Shayne Kandagor

Practical Framework for Migrating JSON Format Course Content from WordPress CMS to Moodle LMS

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Bachelor's Thesis

28 May 2025

Abstract

Author:	Shayne Kandagor
Title:	Practical Framework for Migrating JSON Format Course Content from WordPress CMS to Moodle LMS
Number of Pages:	46 pages
Date:	28 May 2025
Degree:	Bachelor of Engineering
Degree Programme:	Information Technology
Professional Major:	Mobile Solutions
Supervisors:	Toni Spännäri, Senior Lecturer

The objective of this study was to provide structured guidelines for the process of migrating JSON format course content from WordPress CMS to Moodle LMS. This project was carried out for an external customer whose goal was to find out if it was possible to migrate the course to the Moodle platform. Therefore, the case company cooperated with Metropolia University of Applied Sciences (UAS) in pursuit of an affordable and accessible learning platform to which the course could be migrated.

The core of the work involved delving into the evolution of digital learning platforms in the post-COVID-19 period. The work also gives a comparative analysis of CMSs and LMSs. Moreover, it explores LMSs such as Moodle LMS, discussing Moodle's architecture and the Moodle backup file structure. This study is based on the utilisation of AI custom automation scripts to extract course content from JSON files and convert it to Moodle XML files and the use of local Moodle and Metropolia Moodle as the main testing environments.

The outcome of this study is that the course content was successfully migrated to Moodle, except the media files. The implementation of migrating media files was incomplete due to time limitations. Therefore, the media files were transferred manually to the Moodle platform for testing purposes.

The purpose of this study was to help the case company transfer course content to Moodle LMS, an affordable and accessible platform. Another goal was to provide universities, researchers, and organisations with a structured approach for understanding a course migration framework. In the future, the plan is to improve the migration framework to fully automate the whole migration process, including multimedia files, as well as to support multiple courses and bulk course content.

Keywords: Digital Learning Platforms, Moodle, Learning Management Systems, Open Source LMS, JSON, Migration Framework

---

Contents

List of Abbreviations

AI:	Artificial Intelligence
ASCII:	American Standard Code for Information Interchange
COVID-19:	Coronavirus disease 2019
CMS:	Content Management System. Software that helps users create, manage and modify content on a website without the need for technical knowledge.
e-learning:	Electronic learning
HTTP:	Hypertext Transfer Protocol. This is an application layer protocol for transmitting hypermedia documents, such as HTML.
IT:	Information Technology
JSON:	JavaScript Object Notation. A popular, language-independent, standard format for storing and exchanging data.
LAMP:	Linux Apache MySQL PHP
LCMS:	Learning Content Management System
LMS:	Learning Management System. Software application for the administration, documentation, tracking, reporting, and delivery of educational courses, training programs, materials or learning and development programs.
.MBZ:	Moodle backup zip. Compressed archive of a Moodle course that can be used to restore a course within Moodle.

Moodle:	Modular Object-Oriented Dynamic Learning Environment
PHP:	Hypertext Preprocessor
SQTL:	Software Quality and Teaching-Learning Tools
SCORM:	Shareable Content Object Reference Model
UAS:	University of Applied Sciences
WAMP:	Windows Apache MySQL PHP
XML:	Extensible Markup Language

1 Introduction

The World Economic Forum, a leading global online platform, reports that recently there has been a vast increase in the number of students accessing online learning platforms compared to the pre-Corona Virus Disease 2019 (COVID-19) period. Today's constantly evolving world of work has been greatly affected by COVID-19. It has influenced the transition from physical work to remote work. This has led to a rapid rise in the number of people interested in digital learning for the improvement of work skills. As can be seen in Figure 1 below, according to the 2021 Impact Report, there has been a gradual increase in the number of registered learners and enrolments on Coursera, an online learning platform. [1.]

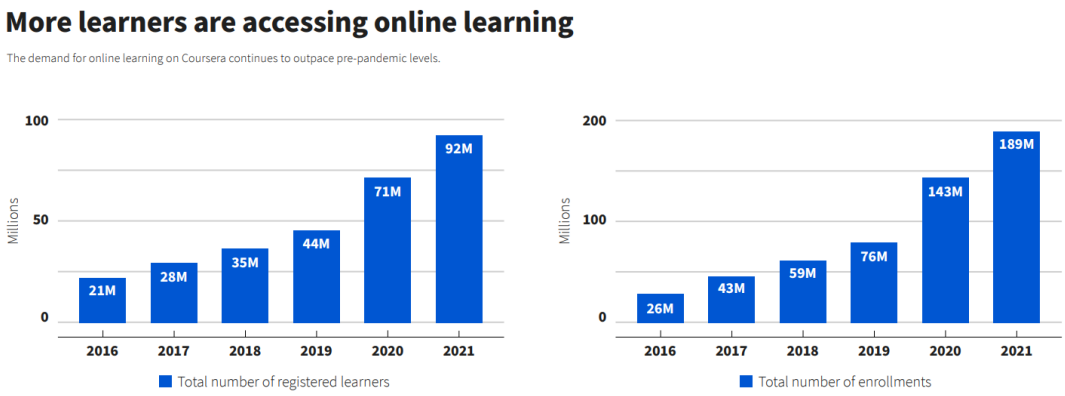

Figure 1. Upward trend in online learning in the post-COVID-19 pandemic period [1].

Figure 1 illustrates that 21 million students registered for Coursera's online courses in 2016. This number rose by around 7 million over the following two years. However, the shift to remote work with the pandemic tripled the increase in the number of new registrations, resulting in 71 million in 2020 and 92 million in 2021. Similarly, course enrolments for online learning also increased in numbers. In 2020, the number of enrolments was twice as high as in the previous year and increased by 32% in the following year, with 189 million enrolments. In summary, this rise reflects the global acceptance of online teaching and learning. [1.]

Over the past years, remote learning and teaching have evolved gradually into digital learning. In the past, learning and teaching materials between students and teachers were exchanged through surface mail, also known as snail mail. Additionally, remote learning materials later continued developing technologically to broadcasting educational programmes on radio and television. [2.]

In the early 1960s, computer-based remote learning training experiments were carried out in the United States, Canada and Europe, making computers part of remote learning and teaching. Thereafter, online courses through the Internet in universities and colleges became popular due to the enhancement of the Internet. In turn, this led to the origin of electronic learning (e-learning) or digital learning. This has become so popular in educational institutions that learning materials are distributed, and programmes are taught through the Internet today. [2.]

In the mid-1990s, schools and universities started developing software systems for managing digital learning platforms. Education systems improved digital learning strategies by introducing Learning Management Systems (LMSs) and Content Management Systems (CMSs) platforms. Digital learning has had a big impact on the development of education. This enables the improvement of the learning abilities of a student and connects students and teachers from various locations and domains to learn the same concepts together. [2, p. 2; 3, p. 1.]

This final year project was carried out for an external customer whose goal was to find out if it is possible to migrate JSON format course content from the WordPress CMS to the Moodle LMS platform. The objective of this study was to provide step-by-step guidelines for the course content migration process. The core of this study consists of the development of a migration framework for JSON format course content migration. The study explores the understanding of digital learning platforms. It also involves discussing the development of custom automation scripts for converting JSON format course content to Moodle XML format. Finally, the study analyses the testing and validation of the course

outcome in both local Moodle and Metropolia Moodle testing environments. The external customer cooperated with Metropolia UAS on a quest for an affordable and highly accessible learning platform, Moodle, to which course content could be migrated. The goal of this study was to offer universities, researchers and organisations a practical and structured approach to course migration as well as help the case company in transferring course content to Moodle LMS.

The overview of the chapters in this thesis covers the structured process of migrating JSON format course content from the WordPress CMS platform to the Moodle LMS platform. Chapter 2 covers the theoretical background and discusses digital learning platforms. It also gives a comparative analysis of CMSs and LMSs. Additionally, it provides the reader with a deep understanding of Moodle as one of the most popular LMS platforms used today. In addition, Moodle's architecture and Moodle's backup file course structure are presented.

In addition, Chapter 3 explores the starting point of the project and the project background. Chapter 4 presents the proposed migration framework. Chapter 5 discusses the development and implementation of the migration framework. Chapter 6 discusses the outcomes of the testing and validation of the courses in the testing environments and provides an analysis of the test outcomes. Finally, the conclusion outlines the key findings and challenges encountered in the project. In addition, recommendations for future work are presented.

2 Digital Learning Platforms

In today's world, the global COVID- 19 pandemic, among others, has been one of the main reasons for the rapid increase of digital learning and training at school and work [3]. The term digital or online learning supports learning through connecting learners to educational materials via the Internet [5]. This kind of learning requires the integration of a student, a curriculum and an Internet connection [5]. Digital learning is also a practice based on the outcome of a learning activity that results in improvements in a learner's learning abilities. This can be achieved through developing digital learning platforms that are

innovative and collaborative software systems for enhancing strategies in education systems. These platforms are vital for learning and evaluation purposes and have evolved into better platforms to enable learners to explore interesting subjects and approach digital learning methods more extensively. [4.]

In the past decade, online courses and opportunities to learn have gradually and steadily increased the popularity of online learning. The benefits of using online platforms offer the possibility to juggle between tasks and commitments such as work, family and community. Furthermore, online platforms may save time and money as they enable studying at home and not having to commute to school, which means that, in some situations, courses and programmes can be completed sooner. [5.]

2.1 Overview of Digital Learning Platforms

Research evidence suggests that the development of technology and the Internet has impacted teaching and learning in many ways over time. Traditionally, teachers and students physically met at an educational institution such as a school, university or college to exchange information on various subjects. In contrast, advancements in technology have made immense changes to traditional teaching and learning methods, as a new model known as Electronic Learning (e-learning) or online learning has been introduced. [2, pp. 1-2.]

Consequently, the recent COVID-19 pandemic has caused digital learning platforms to evolve tremendously. Throughout the world, many educationalists have gained knowledge of the needs and advantages of using digital learning platforms. This has enabled people from various locations and domains to unite and learn common concepts together, and it has allowed blended, personalised and flipped learning for learners' interactions. These platforms have made it easy for students to deepen their knowledge and hold discussions with their colleagues about interesting subjects, gain a deeper understanding and

insights, and find solutions to problems facing the world today. Due to these positive impacts of e-learning, many people have taken the initiative to continue using it either entirely or along with face-to-face interactions through a blended approach. [2, p.1; 3, p.2.]

2.2 Comparative Analysis of Digital Learning Platforms

Digital learning platforms consist of two main platforms, which are a Learning Management System (LMS) and a Content Management System (CMS). These software platforms are used to create interactive, efficient and more engaging learning environments. They were mainly developed to automate educational activities through software systems to manage digital learning platforms for schools and universities. [2.]

As stated earlier, Learning Management Systems (LMSs) were introduced in the mid-1990s. First, this was referred to as software that allowed users to monitor students' records, attendance, tests, grades and transactions. A well-known and successful example of an LMS is Blackboard which is used by many institutions in over 100 countries across the world. This is because it integrates well with various IT aspects such as education, mobile communication, commerce, software and other similar services. Additionally, Moodle, Canvas and Google Classroom are examples of the most used LMSs in higher education institutions. [2, p. 2.]

Content Management Systems (CMSs) are software similar to Learning Management Systems (LMSs). A CMS is a software system used to create, edit or modify, collaborate on, and publish digital learning content. The most popular CMSs used today are Drupal, WordPress and Joomla. Today, most higher education institutions prefer LMSs to CMSs. This is because LMSs have more features and perform more functions than CMSs. [2, p.2.] In contrast, there are many similar features that both LMSs and CMSs share, such as the following:

- Both enable creating, editing, deleting, managing and delivering content online.

- Both can support multiple users and permissions.

- Both are highly scalable and can have various types of users.

- Both enable creating web portals to sign up and access content by users. [2, p.2.]

As noted above, LMSs and CMSs are quite similar and are often assumed to be the same by most people. However, there are also differences between CMSs and LMSs as they also contain different features according to different aspects. The table below presents the differences between an LMS and a CMS.

Table 1. Differences between a CMS and an LMS [2, p. 2].

LMS	**CMS**
It is used to create, manage, administer and track online courses and assessments.	It is used to create, organise and publish content such as personal blogs or website content for businesses.
Its primary objective is to offer seamless e-learning experience through personalised learning paths.	It aims to help users collaborate in creating, modifying, publishing and managing any digital content.
It supports learner interactions with elements such as quizzes, gamification, live sessions and social forums.	It serves solely as a back-end content repository for viewing purposes only.
Generally, it involves a learning curve as it offers various features for online learning and training administration.	It is simpler and faster to deploy due to design implementations.

LMS	CMS
It generates detailed reports on learner activities, courses and instructors.	It is limited to simple activity reports.
End users include learners, teachers or trainers, course administrators and managers.	End users are content managers, content creators and users.

From the above table, it can be concluded that the main function of a CMS can be integrated into an LMS. It shows that a CMS can have an insignificant role in an LMS. As a result, the combination of both is known as a Learning Content Management System (LCMS). Therefore, the main function of an LCMS is learning and testing uploaded materials through a management system. These materials include content uploaded by teachers, such as videos, class notes and reference materials as well as students' content such as assignments, assessments and final test papers. [2, p. 2.]

2.3 Learning Management Systems (LMSs)

The past decade has seen the rapid development of learning management systems through various significant stages. Today, LMSs have emerged as powerful web-based software in the online learning environment. LMSs have been described as the main medium of communication through the Internet in systems of learning management. According to Roza Micha, the need for "class web presences" led to the emergence of LMSs in the late 1990s, where the first generation of LMS software was established. Thereafter, the evolution of LMSs continued, leading to the creation of Blackboard, Desire2Learn, Moodle and later Canvas. [6; 11.]

According to Foreman, Learning Management Systems can be perceived as the solution for all learning challenges in an organisation. However, LMSs can be useful for organisations in a wide range, mostly for organisations that offer training and educational programmes. Therefore, LMSs are broadly referred to

as software applications, often accessible through a web browser for numerous users. They enable many organisations to manage training events, self-paced courses and blended-learning programmes through automation. As a result, it is beneficial that LMSs replace thorough and unaffordable manual work, save time as well as help in organising and managing learning content, data and learners. [7.]

In this study, the original idea of classifying LMS types into three categories, as seen in Figure 2, was derived from Helen Colman's blog, since there are many types of LMSs. Research showed that in different publications, two or three types of LMSs are discussed. The types of LMSs are selected according to the system requirements, needs and finances of a company or educational institution. As the figure below shows, LMSs are categorised into various types depending on the requirements of the management system and the company's needs. [3; 8.]

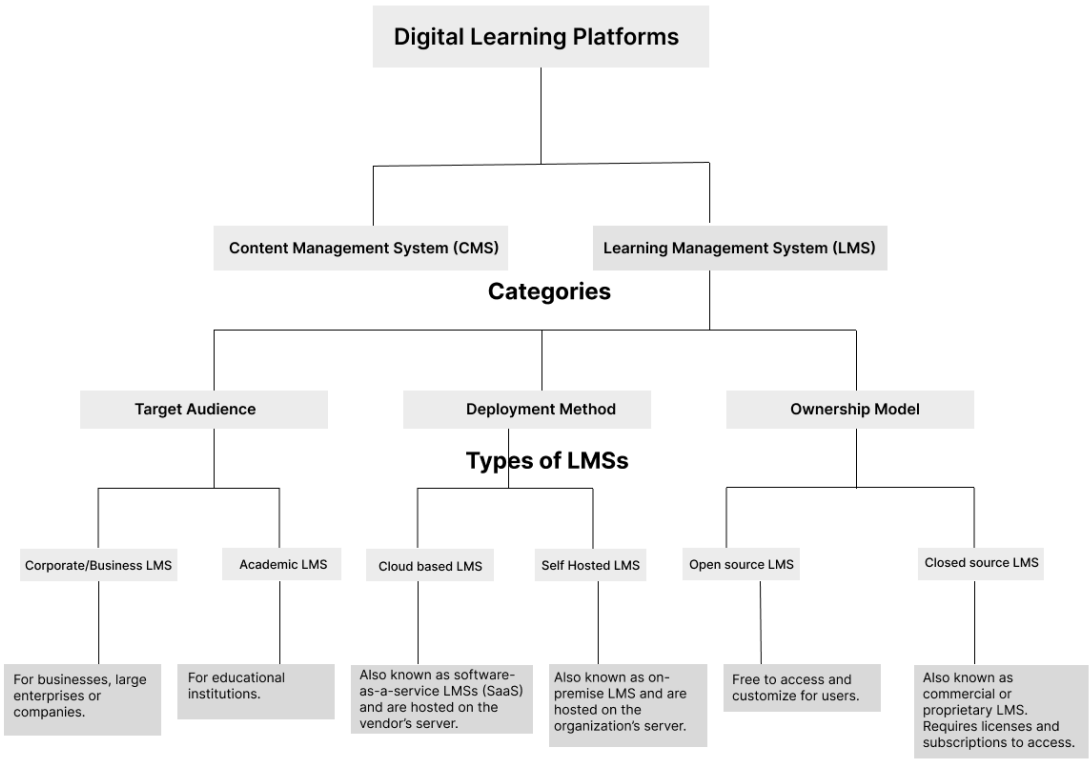

Figure 2. Types of LMSs according to system needs categories.

In an educational institution, the selected LMS depends on the academic needs and the goals and objectives of the learning process [7]. In this study, the types of LMS platforms are grouped according to the categories shown in the list below:

- Target audience
- Deployment method
- Ownership model

Target Audience

The target audience category has two types of LMSs. These are categorised according to the LMS environment and the type of learners. First, Corporate LMSs are used in businesses and large enterprises by employees and customers. Secondly, Academic LMSs are used in educational institutions by students and teachers. Steven Foreman broke down LMS products into two major categories, which are suited to an organisation's needs and requirements. This is discussed further below. [3; 7.]

- Corporate LMS

  Steven Foreman defines a Corporate LMS as a software application that delivers courses for employees and customers in business organisations. A corporate LMS offers courses that an employee of a company or an organisation might need for a particular role, such as customer service, product knowledge, company policies and procedures, personal development, productivity and other related topics. This kind of LMS is mainly intended for short courses that may take from one hour or less to several days. [7.]

- Academic LMS

  According to Foreman, an Academic LMS is a software system that is mostly used by educational institutions. It is mainly used for developing, managing and delivering learning content. Academic LMSs replace a physical classroom as the learning materials are in an online environment, making collaboration between instructors and students easier. The period of the courses in this kind of platform is often a term or semester, depending on the amount of learning content. Additionally, the instructor usually works with the students through various materials and assignments. [7.]

Deployment Environment

The deployment environment category discusses the LMS software installation environment. The deployment environment chosen depends on the budget, the IT infrastructure and the business plans of the organisation [3]. This category consists of two types, which are cloud-based LMSs, which store their data on the cloud and can be modified through an internet connection, and self-hosted LMSs, which store their data on the organisation's server. Some more details are given below.

- Cloud-based LMS

  Cloud-based LMSs are software system products that are hosted on a software-as-a-service (SaaS) platform without requiring an LMS platform installation. Romina Agaci suggests that a computer, a smartphone or a tablet and an accessible Internet connection are the basic requirements to deliver online education whenever and wherever. A cloud-based LMS is usually installed on the vendor's server and can be accessed from anywhere through an Internet connection because data is stored in the cloud. The management can create, upload and edit data via an internet browser. An example of a cloud-based platform is TalentLMS. [8; 9; 11.]

- Self-hosted or Server-based LMS

  Self-hosted or server-based LMSs are software system products that are installed on an organisation's own server. They store data in the local network. Storing data on the organisation's server ensures the safety of the data and ease of customisation. [3; 9; 11.]

Ownership Model

The types of LMSs in the Ownership Model category were chosen based on the budget they require, such as free or paid access and the needs of an organisation such as IT infrastructure [3]. First, open source LMSs are free to access and commercial LMSs require a fee to be accessed. Further details are discussed below.

- Open Source LMS

  Open source LMSs are software systems that are accessible to users under a public free license. Users can edit, modify and personalise the software source code according to their preferred requirements and the specific needs of the organisation. Most open source LMSs are free; however, some may require a fee for services such as hosting, customisation, additional functionalities, management, maintenance and improvement. An example of a popular open-source LMS platform is Moodle LMS. [9; 10.]

- Commercial or Proprietary or Closed Source LMS

  Commercial or proprietary or closed-source LMSs are software systems that require a costly license for access to use. Customers have the advantage of more support and functionality in commercial LMSs; therefore, expenses increase with the increase in features. Additionally,

negotiations of the product's cost can vary according to the number of participants. The presence of a developed infrastructure, such as equipment of buildings with Internet connection and computers as well as platforms installation on the organisation's servers and computers, is essential in commercial LMSs. An example of a commercial LMS platform is Blackboard Learn. [9; 10.]

2.4 Moodle as an LMS

In 2002, Martin Dougiamas invented the Modular Object-Oriented Dynamic Learning Environment (Moodle). Back then, a Content Management System (CMS) in education was utilised to develop Moodle, creating a non-interactive interface, hence unfulfilling teaching and students' learning needs and objectives. Therefore, decisions to further his education and interests in social constructionism grew, resulting in a collaborative learning environment, Moodle LMS. Over the years, Martin and the development team continued to develop Moodle by introducing new technologies and improving older tools and modules, resulting in an easy, recognisable and dependable Moodle interface worldwide. [14.]

According to Susan Nash, research shows that online learning for new students and instructors has resulted in failure. This is due to complex learning experiences, which are difficult to follow and displayed in a single format, such as text only. This was due to the lack of collaboration, engagement and interaction on online learning platforms. Therefore, over the years, Moodle has undergone a series of versions with extensive updates to create a great user experience for users. It has focused on improving learners' interaction, collaboration and engagement, making it easy to navigate. The introduction of new features, such as an integrated dashboard with a built-in calendar and timeline for tracking all deadlines and important dates in an action-oriented interface, has made Moodle one of the world's most well-known and extensively used learning platforms. [13.]

Introduction to Moodle

Moodle (Modular Object-Oriented Dynamic Learning Environment) is a free, online Learning Management System (LMS) that enables instructors to build personal, confidential platforms consisting of adaptive courses that offer easy access to learning whenever and wherever. In an educational setting, Moodle can fulfil the requirements of a teacher, a student and an administrator since it contains various standard features that are customisable. [15.] Moodle's general features that make it a popular LMS platform are discussed below.

- It has a responsive and easy to use interface for both desktop and mobile applications.
- It contains a tailored course page that showcases past, current and future courses.
- It supports a timeline dashboard that contains deadlines and calendar events.
- It consists of forums, wikis, glossaries and database activities where students can work and learn together.
- It has a user-friendly file management system that allows easy drag and drop of files or uploading files from various platforms.
- It contains a simple text editor.
- Users can receive notifications on new tasks or deadlines and can send and receive confidential messages.
- Teachers and students can track progress and completion of individual course activities or resources. [15.]

The official Moodle literature states that Moodle is an open-source software that is free to use without licensing fees, allowing users to create, edit and modify Moodle for educational and commercial projects. Additionally, Moodle is the most widely used and reliable learning platform worldwide, used by learning institutions and enterprises, whether large or small. There have been over 147,000 registered Moodle sites, more than 50 million courses, and above 400 million users in over 250 countries worldwide. According to the statistics, Spain

has the most registered Moodle sites with over 12,000 sites, followed by the United States with over 11,000 sites and Germany with above 8,000 sites worldwide. Therefore, it supports multilingual abilities, translating Moodle into more than 120 languages, allowing users to centralise their Moodle site easily. [15.]

Furthermore, Moodle LMS is mostly used across Finland in educational institutions and enterprises. As per recent statistics on the Moodle official site, Finland has over 356 registered Moodle sites, with over 290 sites private. Well-known universities such as the University of Helsinki and Metropolia University of Applied Sciences (UAS), among others, utilise Moodle as an e-learning platform. They contain an integrated Moodle LMS as an interactive online workspace for degree students or open university courses. [15; 22; 23.]

Comparison of Moodle LMS with other LMS Platforms

In February 2023, a study by Discover Education was conducted to compare 45 LMSs. The study was based on an evaluation of software quality literature from the past 20 years. The goal of the research was to analyse the best LMSs that can be used in higher education institutions, considering that all LMSs have online access to information. The evaluation procedure and tools used in online education were the Software Quality and Teaching-Learning tools (SQTL) for online educational platforms. [16.]

The SQTL study was chosen for LMSs' evaluation as it compares LMSs used in higher educational institutions that offer online courses and use e-learning platforms. In addition, SQTL is composed of three tools, which are learning abilities, communication and productivity, and it can be used in academic and enterprise LMSs. In the SQTL assessment, the maximum score was 10 points and the minimum was 0. [16.] The factors that influenced SQTL comparison were composed of six criteria that include the following:

1. Interoperability. The ability to communicate on various systems with numerous distinct data types and in various formats. Hence, information can be exchanged in various environments.
2. Accessibility. The capacity of the LMS to be accessible on various hardware devices, for a massive number of users around the world. This is based on three factors, which include:

    - Web Navigator - whether the LMS can be accessed on a website.
    - Compatibility on different hardware devices - whether the LMS is compatible with a PC, smartphone or tablet.
    - Languages - whether the LMS supports English, Chinese, Hindi and Spanish.

3. Productivity Tools. A good LMS should contain good productivity tools to create and edit courses, documents, users and grades.
4. Communication Tools. Good communication tools enable effective communication and collaboration between teachers and students in online educational platforms. The communication tools can be categorised into two groups:

    - Synchronous tools such as video conference, chat, virtual blackboard, web seminar and notes.
    - Asynchronous tools such as forum, email, wiki, comments, glossary, documentation and virtual library.

5. Learning Tools. Learning tools are the most important tools that every LMS should use for educational systems. This is divided into three categories:

    - Tools for enhancing understanding of course content, such as surveys, polls, queries, search functionality, video services, document sharing, calendars, support or help features, diaries and predefined surveys.

- Tools for improving interactive and individualised learning, such as forums, wikis, assignments, chats, gamification, workshops, quizzes, blogs, lessons, glossaries and groups.
- Tools for content management and distribution, such as archives or documents, URLs, external tools, folders and Shareable Content Object Reference Model (SCORM).

6. Security and Certifications. Testing the software's reliability and security is a vital aspect in every LMS. This includes authentication, access verification, integrity control and detection of non-authorised users. [16.]

The graph below shows a comparative analysis of the best LMSs used in higher educational institutions based on SQTL results.

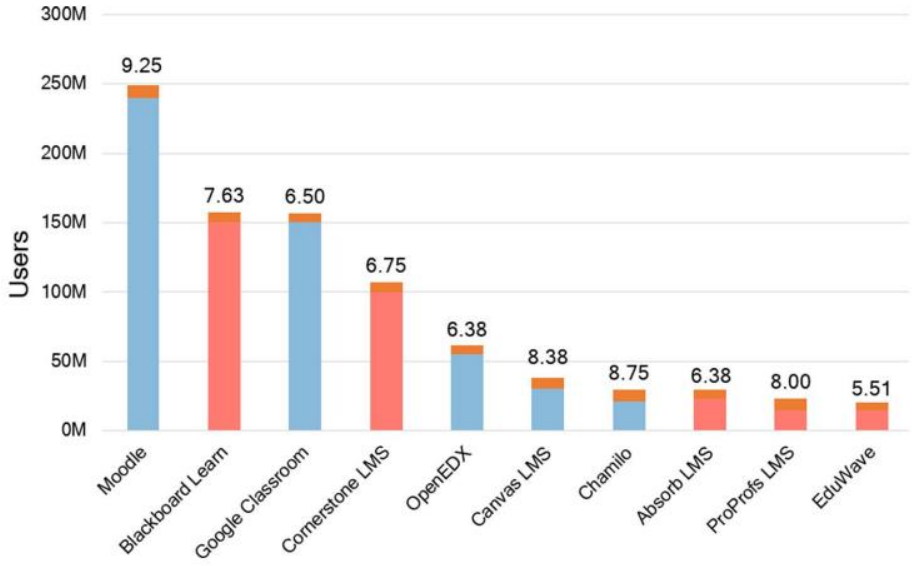

Figure 3. Top 10 LMSs based on the number of users and SQTL results [16, p.11].

As illustrated in the figure above, Moodle is the best LMS according to the number of users and the SQTL score, with a result of 9.25 points and over 250 users. Followed by Blackboard, as the second best LMS based on the number of users and the SQTL result range. All LMSs shown in the graph above have

an SQTL score of above 5 points, which illustrates that most users utilise LMSs with good tools and features. [16.]

2.5 Moodle's Architecture and Course Structure

This section explores the Moodle framework and delves into Moodle's core structure and how it works at a technical level. Moodle is an open-source web application developed in PHP. Moodle installation is simple and can be installed on any server. A successful installation of Moodle comprises PHP as the Moodle executable code. MySQL is needed as the database server, and the Moodle data folder is used as storage for uploaded and generated files. Moodle is designed as a modular system. It is structured as an application core, comprising various plugins for various functionalities. Additionally, it has a customisable and extensible design structure without interference from the core libraries; hence, upgrading to a newer Moodle version is easy. [17, 21.]

LAMP Moodle Architecture

Moodle's broad structural architecture is known as the Linux Apache MySQL PHP (LAMP) or Windows Apache MySQL PHP (WAMP) stack, depending on the operating system. This study focuses on the LAMP architecture, considering it as the most well-known setup by Moodle administrators. [17, 24.] Therefore, the open-source LAMP stack is used in Moodle development, and it consists of, the following:

- Operating system, which is Linux.
- Web server, which is Apache.
- Database, which is MySQL.
- Programming language, which is PHP. [17.]

The figure below shows a simple overview of Moodle's overall stack architecture.

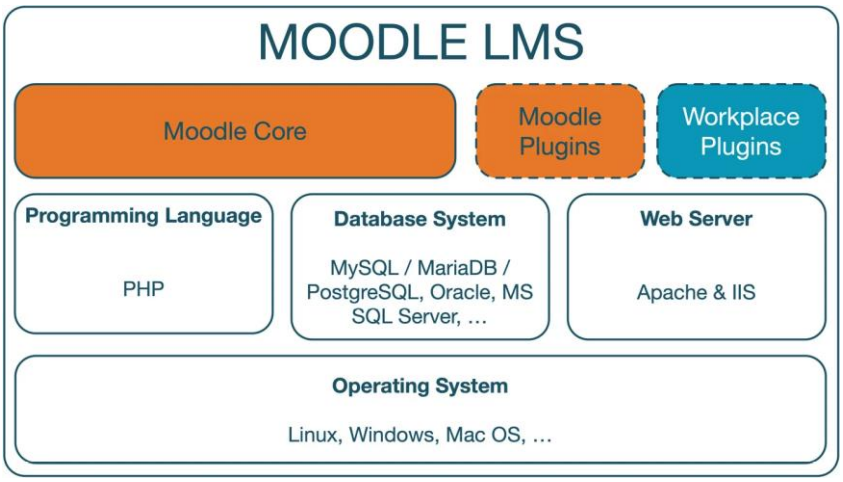

Figure 4. Moodle's overall architecture [17].

The portability and modularity of components in Moodle nature make it suitable for a wide range of operating systems, database systems and webservers. [17.]

Moodle Core Layer

According to Alex Buchner, Moodle's core structure consists of files and a management system. Moodle's core layer is made up of two main building blocks, consisting of code, which is basically developed in PHP, HTML and CSS, as well as the data, which consists of the files and values introduced through the Moodle interface, as shown in the figure below. [17.]

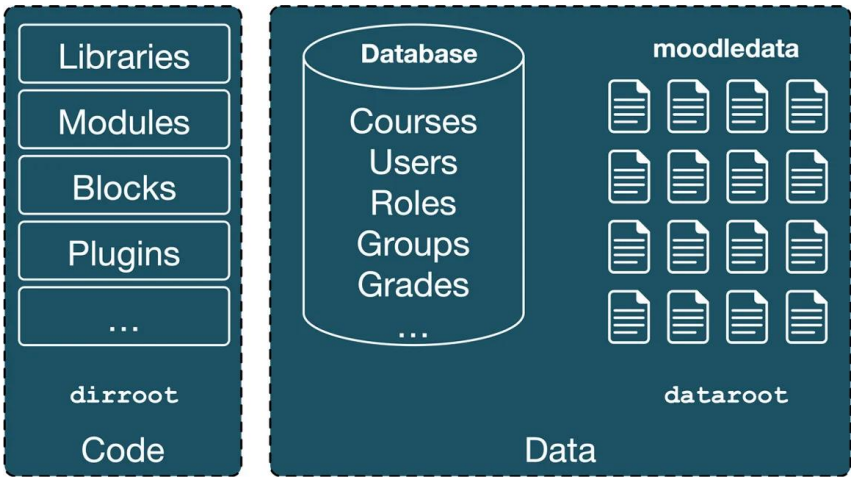

Figure 5. Moodle's core layer building blocks [17].

In the Moodle core layer, Moodle's file system contains two directories where the building blocks are stored, known as **dirroot** and **dataroot**. The code consists of Moodle libraries, modules such as resources and activities, blocks, plugins, admin tools and other entities. This is maintained and stored in the **dirroot** directory and carries out the backend and frontend operations. Secondly, the data building block, often stored in Moodle's database, comprises Moodle courses, users, roles, competencies, learning plans, admin tools and other data such as learning resources created by educators, forum posts created by learners and system settings created by the administrator. [17.]

Nevertheless, not all files are stored in the data building block. User pictures or uploaded assignment files are stored in the **moodledata** directory, in the **dataroot** directory. Additionally, metadata file information such as name, location, date and time modification, license and size are stored in the database. Moodle manages files internally using the SHA1 hashing method, supporting Unicode file names. Therefore, tampering with any files in the **moodledata** directory and storage of duplicate or unnecessary files in different folders can break the application. [17.]

Moodle Backup File

In Moodle, a Moodle course backup file can be created as a replica of the original course. Moodle creates a file that contains all the data about the original course in Moodle Backup Zip (MBZ) file format. These backup files can be restored in every Moodle application worldwide. The restored course has similar functionalities as the original course created in Moodle, so it contains all user data and modifications of the MBZ file and attempts to restore it on Moodle are achievable. [20.]

If the Moodle backup file is restored successfully, it is downloaded and stored on the local server. It is restored from Moodle in MBZ file format structure and in a gzip and tar-packed XML file and folder structure. These files contain all the information about the course and how the files reference each other. As seen in the figure below, every MBZ file structure consists of compulsory files and

folders illustrating the default setting of a Moodle course as shown below. However, when an MBZ file is uploaded to Moodle without all the necessary files, the system will break, resulting in an error message and cancelling the restoration process. [20.]

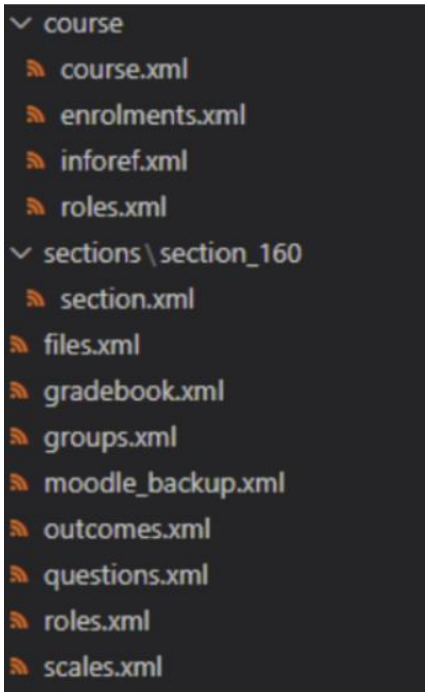

Figure 6. Mandatory XML files and folders in the MBZ file [20].

The figure above shows the files necessary for Moodle to run, known as System files, found in various directories in the root directory of Moodle installation, known as **dirroot**. [20.] The files and directories contained in the MBZ file structure include:

- ‘Course’ contains all the course settings data such as enrolment method, course name, course format, course start and end date.
- ‘Section’ holds all the topics of a Moodle course. Each topic contains its own folder and file. This includes the data about the section such as the section name and number.
- ‘Moodle Backup File’ defines all the details about the course elements, topics and their specific locations in the course. It integrates all the data

from all folders and files and serves as a summary file for the restoration process. [20.]

- ‘Activities’ contains learning elements used in the course to make the learner more active and engaged such as glossaries, assignments, chats, choices, feedback, forums and wikis.
- ‘Resources’ stores learning materials uploaded in the course for passive learning such as reading a document, navigating a link and listening to a podcast or watching a video.
- ‘Files’ is Moodle’s basic file management system where storage of uploaded files takes place, such as course, activities, resources and users’ files like personal or private files.
- ‘Roles’ defines user roles and permissions in Moodle. It outlines what users can or cannot access in the Moodle system. The most common user roles are student, teacher and administrator.
- ‘Gradebook’ contains all the leaners’ grades. It is flexible, powerful and holds an accurate grade management system.
- ‘Outcome’ is mostly used by academic curricula. It stores all the details for expected competencies or goals or outcomes of a course.
- ‘Scales’ defines all the grading or rating scales used in a course, usually offered by the institution’s organisation.
- The Grade history holds tracks for all the modifications made on the gradebook entries.
- ‘Groups’ outlines all the users in groups in their respective courses.
- ‘Completion’ tracks completion of resources and activities in the course.
- ‘Questions’ stores all questions of a course and individual questions in a question bank. [17.]

In summary, Moodle offers a basic file management system. This allows files and directories to be stored, organised and added to Moodle.

3 Project Specifications

3.1 Project Background and Customer Needs

As outlined in the introduction, this project was carried out for an external customer whose goal was to find out if it was possible to migrate JSON format course content from the WordPress CMS platform to the Moodle LMS platform. The original course was hosted by a provider on the WordPress CMS platform. The expenses of hosting the course on the provider's server increased gradually, urging the external customer to determine ways to reduce the costs.

Therefore, the external customer cooperated with Metropolia UAS to migrate the course to Moodle and help in the execution of this project. The purpose of this project was to help the external customer transfer the course content to the Moodle platform and to provide universities, researchers and organisations with a structured approach for understanding the course migration framework.

Metropolia UAS, as an organisation, was responsible for researching and testing the best practices in developing a migration pipeline for migrating the course content. The course was extracted from the CMS platform in JSON format to ease the content migration to the Moodle LMS platform. The objective of this study was to give structured guidelines for the process of developing a migration pipeline for a course existing in JSON format and transferring the course from a WordPress CMS to a Moodle LMS.

The main reason the external customer wanted to migrate the course content to the Moodle LMS platform was making it more accessible to a wide range of users and host the course on an affordable platform. Moodle is an open-source platform, hence free to use and access for users. WordPress, on the other hand, requires plugins for basic functions, such as the creation and development of a course that should be frequently renewed for continuous support. Therefore, the cost of these plugins and hosting a course increases gradually and is expensive. Additionally, Moodle is a highly scalable learning platform that is widely used by

universities and institutions all over the world. In contrast, WordPress is not as scalable as a native LMS and is not e-learning focused. To sum up, WordPress requires separate plugins for each functionality, and the plugins make the website heavy and slow while Moodle has all functionalities in-built and needs less setup for everything to run smoothly. [18; 19.]

The expected outcome of this project for the external customer was successfully migrating all course content to Moodle. The success of this project was determined by an automatic migration pipeline that can transfer a complete JSON format course from the WordPress CMS to the Moodle LMS. The outcome would also enable the migration of multiple courses from the WordPress CMS to the Moodle LMS platform.

However, limitations incurred because media files, such as images, videos and audio files, were not transferable automatically. This is because the timeline offered was insufficient for continuing the development process. Since most of the course content was successfully migrated, the goal of determining whether it was possible to migrate the course content was achieved. In addition, the media files were successfully migrated manually to Moodle for testing purposes. To conclude, the course content was migrated successfully, excluding the media files, due to time limitations.

3.2 Course Content Overview

As mentioned earlier, CMSs are software platforms that resemble LMSs. However, a CMS contains fewer features and functionalities. The course was originally hosted on the WordPress CMS platform. The goal was to migrate and host the course on the Moodle LMS platform. For a successful migration, the course content had to be extracted from the WordPress CMS platform and converted into JSON format to ease the migration process to the Moodle LMS platform.

The course content was successfully converted into JSON format. It consisted of various content types, for instance, text types such as courses, topics and lessons. Secondly, also contains assessment content types, such as quizzes, and thirdly, multimedia content types such as images, audio and video types for handling media files. Lastly, there are also content types such as external source links. In addition, files that outline the course structure and course groups were also included.

3.3 JSON Course Content Format

Sir Douglas Crockford discovered a well-known data interchange format known as JavaScript Object Notation (JSON), which existed in object notation. He was the first to specify and engineer JSON as a standardised format. JSON is a text-based, lightweight, human-readable data format for data exchange between clients and servers. Moreover, JSON is language-dependent, and its data format is supported in popular languages such as C#, PHP, Java, C++, Python and Ruby. In web applications, JSON can be used for transferring data. As shown in the figure below, the browser acting as the client sends a Hypertext Transfer Protocol (HTTP) request to the server, and the server responds to the request; hence, there is data exchange. A serialised string with a composition of key-value pairs in parentheses is utilised in this two-way data exchange communication format, which is JSON. [25.]

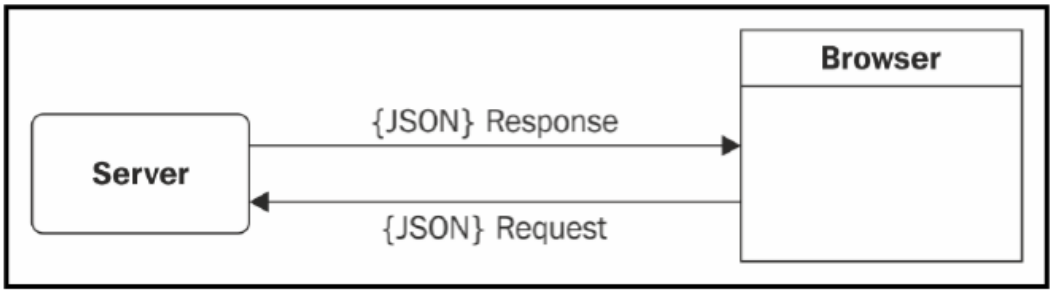

Figure 7. JSON data client-server architecture [25].

Bruno Joseph suggests that PHP is one of the most widely used languages for developing web applications. It enables developers to have a database

connection and execute Create, Read, Update and Read (CRUD) operations. PHP has functionalities comparable to JSON as a data exchange format, which includes managing a JSON request from the server and generating a JSON response to a client. The following figure represents how a JSON format data structure can be generated from PHP using the **json_encode** function. The **student** variable is passed to the function, which converts the variable into a JSON string. Finally, after running the script, a valid response is generated as JSON data for utilisation with other applications. [25.]

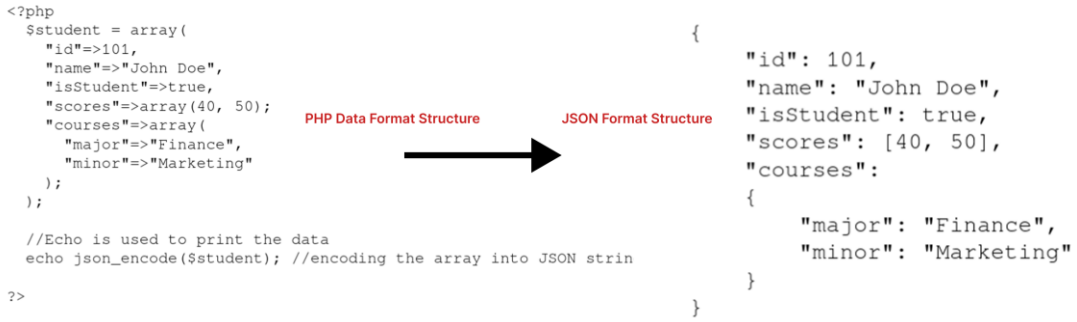

Figure 8. Conversion from PHP to JSON data format [25].

In the context of this study, the course content is converted to JSON format, and the JSON data is in nested arrays and nested objects. Each file contains data in various data types in a key-value pair structure. The key is the name of the value, and the value is the data of the key and can be represented in various data types such as string, number, array object and null. Figure 8 illustrates the conversion process to JSON data format structure and showcases how the course content existed in JSON format to conduct the migration process. [25; 26.]

3.4 Moodle Backup Files Format

Jeff Friesen argues that applications such as Moodle store and exchange data utilising Extensible Markup Language (XML) files. XML encodes documents in both human-readable and machine-readable formats. It is well known and vital because it is defined as a meta-language, a language used to describe other languages and vocabularies, for characterising custom markup languages.

Furthermore, XML vocabulary is the same as in HTML documents since they are text-based. They comprise markup, which is encoded descriptions of a document's logical structure, and content, which is the document text. Markup is represented as tags, and each tag contains a name. Some tags possess attributes that are name or value pairs. [26.]

```
<recipe>
   <title>
      Grilled Cheese Sandwich
   </title>
   <ingredients>
      <ingredient qty="2">
         bread slice
      </ingredient>
      <ingredient>
         cheese slice
      </ingredient>
      <ingredient qty="2">
         margarine pat
      </ingredient>
   </ingredients>
   <instructions>
      Place frying pan on element and select medium heat.
      For each bread slice, smear one pat of margarine on
      one side of bread slice. Place cheese slice between
      bread slices with margarine-smeared sides away from
      the cheese. Place sandwich in frying pan with one
      margarine-smeared side in contact with pan. Fry for
      a couple of minutes and flip. Fry other side for a
      minute and serve.
   </instructions>
</recipe>
```

Figure 9. XML-based recipe for a grilled cheese sandwich [26].

As seen in Figure 9, the recipe data will be used as an illustration of the Moodle XML file format. The figure presents an XML document that explains a grilled cheese sandwich recipe. It showcases how document vocabularies in XML resemble HTML, considering that HTML documents include tags, attributes and content. However, HTML tags are formal according to the HTML language such as <html>, <head>, and <p> while XML tags are informal relating to the recipe language, hence describing its own tags, such as <recipe>, <ingredients> and others. [26.]

Additionally, XML documents often start with an XML declaration, a special markup that represents an XML document to the XML parser. The XML parser ensures the XML file is well structured and does not detect errors [25]. In the above example, the XML declaration is absent, which means that it is optional. However, if present, it must be in the first line of the document. The XML declaration is presented as **<?xml version="1.0" encoding="UTF-8"?>**. It contains a mandatory attribute, version, which is the representation of the XML document version specification. The value of the version attribute is 1.0, which is the initial value of this specification, introduced in 1998 and widely utilised. [26.]

In addition, another attribute contained in the XML declaration is the encoding attribute. XML documents contain characters from the Unicode character set. This illustrates the process of encoding a document's characters into bytes for storage or transmission, such as UTF-8, which is a variable-length encoding character set that is a strict superset of the American Standard Code for Information Interchange (ASCII), such as ASCII text files. [26.]

XML documents also contain elements and attributes. The XML declaration follows a hierarchical tree structure of elements. An element is a section of an XML document denoted by a start tag, such as <name>, and an end tag, such as </name>, which usually contains content and probably other markups. There are also empty element tags, such as <break/>, without any content.

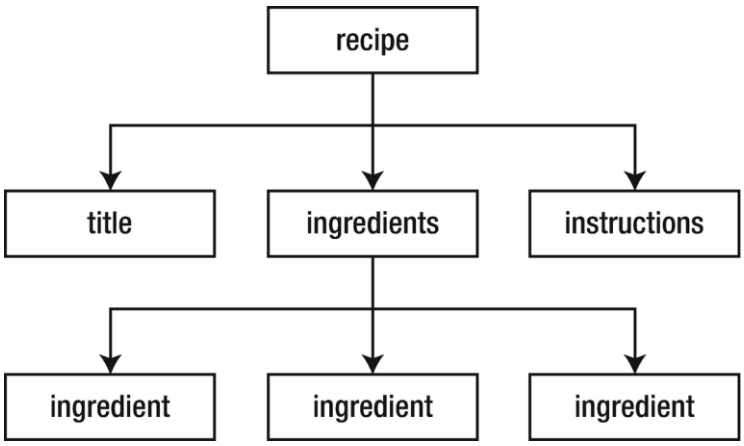

Figure 10. XML-based recipe tree structure [26].

The figure above shows the tree structure of the XML recipe elements. In both HTML and XML documents, the content is enclosed in the topmost element, the root element. In HTML, the root element is <html> with <html> as the opening tag and </html> as the closing tag. In contrast to XML documents, the root elements depend on the name and content of the file. Elements can consist of child elements, content or a combination of content and child elements. As shown above, the root element of the recipe XML document is <recipe>; hence, it has no other parent element. Moreover, the recipe and ingredients elements contain child elements, where the recipe element has title, ingredients and instructions as its child elements and the ingredients element has three ingredient child elements. [26.]

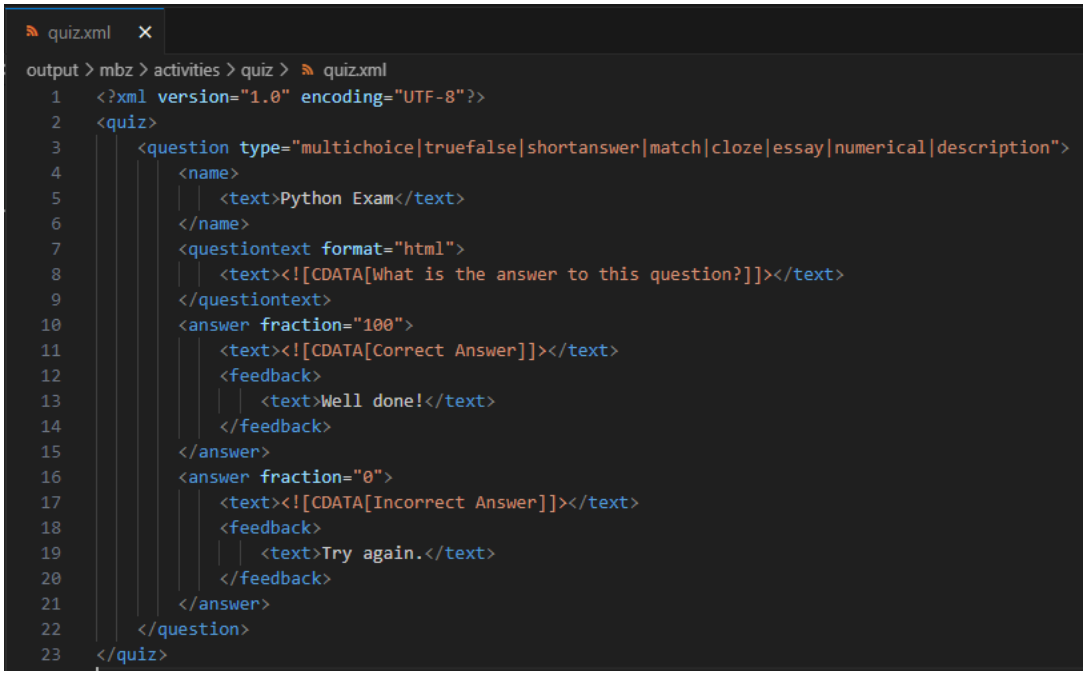

Figure 11. Question file Moodle XML format [27].

Similarly, Moodle backup XML files follow the tree structure as discussed in Figure 10, which describes the parent and child elements in an XML file. In this study, the XML file shown in Figure 11 was utilised as an example of a Moodle XML file. The XML declaration in the first line is vital for every Moodle XML file. The <quiz> tag follows as the parent element. It contains one child element, which is the <question> tag. [27.]

The <question> tag contains the type attributes that specify the type of question, such as multiple choice, true false and more. The <question> tag also contains child elements such as <name>, which states the general name of the test, <question text> and <text> tags, which contain question texts and the content of the question. The <question text> tag contains formatting options of the question text, the selected format being HTML, which is the default, and others including moodle_auto_format, plaintext and markdown. In addition to the child elements, there is the <answer> tag, which is the fraction that contains the grade a student should acquire after answering a question, depending on whether the answer is right or wrong. It also contains the content of the answer and the feedback from the instructor. [27.]

3.5 Comparative Analysis between JSON and Moodle XML File Format

Both JSON and XML are data serialisation formats. Data serialisation is the structuring and exchanging of data across various applications, platforms or systems in a standardised manner. Despite JSON and XML serving similar functions, they also contain key differences. The table below outlines the core differences between JSON and XML data formats. [28.]

Table 2. Core differences between JSON and XML [28].

	JavaScript Object Notation (JSON)	**Extensible Markup Language (XML)**
History	It was released by Douglas Crockford and Chip Morningstar in 2001. They derived JSON from JavaScript.	In 1996, the XML Working Group invented XML, and in 1998, the initial version was released. XML was derived from Standard Generalised Markup Language (SGML).

	JavaScript Object Notation (JSON)	**Extensible Markup Language (XML)**
Format	It uses key-value pairs. The key is a string that acts as the identifier of the pair, and the value is the data of the key.	It uses a tree-like structure to store data starting with a root or parent element, following the child elements.
Syntax	Its syntax is more compact and easier to read and write, as it enables the definition of objects easily.	It uses entity references and tags as its syntax.
Parsing	A JavaScript function can be used to parse JSON, making it more accessible and faster.	It utilises an XML parser to parse XML, making it a slow and complicated process.
Data Types	Supports limited data types such as strings, numbers, objects and Boolean arrays.	XML supports all JSON data types. In addition, it also supports complex data types such as binary data and timestamps.
Ease of use	It contains smaller file sizes; hence, data transmission is faster.	XML is more complex to read and write as it requires a tag structure.

To summarise, the course content was existing in JSON format, and the Moodle backup file structure consists of files in XML format. Therefore, the implementation of the migration framework consisted of the development of custom automation scripts that converted JSON data type into Moodle XML format.

4 Proposed Migration Framework

In general, the term data migration means moving or transferring digital information to a different location, file format, environment, storage system, database, data centre or application [29]. It entails selecting, preparing, extracting, transforming data and moving it permanently from one system to another [29]. As discussed earlier, this study outlines the process of transferring a course from the WordPress CMS platform to the Moodle LMS platform. In the context of this study, the course migration process involved moving the course content from one software application system to another.

4.1 Overview of the Migration Process

When the thought of moving data from one system to another comes to mind, it seems impossible since, in some situations, it is complicated [29]. This is because different systems have different data assets in various states and locations, which makes migration projects more complex and challenging than others [29]. This is why the data migration process needs planning, implementation and validation steps to ensure data is migrated successfully [29]. This is the same process this study took to ensure the course was migrated successfully to the Moodle LMS since the course existed in a different system, the WordPress CMS. Artificial Intelligence (AI) was also utilised in researching the best way to migrate the course since it existed on a different platform.

The first step of data migration is planning [29]. This comprises organising the data to be moved and the organisation responsible for this movement [29]. This process involves gathering needs for data migration, assessing the data and the impact of the data migration on the organisation, which involves communication with the stakeholders, assigning responsibilities and setting the necessary budget for the project [29]. This is believed to be the process the external customer took in planning the data migration process. There was a need for an affordable, easy-to-access and interactive platform for students and instructors. The data to be migrated was a course that involved various content types of text, such as

courses, topics and multimedia files, such as audio, videos and images. The organisation assigned responsibilities to the project stakeholders, who cooperated with Metropolia UAS to aid in the course migration process, transferring the course to the Metropolia Moodle platform.

The next step is the implementation process [29]. This process involves executing the data migration, comprising the data migration process design [29]. This comprises a step-by-step approach from the beginning to the end that the designated team should follow to ensure that data is properly migrated without errors, data quality issues, duplication or variation from the original data [29]. In this project, the external customer cooperated with Metropolia UAS team that would execute this project. Metropolia's responsibility was to find and provide a team for the course migration process and provide a location of the Metropolia Moodle platform to which the course could be migrated. The team implemented the migration pipeline by utilising custom automation scripts with the help of AI. The scripts enabled the conversion of JSON format course content to Moodle XML format.

Finally, the data needs to be validated [29]. After the course migration execution process is completed, the assigned team must inspect and assess the course in the new platform [29]. Validation is done to ensure the data has migrated properly [29]. The validation initiative is taken by business and technical stakeholders and the user customers [29]. In this project, the team was able to migrate almost the entire course to the Moodle LMS platform. The migration of multimedia files automatically was impossible due to time constraints. Hence, the multimedia files had to be migrated manually for testing purposes. Testing and validation were done by the team and Metropolia UAS.

4.2 Data Migration Tools and Technologies

The necessary software and hardware tools and technologies are required for a successful data migration process. The team carrying out the migration process should select the tools to utilise when moving and modifying the data, if needed.

The team is allowed to build their own data migration tools or utilise existing migration tools. To execute the implementation process with personalised tools is more of a disadvantage than an advantage. This is because it consumes more time, more resources, including costs, and requires manual integration and re-implementation, resulting in unscalable issues. On the other hand, if the team chooses to utilise existing data migration software, then moving the data becomes easier, faster and more efficient. Moreover, various software specialising in different data migrations exist, such as moving a Structured Query Language (SQL) database to Azure. [29.]

Moreover, whether the team chooses to utilise software or not, some things need to be considered before the migration process begins [29]. The team needs to understand the data to be moved, assess the amount of data to be transferred and what parts of the data need to be altered and, after the migration is complete, resolve any issues incurred [29]. This is not different from what took place in this project. The team assessed the course on the old platform and the need to have the course content in a format that can be utilised. The course was extracted from the WordPress CMS platform and converted to JSON format to ease the migration process to the Moodle LMS platform. Therefore, having the course content in JSON format was the first step required for the migration process.

In general, the JSON data structure is not compatible with the Moodle XML schema. Therefore, the team needed to build personalised data migration tools. The custom automation scripts were utilised to parse and map JSON files to Moodle XML files to ensure seamless data integration in the Moodle platform. The first required tool was the data, which was in JSON format. The software utilised for writing and editing code was Visual Studio Code. This was chosen because it was a popular code editor and a powerful editing tool. Git version control was also utilised to develop the migration process collaboratively in the team. The programming language used was JavaScript since it is supported in Visual Studio Code. The testing tools used were local Moodle and Metropolia Moodle.

5 Course Migration Implementation

5.1 Development and Implementation Phase

In this chapter, the course migration implementation process will be discussed. The development and implementation phase were successfully executed, achieving the project's goal. The team discovered that JSON format is incompatible with Moodle's XML file format; hence, the practical implementation was executed by custom automation scripts. This approach would suit this situation since tailoring the tools to the specific systems, such as CMS and LMS platforms as well as data formats such as JSON and XML, would be more suitable for this course migration process.

The use of customised scripts would be an asset in the development. The main function of customising the scripts was to ensure seamless integration with different data formats. It would also establish the interoperability of the extracted data in Moodle. Additionally, these scripts would create files in Moodle's backup file structure to ensure that the MBZ backup file is restored to Moodle successfully and without any files missing and that the structure is correct.

5.2 Creation of Empty MBZ files

The first step of the development process was to study the MBZ file structure. These files contain all the information about the course. Understanding the structure and purpose of each file would enable the creation of the files required for the course. As previously mentioned, every MBZ file consists of mandatory files and folders. If mandatory files were missing or an incomplete file was present when an MBZ file was uploaded to Moodle, the system would output an error and cancel the course restoration process.

Since the details of the MBZ file structure were described in only a few publications, the reverse engineering procedure had to be considered. This would enable the team to comprehend the information in each file and the relationship

between each file in the MBZ file structure. This involved creating an empty course, referring to old MBZ course files and assessing the differences between the old and newly created MBZ file structure. The goal was to have an empty custom course that could be restored in the Moodle system for testing purposes, so all the Moodle XML files consisted of empty element tags. When all the required files were created, the files were compressed into a ZIP file and converted to the MBZ extension file structure. Finally, the created empty MBZ course file was tested to see if it can be restored successfully in Moodle.

The development of the scripts began with the creation of empty, structured XML files and directories that follow the architecture of an MBZ file course structure. As shown in the figure below, customisable scripts were written in JavaScript to generate the empty Moodle XML files, placing them in the respective directories of the MBZ file structure. The XML files contained empty root element tags, child element tags and attributes. In addition, each XML file included an XML declaration. Finally, the empty MBZ course file was successfully restored in Moodle and this stage proved the possibility to successfully migrate the course to Moodle. Figure 12, an example of the group's custom script file in the MBZ file structure, generates the group XML file.

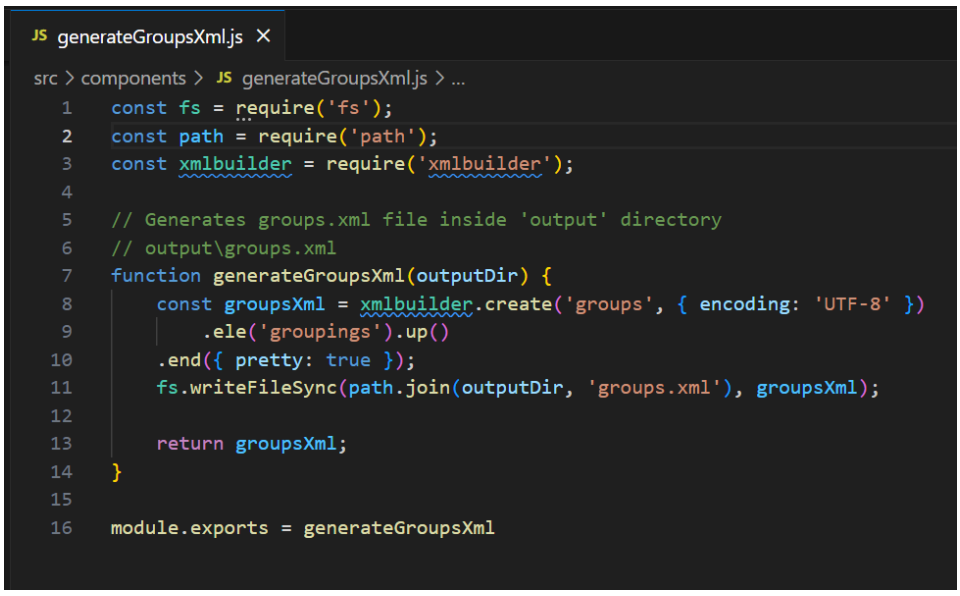

Figure 12. Custom scripts for generation of group's XML file.

Similar scripts were created for all the required XML files, utilising the same approach. This would ensure the MBZ course file is restored successfully, as well as that there is interoperability and integration of the newly created XML files in the Moodle platform. In addition, this also ensured written clean code with separation of concerns, hence easier comprehension of the scripts and the MBZ course file in general.

5.3 JSON Data Parsing and Mapping to XML Format

This part of the implementation process involves the transformation and integration of course content from JSON format into XML format. The extracted JSON data is mapped to the XML files previously generated. This was executed by creating automated data transformation customisable scripts. The main function of these scripts was to ensure automatic data transformation from JSON data format to XML format, utilised in Moodle. The course content in JSON format comprises various content types that resemble Moodle, such as questions, topics, lessons, multimedia files and general course information. Additionally, in JSON format, the data is structured in key-value pairs as discussed earlier, the key being the name of the value and the name of the element tag in XML format. This ensured a simplified data extraction process by mapping the JSON key names with XML element tag names; hence, the data extraction was accurate.

Management Constant Modules

The basic requirement for all scripts was the management modules. These modules should be installed, imported and declared at the beginning of each script. Further details are discussed below.

- The fs module is responsible for managing file system operations such as reading and writing files. It is used in reading JSON files and writing the extracted data into XML files.
- The path module manages file and directory path utilities. It ensures that XML files updated with course content exist in the correct directory.

- The xml2js library module is used for parsing XML into JavaScript objects and building XML from JavaScript objects.

Figure 13 shows the declaration of the required management constant modules declared in the scripts as shown below.

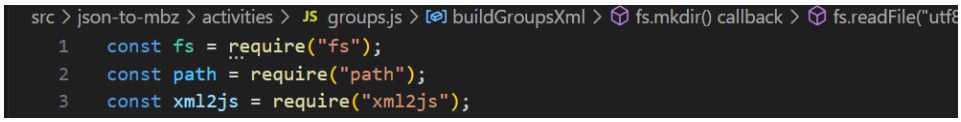

Figure 13. Management script modules.

All the scripts used for extracting JSON data to XML files should contain the above management constant modules for the generation of Moodle-structured, XML files.

Function Declarations and Purpose

The first function declared was the **createContent** function. It comprises object creation from JSON data. A JSON object is created in a function with various properties. The purpose is to extract JSON data as input and transform and integrate it into XML format. The course content is parsed from JSON files and properties in JSON and XML files are mapped. As shown in the figure below, the **wp_data** array in JSON data is mapped and each object in the array is transformed into XML properties such as ID, title and description. In the XML files, default values are fed for some properties if they are absent in the JSON data.

```
// Function to create content from JSON data
const createContent = (data) =>
  JSON.parse(data).wp_data.map(
    ({
      wp_post_id,
      wp_post_title,
      wp_post_name,
      wp_post_content,
      wp_post_date,
      wp_post_modified,
      wp_post_password,
    }) => ({
      id: wp_post_id,
      title: wp_post_title || "",
      description: wp_post_content || "",
      descriptionformat: 1,
      idnumber: "",
      enrolmentkey: wp_post_password || "",
      picture: "",
      hidepicture: 0,
      timecreated: wp_post_date || "",
      timemodified: wp_post_modified || "",
      group_members: [],
    })
  );
```

Figure 13. JSON data extraction and transformation into XML format.

The next function declared is the **updateXmlWithJsonContent** function. It updates the XML-structured files with the course content extracted from JSON data. As illustrated in Figure 15, it takes two parameters, which consist of **xmlData**, the XML data to be updated and **jsonContent**, which is to be integrated into XML. Thereafter, the **xml2js.Parser** library is created to parse XML data and the **xml2js.Builder** to build XML and update XML files with specific options, such as the encoding attribute set to UTF-8 for the XML declaration.

Additionally, the **parser.parseString** method is called to convert the XML string into a JavaScript object. This process is handled asynchronously. If successful, the XML result is processed further. However, the conversion is rejected with an error message if an error occurs. The group property is initialised in the parsed XML objects with an empty group array. The iteration of the new object over each item in the array is executed when the **jsonContent.forEach** function is called. If there are numerous JSON objects, the function iterates over each group in the JSON content array and constructs a new XML group object. Each JSON property is mapped to the corresponding XML element. To conclude, the new

group objects are added to the XML object, and the updated XML file is converted to an XML string.

```
const updateXmlWithJsonContent = (xmlData, jsonContent) => {
  const parser = new xml2js.Parser();
  const builder = new xml2js.Builder({
    xmldec: { standalone: null, encoding: "UTF-8" },
  });

  return new Promise((resolve, reject) => {
    parser.parseString(xmlData, (err, xmlResult) => {
      if (err) {
        return reject(`Error parsing XML: ${err.message}`);
      }

      xmlResult.groups = { group: [] };

      jsonContent.forEach((group) => {
        const newGroup = {
          $: { id: group.id.toString() },
          name: [group.title],
          idnumber: [group.idnumber],
          description: [group.description],
          descriptionformat: [group.descriptionformat],
          enrolmentkey: [group.enrolmentkey],
          picture: [group.picture],
          hidepicture: [group.hidepicture],
          timecreated: [group.timecreated],
          timemodified: [group.timemodified],
          group_members: [group.group_members],
        };
        xmlResult.groups.group.push(newGroup);
      });
```

Figure 14. Updating XML files with JSON data.

According to Figure 16 below, the last function created is the **buildGroupsXml** function. This function is used in reading an XML file and a JSON file, creating course content from JSON data and updating the XML file with the course content. As shown below, it contains two definitions of file paths, which are **inputXmlFilePath**, the path to the original empty XML file, and **outputXmlFilePath**, the path where the updated XML with course content will be saved. Moreover, **fs.existsSync** is used to check if the original XML file exists, and **fs.mkdir** is used to create the main directory, finalDir. The **fs.readfile** is utilised in reading the contents of the original XML file. The **groupsJsonFilePath** is the JSON file path that contains the course's group data and finalDir is the directory where the updated XML file will be saved to emulate the MBZ file structure. However, if an error occurs when these functions are called, an error

message is logged, and the function exits. If there is no error, the new XML file is created in the specified directory.

```
const buildGroupsXml = (groupsJsonFilePath, finalDir) => {
  const inputXmlFilePath = path.join("output", "mbz", "groups.xml"); // Original file
  const outputXmlFilePath = path.join("final-mbz", "groups.xml"); // Updated file

  if (!fs.existsSync(inputXmlFilePath)) {
    console.error("Error: Input XML file does not exist at:", inputXmlFilePath);
    return;
  }

  fs.mkdir(finalDir, { recursive: true }, (err) => {
    if (err) {
      console.error("Error creating directory:", err.message);
      return;
    }

    fs.readFile(inputXmlFilePath, "utf8", (err, xmlData) => {
      if (err) {
        console.error(
          "Error reading XML file. Ensure the XML file exists:",
          err.message
        );
        return;
      }

      fs.readFile(groupsJsonFilePath, "utf8", (err, jsonData) => {
        if (err) {
          console.error(
            "Error reading JSON file. Ensure the JSON file exists:",
            err.message
          );
          return;
        }

        const jsonContent = createContent(jsonData);
```

Figure 15. Reading and writing an XML and a JSON file.

To conclude, the above implementation ensures the necessary XML files are updated with JSON data. The scripts are utilised to ensure that JSON data in the JSON files is extracted and migrated to the XML files successfully. Each script is created for the content in the JSON files, which updates the respective XML file. In addition, all the updated XML files should be saved in their respective directories following Moodle's backup file structure.

After the complete JSON content is extracted and all the XML files are updated with content, the next step is to restore the MBZ file in Moodle. The files are converted to the MBZ Moodle structure and saved on the local computer. Course restoration is the process of importing a course backup file (MBZ) in Moodle as a new course or as an existing course [30]. In this study, the course was

successfully restored as a new course since it had never existed in Moodle. The course was restored on the Metropolia Moodle platform and the local Moodle platform testing environments.

6 Course Migration Testing and Results

As discussed earlier, the last step of the data migration process is validating the course in the new platform. After the course content was migrated, the team was responsible for a systematic review of the course in the new platforms, which was used in testing the locally installed Moodle and Metropolia Moodle and assessing the differences in the structural layout. Additionally, the team reviewed the data in the new layout and examined if it had been migrated correctly. The course's visual layout resembled a generic Moodle course although there were minor differences between the two platforms.

As stated previously, the course was validated by the team and Metropolia stakeholders, who approved the course migration process as successful, and could be a course that can be used by students and instructors. However, there were challenges, which included missing icons, headings and multimedia files. Moreover, the external customer was also not able to test the course in the new platform due to time constraints.

6.1 Testing Environments

Both the local Moodle and the Metropolia Moodle platforms were used as the testing environments in this project. Metropolia Moodle was used because the external customer cooperated with Metropolia in hosting the course on the Metropolia Moodle platform, which is affordable and accessible. Moreover, the implementation team chose to test the course in local Moodle to assess the outcomes and the differences in the course structure layout. As stated earlier, the course's text content was successfully migrated to both Moodle platforms; however, the course's multimedia files were imported manually. In comparison

between the local Moodle and the Metropolia Moodle, there were minor differences in the course structure layout.

6.2 Analysis and Discussions of Test Outcomes

The main challenges incurred in the post-migration phase. This was illustrated in the visual course structure, especially in the Metropolia Moodle platform, which was the main testing environment. First, certain icons and headings were missing on the Metropolia Moodle platform. The figure below shows some of the differences in the structural layout of the course in Metropolia Moodle compared to the local Moodle platform.

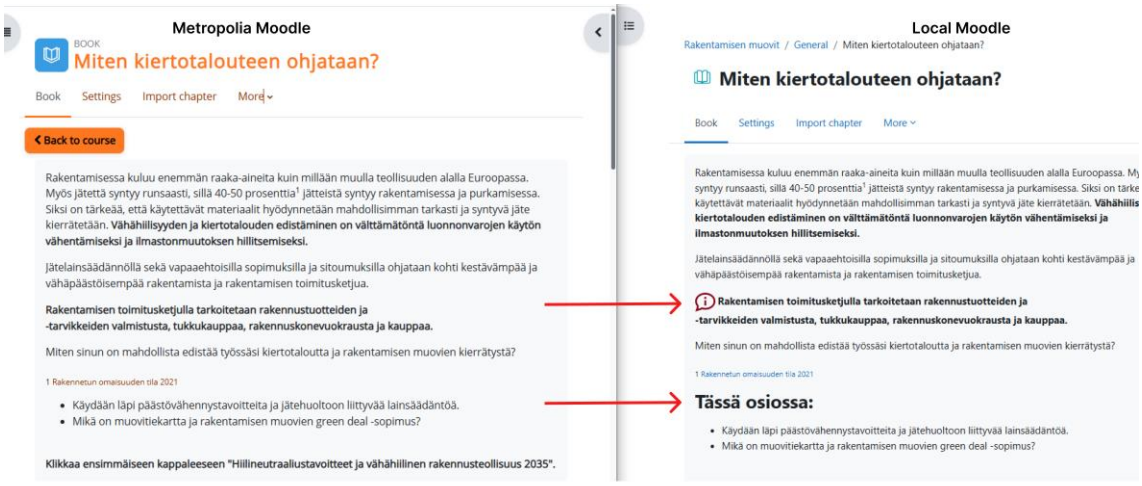

Figure 16. Layout comparison of Metropolia Moodle and local Moodle.

The figure above shows that in Metropolia Moodle, an information icon is missing in front of a heading, so the heading is shown without the icon. Additionally, some headings were not rendered automatically in Metropolia Moodle, which were present in the local Moodle. This is illustrated as one heading was missing in the Metropolia Moodle platform, as showcased in Figure 17. This is because Metropolia Moodle contains customisable themes personalised with the institution's branding and needs. Therefore, these customisations can change and affect the visual layout and organisation of course elements significantly, hence not supporting some headings and icons.

Moreover, while validating the course on both Moodle platforms, the multimedia files were missing. Therefore, images, videos and audios were not correctly displayed, and the following error message was displayed: “The media could not be loaded, either because the server or network failed or because the format is not supported”. This was due to time constraints. Figure 18 illustrates how the audio files in both platforms were displayed.

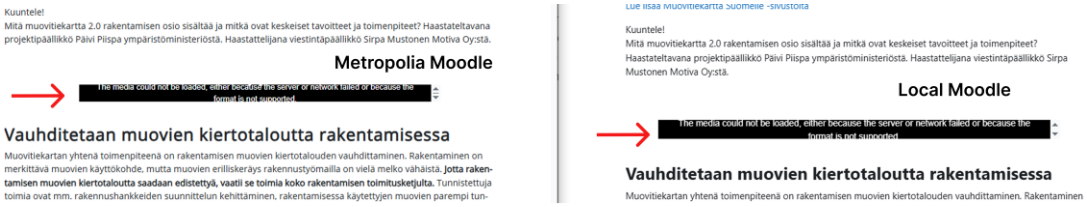

Figure 17. The layout of unsupported multimedia files.

As stated severally, the external customer’s goal was to find out if it was possible to migrate the course to the Moodle platform, and because the course text content migrated successfully, this meant that if there had been no time limitations, the implementation of the scripts dealing with the multimedia files would have worked. However, due to time constraints, the implementations of the scripts dealing with transferring multimedia files were not completed. This resulted in transferring the files by uploading the media files manually in both Moodle platforms for testing purposes to enable proper integration and functionality of the course.

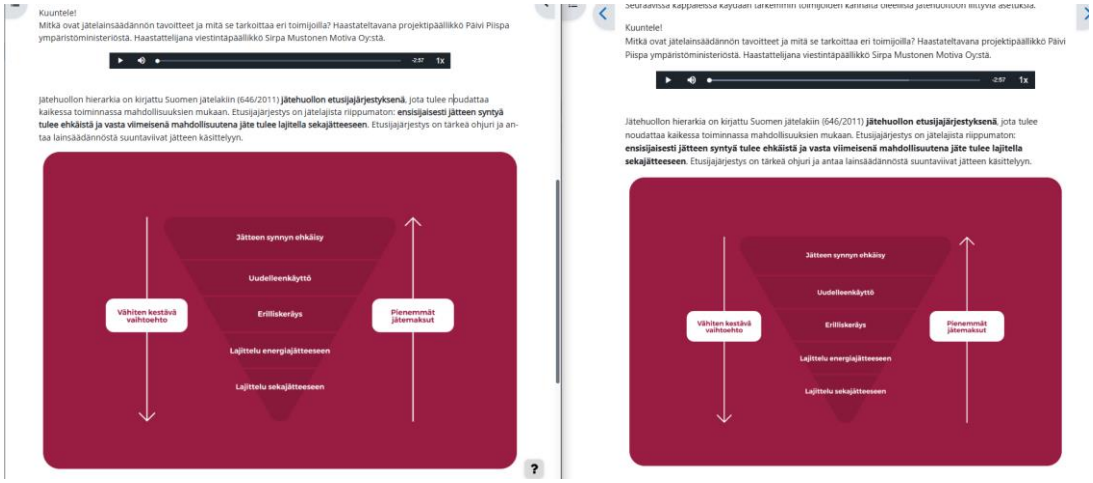

Figure 18. Audio and images in Moodle platforms transferred manually.

As shown above, the audio files and the images were imported manually. Therefore, validation of the course required technical and manual solutions, which enabled the team to make the course in the new platform resemble a Moodle course. It was done ensure that all the course content was transferred correctly and the user experience for the customers resembled the old platform for testing purposes.

To conclude, Moodle's main functionality is similar in both platforms, as it enhances students' learning. However, Metropolia's institutional customisations, such as themes and course formats, caused the differences in the course structure layout between Metropolia Moodle and the locally installed Moodle platform, affecting how the course was structured.

7 Conclusion

The main goal of the external customer for this project was to determine if the course migration process is possible. The objective of this project was to provide a practical and structured guide for migrating a course from WordPress to the Moodle platform. The focus was to develop and implement a migration framework with the course content in JSON data format and convert it into XML format compatible with the Moodle environment.

The implementation of this project resulted in a course content migration pipeline that transferred course text content from the WordPress CMS platform to the Moodle LMS platform. However, various challenges were encountered during the validation process as the course on the Moodle platform was missing icons, headings and multimedia files. The missing media files had to be imported manually for testing purposes. Testing and validating the course in the new platform after the migration showed that testing is a vital part of the process.

The migration of the course from the WordPress CMS platform to the Moodle LMS platform demonstrated the possibility of transferring a course from one platform to another platform with a different data format. Therefore, the case company’s goal was achieved. The successful migration of the course to the Moodle platform was evident in Metropolia Moodle and in locally installed Moodle, which were used as the testing environments.

The aim of this project was to help the case company migrate course content from WordPress CMS to the Moodle platform. This was because the course was hosted on a platform which is expensive to maintain and not easily accessible. In contrast, Moodle is open source and free to access, which makes it affordable and accessible. In addition, Moodle is the best LMS in educational institutions today. Therefore, the external customer chose Moodle as the most suitable LMS to migrate the course to.

In the future, AI tools can be utilised to continue with this study to further automate the whole migration process, to improve efficiency and accuracy. This involves

implementing the scripts that deal with the media files to enable complete course migration, including the transfer of multimedia files such as video and audio files. Furthermore, there is an aim to thoroughly investigate the causes of course content variations in Moodle after the migration process and to implement and develop the best practices in course migrations related to the Moodle platform in the education sector.

To sum up, this project serves as a case study to understand Moodle's backup file structure and the best LMSs to utilise in the education field. It also provides a deep understanding of migrating a course from the WordPress CMS platform to the Moodle LMS platform. Additionally, this study was executed and documented for institutions, researchers and organisations aiming to migrate their courses from one platform to another. This increases understanding of the structured approach of a course migration framework and provides in-depth information to researchers carrying out similar projects, aiming to improve migration frameworks for educational institutions.

References

1 Forum WE [online]. 2022. These 3 Charts Show the Global Growth in Online Learning. < https://www.weforum.org/stories/2022/01/online-learning-courses-reskill-skills-gap >. Accessed 12 March 2025.

2 Firdhous, Mohamed Fazil Mohamed; Elbreiki, Walid & Wijesiriwardana, Charman. 2023. Version Controlling of User Content in Learning Management Systems for Supporting the Teaching/Learning Process. 8th International Conference on Information Technology Research (ICITR), Colombo, Sri Lanka, 2023, pp. 1-6.

3 iSpring Solutions [online]. 2024. 11 Types of LMS Platforms and How to Choose the Best Fit. < https://www.ispringsolutions.com/blog/types-of-lms>. Accessed 10 March 2025.

4 Sangole, Rajkamal; Desai, Darshana & Jain, Anand. 2022. Education 4.0: Case Study on Selection of Digital Learning Platform and Communication Tools for Future Education 4.0 in India. IEEE Pune Section International Conference (PuneCon), Pune, India, 2022, pp. 1-7.

5 Manning, Susan & Johnson, Kevin. 2020. Online Learning for Dummies. For Dummies.

6 Al-Dhief, Fahad Taha; Nasser, Ali Al; Tharikh, Shafazawana Mohamed; Nasser, Hassan Al; Al-Mosleh, Ali AbdulGhaffar; Albadr, Musatafa Abbas Abbood; Alsemawi, Majid Razaq Mohamed. 2024. Review of Learning Management Systems: History, Types, Advantages, and Challenges. Indonesian Journal of Electrical Engineering and Computer Science.

7 Steven, Foreman & Association for Talent Development. 2017. The LMS Guidebook: Learning Management Systems Demystified. Association For Talent Development.

8 Agaçi, Romina. 2017. Learning Management Systems in Higher Education. University for Business and Technology International Conference in Kosovo.

9 Kasabova, Gergana; Parusheva, Silvia & Bankov Boris. 2023. Learning Management Systems as a Tool for Learning in Higher Education. Izvestia Journal of the Union of Scientists-Varna. Economic Sciences Series.

10 Dobre, Iuliana. 2015. Learning Management Systems for Higher Education - an Overview of Available Options for Higher Education Organisations. Procedia - Social and Behavioral Sciences, Volume 180, pp. 313-320.

11 Mohd Kasim, Nurul Nadirah, and Fariza Khalid. 2016. Choosing the Right Learning Management System (LMS) for the Higher Education Institution

Context: A Systematic Review. International Journal of Emerging Technologies in Learning.

12 Micha, Roza. 2019. The Current and Future State of LMSs. Master's Thesis. Aristotle University of Thessaloniki.

13 Smith, Susan Nash. 2022. Moodle 4 E-Learning Course and Development: The Definitive Guide to Creating Great Courses in Moodle 4.0 Using Instructional Design Principles, 5th Edition. Packt Publishing.

14 Dvorak, Radana. 2011. Moodle For Dummies. Wiley Publishing, Inc., Indianapolis, Indiana.

15 Moodle [online]. 2024. Moodle Features. <https://docs.moodle.org/405/en/Features>. Accessed 10 March 2025.

16 Sanchez, Lisseth; Penarreta, Jefferson; Soria Poma, Xavier. 2024. Learning Management Systems for Higher Education: A Brief Comparison. Discover education.

17 Buchner, Alex. 2022. Moodle 4 Administration: An Administrator's Guide to Configuring, Securing, Customizing and Extending Moodle, 4th Edition. Packt Publishing.

18 WooNinjas [online]. 2025. WordPress LMS Vs Moodle: Which Is Best For Your Online Course? < https://wooninjas.com/wordpress-lms-vs-moodle-which-is-best-for-your-online-course/ >. Assessed 30 March 2025.

19 eLearning Industry. 2021. The Pros And Cons Of Using A WordPress Learning Management System. < https://elearningindustry.com/using-a-wordpress-learning-management-system-pros-cons>. Assessed 30 March 2025.

20 Dimitri, Bigler; Hagel, Georg. 2023. Technical Report: Define a Customized Course and Import It into Moodle without Changes to the Configuration of the Moodle System. In Proceedings of the 5th European Conference on Software Engineering Education.

21 Moodle [online]. 2024. Moodle Architecture. < https://docs.moodle.org/dev/Moodle_architecture>. Accessed 14 March 2025.

22 Metropolia UAS: Open UAS - Studying in Open University. < https://www.metropolia.fi/en/academics/open-university/studying-in-open-uas>. Accessed 14 March 2025.

23 University of Helsinki: Online Learning Environments. 2021. < https://teaching.helsinki.fi/instructions/article/online-learning-environments>. Accessed 14 March 2025.

24 Wild, Ian. 2017. Moodle 3.x Developer's Guide. Birmingham: Packt Publishing.

25 D'mello, Bruno Joseph; Sai, Srinivas Sriparasa. 2018. JavaScript and JSON Essentials. Packt Publishing, Limited.

26 Friesen, Jeff. 2019. Java XML and JSON: Document Processing for Java SE. Après.

27 Moodle [online]. 2024. Moodle XML Format. <https://docs.moodle.org/test/Moodle_XML_format> Accessed 24 March 2025.

28 Amazon Web Services [online]. What's the Difference Between JSON and XML? < https://aws.amazon.com/compare/the-difference-between-json-xml/ > Accessed 25 March 2025.

29 Microsoft [online]. What Is Data Migration? 2025. < https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-data-migration>. Assessed 26 March 2025.

30 Moodle [online]. 2024. Course Restore. < https://docs.moodle.org/405/en/Course_restore>. Accessed 31 March 2025.