УДК 004.9

РАЗРАБОТКА МОДУЛЯ ИМПОРТА ЗАДАНИЙ СИСТЕМЫ КОНТРОЛЯ И ОЦЕНКИ ЗНАНИЙ СТУДЕНТОВ LMSDOT

Кузнецов Михаил Михайлович[1], Смирнов Даниил Павлович[2], Дедович Татьяна Григорьевна[3]

[1]Студент;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, д. 19;
e-mail: kmm.19@uni-dubna.ru.

[2]Студент;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, д. 19;
e-mail: smirnov@lmsdubna.ru.

[3]Старший научный сотрудник;
Объединенный институт ядерных исследований;
Россия, 141980, Московская обл., г. Дубна, ул. Жолио-Кюри, д. 6;
Кандидат физико-математических наук, доцент;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, д. 19;
e-mail: tdedovich@jinr.ru.

В статье проведено сравнение импорта заданий в LMS-системах. Сформулированы функциональные и нефункциональные требования. Изучены средства для работы с файлами и выбраны лучшие варианты для внедрения в систему Lmsdot. На основе требований и анализа была спроектирована архитектура модуля импорта заданий. Прототип модуля импорта заданий был реализован и внедрен. Разработанный модуль позволяет: одновременно загружать несколько файлов в текстовом (docs, md и tex) и табличном форматах (xlsx и csv) в систему Lmsdot, разделять данные по вариантам и заданиям, создавать шаблоны заданий и работ, редактировать и сохранять данные.

Ключевые слова: Lmsdot, JS, pandoc, exceljs, импорт, LMS-системы, образование.

Для цитирования:
Кузнецов М. М., Смирнов Д. П., Дедович Т. Г. Разработка модуля импорта заданий системы контроля и оценки знаний студентов Lmsdot // Системный анализ в науке и образовании: сетевое научное издание. 2023. № 2. С. 97-107. EDN: KWUHAJ. URL : https://sanse.ru/index.php/sanse/article/view/582.

DEVELOPMENT OF THE MODULE IMPORT OF TASKS OF THE SYSTEM OF CONTROL AND ASSESSMENT OF STUDENTS' KNOWLEDGE LMSDOT

Kuznetsov Mikhail M.[1], Smirnov Daniil P.[2], Dedovich Tatyana G.[3]

[1]Student;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
e-mail: kmm.19@uni-dubna.ru.

[2]Student;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
e-mail: smirnov@lmsdubna.ru.

[3]Senior Researcher;
Joint Institute for Nuclear Research;
6 Joliot-Curie Str., Dubna, Moscow region, 141980, Russia;
PhD in Physical and Mathematical Sciences, associate professor;

Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
e-mail: tdedovich@yandex.ru.

The article compares the import of tasks in LMS systems. Functional and non-functional requirements are formulated. The tools for working with files were studied and the best options were selected for implementation in the Lmsdot system. Based on the requirements and analysis, the architecture of the job import module was designed. A prototype of the job import module has been implemented and deployed. The developed module allows you to: simultaneously upload several files in text (docs, md and tex) and tabular formats (xlsx and csv) to the Lmsdot system, separate data by options and tasks, create task and work templates, edit and save data.

Keywords: Lmsdot, JS, pandoc, exceljs, import, LMS systems, education.

For citation:
Kuznetsov M. M., Smirnov D. P., Dedovich T. G. Development of the module import of tasks of the system of control and assessment of students' knowledge Lmsdot. *System analysis in science and education*, 2023;(2):97-107 (in Russ). EDN: KWUHAJ. Available from: https://sanse.ru/index.php/sanse/article/view/582.

Введение

LMS (*Learning Management System*) системы являются цифровыми платформами, которые используются для управления обучением и обмена информацией между преподавателями и студентами.

Обязательной частью *LMS*-систем является модуль, позволяющий преподавателям импортировать, экспортировать, создавать и редактировать учебные материалы. Процедура импорта должна гарантировать сохранение различных данных, содержащихся в файлах. К таким данным относятся изображения, таблицы, формулы, списки и текстовые модификаторы (курсив и др.). Функционал такого модуля должен позволять использовать различные форматы файлов, а также включать возможность редактирования перенесенного документа внутри системы.

В университете «Дубна» преподавателями Дедович Т.Г. и Смирновым Д.П. используется система контроля и оценки знаний студентов *Lmsdot*. Данная система требует расширения функционала для создания шаблонов заданий и работ. На данный момент, система *Lmsdot* позволяет создавать шаблоны, используя встроенный редактор.

Поэтому, была поставлена задача: спроектировать и реализовать модуль импорта заданий. Данный модуль позволит создавать шаблоны заданий и работ при помощи загрузки файлов форматов *docs, md, csv, xlsx* и *tex*. При реализации нового модуля проанализированы существующие модули импорта заданий в *LMS*-системах, правильно подобран набор инструментов и встроен модуль в существующую систему.

Анализ импорта данных в LMS-системах

Проведен анализ модуля импорта заданий в разных LMS-системах для определения слабых и сильных сторон данного функционала.

Для анализа были выбраны следующие LMS-системы: как Moodle [1], iSpring Learn [2] и *Docebo* [3]. *Moodle* был выбран, так как используется в университете «Дубна». *ISpring Learn* является одой из самых популярных *LMS*-систем в корпоративном секторе, а также используется вузами. *Docebo* отличается способом создания учебных материалов.

В проанализированных *LMS*-системах присутствует возможность импорта с помощью файла, но для этого требуется:

- Производить редактирование до загрузки файла в систему.
- Знание и использование внутреннего формата данных.
- Использование дополнительных программ, которые упрощают создание внутреннего формата.

- Строгое следование правилам разметки данных внутри файла, с помощью ключей, фигурных скобок и определенного порядка заполнения.
- Знание языка разметки *html*, для добавления изображений, списков, ссылок, таблиц и модификаторов текста.

Ввиду огромных временных затрат на создание такого файла, будет спроектирована отличающаяся модель импорта файлов для создания заданий. Модель должна выполнять следующие условия:

- Приведение файлов к внутреннему формату возьмет на себя система (пользователю нужно только загрузить файл).
- Разметка данных будет проводится с помощью ключей. Данные, после разметки, будут определены в отдельные поля формы шаблона задания.
- Редактирование будет проводится внутри системы после загрузки файла, чтобы преподаватель сразу видел результат разметки данных.

Выбор инструментария для реализации модуля импорта заданий

Система *Lmsdot*, в которую будет встроен модуль импорта заданий, написана на языке программирования *JS* [4]. Язык *JS* имеет различные библиотеки, которые позволяют работать с разными типами файлов, например *docs, xslx, md, csv* и *tex*. Из-за индивидуальной реализации каждой библиотеки, подход к обработке этих типов файлов различен. Поэтому была выбрана утилита *pandoc* [5] за рамками языка *JS*. Данная утилита позволяет приводить множество форматов файлов к одному, например, *json*, который является внутренним представлением данных системы *Lmsdot*.

Формат *json* стандартный текстовый формат для представления структурированных данных на основе синтаксиса объекта языка JS. За счет своей лаконичности, по сравнению с *xml*, формат *json* может быть более подходящим для сериализации сложных структур. Он применяется в веб-приложениях как для обмена данными между браузером и сервером (*AJAX*), так и между серверами (программные *HTTP*-сопряжения).

Кнедостатками pandoc можно отнести отсутствие поддержки *xlsx*. Для работы с файлами формата *xlsx*, можно использовать специализированную библиотеку *exceljs* языка программирования *JS*. Эта библиотека позволяет читать и создавать файлы формата *xlsx* и *csv*.

В итоге была выбрана утилита *pandoc* и библиотека *exceljs*. Данные инструменты будут использоваться на сервере, что позволит передавать клиенту унифицированный формат данных *json*, который позволит обрабатывать данные на клиенте единым образом.

Функциональные требования

Опишем отдельно функциональные требования клиентской и серверной части. На стороне клиентской части преподаватель должен иметь возможность:

- получить заполненные формы шаблонов в редакторе *Lmsdot* при загрузке файлов;
- редактировать и размечать данные в редактора *Lmsdot* перед сохранением шаблона;
- иметь отдельное отображение данных, где преподаватель будет видеть результат после разделения данных по разметке.

Сформулируем требования к серверной части. Сервер должен:

- временно хранить загружаемые документы и их составные части (например изображения);
- по запросу из клиентской части, сервер должен возвращать запрашиваемые файлы;
- периодически очищать временные данные (раз в неделю);
- конвертировать файлы форматов *docs, xlsx, md, csv* и *tex* во внутреннее представление данных формата *json*;
- использовать алгоритм «нахождения островов с пустотами» и фильтровать их по размеру при анализе больших таблиц с пустотами (при работе с файлами формата *xlsx*).

Нефункциональные требования

Сервер должен:

- заимствовать возможности библиотек и технологий, используемые в других сервисах *Lmsdot*;
- при загрузке табличных форматов, должны быть ограничения на размер таблиц (10 столбцов и 30 строк), так как редактор *Lmsdot* корректно отображает таблицы до 10 столбцов.

Проектирование архитектуры модуля

На основе функциональных требований и выбранных инструментов была разработана архитектура модуля импорта и редактирования файлов, которая показана на рис. 1.

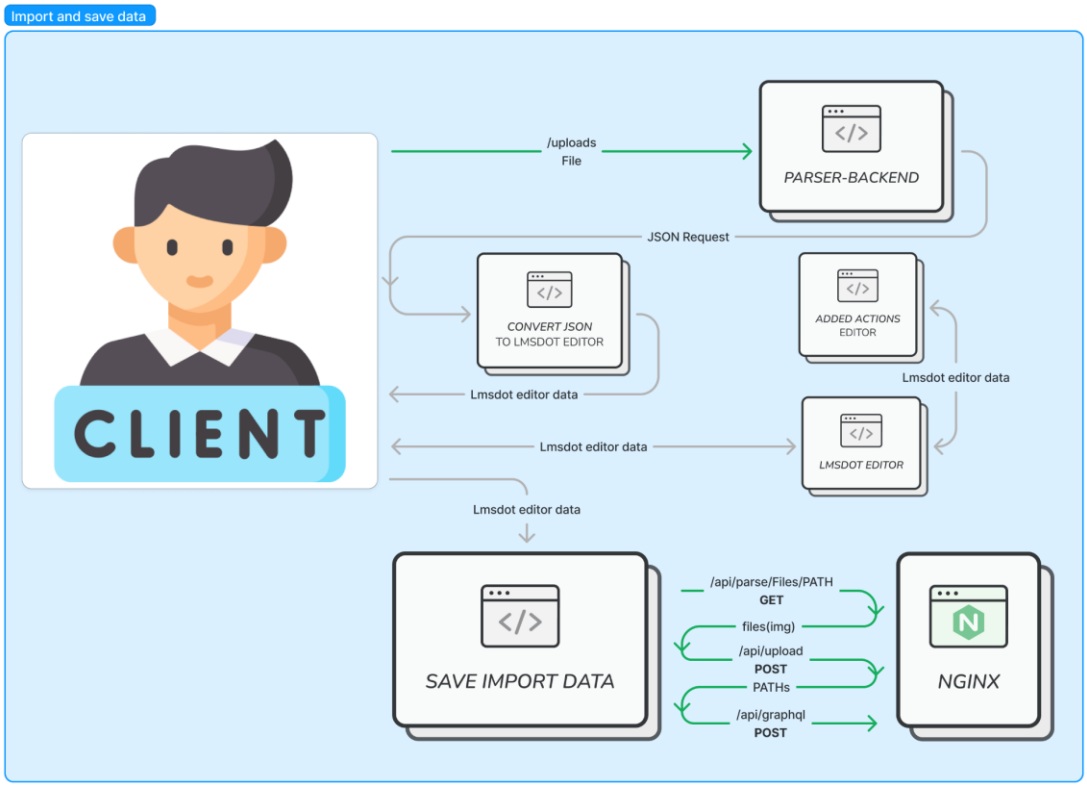

Рис. 1. Процесс импорта файлов, сохранения и редактирования данных, распознаваемых редактором Lmsdot

В начале, файлы через общий сервер *NGINX* [6] отправляются в сервер *PARSE-BACKEND*, который переводит подгружаемые документы в формат *json*. После получения клиентским приложением унифицированных данных в виде *json*, они направляются в модуль конвертирования (*CONVERT JSON TO LMSDOT EDITOR*), который переводит полученные данные в формат, распознаваемый редактором *Lmsdot*.

Клиент может изменять данные, используя редактор системы *Lmsdot* (*LMSDOT EDITOR*). Также он может разделять данные на задания при помощи дополнительного функционала, созданного в режиме импорта файлов (*ADD ACTIONS EDITOR*).

После окончания редактирования, клиент обращается (нажимает на кнопку сохранения) к модулю сохранения данных (*SAVE IMPORT DATA*). Этот модуль переводит изображения из временного файлового хранилища в постоянное. А именно, заменяет в редакторе старые ссылки изображений (сервера *PARSE-BACKEND*) на новые (сервера *UPLOAD-BACKEND*) ссылки. Далее, отправляет запрос на сохранение новых ссылок в основной сервер (*MAIN-BACKEND*). Запросы на получение/загрузку файлов и сохранение данных редактора *Lmsdot* проходят через общий сервер *NGINX*.

Реализация серверной части

Реализацию серверной части представим в виде встраиваемого модуля. Сервер *PARSER-BACKEND* для обработки документа в формат json по пути "/*api*/*parse*" связывается с общим сервером *NGINX*. После внесенных изменений архитектура сервера *Lmsdot* представлена на рис. 2. Подробная модель сервера *PARSER-BACKEND* показана на рис. 3.

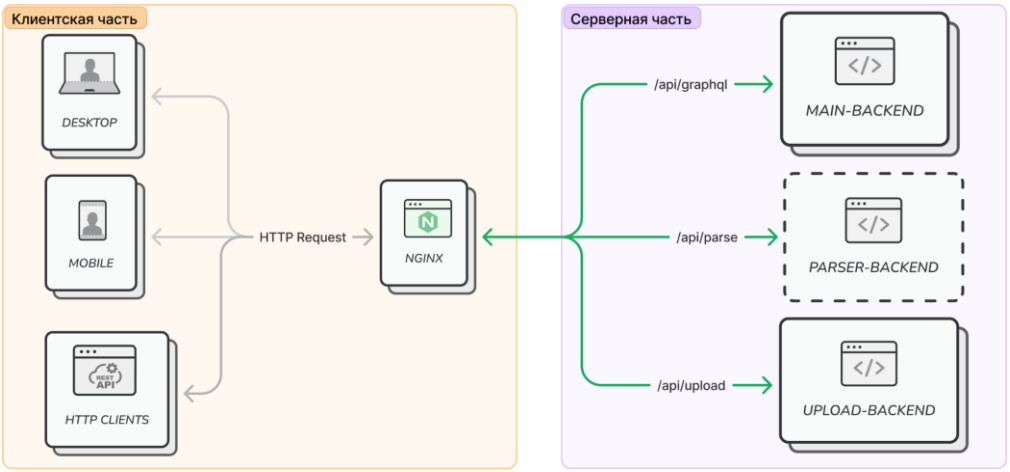

Рис. 2. Дополненная общая модель Lmsdot

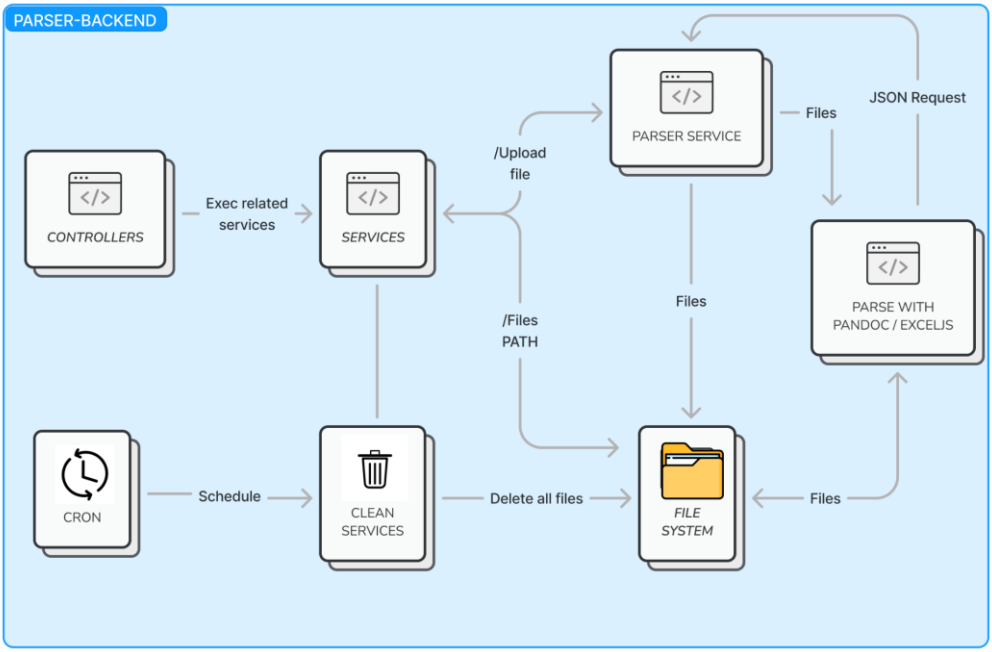

Рис. 3. Модель сервера PARSER-BACKEND для перевода документов в формат json

Внутри *PARSER-BACKEND* запросы маршрутизируются с помощью контроллера (*CONTROLLER*), который распределяет их по определенным сервисам (*SERVICES*). Так клиент имеет возможность обратится к сервису *PARSE SERVICE*, отправив файл. Если расширение файла *xlsx*, то он будет обработан библиотекой *exceljs* с использованием дополнительного кода, реализующего алгоритм «поиска островов с пустотами» (см. Алгоритмическая часть). Данный метод позволяет выделить и разделить друг от друга заполненные участки таблицы. Утилитой *pandoc* обрабатываются файлы с расширениями *docs*, *md*, *csv* и *tex*. Файл формата *json*, полученный от утилиты *pandoc*, будет обработан алгоритмом парсинга (см. Алгоритмическая часть). Данные действия совершаются в модуле *PARSE WITH PANDOC/EXCELJS*, который возвращает клиенту данные в формате *json*.

Утилита *pandoc* при переводе из доступных форматов может переносить изображения в указанную папку файловой системы (*FILE SYSTEM*), при этом записывая относительную ссылку в данные формата *json*. После, при необходимости получения изображения, обращение по выданной сервером ссылке (/*api*/*parse*/*Files*/*), вернет изображение. В *PARSER-BACKEND* также есть сервис очистки временных файлов (*CLEAN SERVICE*), использующий планировщик задач *CRON* [7].

Реализация клиентской части

Так как язык JS позволяет частично обрабатывать веб-страницы на компьютере пользователя без запросов к серверу, то остальной функционал был реализован на стороне клиента.

Модуль, переводящий данные формата json в формат, распознаваемый редактором Lmsdot

Модуль *CONVERT JSON TO LMSDOT EDITOR* переводит данные, получаемые от сервера в формате *json*, в формат, распознаваемый редактором *Lmsdot*, для работы в клиентской части.

Дополнительный функционал редактора Lmsdot

Так как данные в файле не предполагают строгого разделения по полям (название, условие варианта задачи и комментарий), то потребуется использовать ключи, для разделения данных для создания шаблонов, используя редактор *Lmsdot*. Некоторые поля могут содержать только чистый текст, который не учитывает изображения, таблицы и модификаторы шрифтов. Таким образом, в редакторе появятся следующие возможности:

- разделение данных по ключу;
- получения чистого текста из данных.

Отдельное сохранение данных при импорте

Так как возвращаемый формат данных *json* содержит изображения в виде ссылок на сервер с временным хранилищем файлов (*PARSER-BACKEND*), то данные необходимо перенести на сервер с постоянным хранилищем файлов *(UPLOAD-BACKEND*). Для этого необходимо скачать изображение по ссылке из данных формата *json*, загрузить на сервер постоянного хранения файлов *UPLOAD-BACKEND* и изменить ссылки в редакторе *Lmsdot*.

Дальнейшее сохранение данных проходит в стандартном режиме, а именно, отправляется POST запрос на сервер *MAIN-BACKEND* (см. рис. 2).

Такой способ хранения позволит избежать огромного количества изображений в основном хранилище файлов (*UPLOAD-BACKEND*), а также продублировать изображения, которые изначально были ссылкой-изображением на стороннем сайте. При исчезновении ресурса, на который присутствовали ссылки в документе, данные не будут потеряны.

Алгоритмическая часть

Важной частью реализации, являются алгоритмы, использующиеся при обработке файлов. Рассмотрен алгоритм парсинга данных формата *json* получаемых из *pandoc* при обработке файлов форматов *docs*, *md*, *csv* и *tex*. Также рассмотрена обработка файла формата *xlsx*, реализация которого отлична из-за использования библиотеки *exceljs*.

Обработка файлов форматов docs, md, csv и tex

После загрузки файлов и их обработки утилитой *pandoc*, данные будут приведены к *json* формату. Далее, данные формата *json* должны быть приведены к формату редактора *Lmsdot*. Форматом редактора *Lmsdot* является, отличный по структуре, *json*. Перевод будет производить рекурсивная функция, которая принимает массив элементов в виде *json*-объекта. Возвращаемыми данными является массив элементов редактора *Lmsdot*.

Структура *json*, получаемого из pandoc имеет древовидную структуру. Эта структура содержит блочные (таблица, списки, модификаторы текста и др.) и встроенные (изображения, текст, формулы и др.) элементы. Блочные элементы могут содержать в себе блочные и встроенные элементы. Встроенный элемент не имеет вложенности, поэтому он будет крайним элементом, гарантирующим выход из рекурсии.

Обработка файла формата xlsx

Так как *pandoc* не поддерживает формат *xlsx*, то для его обработки была использована библиотека *exceljs*. Данный файл можно представить, как массив страниц. Каждая страница представляет собой двумерный массив [строка, столбец]. Таким образом, мы можем работать с отдельными страницами файла, как с двумерным массивом. Если считать, что таблица начинается с ячейки [0,0], а заканчивается крайней заполненной ячейкой [n,m], то таблица может содержать много пустых строк и столбцов, а ее размеры могут выходить за рамки ограничений. Эту проблему может решить «алгоритм нахождения островов с пустотами», который работает на двумерном массиве и выделяет заполненные элементы таблицы. Таким образом, мы можем получить как, очищенную от пустых строк и столбцов таблицу, так и множество меньших по размеру таблиц, которые находились на расстоянии друг от друга. Данный алгоритм представляет собой модернизированный алгоритм поиска в глубину на двумерной плоскости, для поиска компонент связности.

Опишем основные шаги алгоритма:
Определяем массив, который, по окончанию работы, будет содержать координаты (левый-верхний и правый-нижний) найденных островов.
Выбрать ячейку, которая не была посещена и является заполненной. Отметить эту ячейку как посещенную. Определить и записать максимальный и минимальный индексы по столбцу и строке (индексы будут содержать координаты выбранной точки [i,j]).
В цикле рассмотреть смежные ячейки (верхний [i+1,j], нижний [i-1,j], левый [i,j-1], правый [i,j+1]) для искомой ячейки. Для каждой ячейки выполнить следующие действия:
Если координаты данной ячейки выходят за рамки матрицы, то она не обрабатывается.
Если ячейка посещена, то пропустить ее.
Если ячейка не посещена и не заполнена, то отметить ее как посещенную.
Если ячейка заполнена и не посещена, то обновить максимальный или минимальный индекс по столбцу и строке. Отметить ячейку как посещенную. И вызвать для данной ячейки п.3.
Создать из максимальных и минимальных индексов левое-верхнее и правое-нижнее значение острова. Добавить это значение в массив координат островов.
Если есть не посещенные и заполненные ячейки, то вернутся к п.2.
Вернуть список с индексами крайних элементов островов.

В результате выполнения алгоритма, будет получен массив пар крайних элементов островов. Их можно использовать, чтобы создать таблицы для редактора *Lmsdot*.

Интерфейс

Импорт файла предполагает новый способ взаимодействия с системой. Поэтому необходима отдельная страница или функциональный блок, который лаконично встраивается в существующую часть клиентского приложения.

На рис. 4 представлен дизайн начальной страницы импорта файла.

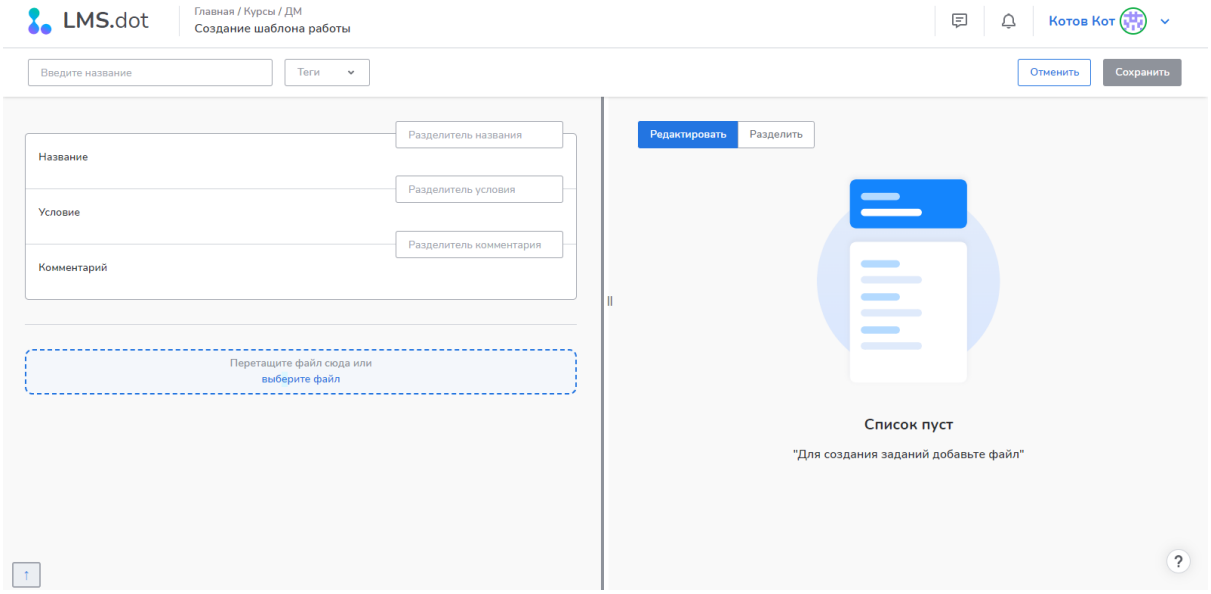

Рис. 4. Интерфейс начальной страницы импорта файла

Смысловые блоки разделены линиями и имеют контрастирующий с фоном цвет. Страница содержит:

- блок с областью выбора и перемещения файла;
- блок ввода ключей, по которым будут разделены данные документа, содержащиеся в редакторе *Lmsdot*;
- редактор *Lmsdot*.

В правой части страницы содержится подсказка для дальнейших действий, например, «для создания заданий добавьте файл». При импортировании файла, не предусмотренного формата, или при выполнении не предусмотренного действия, преподаватель будет оповещен об ошибке во всплывающем окне.

После загрузки файла, преподавателю доступны режимы редактирования и разделения. В первом режиме преподаватель видит все данные документа и имеет возможность их редактировать. Он может указать ключи для параметров, и вставить их в редактируемые данные. Перейдя в режим разделения, клиент разделит редактируемые данные по ключам и получит готовый шаблон задания. После указания названия он может сохранить данные в системе.

Варианты использования

Пошаговые инструкции позволяют пользователям легче ориентироваться в сложных процессах и сокращают время, затрачиваемое на выполнение задач, а также знакомят человека с новыми возможностями системы. Ввиду этого, была создана краткая инструкция, которая познакомит пользователя с функционалом, который дает модуль импорта заданий.

Для успешного создания шаблона работы и задания, пользователь должен зайти на страницу импорта заданий из файла.

Для выбора импортируемых файлов, пользователь может нажать на блок «выбор файла» (1) и выбрать файлы в открывшемся проводнике (2), либо открыть проводник самостоятельно и переместить импортируемые файлы в блок выбора файлов (1) (см. рис. 5).

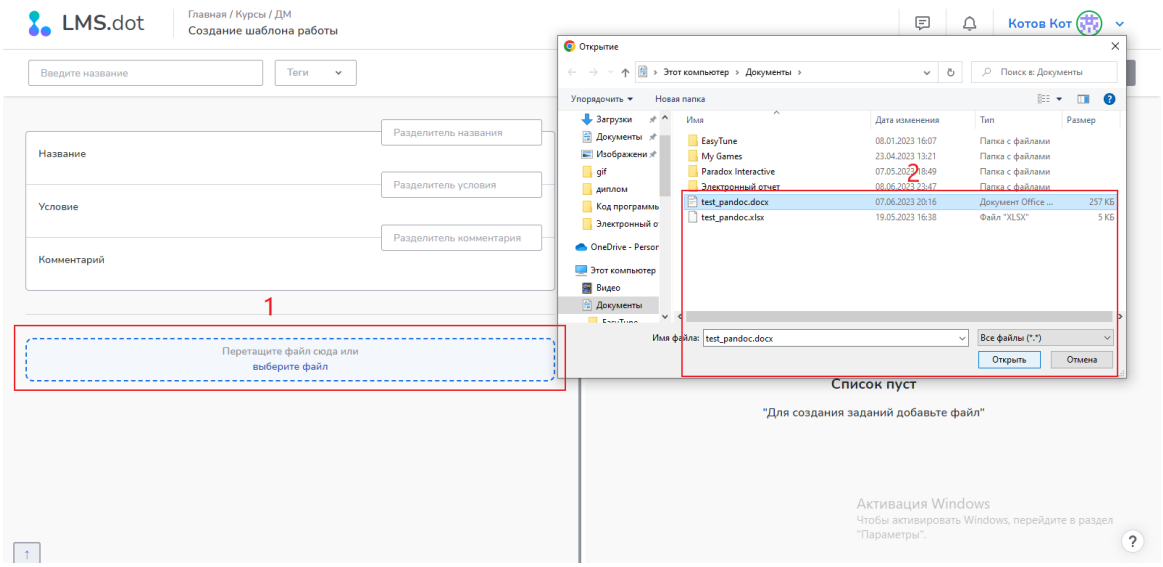

Рис. 5. Выбор импортируемого файла

После импорта файла, пользователь может редактировать конвертированные данные в редакторе Lmsdot (4) (см. рис. 6). Если шаблон задания имеет одно условие или вариант, он может пропустить этап разделения в редакторе и перейти к пункту (5).

Если пользователю нужно создать шаблон работы из нескольких заданий или задание с несколькими вариантами, пользователь вводит ключи в левый верхний блок (3) (см. рис. 6). После, эти ключи, следует вставить перед данными в редакторе (4). Ключи охватывают всю область до следующего ключа. После того как пользователь разметил данные в редакторе, он может разделить их, перейдя в режим «Разделить» (5). Таким образом, можно увидеть корректность разметки (см. рис. 7).

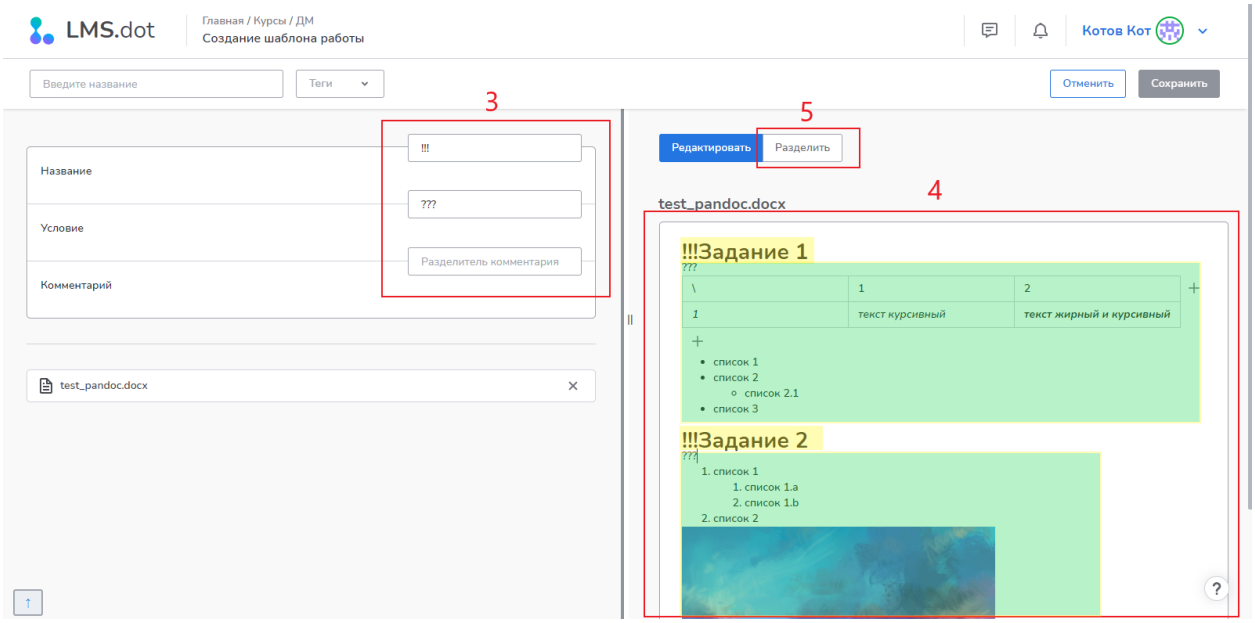

Рис. 6. Разметка данных в редакторе Lmsdot при импорте файла

Перед сохранением данных, все обязательные поля должны быть заполнены (6). Последним шагом будет сохранение данных (7).

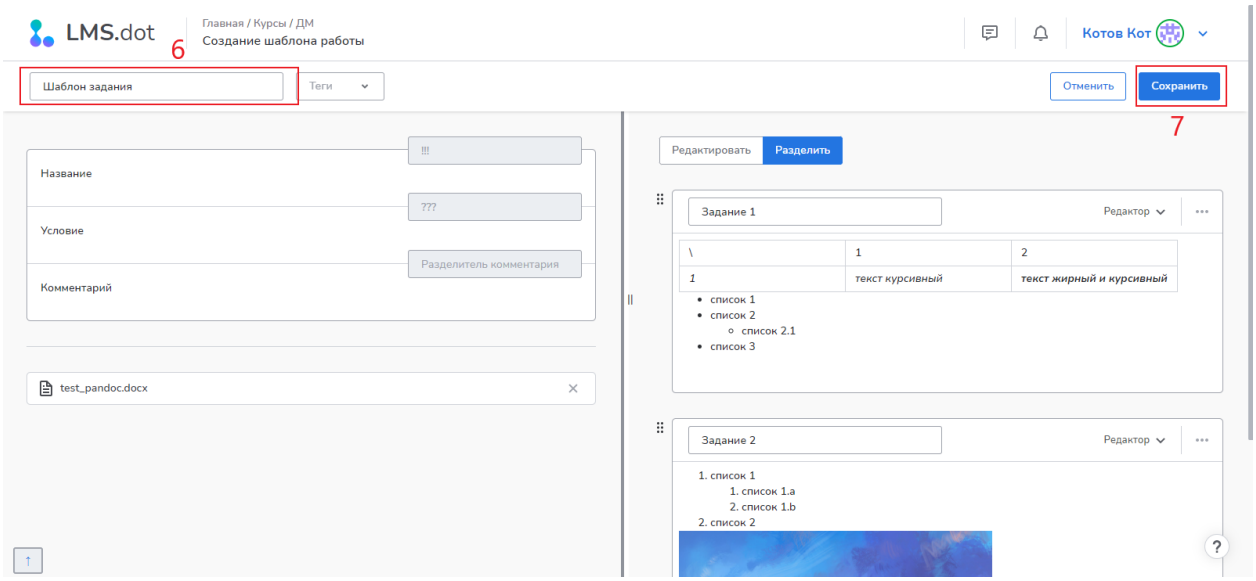

Рис. 7. Сохранение шаблона работы или задания

Заключение

Проведено сравнение импорта заданий в популярных *LMS*-системах, таких, как *Moodle*, *iSpring Learn* и *Docebo*. Сформулированы функциональные и нефункциональные требования к разрабатываемому модулю, а также требования к интерфейсу. Проведен анализ структуры системы *Lmsdot*, технологий и библиотек, использующихся в ней. Изучены средства для работы с файлами и выбраны лучшие варианты для внедрения в систему *Lmsdot*, а именно утилита *pandoc* и библиотека *exceljs* языка программирования *JS*.

На основе поставленных требований и проведенного анализа была спроектирована и внедрена архитектура модуля импорта заданий. Прототип модуля импорта заданий был внедрен в систему контроля и оценки знаний студентов *Lmsdot*.

Разработанный модуль позволяет: одновременно загружать несколько файлов в текстовом (*docs*, *md* и *tex*) и табличном форматах (*xlsx* и *csv*) в систему *Lmsdot*, разделять данные по вариантам и заданиям, создавать шаблоны заданий и работ, редактировать и сохранять данные.

Список источников

1. Moodle – Open-Source learning platform | Moodle.org. – URL: https://moodle.org/ (дата обращения: 22.11.2022).
2. Программы для онлайн-обучения iSpring. – ООО «Ричмедиа», 2001-2023. – URL: https://www.ispring.ru/ (дата обращения: 22.11.2022).
3. Docebo Learning Suite: Learn. Develop. Succees. – Docebo, 2023. – URL: https://www.docebo.com/ (дата обращения: 22.11.2022).
4. JavaScript : [сайт]. – JavaScript.com, 2016 – 2023. – URL: https://www.javascript.com/ (дата обращения: 22.04.2022).
5. Pandoc : a universal document converter. – John MacFarlane, 2006–2022. –URL: https://pandoc.org/ (дата обращения: 25.11.2022).
6. Nginx: что это за сервер, как работает и для чего нужен // Skillfactory media –Честные истории о карьере в IT : [блог-сайт]. – Skillfactory media, 2023. – Дата публикации: 27.03.2023. – URL: https://blog.skillfactory.ru/glossary/nginx/.

7. cron : [демон] // Википедия – свободная энциклопедия. – Wikimedia Foundation, Inc. – Дата обновления: 1 ноября 2022. – URL: https://ru.wikipedia.org/wiki/Cron (дата обращения: 23.04.2022).