International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I8P133

Volume 74 Issue 8, 496-507, August 2026

Original Article

Design and Deployment of an Open-Source LMS Architecture for Hybrid Pre-Laboratory Instruction: An Open edX Implementation Case Study

S. Aimara[1*], M. Radid[1], G. Chemsi[2], M. Khiati[3], S. Nallusamy[4]

[1] Laboratory of Analytical and Molecular Chemistry, Faculty of Sciences Ben M'Sick, Hassan II University, Casablanca, Morocco.
[2]Laboratory for Mathematics, Artificial Intelligence and Digital Learning, Faculty of Sciences Ben M'Sick, Hassan II University, Casablanca, Morocco.
[3]Applied Economics and Finance Laboratory, Faculty of Sciences, Ben M'Sick, Hassan II University, Casablanca, Morocco.
[4]School of Engineering Management and Continuing Education, Jadavpur University, Kolkata -700032, India.

[*]Corresponding Author : salma.aimara123@gmail.com

Received: 22 June 2026 Revised: 17 August 2026 Accepted: 21 August 2026 Published: 29 August 2026

Abstract *- This paper reports the design, implementation and deployment of an open-source Learning Management System (LMS) architecture supporting hybrid pre-laboratory instruction in a resource-constrained university setting. The platform was deployed with Tutor, the Docker-based distribution of Open edX, on a single virtual private server with four virtual CPU cores and 4 GB of RAM, with DNS and TLS resolution provided by Cloudflare. The architecture is layered across infrastructure, platform, course content and data processing. Two pre-laboratory modules were delivered through short instructional videos, auto-graded formative quizzes and an embedded interactive serious game, and the engineering trade-offs of third-party tool integration without a Learning Tools Interoperability bridge are analysed. A four-stage learning analytics pipeline extracts learner activity from the LMS gradebook, selects valid engagement signals and computes a composite engagement index. Operational validation over a continuous 58-day production period shows a mean edge response time of 67.3 ms (SD 14.0 ms, n = 20), an application-tier saturation throughput of approximately 74 requests per second, and zero failed requests across 1,600 load-test requests at concurrency levels from 5 to 50. Across the deployed cohort, 82.4% of learners in Module CS101 (n = 17) and 77.3% in Module CS102 (n = 22) attempted at least one formative quiz, and the composite engagement index (n = 17) reached a mean of 76.5% (SD 33.8%, median 88.9%). The result is a low-cost, reproducible blueprint for hybrid digital instruction, together with a quantified account of its capacity envelope and its limits.*

Keywords *- Learning Management System, Serious game integration, Educational data pipeline, Engagement analytics, System implementation, Resource-Constrained Deployment.*

1. Introduction

Hands-on laboratory work is an essential component of undergraduate science education [1]; however, effective implementation is increasingly difficult in large public universities with large numbers of students, limited laboratory time, and limited budgets [2, 3]. In this context, hybrid instruction, which integrates online preparation with face-to-face laboratory sessions, has emerged as a pragmatic strategy [4]. However, most hybrid solutions rely on access to commercial platforms or major technical resources not readily available in resource-limited institutions.

This paper deals with the problem from an engineering point of view. The question is not whether digital pre-laboratory preparation is pedagogically useful, but how to design, build and deploy a complete, functional and reproducible hybrid learning environment at minimal cost. The work reported here concerns an open-source LMS architecture based on Open edX, supporting pre-laboratory preparation for first-year biology practical work at the Faculty of Sciences, Ben M'Sick, Hassan II University, Casablanca.

Three contributions are offered. First, a complete containerized open-source architecture is described that runs on a single modest server, together with a quantitative characterisation of its performance envelope under load. Second, the integration of external interactive resources - a serious game and instructional videos - is presented, and the engineering trade-offs arising from the absence of a Learning Tools Interoperability bridge are analysed. Third, a

lightweight learning-analytics pipeline is proposed that transforms raw gradebook exports into a usable engagement index, with an explicit account of the statistical properties and the limitations of the resulting signal. The remainder of the paper covers related work, requirements, architecture, implementation, the data pipeline, deployment and performance results, and a discussion of reproducibility, scalability and limitations.

2. Related Work

2.1. Learning Management Systems for Hybrid and Blended Instruction

Learning Management Systems (LMS) have become the backbone of hybrid and blended instruction, providing a single environment in which course content, assessment, and learner tracking are centralized [6]. Among the available solutions, open-source platforms are particularly attractive for institutions operating under budget constraints, since they remove licensing costs and allow full control over hosting and customization [5].

The most popular open source LMS are Moodle and Open edX. Moodle is a mature plugin-based platform optimized for institution-wide course management and the default choice for general academic administration. Built in 2013 by edX and MIT, Open edX was designed for large-scale, structured, sequential learning.

It has a distinct separation of the authoring environment (Studio) from the delivery platform, and a containerized deployment path through Tutor. In this case, Open edX was a more natural content model and a more reproducible, container-based deployment for a project focused on a small number of carefully sequenced pre-laboratory modules combining videos, embedded interactive resources, and formative quizzes.

Open edX is consequently well suited to online and MOOC settings, and most reported deployments occur in exactly those settings. Comparatively little has been documented about lightweight self-hosted deployments intended specifically to support in-person laboratory teaching in resource-limited universities, which is the gap addressed here.

2.2. Integration of Third-Party Interactive Tools into the LMS

One of the recurrent engineering challenges in LMS-based instruction is the integration of external interactive tools, e.g. serious games, simulation or video platforms, into the course environment [8]. There are two main approaches: The simplest is to embed the external resource directly using an HTML iframe. This will display the tool within the course page, but it will be separate from the LMS: no identity, context or activity data will be passed between the two systems. Another approach relies on the Learning Tools Interoperability (LTI) standard maintained by 1EdTech (formerly IMS Global) [7].

LTI is a secure protocol for an LMS and an external tool to authenticate users and exchange data, including returning grades and completion status to the LMS gradebook. LTI has much richer integration, but it requires the external tool to support the standard, and both ends to be correctly configured. Many design-oriented or lightweight authoring tools used to build serious games do not expose an LTI interface, which forces practitioners to rely on iframe embedding and to capture engagement by other means. This trade-off is directly relevant to the present work and is revisited in Section 8.

2.3. Learning Analytics Pipelines in Open edX

Learning analytics refers to the measurement, collection, analysis, and reporting of data about learners in order to understand and improve learning [9]. LMS platforms generate large volumes of trace data, which can be used to predict academic outcomes [10, 13], but this raw data is rarely usable as-is: it must be extracted, cleaned, and transformed before it can yield meaningful indicators of engagement or performance.

This processing gap has been highlighted repeatedly in the Open edX context. Ruipérez-Valiente and colleagues, for instance, showed that the native analytics support of Open edX is limited and developed dedicated tools to extract and process platform data into actionable engagement and performance indicators [11, 12].

Open edX exposes learner data primarily through the instructor dashboard, from which grade and activity reports can be downloaded as CSV files for further analysis. Most existing work in this area targets large-scale MOOC environments; the construction of a simple, reproducible engagement pipeline for a small, self-hosted hybrid deployment, as described in this paper, has received far less attention.

2.4. Positioning of the Present Work

Table 1 positions the present deployment against the two reference points identified above. The novelty claimed here is not a new LMS, a new algorithm or a modification of Open edX itself.

It is an integrated engineering result: the combination of a minimal containerized Open edX footprint sized for a single low-cost server, a documented iframe-based integration path for authoring tools that do not expose LTI, and a transparent four-stage analytics pipeline requiring no additional analytics infrastructure - validated together in a live teaching setting and characterised quantitatively, including its saturation point and the statistical limits of the engagement signal it produces.

Table 1. Positioning of the present deployment relative to existing approaches

Dimension	Institution-wide Moodle	Reported Open edX / MOOC deployments	Present work
Target scale	Institution-wide, thousands of users	Thousands to hundreds of thousands	Single cohort, tens of learners
Infrastructure	Institutional servers or managed hosting	Clustered, multi-node	Single 4-vCPU / 4 GB VPS
Deployment path	Plugin-based, manual	Kubernetes or distributed Tutor	Single-host Tutor, 11 containers
Analytics	Plugin reports	Dedicated tooling (e.g. ANALYSE [11, 12])	Four-stage CSV pipeline, no extra infrastructure
External tool integration	LTI or plugin	LTI	Documented iframe path with stated trade-offs
Primary purpose	General course administration	Online or MOOC delivery	Preparation for in-person laboratory sessions
Performance characterisation	Rarely reported	Reported at cluster scale	Measured saturation point on one host (Section 7.1)

Table 2. Mapping of key requirements to design decisions

Requirement	Design decision	Rationale
Zero licensing cost	Open edX + Tutor, fully open-source stack	No budget for commercial platforms
Modest infrastructure	Single Contabo VPS, containerized services	Reproducible on low-cost hardware
Short development window	Iterative SAM-style development	Rapid prototyping with instructor feedback
Limited student time	Video capsules of 5–10 minutes	Needs analysis: insufficient preparation time
Activity tracking	Native auto-graded quizzes + CSV export	Provides a machine-readable engagement signal
Secure access	HTTPS via Cloudflare (DNS + TLS)	Protects learner access, simplifies admin
Controlled cohort	Pre-provisioned accounts, no self-registration	Restricts access to enrolled students

3. System Requirements and Design Goals

The design of the system was guided by a prior needs analysis carried out with first-year biology students at the same faculty. That survey identified insufficient preparation time as a primary obstacle to laboratory success and showed strong student interest in short digital pre-laboratory resources. These findings translated directly into the functional and non-functional requirements of the system.

Functionally, the system had to host short instructional videos, embed an interactive serious game, offer formative quizzes with automatic grading, show a threshold for passing the course and track learner activity for further analysis.

The needs analysis also informed a particular design choice. Given that students reported limited time, the instructional video capsules were kept short (five to ten minutes each), in line with multimedia learning principles that recommend short, segmented content [14].

On the non-functional side, the requirements reflected the constraints of the institution. The system had to be built primarily with free and open-source tools, deployable on a modest server, maintainable by a small team, and operational within a short development window.

It also had to remain accessible on both mobile devices and personal computers under limited bandwidth conditions. These goals shaped every subsequent architectural choice described below. Table 2 maps the main requirements to the corresponding design decisions and their rationale.

4. System Architecture

The system was designed as a layered architecture, from the hosting infrastructure up to the learning analytics data layer. Figure 1 gives an overview of the four layers and of how external resources are embedded inside the platform. Each layer is described in the subsections below.

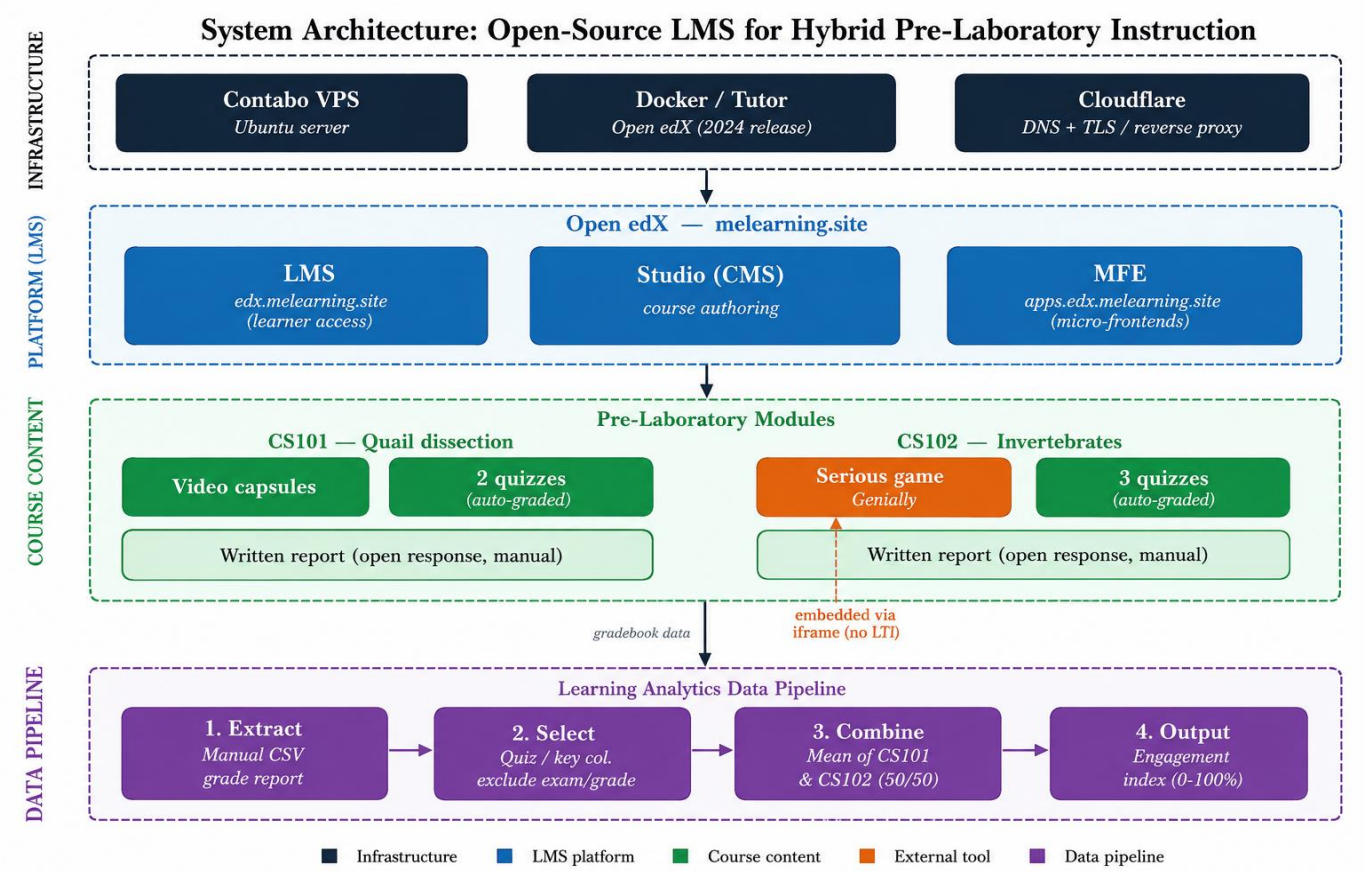


Fig. 1 Layered architecture of the deployed system : infrastructure (Contabo VPS, Docker/Tutor, Cloudflare), Open edX platform (LMS, Studio, micro-frontends), pre-laboratory course content (CS101, CS102) with the Genially serious game embedded

4.1. Infrastructure Layer

The platform was deployed on a virtual private server hosted by Contabo, provisioned with four virtual CPU cores, 4 GB of RAM and 200 GB of SSD storage, running Ubuntu 24.04.2 LTS (Noble Numbat).

The learning environment was based on Open edX and deployed using Tutor, the official Docker-based distribution of the platform; every service therefore ran as an isolated Docker container, which ensured portability, reproducibility, and ease of maintenance.

The deployment comprised eleven containers organised into three groups (Figure 2). The application services comprised the Open edX LMS and its Celery worker (the learner-facing platform) and Studio (CMS) with its own worker (the authoring platform).

The data and infrastructure services comprised a MySQL 8.4 relational database, a MongoDB 7.0 document database, a Redis 7.2 cache and message broker, a Meilisearch 1.8 search engine, and an Exim SMTP relay for transactional email. The frontend and gateway layer comprised the Open edX Micro-Frontend (MFE) services and a Caddy web server acting as a reverse proxy.

Domain name resolution and SSL/TLS certificate provisioning were handled through Cloudflare, which provided secure HTTPS access and simplified domain administration. Orchestrating all of these components through Tutor kept application services, databases, caching, search, and frontend cleanly separated while preserving a single, reproducible deployment that can be redeployed on comparable hardware with minimal effort.

4.2. Platform Layer

Courses were authored and maintained in Studio, the course-authoring environment of Open edX. The platform exposes a learner-facing micro-frontend, where students access their enrolled courses, and Studio, where the teaching team built and edited the course structure. Rather than relying on open self-registration, student accounts were provisioned in advance. Each student first completed a short form collecting first name, last name, and email address; this list was then uploaded to the platform in bulk so that each student only had to activate a pre-created account. This approach kept the cohort under administrative control and simplified onboarding. The course passing threshold of 76%, displayed to learners on the dashboard, indicated the score required to validate each module (Figure 3).

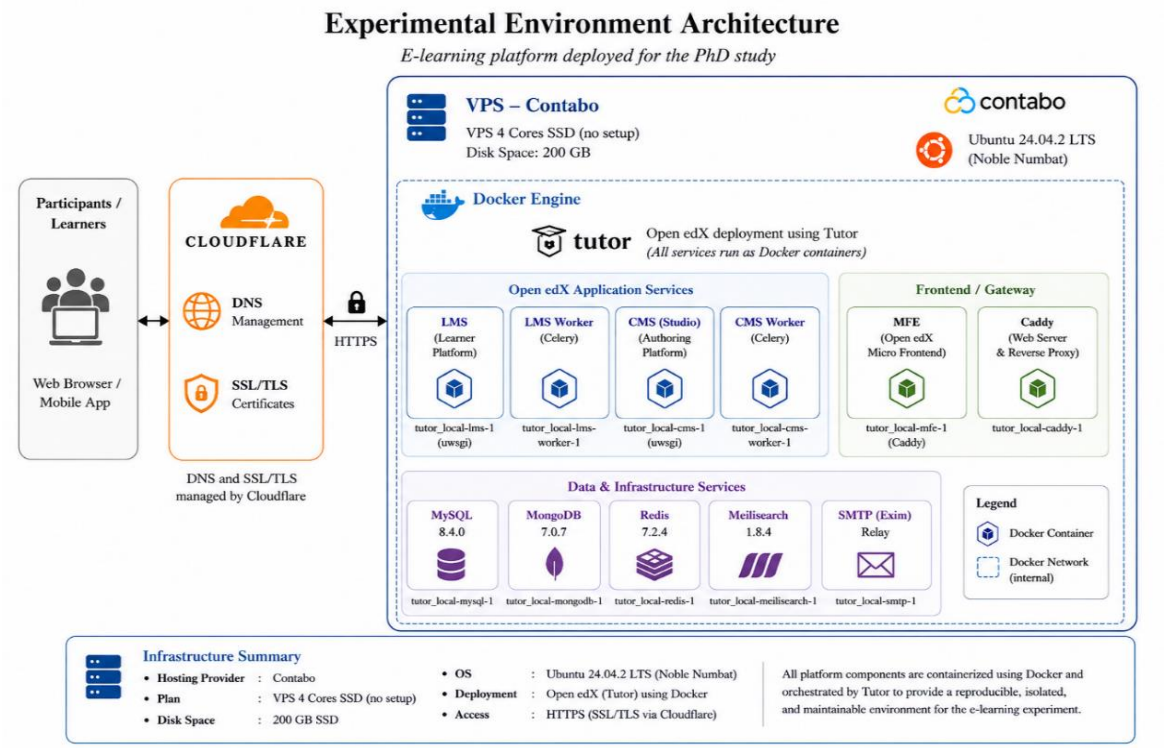


Fig. 1 Detailed infrastructure of the deployed environment: a single Contabo VPS running Ubuntu 24.04 LTS, with all Open edX services orchestrated as Docker containers through Tutor (application services, data and infrastructure services, and frontend/gateway), and DNS and SSL/TLS managed through Cloudflare.

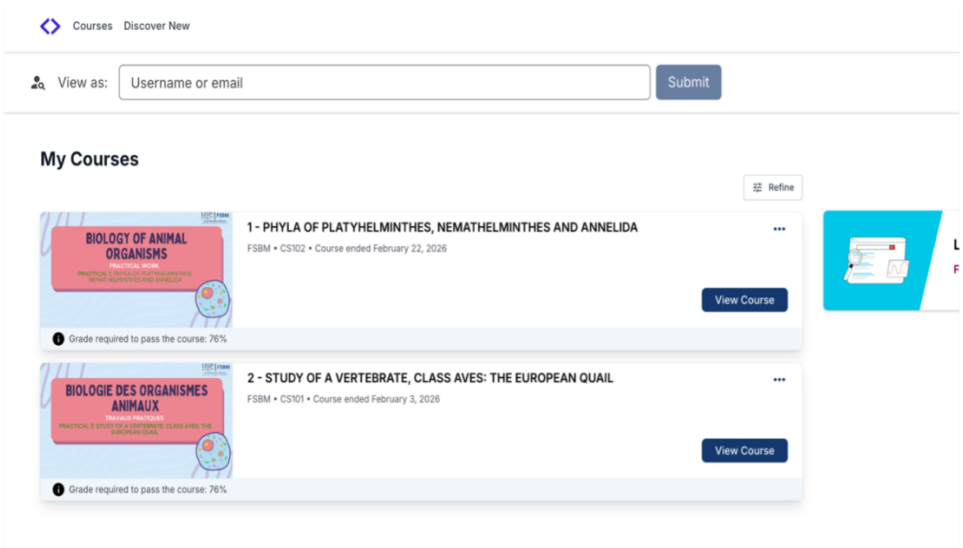


Fig. 2 Learner dashboard showing the two pre-laboratory modules (CS102 and CS101) and the 76% passing threshold displayed for each course

4.3. Course Content Layer

Two pre-laboratory modules were deployed, each combining several types of resources. Module CS102 (Invertebrates: Platyhelminthes, Nemathelminthes, and Annelida) is organized in two parts, covering the organization of Metazoa and the study of Taenia together with microscopic and macroscopic observations. Module CS101 (Vertebrate dissection: the European quail, Coturnix) is organized in three parts: an introduction to vertebrates, a step-by-step dissection covering the circulatory, respiratory, digestive, and urogenital systems, and a final synthesis and evaluation stage. Within these modules, the content types include explanatory and practical videos, formative quizzes, interactive activities such as drag-and-drop exercises, and a short written report completed at the end of each course.

4.4. External Content Integration

Two clarifications are required regarding the open-source character of the deployment. The platform stack (Open edX, Tutor, Docker, MySQL, MongoDB, Redis, Meilisearch, Caddy and Ubuntu) is entirely free and open-source, and this is the component that constitutes the reproducible blueprint. The authoring of one content artefact, the serious game, relied on Genially, a proprietary hosted service used under its free tier; Vimeo, used for video hosting, is likewise a proprietary service. The claim advanced in this paper is therefore that the infrastructure and platform layers carry no licensing cost and can be reproduced without any proprietary dependency, not that every content artefact was produced with open-source tooling. Section 8 identifies open-source substitutes for both external services.

5. Implementation

The system was implemented by a small team following an iterative development process. Rather than specifying the full content in advance, the team refined it progressively through successive validation sessions with the instructor responsible for the practical work. Across roughly seven such sessions, the course plan, the serious game, the quizzes, and the supporting resources were reviewed and adjusted until they matched the objectives of each laboratory session. This iterative, feedback-driven approach is consistent with the Successive Approximation Model of instructional design [15, 16].

Each validation cycle targeted a specific deliverable. Early sessions fixed the overall structure of the two modules and the mapping between digital resources and the competencies assessed in the laboratory. Subsequent sessions reviewed the storyboard and filming plan for the dissection videos, validated the biological accuracy of the recorded footage, and refined the serious game so that its difficulty and terminology matched the first-year curriculum. The final sessions focused on the formative quizzes: the wording of items, the correctness of answer keys, and the alignment of each quiz with the content of its module. This staged validation ensured that every resource released to students had been checked for both technical functioning and disciplinary correctness before deployment.

Content production combined several activities. The anatomy videos for the quail dissection module were filmed during dedicated recording sessions, carried out in collaboration with an audiovisual team at the faculty; keeping the capsules short and focused is consistent with evidence on the role of video length in STEM instruction. The serious game and the remaining digital resources were created separately and then assembled inside the platform. The instructional video capsules were kept to five to ten minutes, in line with the requirements derived from the needs analysis. The whole environment was deployed in approximately six weeks using standalone, free, and open-source tools.

Within the platform, each module was structured into parts and units in Studio. Formative quizzes were implemented as native Open edX problems, automatically graded on a three-item scale, and a written report was added at the end of each course as an open-response activity. Because the platform did not automatically grade this open-response task, the written reports were exported and reviewed manually by the responsible instructor, who corrected and scored them using a spreadsheet prepared for that purpose. Quizzes were configured with unrestricted attempts, in line with their formative purpose; the consequences of this configuration for the engagement signal are analysed in Sections 6.2.

6. Learning Analytics Data Pipeline

A central engineering component of the system is the pipeline that transforms raw LMS data into a usable engagement indicator. It comprises four stages: extraction, selection, combination and output.

6.1. Extraction and Selection

In the extraction stage, the grade report of each module was downloaded manually as a CSV file from the Open edX instructor dashboard. In the selection stage, the data were cleaned. The native Quiz (Avg) column, which records the average score across the formative quizzes, was retained; two other columns were discarded.

The final examination column was excluded because the examination was not taken on the platform and therefore returned zero for every student. The native overall Grade column was also discarded, and the exported data provide direct evidence for that decision: in Module CS101, the Grade column returned 0.00 for every student in the cohort, including the fourteen students who had scored 100% on both formative quizzes, whereas in Module CS102, the same column ranged from 0.00 to 0.50. The two modules carried different grading-policy weightings, and in neither case did the column reflect formative activity. Selecting the quiz average, therefore, provided the only internally consistent behavioural signal across both modules.

6.2. Justification of the Engagement Index

The use of formative quiz scores as an engagement measure requires explicit justification, since a quiz score is ordinarily interpreted as an achievement measure rather than a behavioural one. Three considerations support the interpretation adopted here.

First, the quizzes are formative and carry unrestricted attempts. They are not summative assessments and contribute nothing to the students' final marks. Under these conditions, a score reflects primarily whether a student engaged with the preparatory material and persisted through the items, rather than the level of prior knowledge; this is the sense in which

behavioural engagement is defined in the learning-analytics literature [9].

Second, the quizzes are the only instrumented interaction points in the deployment. As explained in Section 4.4, the serious game returns no data through the iframe, and video-viewing events were not collected. The quiz score is therefore the sole machine-readable trace of learner activity available in this architecture - a constraint of the design rather than a preference.

Third, and importantly, the resulting signal is not uniformly graded. As reported in Section 7.3, the CS101 quiz score exhibits a complete ceiling effect: every active student reached 100%. In that module, the measure functions as a binary participation indicator rather than as a graded measure of engagement intensity. Only in CS102 does the score vary continuously. The composite index must therefore be read as a combination of one participation indicator and one graded score, and this paper does not claim that it measures engagement intensity uniformly across both modules.

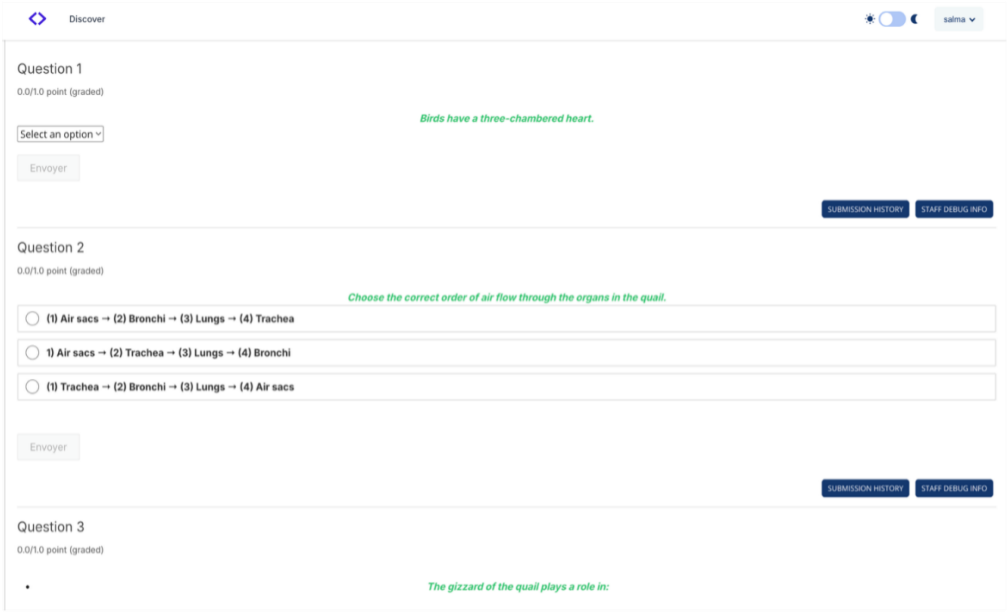


Fig. 4 Native Open edX formative quiz (auto-graded), the source of the engagement signal used by the data pipeline

6.3. Combination and Output

In the combination stage, a single engagement score was computed for each student as the unweighted mean of the two module quiz averages, as expressed in Equation (1):

$$Engagement_i = (Quiz(Avg)_{CS101,i} + Quiz(Avg)_{CS102,i}) / 2 \quad (1)$$

Equal weighting was preferred over quiz-count weighting, which would have given Module CS102 (three quizzes) more influence than Module CS101 (two).

The rationale is pedagogical rather than arithmetic: each module corresponds to one full laboratory session, so each should contribute equally to a measure of preparation effort. This keeps the index interpretable as average preparation across the two sessions rather than as an artefact of the number of quiz items per module.

A worked example illustrates the transformation. In Module CS102, one student obtained 33.3%, 100% and 66.7% across the three formative quizzes, yielding a module average of 66.7%; the same student obtained 100% in Module CS101. The composite index for that student is therefore (100 + 66.7) / 2 = 83.3%.

In the output stage, the index, expressed on a 0–100 scale, provides a per-student measure of activity with the formative content. The pipeline is deliberately simple and fully transparent: each stage relies on a documented export and an explicit, reproducible rule, so that the procedure can be re-applied to other modules or cohorts without specialised analytics infrastructure.

7. Deployment Results and Validation

This section reports operational, performance and engagement results. It does not address the pedagogical effectiveness of the intervention, which is the subject of a separate study.

7.1. Platform Performance and Capacity

The deployment ran continuously for 58 days in production, with all eleven containers reporting uninterrupted uptime over that period and no unplanned restarts. Resource consumption at rest is reported in Table 3.

Table 1. Resource consumption of the containerized stack at rest

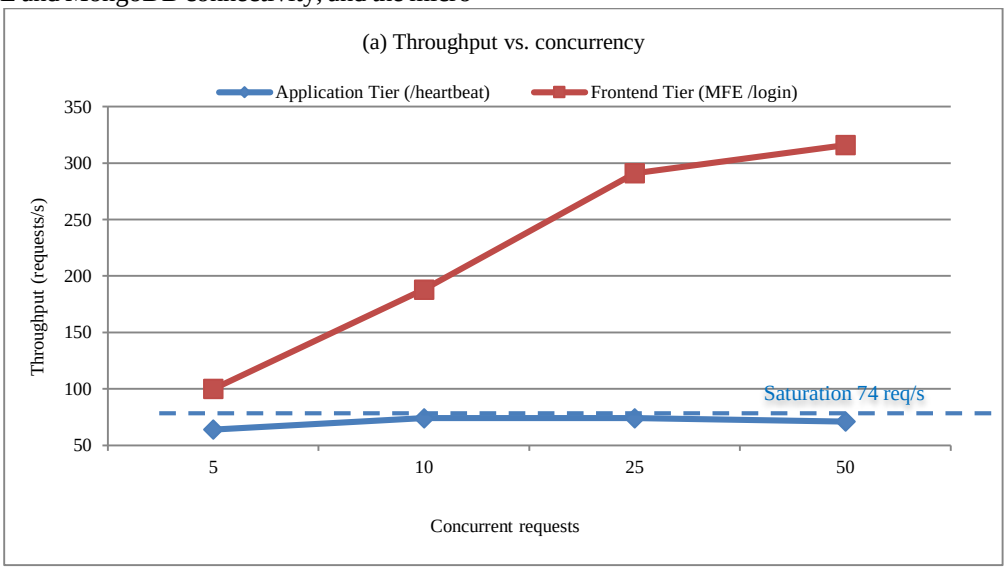
Service group	Container	CPU (%)	Memory (MiB)
Application	LMS	0.38	657.1
Application	CMS (Studio)	0.23	544.5
Application	LMS worker	0.39	392.2
Application	CMS worker	0.29	330.7
Data	MySQL 8.4	0.95	197.4
Data	MongoDB 7.0	0.65	46.6
Data	Meilisearch 1.8	0.13	30.4
Data	Redis 7.2	0.38	5.2
Gateway	Caddy	0.00	39.4
Gateway	MFE	0.00	15.3
Gateway	SMTP (Exim)	0.00	18.6
Total	11 containers	3.40	2,277.4 (2.22 GiB)

At rest, the full stack consumes 2.22 GiB of the 3.73 GiB available, that is 59.5% of physical memory, and 3.40% of aggregate CPU across four cores. Host-level monitoring recorded 1.8 GiB of swap in use, indicating that the 4 GB memory allocation represents the practical lower bound for this configuration rather than a comfortable margin; this observation is developed in Section 8. Response time was measured over 20 successive HTTPS requests to the platform entry point, yielding a mean of 67.3 ms (SD 14.0 ms, median 66.2 ms, 95th percentile 87.9 ms, range 53.0–101.9 ms).

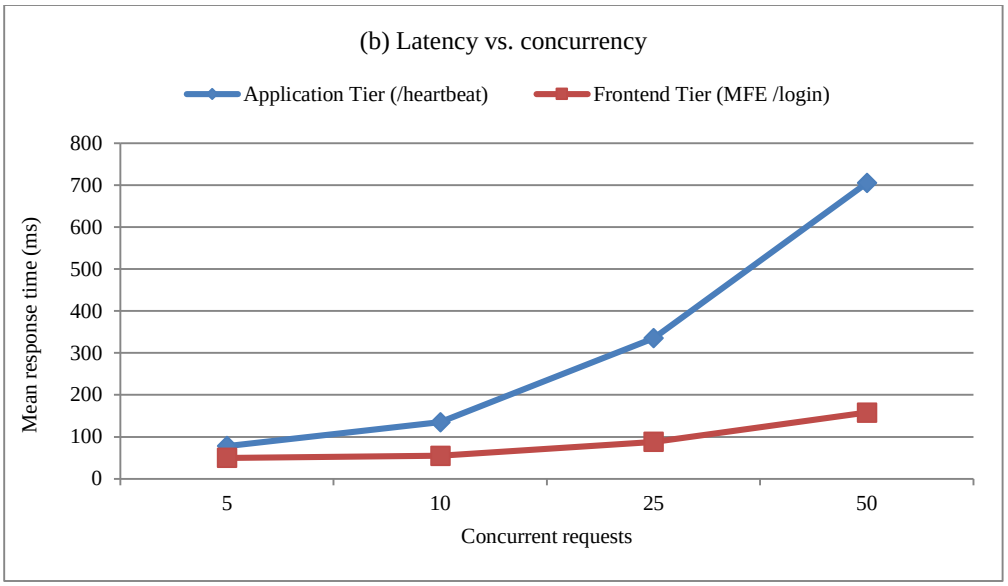
Table 2. Load-test results by concurrency level (200 requests per run)

Endpoint	Concurrency	Throughput (req/s)	Mean latency (ms)	p50	p95	p99	Failed
Application (/heartbeat)	5	64.4	77.7	73	86	122	0
Application (/heartbeat)	10	74.3	134.7	127	147	212	0
Application (/heartbeat)	25	74.6	335.3	312	355	418	0
Application (/heartbeat)	50	71.0	704.6	646	682	756	0
Frontend (MFE /login)	5	101.2	49.4	47	52	58	0
Frontend (MFE /login)	10	187.3	53.4	48	58	73	0
Frontend (MFE /login)	25	290.5	86.1	76	92	113	0
Frontend (MFE /login)	50	315.7	158.4	127	207	208	0

Capacity was characterised with ApacheBench at four concurrency levels against two distinct endpoints: the LMS /heartbeat endpoint, which exercises the full application path including MySQL and MongoDB connectivity, and the micro-frontend login page, which exercises static delivery through Caddy. Each run issued 200 requests. Results appear in Table 4 and Figure 5.



(a)



(b)

Fig. 3 Platform behaviour under increasing concurrency: (a) throughput saturates at approximately 74 requests per second on the application tier while the frontend tier continues to scale; and (b) latency rises linearly on the application tier once saturation is reached

Three findings follow. First, application-tier throughput saturates at approximately 74 requests per second and remains flat from concurrency 10 to 50, while mean latency rises linearly by a factor of 5.2 over the same interval.

This is the signature of a fixed service capacity in which additional concurrent requests queue rather than being served faster; the measurements satisfy Little's law across all four points to within measurement precision, confirming their internal consistency. The saturation point lies between concurrency 5 and 10.

Second, the frontend tier had not saturated at a concurrency of 50, still delivering 315.7 requests per second. The capacity constraint of this deployment therefore resides in the Django application tier, not in static delivery - a specific and actionable architectural finding for anyone reproducing the blueprint.

Third, no request failed across all 1,600 load-test requests, and no non-2xx response was returned at any concurrency level. The system degrades by increasing latency rather than by dropping requests.

An important qualification applies to these figures. The /heartbeat endpoint is lightweight: it verifies database connectivity without rendering course content. Rendering a full course page with XBlock content is more expensive, so 74 requests per second constitutes an upper bound on application-tier throughput rather than a measured page-serving rate. No conversion to a number of "concurrent learners" is offered, as such a figure would require an assumption about inter-request think time that was not measured.

7.2. Account Provisioning and Participation

Accounts were pre-created from uploaded email lists and activated by students before the sessions. Three accounts appearing in the raw exports belonged to the research team and were used for platform testing; these were excluded from all analyses reported below, leaving 17 students in Module CS101 and 22 in Module CS102.

The difference between the two module cohorts arises because five students enrolled in CS102 only; they are consequently absent from the composite index, which requires data from both modules. Participation is defined as at least one attempted quiz item, a definition that includes one CS102 student who attempted the first quiz and scored zero.

Table 3. Operational and participation metrics by module

Metric	CS101 (Quail)	CS102 (Invertebrates)
Students in grade report (after exclusions)	17	22
Students attempting ≥ 1 quiz	14	17
Participation rate	82.4%	77.3%
Quiz average, whole cohort - mean (SD)	82.4% (39.3)	57.1% (38.5)
Quiz average, active students - mean (SD)	100.0% (0.0)	73.9% (25.1)
Quiz average, active students - median	100.0%	77.8%
Quiz average, active students - range	100.0-100.0%	0.0-100.0%

7.3. Engagement Index and the CS101 Ceiling Effect

The composite engagement index was computed for the 17 students present in both modules. It yielded a mean of 76.5% (SD 33.8%), a median of 88.9%, a first quartile of 83.3% and a third quartile of 94.4%, spanning the full 0–100% range. Fourteen of the 17 students reached or exceeded the 76% passing threshold. The distribution is shown in Figure 6.

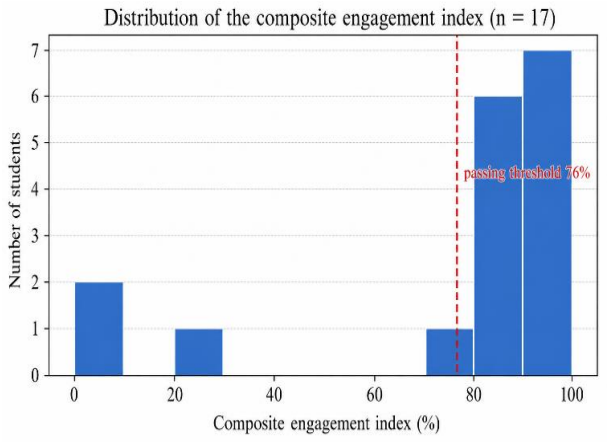


Fig 4. Distribution of the composite engagement index across the 17 students present in both modules, with the 76% passing threshold indicated

The distribution is strongly left-skewed and bimodal: a small group of non-participants at 0%, and a dense cluster between 77% and 100%. This structure is a direct consequence of a ceiling effect in Module CS101, which must be reported explicitly. All fourteen active CS101 students obtained exactly 100% on both quizzes, giving a standard deviation of 0.00. With unrestricted attempts on two three-item formative quizzes, students who engaged at all converged on the maximum score.

Consequently, the CS101 component of the index does not discriminate among active students; it discriminates only between participants and non-participants. All observed variation among active students originates in CS102, where the standard deviation is 25.1. The high inter-module correlation ($r = 0.912$) is likewise an artefact of the CS101 ceiling and is not interpreted here as evidence of a consistent engagement trait.

This paper, therefore, does not claim that the index provides a finely graded measure of engagement intensity in its present form. It claims something narrower and defensible: that the pipeline reliably extracts, cleans and combines platform data into a per-student value spanning the full scale, that the resulting value separates participants from non-participants without exception, and that it captures graded variation wherever the underlying assessment permits it. Section 8 specifies the configuration change required to remove the ceiling in future deployments.

8. Discussion: Reproducibility, Scalability and Limitations

From a reproducibility standpoint, the platform and infrastructure layers are built entirely on a free and open-source stack. Because the deployment is containerized through Tutor and Docker, the same environment can be redeployed on comparable hardware, and the blueprint transfers to other modules or disciplines with limited additional effort. The four-stage data pipeline is equally simple to reproduce, relying only on a manual CSV export and a transparent aggregation rule.

Scalability can now be stated quantitatively rather than asserted. The measurements in Section 7.1 establish that this single-server deployment saturates at approximately 74 requests per second on the application tier, with the bottleneck in the Django application layer rather than in static delivery. The deployment served its cohort without any failed requests, and it degrades gracefully under overload by increasing latency rather than by refusing connections. Scaling beyond this envelope would require distributing application services across additional resources, which Tutor supports, but which was beyond the scope of this work. Memory rather than CPU is the binding constraint on the present host: aggregate CPU usage at rest was 3.40%, while 2.22 GiB of 3.73 GiB of physical memory were committed, and 1.8 GiB of swap were in use. A deployment reproducing this blueprint should therefore provision at least 8 GB of RAM to obtain a comfortable operating margin.

Several limitations should be acknowledged.

Instrumentation: Because the serious game is embedded through an iframe without an LTI bridge, no fine-grained interaction data could be captured from the game itself, and video-viewing events were not collected. The engagement signal rests on quiz data alone.

Measurement ceiling: As reported in Section 7.3, unrestricted quiz attempts produced a complete ceiling effect in Module CS101, reducing that component of the index to a participation indicator. Future deployments should either limit attempts on at least one instrument, or record first-attempt scores alongside best-attempt scores, or complement quiz data with time-on-task traces, in order to recover graded variation.

Sample size: The analysis covers 17 students in CS101 and 22 in CS102, with 17 present in both modules. These numbers support descriptive statistics and operational validation, but are too small for inferential testing, and no significance claims are made. The five students enrolled in CS102 are absent from the composite index.

Manual stages: Extraction of engagement data was performed through manual CSV export, which is not

automated, is prone to human error and does not operate in real time. Written reports were graded manually, introducing a further bottleneck.

Proprietary dependencies: Two external services are proprietary: Genially for the serious game and Vimeo for video hosting. Neither is required by the architecture. H5P provides an open-source alternative for interactive content and integrates with Open edX through an XBlock, and PeerTube offers a self-hostable video alternative; both would remove the remaining proprietary dependencies at the cost of additional deployment effort.

Security and privacy: The deployment implements several protective measures. All traffic is served over HTTPS, with TLS terminated at Cloudflare and certificates managed automatically. Self-registration is disabled, and accounts are pre-provisioned from a controlled list, so platform access is restricted to enrolled students. Open edX role-based access control separates learner, staff and administrator privileges, and Studio authoring access is limited to the teaching team. Data at rest resides on a single server under institutional control, with no transfer to third-party analytics services. Personal data collected is limited to first name, last name and email address-the minimum required for account provisioning-and grade exports are handled by the responsible instructor.

Two weaknesses remain: First, embedding external content through an iframe means learner browsers contact Genially and Vimeo directly so that those providers may observe access patterns; this cannot be prevented without self-hosting the content. Second, the CSV export workflow places files containing student identifiers and scores on staff workstations, where their protection depends on local practice rather than on platform controls. A production deployment should add automated encrypted backups, formal retention limits, pseudonymisation of exported identifiers, and a documented processing basis under Moroccan Law 09-08 on the protection of personal data.

Future work should address these points by adopting LTI-based integration, automating extraction through the Open edX API or an xAPI/Caliper layer, and providing the instructor with a real-time engagement dashboard, since timely engagement feedback is known to support online learning [17, 18].

9. Conclusion

This paper reports the design and deployment of an open-source LMS architecture for hybrid pre-laboratory instruction in a resource-constrained university. The contribution is an engineering one: a containerized, low-cost stack based on Tutor and Open edX, a documented integration path for external interactive resources that do not expose an LTI interface, and a lightweight learning-analytics pipeline that converts raw gradebook data into a per-student engagement index. The complete environment was designed and deployed in approximately six weeks by a small team using free and open-source tools for the platform and infrastructure layers.

The deployment has been characterised quantitatively rather than described qualitatively. It ran for 58 continuous days without unplanned restart, responded in 67.3 ms on average at rest, sustained approximately 74 requests per second on the application tier, and returned no failed requests across 1,600 load-test requests spanning concurrency levels from 5 to 50. The capacity constraint was localised to the Django application tier, and memory rather than CPU was identified as the binding resource on a 4 GB host. The analytics pipeline produced a usable per-student index across the deployed cohort, and its principal measurement limitation - a ceiling effect arising from unrestricted quiz attempts - has been identified along with the configuration changes required to remove it.

These results are those of a single deployment serving a small cohort, and they are not presented as evidence of scalability to institutional scale or of pedagogical effectiveness, which is examined in a separate study. Within that scope, the work offers a reproducible blueprint together with a measured account of its capacity envelope, so that practitioners delivering hybrid digital instruction under tight infrastructure and budget constraints can judge in advance whether it fits their setting. Future work will focus on automating the data pipeline, adopting standard tool-integration protocols, replacing the remaining proprietary dependencies, and extending the deployment across additional modules and institutions.

Declarations

CRediT author statement, declaration of competing interests, funding, and data availability statements to be added according to the journal template. Author ORCID identifiers to be completed for all co-authors.

References

[1] Avi Hofstein, and Vincent N. Lunetta, “The Laboratory in Science Education: Foundations for the Twenty-First Century,” *Science Education*, vol. 88, no. 1, pp. 28-54, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[2] Fredrick S. Simasiku et al., “Exploring the Efficacy of Practical Work on Learners’ Academic Achievements in Biology at Senior Secondary Level in Namibia,” *European Journal of Health and Biology Education*, vol. 12, no. 1, pp. 1-9, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[3] Ram Babu Pareek, "An Assessment of Availability and Utilization of Laboratory Facilities for Teaching Science at Secondary Level," *Science Education International*, vol. 30, no. 1, pp. 75-81, 2019. [Google Scholar] [Publisher Link]

[4] Xiaoran Wang et al., "Hybrid Teaching after COVID-19: Advantages, Challenges and Optimization Strategies," *BMC Medical Education*, vol. 24, no. 1, pp. 1-10, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[5] Mohammed Ouadoud, Nouha Rida, and Tarik Chafiq, "Overview of E-Learning Platforms for Teaching and Learning," *International Journal of Recent Contributions from Engineering, Science and IT*, vol. 9, no. 1, pp. 50-70, 2021. [Google Scholar]

[6] A.W. (Tony)Bates, Teaching in a Digital Age: Guidelines for Designing Teaching and Learning, 2nd ed., BCcampus, 2019. [Online]. Available: https://open.umn.edu/opentextbooks/textbooks/221

[7] 1EdTech, Learning Tools Interoperability (LTI), 1EdTech, 2019. [Online]. Available: https://www.1edtech.org/standards/lti

[8] Lea C. Brand, and Andreas Schrader, "Serious Games in Higher Education in the Transforming Process to Education 4.0—Systematized Review," *Education Sciences*, vol. 14, no. 3, pp. 1-12, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[9] Florence Martin, and Doris U. Bolliger, "Engagement Matters: Student Perceptions on the Importance of Engagement Strategies in the Online Learning Environment," *Online Learning*, vol. 22, no. 1, pp. 205-222, 2018. [Google Scholar]

[10] Alberto Rivas et al., "Artificial Neural Network Analysis of the Academic Performance of Students in Virtual Learning Environments," *Neurocomputing*, vol. 423, pp. 713-720, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[11] José A. Ruipérez-Valiente et al., "Scaling to Massiveness with ANALYSE: A Learning Analytics Tool for Open edX," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 6, pp. 909-914, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[12] José A. Ruipérez-Valiente et al., "Evaluation of a Learning Analytics Application for Open edX Platform," *Computer Science and Information Systems*, vol. 14, no. 1, pp. 51-73, 2017. [Google Scholar] [Publisher Link]

[13] Yinghui Sh et al., "College Students' Cognitive Learning Outcomes in Technology-Enabled Active Learning Environments: A Meta-Analysis of the Empirical Literature," *Journal of Educational Computing Research*, vol. 58, no. 4, pp. 791-817, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[14] Richard E. Mayer, and Logan Fiorella, *The Cambridge Handbook of Multimedia Learning*, Cambridge University Press, 2021. [Google Scholar] [Publisher Link]

[15] Michael Allen, and Richard Sites, *Leaving ADDIE for SAM: An Agile Model for Developing the Best Learning Experiences*, ATD Press, 2012. [Google Scholar] [Publisher Link]

[16] Hyojung Jung et al., "Advanced Instructional Design for Successive E-Learning based on the Successive Approximation Model (SAM)," *International Journal on E-Learning*, vol. 18, no. 2, pp. 191-204, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[17] Khe Foon Hew et al., "Meta-Analyses of Flipped Classroom Studies: A Review of Methodology," *Educational Research Review*, vol. 33, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[18] Emtinan Alqurashi, "Predicting Student Satisfaction and Perceived Learning within Online Learning Environments," *Distance Education*, vol. 40, no. 1, pp. 133-148, 2019. [CrossRef] [Google Scholar] [Publisher Link]