МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА API ВЗАИМОДЕЙСТВИЯ БИБЛИОТЕЧНОЙ СИСТЕМЫ ИРБИС И IPSILONUNI

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Акчурина Артема Викторовича

Научный руководитель
зав. каф. техн. пр., к. ф.-м. н. _____ И. А. Батраева

Заведующий кафедрой
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2022

ВВЕДЕНИЕ

Актуальность темы. Библиотека — одна из важнейших структур университета. Она участвует во многих процессах: от создания учебных планов до предоставления литературы для обучения студентам и преподавателям. Большинство процессов университета так или иначе связано с ней.

В 21 веке все сферы жизни постепенно переходят к цифровизации, и структуры университета не стали исключением. Это обуславливается удобством, ведь можно через интерфейс одного устройства проводить различные действия: как искать и изучать необходимую литературу, так и проводить ее учет.

Так как каждая система очень часто представляет собой целостное программное обеспечение, очень неудобно осуществлять какие-то действия между системами. Теряются все преимущества технологий взаимодействия и многое приходится делать вручную. Именно вопрос взаимодействия между системами является слабым местом многого программного обеспечения.

Цель бакалаврской работы — создание и внедрение в систему функционала заполнения учебно-методических материалов для рабочих программ, объединяющего платформу университета IpsilonUni и библиотечную систему ИРБИС. Новые функции позволят упростить взаимодействие между преподавателями университета и сотрудниками библиотеки, а также смогут предоставить одну среду работы с обеими системами, тем самым избавят сотрудников от дополнительных нагрузок и сэкономят много времени.

Поставленная цель определила **следующие задачи**:

- изучение принципов работы библиотечной системы ИРБИС;
- написание API взаимодействия с сервером библиотеки;
- создание сервиса взаимодействия между системами IpsilonUni и ИРБИС.

Методологические основами в процессе разработки API взаимодействия библиотечной системы ИРБИС и IpsilonUni послужили работы Д. Чамберса, Д. Пэкетта, С. Тиммса, С. Руби, Д. Томаса, Д. Хенссона, а также С. Г. Синицы.

Теоретическая значимость бакалаврской работы заключается в анализе и формировании основных особенностей и подходов при разработке систем, взаимодействующих между собой, а также выбору технологий для этого.

Практическая значимость бакалаврской работы заключается в разработке уникального сервиса взаимодействия систем, который к тому же может быть внедрен в систему и использоваться.

Структура и объём работы. Бакалаврская работа состоит из введения, 2 разделов, заключения, списка использованных источников и 1 приложения. Общий объем работы — 62 страницы, из них 37 страниц — основное содержание, включая 19 рисунков и 1 таблицу, приложение, список использованных источников информации — 21 наименование.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Использование API и REST при разработке ПО» посвящён рассмотрению различных подходов к проектированию приложений.

API. API (Application Programming Interface) представляет собой набор определенных и стандартизированных правил. Они позволяют разделить работу приложения на слои и компоненты, тем самым помогая создать правильную структуру приложения, с которой удобно работать.

У API имеются следующие преимущества:

— Интеграция новых систем с уже существующими. Разработчику не нужно заново писать все функции, он может использовать доступные.

— Инновации. С появлением новых сервисов, компаниям не нужно переписывать весь код своих приложений, достаточно внести изменения на уровне API.

— Расширение. API позволяет приложениям выполнять свои функции на разных платформах. Таким образом, любая компания имеет возможность предоставить доступ к своей внутренней структуре с помощью использования аналогичного интерфейса взаимодействия.

— Простота обслуживания. API представляет собой шлюз между системами. И каждая из систем должна изменять свое внутреннее устройство без затрагивания самого API, чтобы изменения одной системы не повлияло на работу другой.

Архитектурный стиль REST. REST (Representational State Transfer) — архитектурный стиль, обеспечивающий стандарт между взаимодействием систем в сети. Он характеризуется тем, что не сохраняет состояние. Это означает, что серверу не нужно знать о клиенте никакой информации и все необходимое находится в самом запросе.

Также RESTful системы разделяют сервер и клиента. Клиенты отправляют запросы на определенные действия, а серверы их получают, обрабатывают и отправляют обратно ответ. Схема архитектурного стиля REST представлена на изображении 1.

Как правило, запросы состоят из следующих частей:

— HTTP-метода, который определяет тип операции;

— заголовка;

— пути к ресурсу;

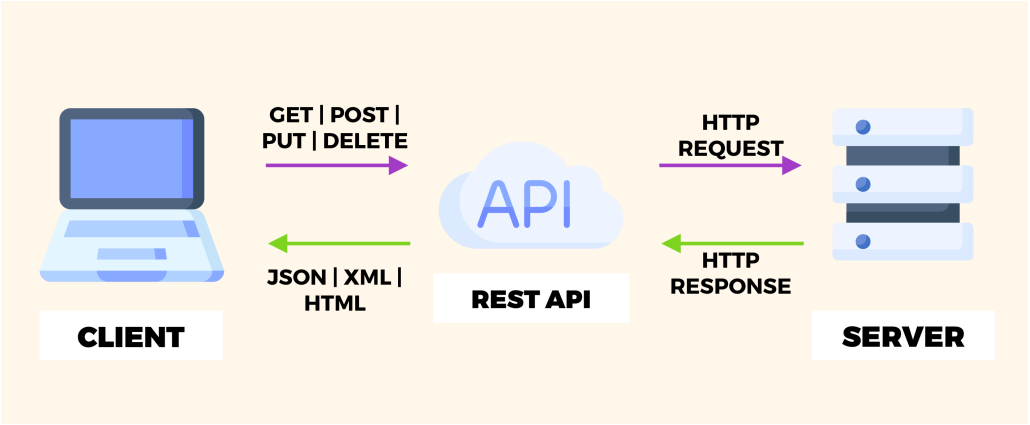

Рисунок 1 – Схема взаимодействия RESTful API

— тела сообщения, в котором содержатся данные.

Основными методами по взаимодействию с ресурсами в RESTful системах являются:

— GET — получение ресурса.
— POST — создание ресурса.
— PUT — обновление уже существующего ресурса.
— DELETE — удаление определенного ресурса.

А одним из наиболее популярных форматов для передачи данных является JSON (JavaScript Object Notation). Он представляет собой формат, в котором хранится информация об объектах в удобном формате вида «ключ-значение».

Платформа ASP.NET. ASP.NET представляет собой платформу, которая служит для создания веб-приложений. Она является расширением .NET Core, включает в себя множество инструментов для создания серверных веб-приложений. Также предлагает кросс-платформенность, масштабируемость и поддержку Web API.

У платформы имеются следующие особенности. Первым из них является наличие множества доступной функциональности, предоставляемой встроенной библиотекой классов.

Следующее важное отличие заключается в том, что код ASP.NET компилируется, а не интерпретируется. Приложения проходят через два этапа компиляции. Сначала код, написанный на языке программирования высокого уровня компилируется в код на промежуточном языке MSIL/IL. Именно за счет промежуточного языка и обеспечивается возможность использовать в различные языки. И только уже при выполнение страницы происходит второй этап, где код на языке IL компилируется в низкоуровневый код на машинном языке.

Помимо этого ASP.NET предоставляет следующие преимущества в виде расширяемости метаданных, безопасности типов, многопоточности и структурированной обработке ошибок. Также он позволяет с легкостью совершать развертывание и конфигурирование приложений на серверах.

ASP.NET во многом использует подход MVC, который рассмотрим сейчас подробнее.

Подход MVC. MVC (Model-View-Controller) — популярный архитектурный подход для веб-приложений. Представляет собой схему взаимодействия отдельных частей приложения, разделяет логику работы на составляющие и разграничивает сферы ответственности. Ее основными частями являются:

1. Model (модель) — представление, структура данных предметной области.
2. View (представление) — визуальное отображение данных.
3. Controller (контроллер) — взаимодействие между моделью и представлением.

А их общая идея взаимодействия описывается следующим образом. Она представляет собой страницы-представления, где отображается информация из моделей (объектов базы данных), за доставку которых отвечают контроллеры.

Общая схема взаимодействия частей MVC представлена на рисунке 2.

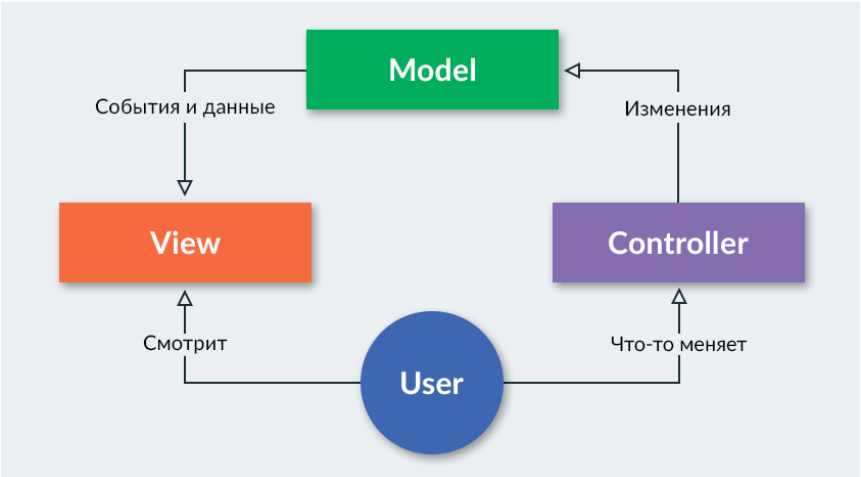

Рисунок 2 – Схема взаимодействия частей MVC

Преимущество подхода MVC состоит еще в том, что модификация каждого компонента может производиться независимо от других, что значительно повышает эффективность разработчиков. Можно изменять, например, только обработку данных и совершенно не затрагивать их отображение. Именно в таких задачах подход наиболее доказывает свою эффективность.

Второй раздел «Реализация сервиса взаимодействия ИРБИС и IpsilonUni» посвящён реализации интеграции библиотечной системы ИРБИС и системы дистанционного обучения IpsilonUni. В нем описывается получение и согласование технического задания, проработка подхода к его решению, процесс написания программы и устранение возникающих в ходе этого проблем, а также финальное интеграционное тестирование.

Постановка задачи. IpsilonUni является дистанционным образовательным порталом для обучения, контроля успеваемости студентов и общения с преподавателями. Он предоставляет функционал для работы с учебно-методическими курсами и тестами, учебными планами и рабочими программами, фиксирует ход процесса обучения и многое другое.

ИРБИС представляет собой систему автоматизации библиотек, занимающуюся всеми оборотами библиотечных изданий, учебников и другой литературой. Она позволяет как хранить необходимую информацию в базах данных, так и предоставляет большой функционал для описания и обработки любых видов изданий.

Основной проблемой этих систем стал неудобный процесс работы с учебно-методическими материалами для рабочих программ. На текущий момент, весь этот процесс происходит в обход систем: преподаватели составляют список, вручную согласуют его с библиотекой, и только в самом конце отдают его администраторам платформы UpsilonUni, которые просто размещают его на сайте.

Для упрощения процесса и приведения его к единому стандарту согласования, было принято решение внедрить в IpsilonUni новый функционал, затрагивающий обе платформы.

Были поставлены основные пункты технического задания. Преподаватель, во время создания рекомендуемого списка литературы для рабочей программы, должен видеть доступные в библиотеке учебники и иметь возможность выбрать необходимые ему. Вследствие выбора набора литературы, этот список должен быть виден как преподавателю, так и сотрудникам библиотеки, которые этот список должны будут проверить. Проверяющий сможет либо утвердить список, либо сделать какие-то определенные замечания по конкретным позициям и вернуть на доработку.

Так как для этого функционала нужны были данные из обеих систем, то главной задачей было осуществить процесс интеграции между сервисами:

IpsilonUni, где производятся все необходимые действия по выбору и утверждению списка литературы из рабочих программ, и библиотечными сервисами ИРБИС, в которых находится вся информация о существующих учебно-методических пособиях.

Подход к решению.

В системе IpsilonUni на текущий момент уже существовали рабочие программы. В них был и раздел со списком рекомендуемой литературы. Но на тот момент он представлял собой лишь текстовое поле, которое просто заполнялось администратором после получения от преподавателей списка, одобренного библиотекой.

Необходимо было сделать какую-то форму, в которой преподаватель должен был вводить автора или название учебной литературы и получать список учебных пособий, содержащих искомые термины. Затем он бы выбирал нужную ему позицию и добавлял ее в список. После проведения процедур по формированию необходимого набора рекомендуемой литературы, он должен был сохранить его нажатием соответствующего элемента на странице.

Все запросы, которые будут уходить со стороны IpsilonUni при поиске литературы, должны кем-то обрабатываться. Для этого необходимо было сделать специальный сервис, который бы все эти запросы принимал, каким-то образом обрабатывал и возвращал нужный ответ. Было принято решение вынести это функционал в отдельный промежуточный сервер.

Сам сервис должен был использовать API и обращаться к серверу и базам данных ИРБИС для получения необходимой информации. После ее получения и обработки, нужно было сформировать данные определенным образом и отправить обратно на сторону IpsilonUni.

Таким образом общий вид предполагаемой схемы представлен на рисунке 3.

API взаимодействия с библиотекой.

Первой задачей, которую необходимо решить, является написание API взаимодействия с библиотекой. Прежде всего нужно определить схему коммуникации с их сервером. В процессе изучения документации была найдена библиотека функций «irbis64_client.dll», которая предназначалась для полнофункционального доступа к базам данных ИРБИС на основе архитектуры взаимодействия клиента и сервера.

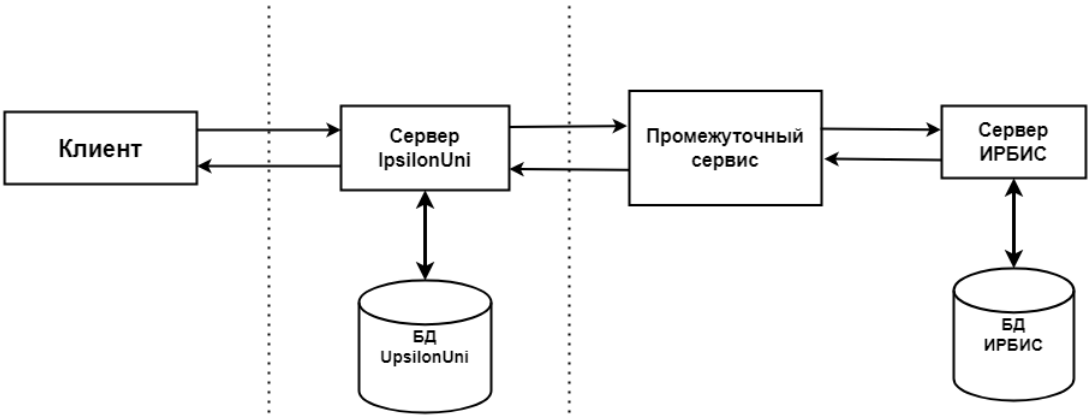

Рисунок 3 – Общая схема процесса

Следующим этапом, для большего удобства представления работы, будут написаны специальные «функции-обертки», позволяющие проводить определенные действия в системе ИРБИС.

После этого необходимо написать функционал регистрации пользователей в системе. Далее, для чтения записей из баз данных системы ИРБИС, были написаны специальные классы. Следующим этапом была реализация поиска, основанная на словарях, которая бы позволяла искать необходимые элементы среди всего набора данных библиотечной системы. А после получения информации ее было необходимо привести в правильный вид, удовлетворяющий стандартам, проведя форматирование.

Также стоит помнить, что помимо самого описания учебно-методических пособий, нужно приложить ссылку на это издание в системе library.sgu.ru. Для этого ее необходимо правильно сформировать, для чего были написаны отдельные функции. В ее параметрах содержится название базы данных, тип запроса, специальное выражение, пользователь, под которым необходимо совершить авторизацию на сайте и другая информация. Само поисковое выражение представляет собой поиск по определенному заданному индексу. Для реализации этой логики написана дополнительная функциональность.

В конце, всю эту информацию нужно было правильно сформировать в массив json-элементов для дальнейшей отправки на сторону IpsilonUni.

В общем итоге, в ходе разработки API были написаны следующие классы:

1. с логикой работы — BooksController, BookService, Irbis;
2. вспомогательные классы — Book, BookFormat, SearchExpression, BookLink, Util;

3. классы настроек и типов — ARM, Field, ReturnCode, TypeSearch, Settings.

Создание RESTful сервиса. Для того, чтобы обрабатывать запросы, нужно правильно настроить сервер. Логика работы будет заложена в WEB API-приложение. Для этого необходимо написать RESTful сервис, который будет обрабатывать поступающие на сервер запросы. Само решение будет создано на платформе ASP.NET.

Для нашего функционала нужен метод поиска книг. Вполне достаточно реализовать один метод, но в котором благодаря параметрам будет возможность конфигурирования типа поиска. Из интересующих нас: поиск по автору, названию, идентификатору, ключевым словам и по всем этим пунктам сразу. Ожидаемый формат запроса будет иметь вид «host:port/books?typeSearch=x&query=y», где host:port/books — путь к ресурсу, параметр typeSearch будет отвечать за тип поиска, а query представлять собой сам поисковый запрос.

В зависимости от типа запроса будет выбираться определенное поведение, заложенное в реализованное ранее API взаимодействия. Результатом выполнения будет сформированный ответ.

Таким образом общий вид предполагаемой схемы взаимодействия основных классов в процессе интеграции представлен на изображении 4.

Тестирование интеграции. Теперь, когда весь функционал готов, протестируем написанное приложение в полном виде. На стороне IpsilonUni при создании списка учебно-методических материалов в элемент поиска введем название пособия, например, «java», как изображено на рисунке 5.

В этот момент IpsilonUni отправит GET запрос вида localhost:5001/books?typeSearch=keyWord&query=java, где в параметрах передается тип запроса и само поисковое выражение. Промежуточный сервер обработает его, взяв необходимые данные со стороны ИРБИС, и вернет в теле ответа на сторону IpsilonUni список литературы в формате массива json, каждый элемент которого хранит идентификатор, описание и ссылку. Формат ответа приведен на изображении 6.

Далее на стороне IpsilonUni будет произведено расформатирование данных и установка полученного результата в необходимые места. Список отобразится на странице заполнения учебно-методических материалов в виде выпадающего списка библиографических описаний. После выбора нужной позиции и нажатия элемента «Добавить», описание и ссылка на материал в электронном каталоге library.sgu.ru будут добавлены в таблицу, как на изображении 7.

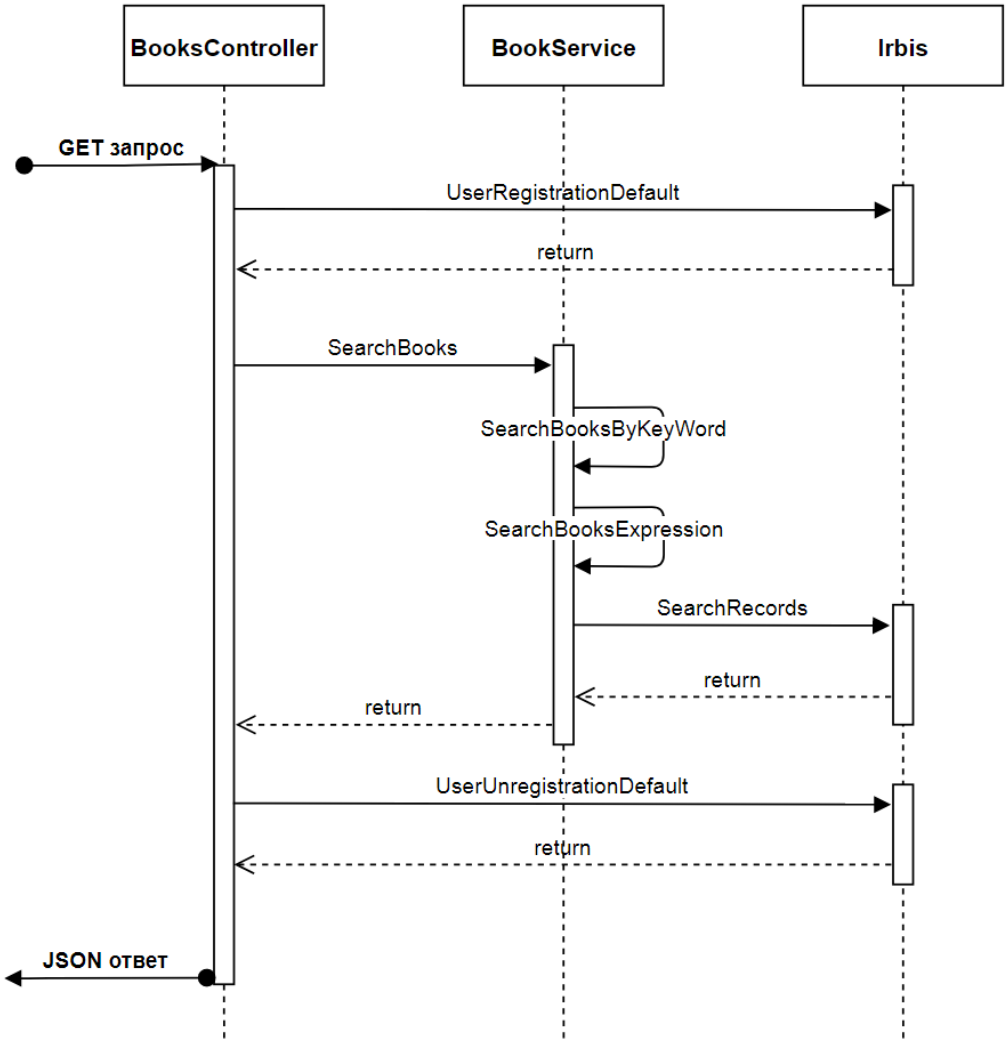

Рисунок 4 – Общая схема процесса

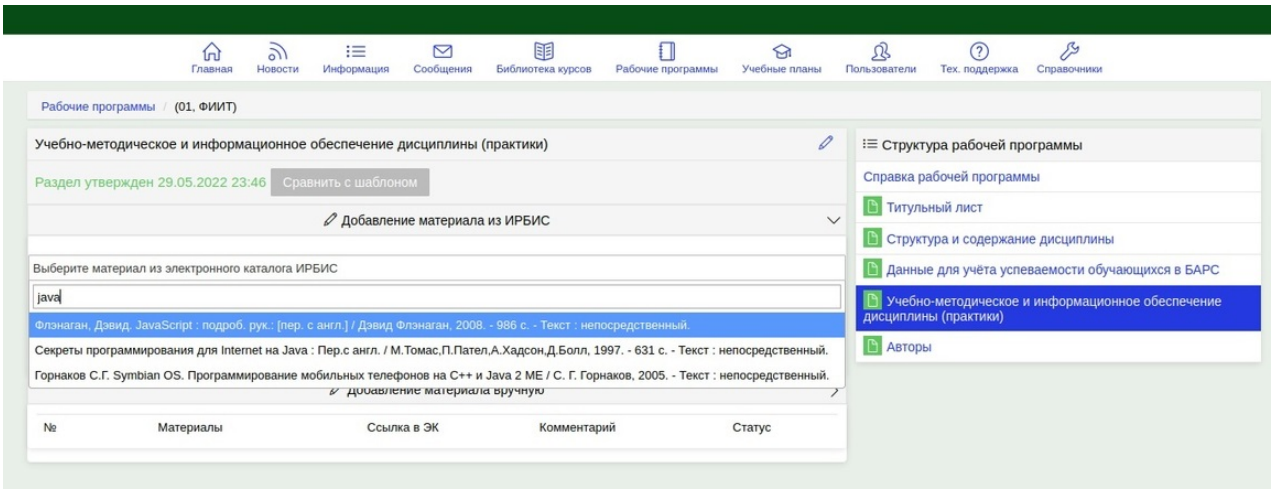

Рисунок 5 – Форма ввода названия материала

```
[
    {
        "id": 15,
        "description": "Флэнаган, Дэвид. JavaScript : подроб. рук.: [пер. с англ.] / Дэвид Флэнаган, 2008. - 986 с. - Текст : непосредственный.",
        "url": "http://library.sgu.ru/cgi-bin/irbis64r_plus/cgiirbis_64_ft.exe?S21FMT=briefHTML_ft&C21COM=S&I21DBN=NIKA_FULLTEXT&P21DBN=IBIS&Z21ID=GUEST&S21ALL=%3C.%3EI=15%3C.%3E"
    },
    {
        "id": 17,
        "description": "Секреты программирования для Internet на Java : Пер.с англ. / М.Томас,П.Пател,А.Хадсон,Д.Болл, 1997. - 631 с. - Текст : непосредственный.",
        "url": "http://library.sgu.ru/cgi-bin/irbis64r_plus/cgiirbis_64_ft.exe?S21FMT=briefHTML_ft&C21COM=S&I21DBN=NIKA_FULLTEXT&P21DBN=IBIS&Z21ID=GUEST&S21ALL=%3C.%3EI=17%3C.%3E"
    },
    {
        "id": 21,
        "description": "Горнаков С.Г. Symbian OS. Программирование мобильных телефонов на C++ и Java 2 ME / С. Г. Горнаков, 2005. - Текст : непосредственный.",
        "url": "http://library.sgu.ru/cgi-bin/irbis64r_plus/cgiirbis_64_ft.exe?S21FMT=briefHTML_ft&C21COM=S&I21DBN=NIKA_FULLTEXT&P21DBN=IBIS&Z21ID=GUEST&S21ALL=%3C.%3EI=21%3C.%3E"
    },
]
```


Рисунок 6 – Формат ответа сервера

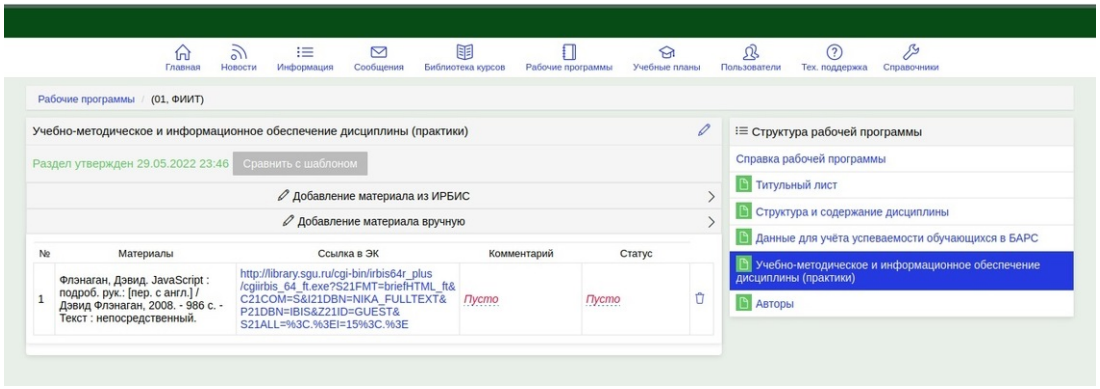

Рисунок 7 – Заполнение таблицы учебно-методических материалов

В дальнейшем сотрудники библиотеки смогут проверить список, созданный преподавателем, подтвердить подходящие позиции, отметив это в соответствующем поле таблицы, или написать комментарий к неподходящим пособиям.

ЗАКЛЮЧЕНИЕ

В процессе выполнения работы было произведено изучение проблемы, формирование технического задания, общение между сторонами и опыт разработки в команде. Итогом проделанной работы стало написание API взаимодействия с библиотекой ИРБИС и сервис интеграции между системами IpsilonUni и ИРБИС, которая в будущем значительно облегчит работу как преподавателям, так и сотрудникам библиотеки.

Основные источники информации:

1. *Чамберс, Д.* ASP.NET Core. Разработка приложений / Д. Чамберс, Д. Пэкетт, С. Тиммс. — СПб.: Питер, 2018. — С. 464. — Яз. рус.
2. *Руби, С.* Rails 4. Гибкая разработка веб-приложений / С. Руби, Д. Томас, Д. Хенссон. — СПб.: Питер, 2014. — С. 448. — Яз. рус.
3. *Синица, С. Г.* Веб-программирование и веб-сервисы / С. Г. Синица. — Краснодар: Кубанский государственный университет, 2013. — С. 158. — Яз. рус.
4. Система автоматизации библиотек ИРБИС64. Общее описание системы. — М.: Ассоциация ЭБНИТ, 2018. — С. 498. — Яз. рус.
5. Учебник. Создание веб-API с помощью ASP.NET Core [Электронный ресурс]. — URL: https://docs.microsoft.com/ru-ru/aspnet/core/tutorials/first-web-api?view=aspnetcore-6.0&tabs=visual-studio (Дата обращения 18.05.2022). Загл. с экр. Яз. рус.