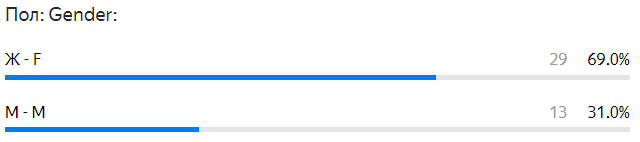
* None
* Google Colab
* Kaggle
* PyCharm
* Leetcode
* Your option in the comments

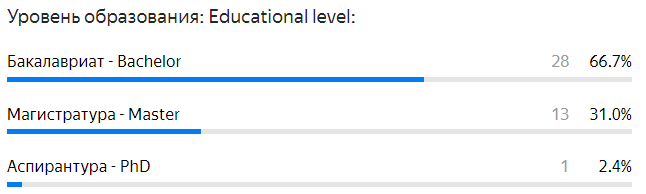
1. What type of tool do you prefer?

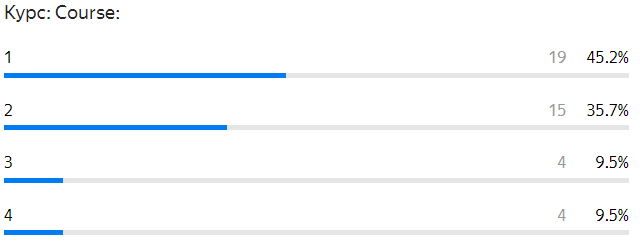
* Cloud-based
* Own laptop
* Computer class

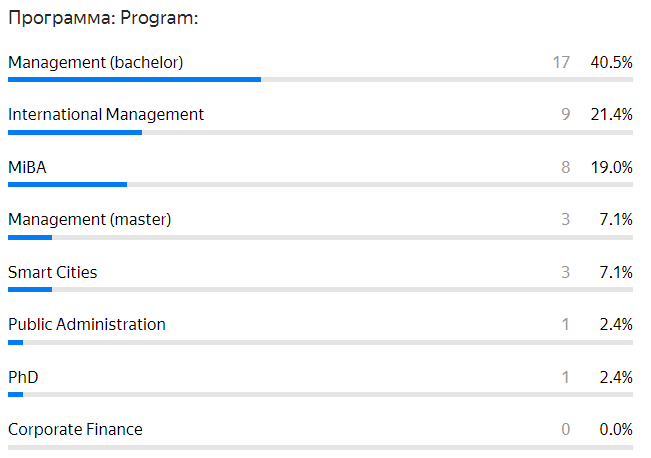
## Annexe 2

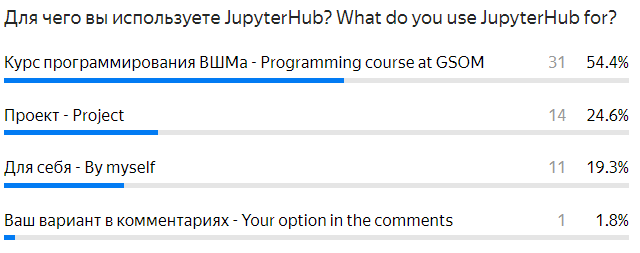
## Survey results

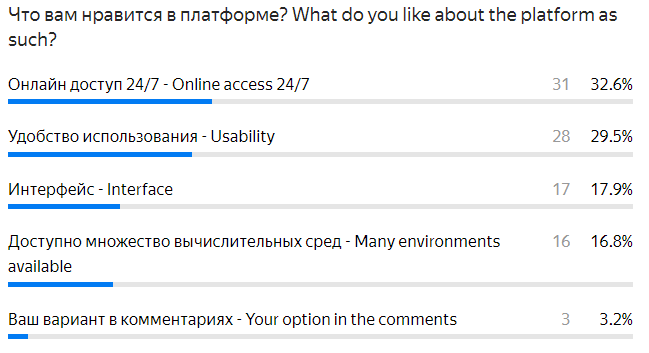


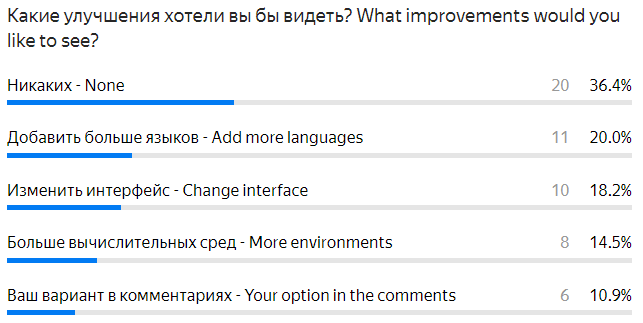


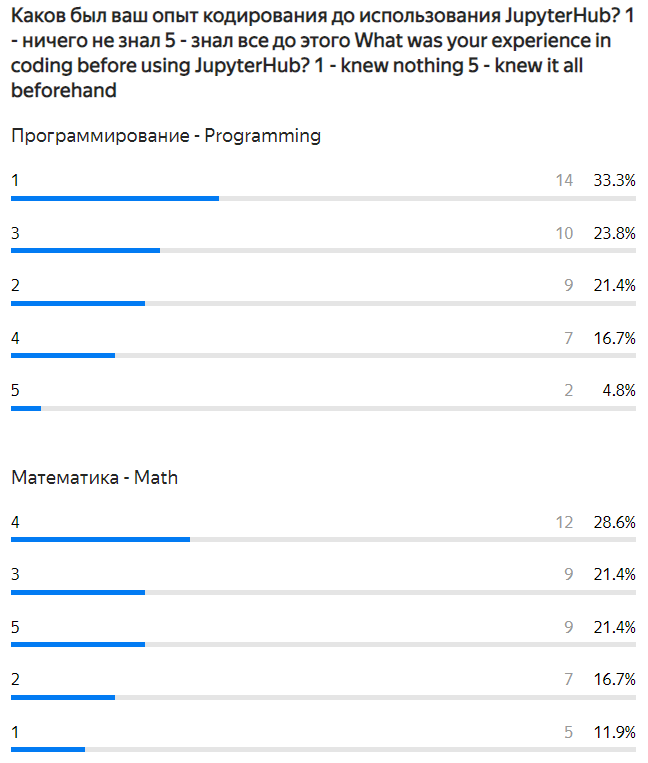


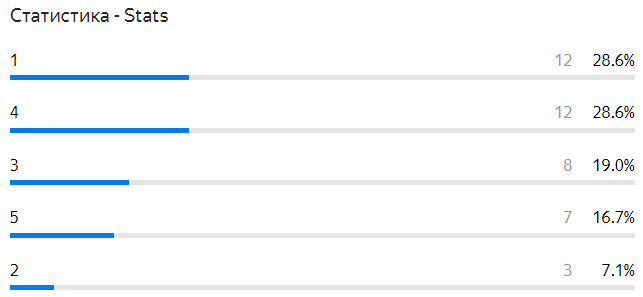


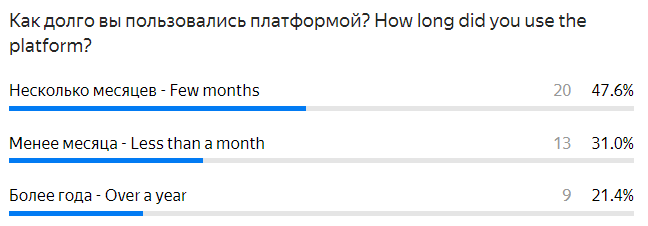


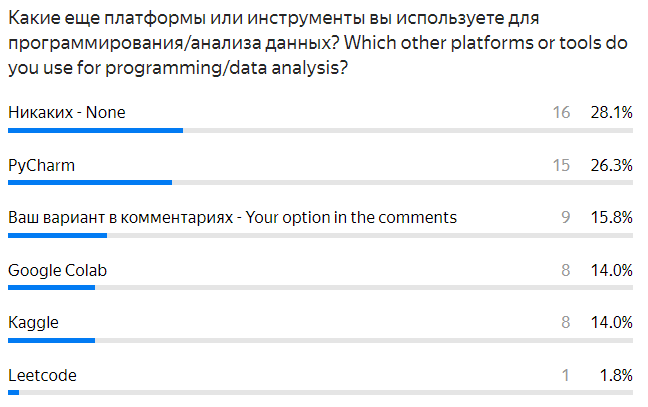


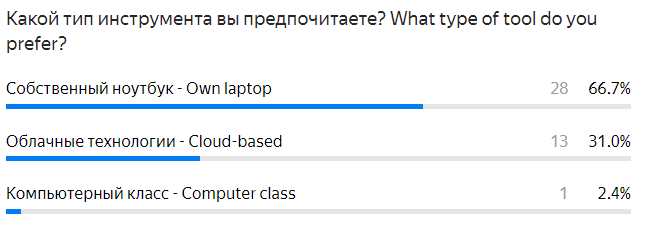












## Annexe 3

## Log codes

* '43': 'logged out’.
* '757': 'logged in’.
* '402': 'pending spawn'.
* '1875': 'attempt to create pvc with timeout’.
* '1887': 'pvc already exists’.
* '1840': 'attempting to create pod with timeout’.
* '1344': 'failing suspected API request to not running server’.
* '380': 'previous spawn failed’.
* '567': 'stream closed while handling’.
* '681': 'server failed to start’.
* '1997': 'deleting pod’.
* '689': 'unhandled error starting with timeout’.
* '1961': 'restarting pod reflector’.
* '2044': 'restarting pod reflector’.
* '257': 'adding user to proxy’.
* '664': 'server is ready’.
* '61': 'spawning server with advanced configuration option'.
* '85': 'spawning server with advanced configuration option'.
* '1143': 'server is slow to stop’.
* '2077': 'still running’.
* '167': 'server is already active’.
* '1067': 'user server stopped with exit code 1’.
* '2022': 'user server stopped with exit code 1’.
* '1857': 'found existing pod and attempting to kill’.
* '1861': 'killed pod and will try starting single user pod again’.
* '738': 'server never showed up and giving up’.
* '2069': 'user does not appear to be running and shutting it down’.
* '148': 'user is running’.
* '1415': 'admin requesting spawn on behalf’.
* '1437': 'user requested server which user do not own’.

## Annexe 4

## Code link

<https://github.com/Az-Vanya/Comparison-and-Optimization-of-IT-Coding-Platforms-in-Educational-Context.git>

1. <https://www.semanticscholar.org/> [↑](#footnote-ref-1)
2. <https://scholar.google.com/> [↑](#footnote-ref-2)