

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ КЫРГЫЗСКОЙ РЕСПУБЛИКИ**  
**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ**  
**ФЕДЕРАЦИИ**

**Государственное образовательное учреждение**

**высшего профессионального образования**

**КЫРГЫЗСКО-РОССИЙСКИЙ СЛАВЯНСКИЙ УНИВЕРСИТЕТ**

**имени Первого Президента Российской Федерации Б.Н. Ельцина**

**ЕСТЕСТВЕННО – ТЕХНИЧЕСКИЙ ФАКУЛЬТЕТ**

**Кафедра информационных и вычислительных технологий**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**на тему:**

**Разработка WEB-сайта национальной электронной библиотеки периодических**  
**изданий ВАК КР**

**Выполнил студент группы**

**ЕПИ-2-18**

**Лазарев Дмитрий Денисович**



**Руководитель**

**к.т.н., доцент Верзунов С.Н.**



**Работа к защите допущена**

**заведующей кафедрой ИВТ**

**д.т.н., проф. Лыченко Н.М.**



**БИШКЕК 2022**

## **Аннотация**

Данная выпускная квалификационная работа посвящена разработке Web-сайта для сбора, анализа, хранения статей и метаданных научно-периодических изданий Кыргызской Республики.

Для решения поставленной задачи нами были выполнены следующие работы:

- Провести анализ предметной области.
- Разработать web-платформу, предоставляющая возможность научным работникам получать полную информацию о статьях и журналах
- Реализовать web-платформу с простым и понятным интерфейсом
- Реализовать алгоритмы импорта данных.
- Реализовать создание и сохранение метаданных статей в системе.

Разработанная система выполняет следующие функции:

- Создание новых журналов.
- Создание новых выпусков
- Создание новых статей
- Загрузка метаданных статей из файла в формате OJS
- Редактирование метаданных статей и журналов вручную.
- Сохранение метаданных в системе.

Средства разработки:

- Среда разработки: PyCharm, Visual Studio Code
- Язык программирования: Python, HTML (JS, CSS)
- Фреймворк: Django
- База данных: MySQL

Объем пояснительной записки: 49 стр.

Количество рисунков: 17 шт.

Количество таблиц: 15 шт.

## Аннотация

Бул жыйынтыктоочу квалификациялык иш Кыргыз Республикасынын илимий мезгилдүү басылмаларынын макалаларын жана метамаалыматтарын чогултуу, талдоо, сактоо үчүн веб-сайтты иштеп чыгууга арналган.

Милдетти чечүү үчүн төмөнкү иштерди аткаруу зарыл:

- Предметтик чөйрөгө талдоо жүргүзүү.
- Изилдөөчүлөргө макалалар жана журналдар тууралуу толук маалымат алуу мүмкүнчүлүгүн берген веб-платформаны иштеп чыгуу
- Жөнөкөй жана интуитивдик интерфейси бар веб-платформаны ишке ашыруу
- Маалыматтарды импорттоо алгоритмдерин ишке ашыруу.
- Системада макаланын метаберилиштерин түзүү жана сактоону ишке ашыруу.

Иштелип чыккан система төмөнкү функцияларды аткарат:

- Жаңы журналдарды түзүү.
- Жаңы чыгарылыштарды түзүү
- Жаңы макалаларды түзүнүз
- OJS форматындагы файлдан макала метадайындары жүктөлүүдө
- Макалалардын жана журналдардын метаберилиштерин кол менен түзөтүү.
- Системада метаберилиштерди сактоо.

Иштеп чыгуу куралдары:

- Иштеп чыгуу чөйрөсү: PyCharm, Visual Studio Code
- Программалоо тили: Python, HTML (JS, CSS)
- Framework: Django
- Маалыматтар базасы: MySQL

Түшүндүрмө каттын көлөмү: 49 барак.

Сүрөттөрдүн саны: 17 даана.

Үстөлдөрдүн саны: 15 даана.

## **Annotation**

This final qualification work is devoted to the development of a Web site for the collection, analysis, storage of articles and metadata of scientific periodicals of the Kyrgyz Republic.

To solve the task, it is necessary to perform the following work:

- Conduct an analysis of the subject area.
- Develop a web-platform that provides an opportunity for researchers to receive complete information about articles and journals
- Implement a web-platform with a simple and intuitive interface
- Implement data import algorithms.
- Implement the creation and saving of article metadata in the system.

The developed system performs the following functions:

- Creation of new journals.
- Creation of new releases
- Create new articles
- Loading article metadata from a file in OJS format
- Editing the metadata of articles and journals manually.
- Saving metadata in the system.

Development tools:

- Development environment: PyCharm, Visual Studio Code
- Programming language: Python, HTML (JS, CSS)
- Framework: Django
- Database: MySQL

The volume of the explanatory note: 49 pages.

Number of drawings: 17 pcs.

Number of tables: 15 pcs.

## Оглавление

Введение.....	6
1.1 Введение .....	10
1.2. Позиционирование .....	11
1.3. Описания пользователей .....	12
1.4. Краткий обзор изделия .....	15
1.5. Возможности продукта.....	16
1.6. Ограничения .....	16
1.7. Показатели качества .....	16
1.8. Другие требования к Web-платформе .....	17
1.9. Требования к документации.....	17
2. Спецификация требований к программному продукту .....	18
2.1. Введение .....	18
2.2. Общее описание.....	19
2.3. Специфические требования.....	27
3. Проектирование и конструирование программного обеспечения .....	29
3.1 Разработка диаграммы классов.....	29
3.2 Определение классов-сущностей.....	29
3.3 Разработка диаграмм компонентов.....	31
3.4. Разработка диаграмм последовательностей .....	32
4. Разработка тестов и тестирование программного продукта .....	34
4.1. Разработка плана тестирования .....	34
4.2. Модульное тестирование .....	35
4.3. Системное тестирование .....	37
5. Руководство пользователя .....	39
5.1. Назначение системы.....	39
5.2. Условия применения системы .....	39
5.3. Подготовка системы к работе .....	39
Заключение.....	46
Список литературы .....	47
Приложение 1. Глоссарий .....	48
Приложение 2. Листинг .....	50

## Введение

В настоящее время для публикации научных работ в национальной академии наук используется система разметки метаданных и цифрового сопровождения научных журналов. Программа разметки OJS предназначена для подготовки выпусков и журналов для размещения статей на сайтах научных организаций в то время, как нашей задачей является централизованное хранение и обеспечение бесперебойного доступа к научным статьям. Тем не менее OJS является наиболее часто используемой системой цифровой поддержки научных журналов в мире, поэтому на первый план выходит задача обеспечения импорта метаданных научных статей из системы OJS в разрабатываемый нами репозиторий, как правило, эти данные хранятся с расширением .xml.

В настоящий момент в мире насчитывается более десятка систем поддержки цифровых хранилищ (институциональных репозитриев). Наиболее популярные из них:

**DSpace** – это самое популярное в академической среде ПО для создания архива электронных ресурсов (институционального цифрового репозитория). DSpace обеспечивает платформу для долгосрочного хранения цифровых материалов (изображения, медиафайлы, документы в различных форматах и т. п.), используемых в академических исследованиях. Платформа DSpace разрабатывалась совместно компанией HewlettPackard и библиотеками MIT (Massachusetts Institute of Technology). Движение Scholarly Communication оказало влияние на развитие DSpace, вследствие чего конфигурация по умолчанию направлена на поддержку научных публикаций. Для базовой организации данных в DSpace зафиксирована определенная модель данных, основанная на схеме Dublin Core и ее расширениях. Система хранит (конвертирует) и индексирует метаданные в разнообразных форматах (DIM, MODS, METS, QDC, XOAI, MARC, RUSMARC, МЕКОФ, ORE и др.). Список форматов может быть расширен добавлением новых, в том числе собственной генерации, конвертеров. Поддерживается резервное копирование контента и система LOCKSScompliant для организации надежного хранилища данных. Система хранит информацию о пользователях системы и поддерживает авторизацию и разграничивает доступ к содержимому репозитория. Кроме того, такие функции, как депонирование и редакторская проверка, привязаны к пользователям. DSpace работает со стандартными для библиотечной сферы протоколами OAI-PMH, OpenURL (ANSI/NISO Z39.88-2004 (R2010) – The OpenURL Framework for Context-Sensitive Services. National Information Standards Organization) и SWORD C 2009 г. DSpace поддерживается сообществом

DURASpace, которое образовано путем слияния двух проектов цифровых репозиториях DSpace Foundation и Fedora Commons. Fedora (Flexible Extensible Digital Object Repository Architecture) – репозиторий разработан исследователями из Корнельского университета в 1997 г. в качестве платформы для хранения, управления и доступа к цифровому контенту (цифровым объектам) [9]. Открытое (лицензия Apache License, Version 2.0) Java приложение поддерживает резервное копирование контента и программу LOCKSS-compliant для обеспечения надежного хранения ресурсов. Fedora определяет набор абстракций для выражения цифровых объектов, отношения между цифровыми объектами, их связи и поведение. В отличие от DSpace, Fedora больше подходит для хранения произвольных цифровых объектов, например, ПО. Ядро репозитория Fedora предоставляет набор веб-сервисов с четко определенными API. Кроме того, Fedora предоставляет широкий спектр вспомогательных сервисов и приложений, включая поиск, поддержку протоколов OAI-PMH и SWORD, обмен сообщениями, управление клиентами и многое другое. Метаданные представлены в формате Dublin Core. Метаданные могут быть преобразованы в форматы METS, FOXML, Atom. Обеспечивается поддержка RDF. [1]

**Fedora** (Flexible Extensible Digital Object Repository Architecture) – репозиторий разработан исследователями из Корнельского университета в 1997 г. в качестве платформы для хранения, управления и доступа к цифровому контенту (цифровым объектам) [9]. Открытое (лицензия Apache License, Version 2.0) Java приложение поддерживает резервное копирование контента и программу LOCKSS-compliant для обеспечения надежного хранения ресурсов. Fedora определяет набор абстракций для выражения цифровых объектов, отношения между цифровыми объектами, их связи и поведение. В отличие от DSpace, Fedora больше подходит для хранения произвольных цифровых объектов, например, ПО. Ядро репозитория Fedora предоставляет набор веб-сервисов с четко определенными API. Кроме того, Fedora предоставляет широкий спектр вспомогательных сервисов и приложений, включая поиск, поддержку протоколов OAI-PMH и SWORD, обмен сообщениями, управление клиентами и многое другое. Метаданные представлены в формате Dublin Core. Метаданные могут быть преобразованы в форматы METS, FOXML, Atom. Обеспечивается поддержка RDF. [1]

**EPrints** – это вторая в академическом мире по популярности после DSpace система, которая используется для формирования и управления открытыми архивами и предназначена для поддержания институциональных репозиториях открытого доступа и создания архивов научных исследований с большим разнообразием ИР (научные статьи, отчеты, диссертации, монографии, учебно-методические пособия, материалы конференций, данные результатов экспериментов и наблюдений и т. п.). EPrints был разработан при университете Саутгемптона. Модель данных EPrints отличается от DSpace, использующего строгую иерархическую

систему организации данных, которая позволяет отразить структуру организации тем, что все записи эквивалентны и являются одноуровневыми. Открытые архивы, созданные в среде EPrints, поддерживают протоколы обмена метаданными OAI-PMH, SWORD, которые обеспечивают глобальные услуги доступа и поиска. [1]

**Greenstone** – свободно распространяемое ПО (выпускается под лицензией GNU General Public License) для создания и поддержания институциональных репозитиев открытого доступа (цифровых онлайн библиотек). Оно разработано в рамках Проекта новозеландской цифровой библиотеки при Университете Вайкато в сотрудничестве с ЮНЕСКО и Неправительственной организацией гуманитарной информации. Основная схема данных Dublin Core с квалификаторами, основные форматы документов HTML и MS Word. Для программного доступа к ресурсам имеется собственный API, отличный от стандартных протоколов доступа к цифровым репозиториям, таких как OAI-PMH или SRU / SRW (<http://www.loc.gov/standards/sru>) (Search/Retrieve via URL / Search/Retrieve Web service), однако есть поддержка протокола Z39.50 [11]. Для организации просмотра материала предусмотрено использование внутренних классификаторов. [1]

**CDS Invenio** (ЦЕРН / CERN (<http://cds.cern.ch>), Швейцария) – интегрированная электронная библиотечная система, которая представляет собой набор приложений для построения и управления автономным сервером ЭБ. ПО бесплатное, распространяемое под лицензией GNU General Public License. Технология, предлагаемая данным продуктом, покрывает все аспекты поддержки ЭБ, поддерживает протокол OAI-PMH, использует формат MARC21 как основной библиографический стандарт. CDS Invenio является комплексным решением управления репозиториями документов средних и больших объемов. Посредством CDS Invenio создан и поддерживается архив публикаций сервера документов CERN (CERN Document Server). В CERN CDS Invenio управляет более чем 500 коллекциями данных, состоящих из более чем 800 000 библиографических записей и 350 000 полнотекстовых документов, покрывая препринты, статьи, книги, журналы, фотографии, видеоматериалы и др. Помимо CERN, CDS Invenio в настоящее время инсталлирована и используется в 14 научных и образовательных учреждениях мира.[1]

Целью данной выпускной квалификационной работы является обеспечение возможности автоматизированного импорта метаданных научных статей из системы Open Journal Systems (OJS) в базу данных WEB-сайта



Пояснительная записка состоит из 5 глав, введения, заключения, списка литературы и приложения.

В первой главе представлено видение проекта. В данном разделе обсуждаются высокоуровневые требования (возможности, свойства) к программному продукту и наиболее существенные ограничения, т.е. производится выявление и анализ бизнес-требований.

Во второй главе спецификация требований к программному продукту. В данном разделе обсуждаются специфические требования, присущие данному программному продукту.

В третьей главе Проектирование и конструирование программного обеспечения. Данный раздел описывает процесс разработки ПО.

В четвертой главе выполнено тестирование программного обеспечения с применением модульного тестирования.

В пятой главе представлено руководство пользователя с описанием основных действий при работе с программой.

## **Глава 1**

### **Видение программного продукта**

#### **1.1 Введение**

В настоящее время в Кыргызской Республике существуют определенные трудности доступа к научным периодическим изданиям Кыргызской Республики, в частности отсутствует централизованное хранилище научных статей, которая бы обеспечивала надежное хранение и бесперебойный доступ к статьям. Несмотря на то, что активно функционирует Национальная Электронная Библиотека Российской Федерации, на наш взгляд, в связи сложившейся в настоящее время общественно-политической обстановкой, этот репозиторий не является надежным из-за сложности поддержания аппаратной базы в работоспособном состоянии в условиях ограничения импорта в РФ высокотехнологичной продукции.

##### **1.1.1. Цель**

Целью создания этой системы является создание централизованного хранилища научных статей для обеспечения долговременного и бесперебойного доступа к научным журналам КР, при этом полные тексты статей будут размещаться в открытом доступе для обеспечения свободного использования научной информации для поддержки научных исследований, развития науки и технологического прогресса в нашей стране при обязательном соблюдении прав автора. А также обеспечение возможности автоматизированного импорта метаданных научных статей из системы Open Journal Systems (OJS) в базу данных WEB-сайта

##### **1.1.2. Контекст**

Настоящий документ разрабатывается в рамках проекта создания централизованного надежного хранилища цифровых научных журналов КР.

##### **1.1.3. Определения, акронимы и сокращения**

Основные определения приведены в документе «Приложение 1. Глоссарий проекта».

##### **1.1.4. Краткое содержание**

Документ описывает требования высокого уровня для репозитория научных журналов. Указаны основные коммерческие преимущества решения, предусмотренного в Видении, сформулированы ключевые проблемы и методы их решения, характеристики пользователей

системы, возможности системы, ограничения, показатели качества и другие требования продукт указан.

## 1.2. Позиционирование

### 1.2.1. Деловые преимущества

В настоящее время имеющееся в Высшей Аттестационной Комиссии (ВАК) КР, репозиторий научных журналов не обеспечивает импорта метаданных статей из системы OJS. Новое решение позволит обеспечить более удобный режим доступа заинтересованных лиц к информации, повысить быстродействие, обеспечить надёжное хранение данных и более полный охват функций, подлежащих автоматизации.

Использование системы автоматизированной конвертации метаданных сократит время ответственного секретаря журналов на подготовку статьи, сведя его функции к проверке и корректировке вводимой информации, что позволит повысить производительность труда и достигнуть конкурентного преимущества на рынке научных публикаций и привлечь журнал наиболее ценных авторов.

### 1.2.2. Определение проблемы

Проблема описана в табл. 1.1.

Таблица 1.1. Определение проблемы

Проблема	Потеря доступа к научным статьям
затрагивает	Ученных в целом
Ее следствием является	Отставание отечественной науки от мирового уровня
Успешное решение	Разработка централизованного репозитория научных статей
Проблема	Отсутствие автоматизированного импорта метаданных статей из популярных систем цифровой поддержки научных журналов (в частности OJS)
затрагивает	Ответственных секретарей научных журналов

Ее следствием является	Повторная разметка метаданных статей для различных репозиторийев научных журналов
Успешное решение	Обеспечение автоматизированного импорта метаданных статей из системы OJS

### 1.2.3 Определение позиции изделия

Позиция изделия описана в табл. 1.2.

Таблица 1.2. Определение позиции изделия

Для	ВАК КР
которой	Требуется обеспечить централизованное хранение научных статей
(Название продукта)	Цифровой репозиторий научных журналов
который	Обеспечивает долговременное хранение данных
В отличие от	Существующего репозитория научных журналов КР
наш продукт	Обеспечивает автоматизированный импорт метаданных научных статей

## 1.3. Описания пользователей

### 1.3.1. Сведения о пользователях

У системы существуют три основных пользователя: читатель, ответственный секретарь журнала, администратор.

Читатель – может зайти на WEB-сайт, ознакомиться с опубликованными журналами, посмотреть их подробную информацию

Ответственный секретарь журнала – занимается формированием метаданных для выпуска статьи на портал, а также редактирование метаданных статей в журнале.

Администратор – занимается добавлением и редактированием данных новых редакторов журнала, добавление новых категорий и разделов

### 1.3.2. Пользовательская среда

Пользователи сайта это в основном люди в возрасте 18-70 лет. Возможно, пользователи плохо владеют компьютером, для этого необходимо разработать интуитивно простой и удобный интерфейс сайта. Для удобного поиска журналов необходимо реализовать поиск по сайту.

Администратор сайта должен уметь на базовом уровне пользоваться компьютером. Панель администратора тоже должна быть интуитивно понятной и удобной в управлении, иметь возможность управлять сайтом без участия программиста

### 1.3.3. Профили пользователей

Профиль пользователя Читатель описан в следующей табл. 1.3.

Таблица 1.3. Профиль пользователя. Читатель

Типичный представитель	Читатель
Описание	Пользователь системы, наделенный правами на просмотр и чтение научных публикаций
Тип	Пользователь
Ответственности	Нет ответственности
Критерий успеха	Возможность получения необходимой информации.

Профиль пользователя Ответственный секретарь журнала описан в следующей табл. 1.4.

Таблица 1.4. Профиль пользователя. Ответственный секретарь журнала

<b>Типичный представитель</b>	<b>Ответственный секретарь журнала</b>
<b>Описание</b>	Пользователь системы, наделенный правами на редактирование статей и журналов
<b>Тип</b>	Пользователь
<b>Ответственности</b>	Загрузка метаданных в систему
<b>Критерий успеха</b>	Умение работать с системой.

Профиль пользователя Администратор описан в следующей табл. 1.5.

Таблица 1.5. Профиль пользователя. Администратор

<b>Типичный представитель</b>	<b>Администратор</b>
<b>Описание</b>	Администратор – ответственное лицо, отвечающее за корректную работу сайта. Он может добавлять, редактировать и удалять публикации. Регистрировать новых научных редакторов, загружать метаданные из файла с расширением .xml
<b>Тип</b>	Пользователь
<b>Ответственности</b>	Добавляет на сайт актуальные данные, редактирует, удаляет устаревшие. Отслеживает корректную работу сайта, исправляет имеющиеся ошибки.
<b>Критерий успеха</b>	Возможность грамотно отобразить информацию, тем самым дать пользователю полное представление проблемы. Возможность осведомлять пользователей об актуальных новостях.

### 1.3.4. Ключевые потребности пользователей

Ответственный секретарь журнала затрачивает большое количество времени на обрабатывание метаданных статей.

## 1.4. Краткий обзор изделия

### 1.4.1. Контекст использования системы

Система является законченной независимой разработкой. В перспективе возможно развитие функционала. Весь контент на Web-портале может редактировать заказчик как администратор сайта, без участия программиста.

### 1.4.2. Сводка возможностей

Возможности программы описаны в табл. 1.6.

Таблица 1.6. Сводка возможностей продукта

Выгоды заказчика	Поддерживающие возможности
Упрощение редактирования метаданных	Автоматическая конвертация метаданных.
Ускорение обращения информации в процессе издания журнала	Система позволит автоматически брать метаданные из системы OJS предоставляющая метаданные в виде .xml файлов
Формирование единой базы публикаций	Все заинтересованные пользователи со своих рабочих мест имеют доступ ко всем опубликованным на портале научным статьям и журналам

### 1.4.3. Предположения и зависимости

Для того чтобы пользоваться Web-платформой у пользователей должен быть, как минимум, смартфон, компьютер или другое устройство, подключенное к интернету.

## **1.5. Возможности продукта**

### **1.5.1. Для типа пользователя – Читатель**

- Возможность посмотреть информацию на портале, просмотреть подробную информацию журналов, выпусков и статей.
- Возможность найти журналы при помощи поисковой системы

### **1.5.2. Для типа пользователя – Ответственный секретарь журнала**

- Возможность редактирования метаданных статей
- Возможность получить XML файл из внешних источников
- Возможность редактировать размещенную информацию на портале
- Возможность загрузить файл с метаданными на портал

### **1.5.3. Для типа пользователя – Администратор**

- Возможность зарегистрировать ответственного секретаря журнала
- Возможность редактирования категории или раздела

## **1.6. Ограничения**

Внедрение системы не должно занимать более 3 месяцев. В ядре системы должна быть представлена промышленная СУБД реляционного доступа.

## **1.7. Показатели качества**

### **1.7.1. Применимость**

- Интуитивно понятный интерфейс
- Удобный поиск по сайту
- Время необходимое для обучения обычных пользователей – два рабочих дня (16 часов),  
для обучения продвинутых пользователей один рабочий день (8 часов)



### **1.7.2. Надежность**

- Доступ к панели администратора предоставляется только после ввода логина и пароля.
- Пароль администратора хранится в зашифрованном виде, если даже кто-то скачает базу данных, реального пароля там не будет.
- Интерфейс программного продукта должен быть интуитивно понятным для пользователя, и требовать минимальное количество времени на обучение.
- Время отклика системы – не более 5 секунд.

## **1.8. Другие требования к Web-платформе**

### **1.8.1. Применяемые стандарты**

Система должна соответствовать всем стандартам интерфейса пользователя Microsoft Windows.

### **1.8.2. Системные требования**

Минимальные системные требования:

- 1 GB памяти
- 5 GB свободного дискового пространства
- процессор с тактовой частотой 1 GHz
- Операционная система Windows 7, 8, 8.1, 10.

### **1.8.3. Эксплуатационные требования**

- Хорошее подключение к интернет сети
- Максимально простой и понятный интерфейс
- Поиск по категориям и разделам

## **1.9. Требования к документации**

### **1.9.1. Руководство пользователя**

В документации должны быть представлены Руководства пользователей (по типам пользователей). Они должны содержать расшифровку всех используемых терминов, описания

основных вариантов использования, включая альтернативные сценарии, а также подробный обзор интерфейса Web-платформы.

### **1.9.2. Интерактивная справка**

Интерактивная справка нужна для разрешения появившийся во время работы вопросов. В справке должна быть реализована возможность поиска информации, по ключевым словам, а также вариант представления информации по отдельным позициям меню программы. Справка должна содержать максимально полную и подробную информацию по работе Web-платформы.

## **2. Спецификация требований к программному продукту**

### **2.1. Введение**

#### **2.1.1. Назначение**

Назначение документа “Спецификация требований к Web-платформе «Разработка WEB-сайта национальной электронной библиотеки периодических изданий ВАК КР» – определение требований к системе. Документ включает общее видение продукта. Данный документ является достаточным для разработки программного продукта.

#### **2.1.2. Цели создания системы, решаемые задачи и область применения**

##### **2.1.2.1. Краткое описание деятельности заказчика и существующей технологии**

Данный документ предназначен для определения, спецификации и согласования с заказчиком требований к системе **Open Journal Systems**. Документ включает общее видение продукта, формализованные требования, описание функционала и объектов системы. Данный документ является достаточным для разработки программного продукта.

##### **2.1.2.2. Цели создания системы. Эффекты от ее внедрения**

Цель создания этого раздела состоит в том, чтобы собрать, проанализировать и определить высокоуровневые потребности и возможности системы конвертации метаданных на основе платформы Open Journal Systems. Документ акцентирует внимание на возможностях, необходимых совладельцам и целевым пользователям, и на том, почему эти потребности существуют. Подробности того, как система Open Journal Systems выполняет эти потребности, будут детализированы в прецедентах и дополнительных спецификациях.

### **2.1.2.3. Назначение системы и область применения**

Web-платформа «Разработка WEB-сайта национальной электронной библиотеки периодических изданий ВАК КР» является инструментом для загрузки статей на портал

### **2.1.3. Определения, акронимы и сокращения**

Основные определения приведены в документе «Приложение 1. Глоссарий проекта».

### **2.1.4. Ссылки**

Требования к Web-платформе «Разработка WEB-сайта национальной электронной библиотеки периодических изданий ВАК КР»

### **2.1.5. Краткое содержание**

Документ описывает высокоуровневые требования к системе «Разработка WEB-сайта национальной электронной библиотеки периодических изданий ВАК КР». Указаны основные деловые преимущества рассматриваемого в Видении решения, сформулированы ключевые проблемы и способы их решения, приведены характеристики пользователей системы, возможности системы, ограничения, показатели качества и другие требования к продукту.

## **2.2. Общее описание**

### **2.2.1. Позиционирование программного продукта**

Данный программный продукт является расширением для системы Open Journal Systems и наличия в нем журналов и статей относящихся к работам КР. Также данная система зависит от наличия интернет-соединения и веб-браузера

### **2.2.2 Функциональность продукта**

Функционал Web-платформы «Разработка WEB-сайта национальной электронной библиотеки периодических изданий ВАК КР» представлен в виде диаграммы вариантов использования, которая показана на рис. 2.1.



В табл. 2.2. предоставлена краткая формулировка платформы акторов

Таблица 2.2. Выявление вариантов использования

Основной актёр	Наименование	Формулировка
Администратор	Редактирование категорий и разделов на сайте	Администратор через админ панель будет добавлять, изменять категории и разделы на сайте
Администратор	Регистрация новых ответственных секретарей журнала	Администратор через админ панель будет добавлять ответственных секретарей журналов
Ответственный секретарь журнала	Авторизация	Ответственный секретарь журнала может авторизоваться на сайте для дальнейших доступных действий
Ответственный секретарь журнала	Редактирование метаданных статей	Ответственный секретарь журнала может добавлять метаданные на сайт в виде xml файла и также их редактировать
Ответственный секретарь журнала	Редактировать размещенную информацию на портале	Ответственный секретарь журнала может изменять данные на портале
Ответственный секретарь журнала	Получить XML файл	Ответственный секретарь журнала получает XML файл из внешних источников
Ответственный секретарь журнала	Загрузить файл с метаданными на портал	Ответственный секретарь журнала может загрузить метаданные в журнал
Читатель	Просматривать информацию на портале	Читатель может просматривать информацию на портале а также воспользоваться функцией поиска необходимых журналов по критериям

Читатель	Получить информацию о журналах и статьях	Читатель может просматривать доступную информацию о всех журналах и научных статьях на сайте
----------	--	--

В таблице 2.3 предоставлен реестр вариантов использования Web-платформы.

Таблица 2.3. Реестр вариантов использования

Код	Основной актер	Наименование	Формулировка
<b>A1</b>	Администратор	Редактирование разделов и категорий на сайте	Администратор через админ панель будет редактировать разделы и категории на сайте
<b>A2</b>	Администратор	Добавление ответственных секретарей журналов	Администратор через админ панель будет добавлять ответственных секретарей журналов
<b>B1</b>	Ответственный секретарь журнала	Авторизация	Ответственный секретарь журнала может авторизоваться на сайте для дальнейших доступных действий
<b>B2</b>	Ответственный секретарь журнала	Редактирование метаданных статей	Ответственный секретарь журнала может добавлять метаданные на сайт в виде xml файла и также их редактировать
<b>B3</b>	Ответственный секретарь журнала	Редактировать размещенную информацию на портале	Ответственный секретарь журнала может изменять данные на портале
<b>B4</b>	Ответственный секретарь журнала	Получить XML файл	Ответственный секретарь журнала получает XML файл из внешних источников
<b>B5</b>	Ответственный секретарь журнала	Загрузить файл с метаданными на портал	Ответственный секретарь журнала может загрузить метаданные в журнал

<b>C1</b>	Читатель	Просмотр информации на портале	Читатель может просматривать информацию на портале
<b>C2</b>	Читатель	Получить информацию о журналах и статьях	Читатель может просматривать доступную информацию о всех журналах и научных статьях на сайте

### 2.2.3. Характеристики пользователей

Читатель – это в основном люди в возрасте 20-70 лет. Возможно, пользователи плохо владеют компьютером и телефоном.

Администратор – Продвинутый пользователь компьютера

Ответственный секретарь журнала - человек умеющий работать с компьютером на базовом уровне

Таблица 2.4. Характеристики пользователей

#### Основное действующее лицо: Администратор

A1	Администратор	Редактирование разделов и категорий на сайте	Администратор через админ панель будет редактировать разделы и категории на сайте
----	---------------	--	---

**Основное действующее лицо:** Администратор

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:** включает варианты использования:

"A3. Изменить категорию / раздел"

"A4 Добавить категорию / раздел"

**Краткое описание.** Администратор через админ панель будет добавлять и менять категории и разделы на сайте

**Основное действующее лицо:** Администратор

A2	Администратор	Добавление ответственных секретарей журналов	Администратор через админ панель будет добавлять ответственных секретарей журналов
----	---------------	--	--

**Основное действующее лицо:** Администратор

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:**

**Краткое описание.** Администратор через админ панель будет добавлять новых ответственных секретарей журнала

**Основное действующее лицо:** Ответственный секретарь журнала

B1	Ответственный секретарь журнала	Авторизация	Ответственный секретарь журнала может авторизоваться на сайте для дальнейших доступных действий
----	---------------------------------	-------------	---

**Основное действующее лицо:** Ответственный секретарь журнала

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:** включает варианты использования:

"B6. Выход из системы"

**Краткое описание.** Ответственный секретарь журнала имеет возможность авторизоваться на сайте для дальнейших доступных действий с последующей возможностью выхода из системы

**Основное действующее лицо:** Ответственный секретарь журнала

B2	Ответственный секретарь журнала	Редактирование метаданных статей	Ответственный секретарь журнала может добавлять метаданные на сайт в виде xml файла и также их редактировать
----	---------------------------------	----------------------------------	--

**Основное действующее лицо:** Ответственный секретарь журнала

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:** включает варианты использования:

"B4. Получить XML файл"

"B5. Загрузить файл с метаданными на портал"



**Краткое описание.** Ответственный секретарь журнала может редактировать метаданные на сайте после их получения из внешних источников

**Основное действующее лицо: Ответственный секретарь журнала**

B3	Ответственный секретарь журнала	Редактировать размещенную информация на портале	Ответственный секретарь журнала может изменять данные на портале
----	---------------------------------	---	--

**Основное действующее лицо:** Ответственный секретарь журнала

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:** включает варианты использования:

"B7. Изменить информацию на портале"

"B8. Редактирование журналов и статей"

**Краткое описание.** Ответственный секретарь журнала может редактировать размещенную информацию на портале

**Основное действующее лицо: Ответственный секретарь журнала**

B4	Ответственный секретарь журнала	Получить XML файл	Ответственный секретарь журнала получает XML файл из внешних источников
----	---------------------------------	-------------------	---

**Основное действующее лицо:** Ответственный секретарь журнала

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:**

**Краткое описание.** Ответственный секретарь журнала может получить XML файл с данными из внешних источников

**Основное действующее лицо: Ответственный секретарь журнала**

B5	Ответственный секретарь журнала	Загрузить файл с метаданными на портал	Ответственный секретарь журнала может загрузить метаданные в журнал
----	---------------------------------	--	---

**Основное действующее лицо:** Ответственный секретарь журнала

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:** включает варианты использования:

"B4. Получить XML файл"

"B2. Редактирование метаданных статей"

**Краткое описание.** Ответственный секретарь журнала может загрузить метаданные на портал

**Основное действующее лицо:** Читатель

C1	Читатель	Просмотр информации на портале	Читатель может просматривать информацию на портале
----	----------	--------------------------------	--

**Основное действующее лицо:** Ответственный секретарь журнала

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:** включает варианты использования:

"C3. Поиск по категориям"

**Краткое описание.** Читатель может просматривать информацию на портале, а также воспользоваться системой поиска

**Основное действующее лицо:** Читатель

C2	Читатель	Получить информацию о журналах и статьях	Читатель может просматривать доступную информацию о всех журналах и научных статьях на сайте
----	----------	--	--

**Основное действующее лицо:** Ответственный секретарь журнала

**Другие участники прецедента:** отсутствуют

**Связи с другими вариантами использования:** включает варианты использования:

"C1. Просмотр информации на портале"

**Краткое описание.** Читатель может просматривать подробную информацию о всех журналах, выпусках и научных статьях на сайте

#### 2.2.4. Ограничения

Метаданные, которые подготавливаются, должны быть тщательно подобраны исходя из требований программы для получения корректных результатов.

### **2.2.5. Предположения и зависимости**

Чтобы пользоваться Web-платформой у пользователей должен быть, как минимум, смартфон или компьютер и подключение к интернету, также система будет функционировать правильно в том случае, когда все необходимое для нее оборудование и программное обеспечение доступно и находится в рабочем состоянии.

### **2.2.6. Распределение требований**

Эксплуатационные требования:

- Подключение к интернет сети
- Максимально простой интерфейс
- Поиск по сайту

### **2.3. Специфические требования**

#### **2.3.1. Внешние требования к интерфейсам**

Сайт должен:

- Должен правильно работать во всех в разных браузерах.
- Работать под разное разрешение монитора и диагональ.
- Должен включать в себя только веб безопасные шрифты.
- Должен быть продуман с точки зрения возможности последующей программной реализации задуманного.
- Текстовые материалы должны быть читабельными
- Используемые графические элементы должны быть легальными, понятными и однозначными

#### **2.3.2. Функциональные требования**

#### **2.3.3. Требования к производительности**

- Скорость загрузки сайта должна быть как можно меньше и не нагружать компьютер пользователя
- Сайт должен одинаково отображаться во всех современных браузерах. В старых версиях браузеров вероятны незначительные различия в оформлении блоков, без потери их структуры и функциональности.

- Все несуществующие или неработающие ссылки должны отдавать код ошибки 404.
- Сайт и все его информационные материалы должны быть доступны всем пользователям без прохождения операции аутентификации.
- Административный интерфейс должен быть доступен только определенной группе пользователей после прохождения процедуры авторизации.

#### **2.3.4. Логические требования к базе данных**

Для разработки базы данных используется СУБД SQLite

#### **2.3.5. Ограничения дизайна**

- Должен использоваться простой интуитивно понятный интерфейс
- Сайт должен работать на устройствах с разными размерами экранов(адаптивность)
- Сайт должен правильно открываться в разных браузерах(кроссбраузерность)

#### **2.3.6. Ограничения проектирования**

#### **2.3.7. Атрибуты программного продукта**

**Программный продукт обладает следующими атрибутами, которые определяет соответствующие требования:**

##### **Надежность:**

- Доступ к панели администратора предоставляется только после ввода логина и пароля.
- Пароль администратора хранится в зашифрованном виде, если даже кто-то скачает базу данных реального пароля там не будет.

##### **Применимость:**

- Интерфейс пользователя и администратора должен быть интуитивно понятным.

### 3. Проектирование и конструирование программного обеспечения

#### 3.1 Разработка диаграммы классов

Диаграмма классов (class diagram) используется для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, различные отношения между отдельными объектами тематической области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. Эта диаграмма не показывает информацию о временных аспектах работы системы (рис. 3.1). С этой точки зрения диаграмма классов является дальнейшим развитием концептуальной модели проектируемой системы.

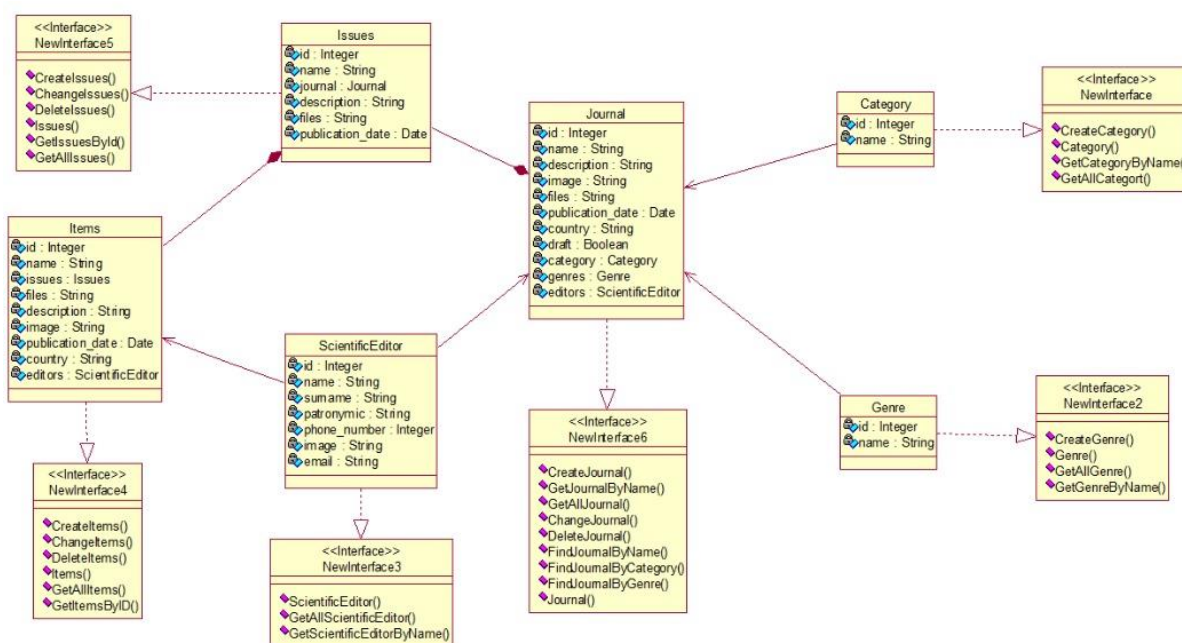


Рисунок 3.1. Диаграмма классов

#### 3.2 Определение классов-сущностей

Класс - сущность содержит информацию, которая должна храниться постоянно и не уничтожается с уничтожением объектов данного класса или после прекращения работы моделируемой системы. Этот класс соответствует отдельной таблице базы данных. В этом случае его атрибуты являются полями таблицы, а операции – присоединенными или хранимыми процедурами. Этот класс только принимает сообщения от других классов модели.

[2]

В данной системе в виде классов-сущностей представлены классы, которые отвечают за хранение данных о журнале, авторах и пользователях.

В табл. 3.1 перечислены основные классы сущности и их краткие описания.

Таблица. 3.1 Основные классы сущности и их краткие описания

Название класса	Краткое описание
library_journal	Класс, содержащий все данные журнала.
library_Issues	Класс, содержащий все данные выпуска
library_Items	Класс, содержащий все данные статей
library_genre	Класс, содержащий данные разделов
library_journal_genres	Класс, содержащий данные, связывающие раздел и журнал.
library_category	Класс, содержащий данные категорий, связывающийся с журналом.
library_journal_category	Класс, содержащий данные, связывающие категорию и журнал.
library_scientificeditions	Класс, содержащий информацию редакционной коллегии
library_items_editors	Класс, содержащий данные, связывающие редакционную коллегию и статью.
library_journal_editors	Класс, содержащий данные, связывающие редакционную коллегию и журнал.

На рис. 3.2 представлена диаграмма классов-сущностей базы данных и связи и между ними.

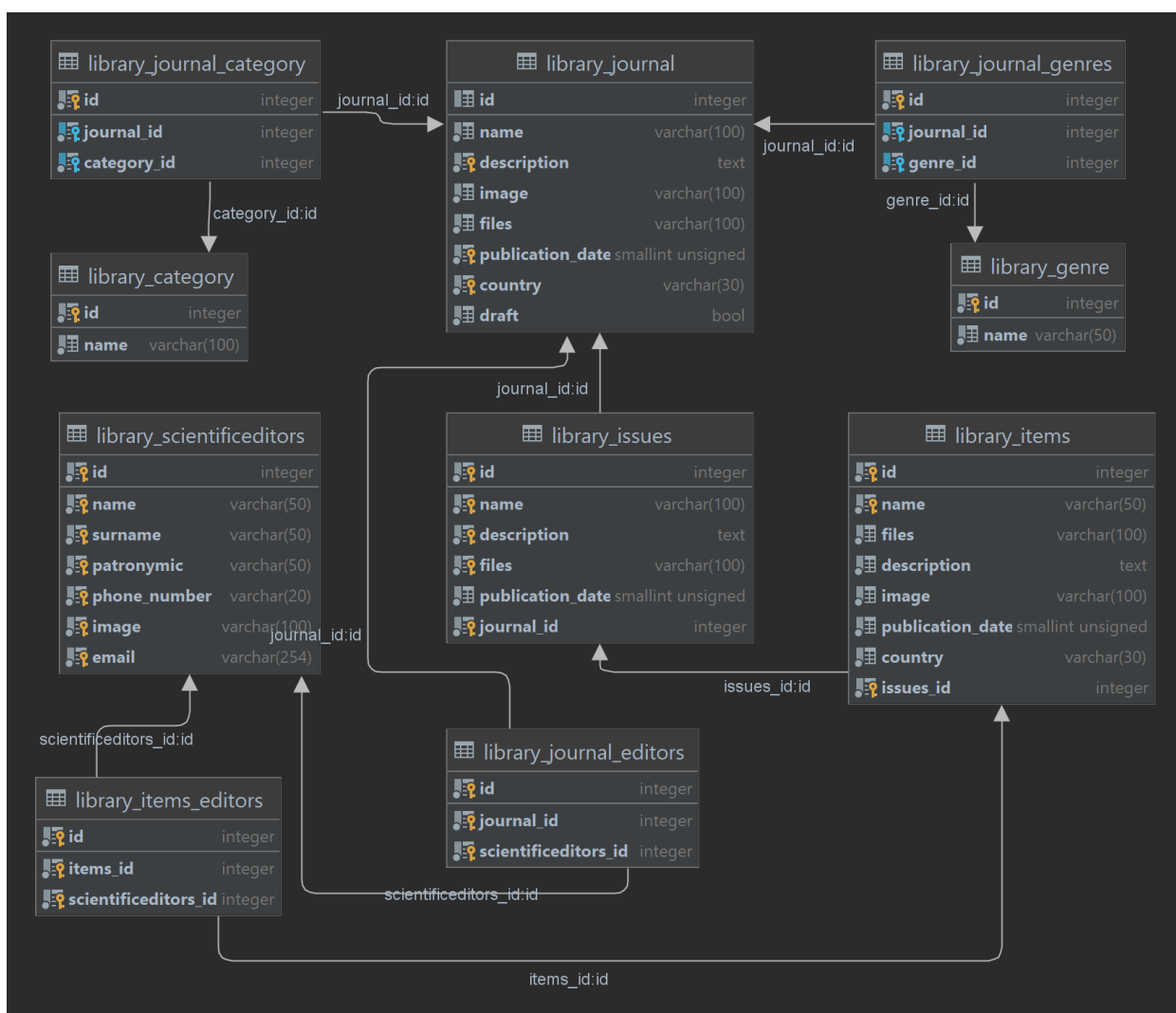


Рисунок 3.2. Классы-сущности и связи между ними

### 3.3 Разработка диаграмм компонентов

Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. Пунктирные стрелки, соединяющие модули, показывают отношения взаимозависимости, аналогичные тем, которые имеют место при компиляции исходных текстов программ. Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости между ними [3].

Диаграмма компонентов разрабатывается для следующих целей:

- Визуализации общей структуры исходного кода программной системы.
- Спецификации исполнимого варианта программной системы.

- Обеспечения многократного использования отдельных фрагментов программного кода.
- Представления концептуальной и физической схем баз данных.

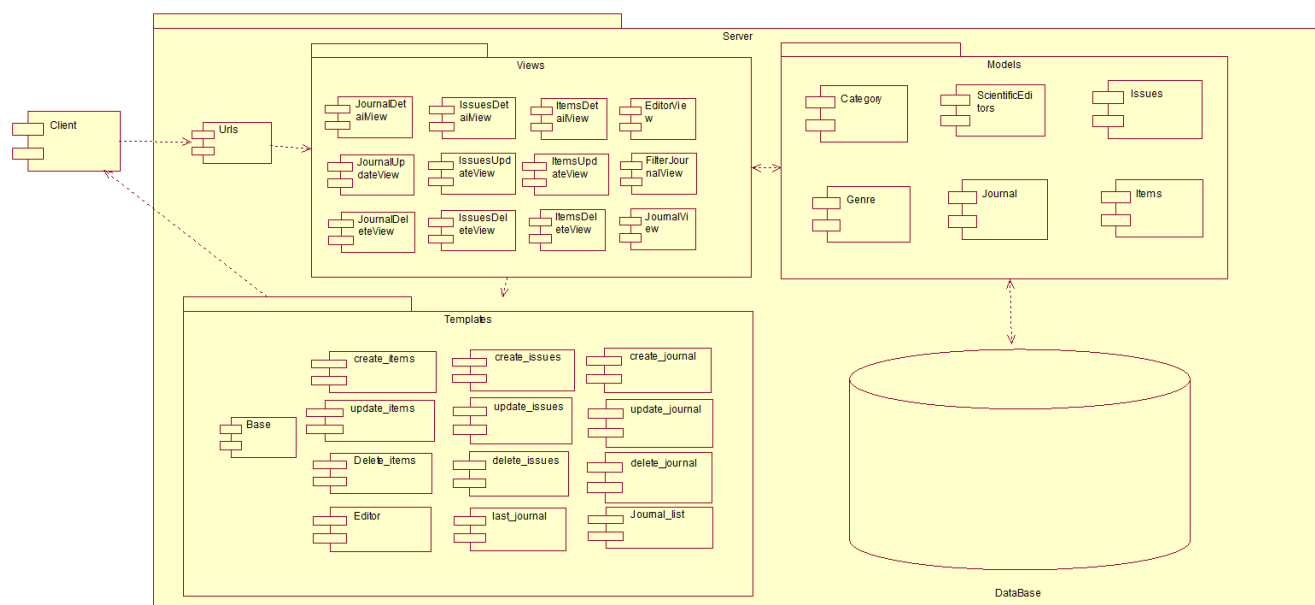


Рисунок 3.3. Диаграмма компонентов системы

### 3.4. Разработка диаграмм последовательностей

Диаграмма последовательности отражает сценарий поведения в системе, показывает взаимодействие между объектами и их отношения, а также обеспечивает наглядное представление порядка передачи сообщений, то есть выделяет упорядочение сообщений по времени.

На рисунках 3.4, 3.5 представлены диаграммы последовательностей процессов авторизации, поиска статьи по данным.

#### Описание последовательности:

1. Пользователь вводит данные для поиска.
2. Идет запрос данных в базе данных.
3. Происходит поиск данных по журналу.
4. Данные передаются пользователю



## Поиск журнала по данным

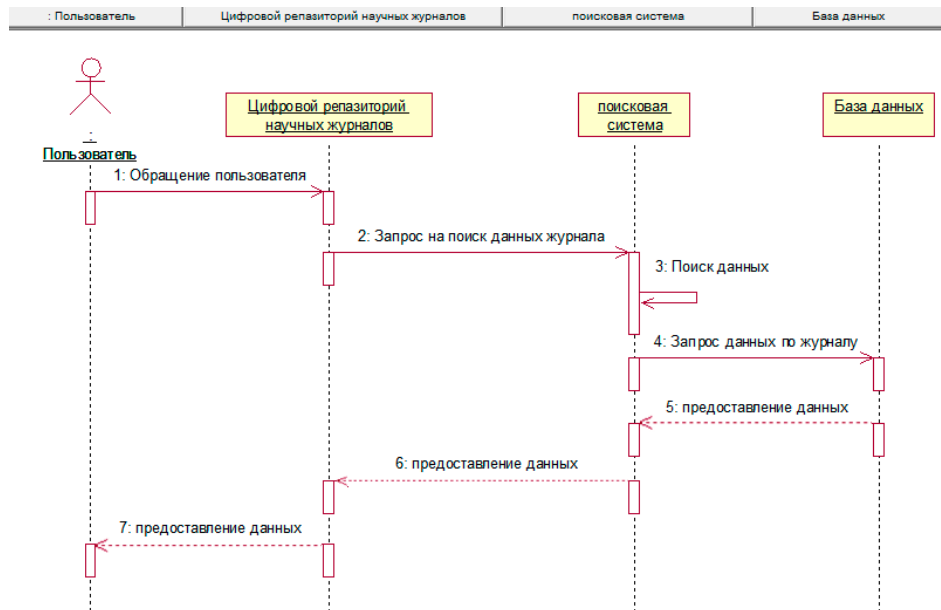


Рисунок 3.4. Диаграмма последовательности поиска статьи

### Описание последовательности:

1. Отображение формы для входа.
2. Ввод логина и пароля в форму.
3. Проверить корректность данных.
  - 3.1. Вернуть сообщение об ошибке.
4. Переход на главную страницу.

## Авторизация

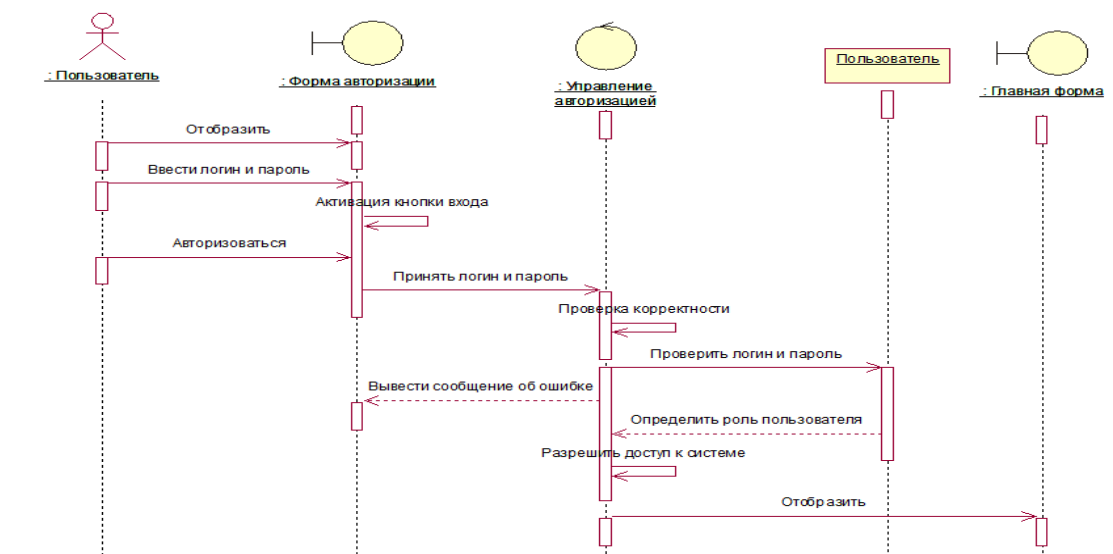


Рисунок 3.5. Диаграмма последовательности авторизации

## **4. Разработка тестов и тестирование программного продукта**

### **4.1. Разработка плана тестирования**

#### **4.1.1. Введение**

Данный документ описывает план тестирования для системы «Разработка WEB-сайта национальной электронной библиотеки периодических изданий ВАК КР». согласно спецификации. Полная стратегия тестирования программного обеспечения состоит из следующих типов испытаний и выполняется в следующем порядке:

1. Тестирование компонентов (модульное тестирование). Тестируются все программные компоненты (при этом проверяется покрытие кода тестами). Анализ кода.
2. Тестирование интеграции. Тестируется программное обеспечение, чтобы гарантировать, что компоненты взаимодействуют правильно.

#### **4.1.2. Область тестирования**

Целью этого тестирования является определить, соответствует ли разработанное программное обеспечение заявленным функциональным требованиям, а также выявить ошибки и представить их исправления, улучшив тем самым качество разработанного программного обеспечения.

#### **4.1.3. Начальные условия**

Задачи, которые должны быть решены перед началом тестирующей деятельности:

1. Имеется законченная программная спецификация.
2. Работающее программное обеспечение.
3. Все необходимые компоненты находятся в рабочем состоянии.

Составлен следующий набор тестов (табл. 4.1):

Таблица 4.1. Набор тестов для тестирования системы

№	Тестовый вариант	Ожидаемый результат
1	Добавление ответственного секретаря журналов	Пользователь успешно добавлен в базу данных
2	Авторизация пользователя	Пользователь успешно вошел в систему

3	Сохранение данных журнала после редактирования пользователем	Данные корректно сохранены в базу данных
5	Добавление журнала	Журнал успешно добавлен
6	Добавление выпуска	Выпуск успешно добавлен
7	Добавление статьи	Статья успешно добавлена

#### 4.1.4. Приоритеты тестирования

Проверки перечислены в порядке уменьшения приоритетного уровня:

1. Функционал - все ли заданные функции Web-платформы выполняются, как ожидалось?
2. Удобство и простота - реально ли Web-платформа удобна и легко понимаема для пользователя?
3. Защита – тщательно ли обеспечивается защита данных?
4. Выполнение – Web-платформа соответствует всем критериям выполнения?

#### 4.1.5. Методы тестирования

Будут использоваться **тестовые сценарии** – сценарии вариантов использования (с предопределенным вводом и ожидаемыми выходными данными), варианты использования взяты из диаграммы вариантов использования.

#### 4.1.6. Среда тестирования

Для тестирования данного программного продукта необходимо следующее оборудование:

Персональный компьютер со следующей конфигурацией:

- Ryzen 7 2700, 16Gb
- Операционная система Microsoft Windows 10
- Установленные программы JetBrains PyCharm, Google Chrome

#### 4.2. Модульное тестирование

**Модульное тестирование** — это процесс программирования, который позволяет проверять правильность блоков исходного кода, правильную работу программных модулей, а также наборов одного или нескольких программных модулей вместе с соответствующими управляющими данными. [4]

Суть метода заключается в создании методов тестирования для тестирования отдельного метода в программе. Таким образом, можно проверить правильность кода при его изменении и добавлении функциональности в программу. Кроме того, модульные тесты позволяют обнаруживать ошибки в работе программных алгоритмов, что, в свою очередь, повышает качество и надежность программного обеспечения.

Цель модульного тестирования — выделить отдельные модули программы и проверить работу этих модулей в соответствии с определенными входными данными.

#### 4.2.1. Тестовые случаи

Шаги тестирования:

1. Администратор авторизовывается на сайте и нажимает кнопку добавить нового редактора
2. Вводит данные Логин и пароль

##### Описание тестового случая:

Тестирование метода register() – проверка создания нового пользователя в базе данных (табл. 4.2).

Таблица 4.2. Тестовые варианты метода register()

№	Входные данные	Что проверяется	Ожидаемый результат
1	Логин: user	Наличие в базе данных пользователя с таким же логином <ul style="list-style-type: none"> <li>Пользователь с таким логином не найден</li> </ul>	Сообщение об успешной регистрации.
2	Логин: user	Наличие в базе данных пользователя с таким же логином <ul style="list-style-type: none"> <li>Пользователь с таким логином найден</li> </ul>	Сообщение, что пользователь с таким логином уже существует.

##### Описание тестового случая:

Шаги тестирования:

1. Администратор заходит на сайт и нажимает кнопку авторизоваться
2. Вводит данные Логин и пароль

Тестирование метода login() – проверка создания нового пользователя в базе данных (табл. 4.3).

Таблица 4.3. Тестовые варианты метода login()

№	Входные данные	Что проверяется	Ожидаемый результат
1	Логин: user	Корректность пароля <ul style="list-style-type: none"> <li>Пароль соответствует</li> </ul>	Переход на главную страницу.
2	Логин: user	Корректность пароля <ul style="list-style-type: none"> <li>Пароль не соответствует</li> </ul>	Сообщение, что пароль не верный.

#### Описание тестового случая:

Тестирование метода create\_journal () – проверка добавления журнала в базу данных (табл. 4.4).

Таблица 4.4 Тестовые варианты create\_journal ()

№	Входные данные	Что проверяется	Ожидаемый результат
1	name	Есть ли в базе данных уже журнал с таким же именем <ul style="list-style-type: none"> <li>Такого журнала нет</li> </ul>	Добавление содержимого в бд.
2	name	Есть ли в базе данных уже журнал с таким же именем <ul style="list-style-type: none"> <li>Такой журнал уже есть</li> </ul>	Сообщение, что такие данные уже есть

### 4.3. Системное тестирование


Системное тестирование — это тестирование программного обеспечения, выполняемое на полной, интегрированной системе, с целью проверки соответствия системы исходным требованиям.

Системное тестирование выполняется методом “черного ящика”.

Было проведено тестирование соответствия системы функциональным требованиям, изложенным в спецификации вариантов использования.

Тестирование системы показало ее соответствие функциональным требованиям.

Система правильно реагирует на неверный ввод данных пользователем, выдавая ему подсказки. На рис. 4.1 показан пример неверного ввода с последующим выводом сообщений об ошибках.



- Имя пользователя и/или пароль не верны.

Имя пользователя:

Имя пользователя

Пароль:

Пароль

Запомнить меня:

☐

[Забыли пароль?](#)

Войти

Рисунок 4.1. Попытка входа с неверными данными

## 5. Руководство пользователя

### 5.1. Назначение системы

Автоматизированная система конвертации метаданных предназначена для подготовки публикации метаданных из системы Open Journal Systems в базу данных WEB-сайта. Web-платформа обладает простым, удобным и интуитивно понятным интерфейсом.

### 5.2. Условия применения системы

Пользователям системы необходимо иметь персональный компьютер со следующими минимальными системными требованиями:

- 1 GB оперативной памяти
- 10 GB свободного дискового пространства
- процессор с тактовой частотой 1 GHz
- Операционная система Windows 7, 8, 8.1, 10.
- Веб-браузер Opera, Mozilla Firefox, Google Chrome. Версии должны быть не раньше 2019 года.

Также пользователи должны обладать базовыми навыками работы с персональным компьютером

### 5.3. Подготовка системы к работе

#### Запуск системы

Перед началом работы, пользователю потребуется открыть браузер и в окне для ссылки ввести следующие данные «http://127.0.0.1:8000» после этого откроется главная страница веб-приложения (рис. 5.1).

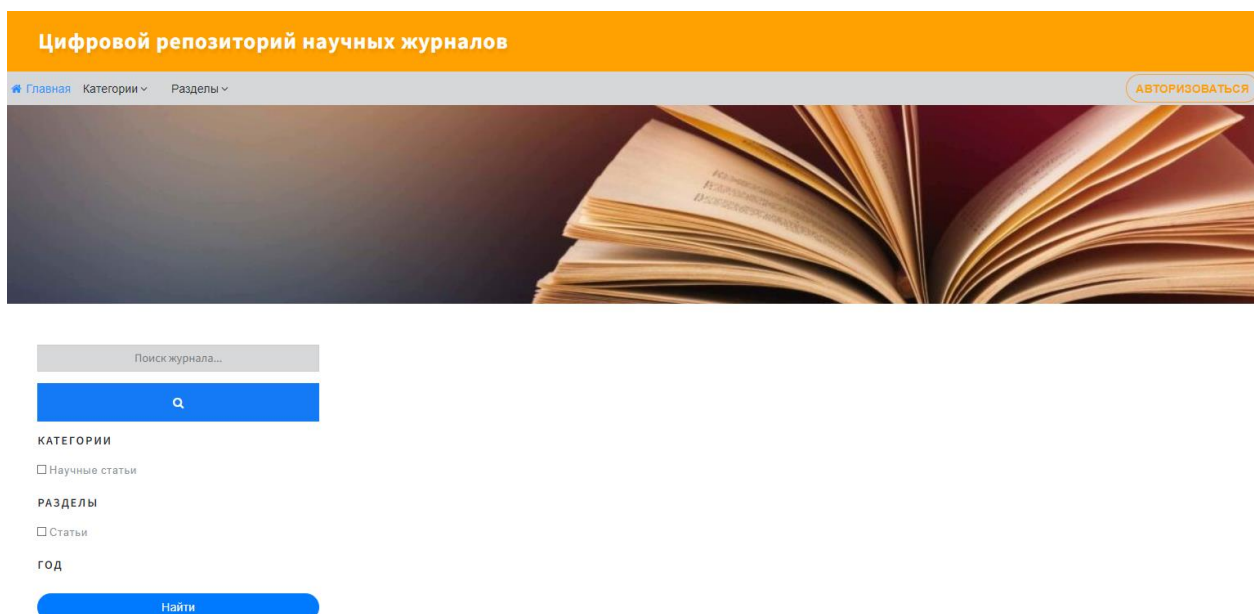



Рисунок 5.1. Главная страница

### Вход

Пользователь, будучи администратором или ответственным секретарем, может пройти к окну авторизации и ввести свои данные «логин пароль» (рис. 5.2) для дальнейшего входа в систему и получения доступного им функционала

## Авторизация



Имя пользователя:

Пароль:

Запомнить меня:

☐

[Забыли пароль?](#)

Рисунок 5.2. Вход в систему

### Главная страница

После успешного входа в систему, открывается главная страница сайта, но уже с возможностью взаимодействия панелью администратора (рис. 5.3) или ответственного секретаря журнала (рис. 5.4).



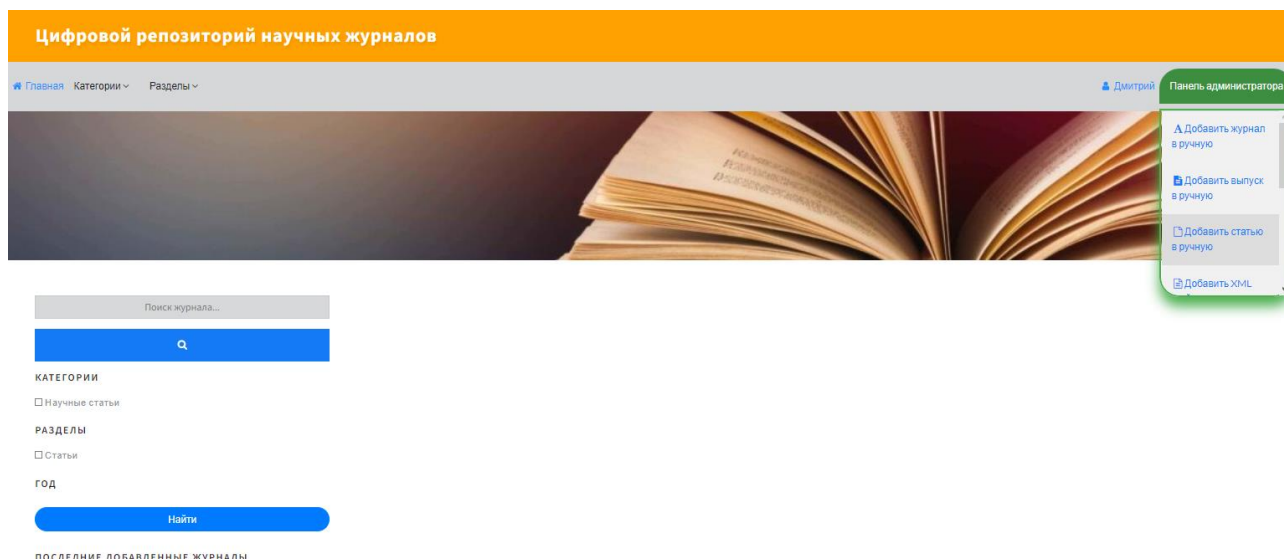


Рисунок 5.3. Главная страница от лица администратора

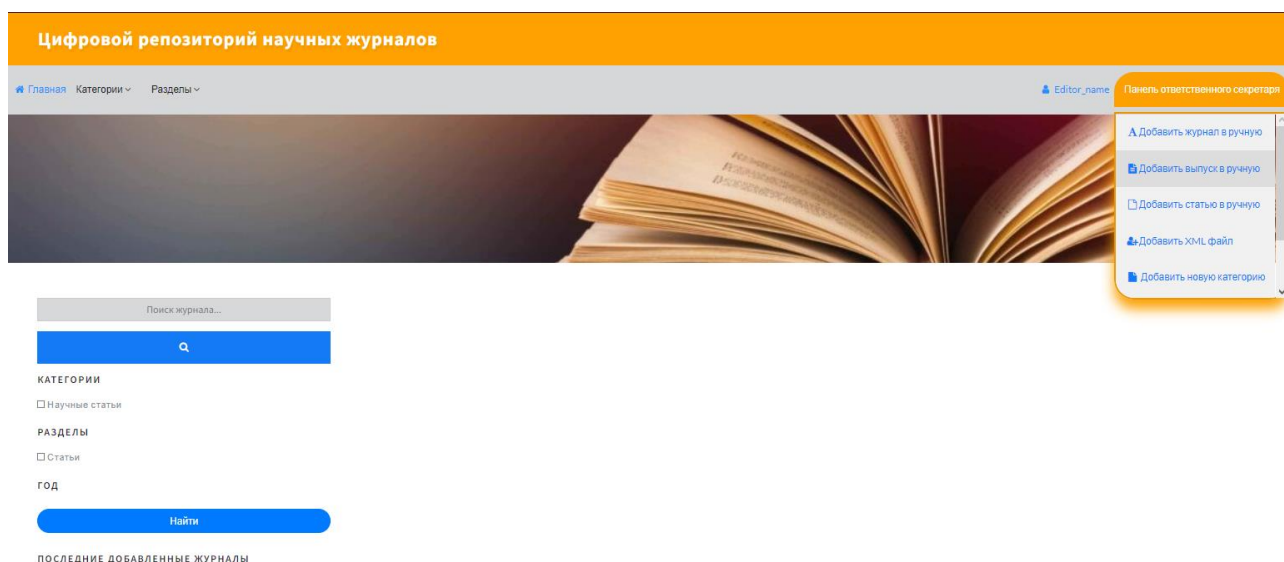


Рисунок 5.4. Главная страница от лица редактора

## Добавление нового редактора

После авторизации, у администратора появляется возможность через административную панель создать нового редактора. Для этого потребуется заполнить форму, где нужно ввести логин, пароль и почту редактора (рис. 5.5).

The screenshot shows a web form titled "Создание нового ответственного секретаря" (Creating a new responsible secretary). Below the title is a subtitle: "Введите все данные для регистрации нового пользователя:" (Enter all data for registering a new user:). The form contains several input fields and a submit button:

- Имя пользователя:** (Username) with a text input field containing "Editor".
- Обязательное поле. Не более 150 символов. Только буквы, цифры и символы @/./+/-/\_.** (Mandatory field. No more than 150 characters. Only letters, digits, and symbols @/./+/-/\_).
- Имя:** (Name) with a text input field containing "Editor\_name".
- Адрес электронной почты:** (Email address) with a text input field containing "Editor\_name@editor.ru".
- Password:** with a password input field containing "\*\*\*\*\*".
- Repeat password:** with a password input field containing "\*\*\*\*\*".
- Создать** (Create) button, highlighted in green.

Рисунок 5.5. Создание редактора в системе

## Добавление нового журнала

После авторизации под учетной записью администратора появляется возможность добавление нового журнала, для этого нужно заполнить соответствующую форму (рис. 5.6).

**Название журнала**

Введите название журнала...

**Категория**

Категория не выбрана  
Научные статьи

**Раздел**

Раздел не выбран  
Статьи

**Описание журнала**

**Обложка журнала**

Выберите файл | Файл не выбран

**Файл**

Выберите файл | Файл не выбран

**Год публикации журнала**

2022

**Страна публикации**

-----

**Редакционная коллегия**

Лазарев Дмитрий Денисович  
Верзунев Сергей Николаевич

**Черновик**

☐

Добавить

Рисунок 5.6. Создание журнала

## Подробный просмотр журнала

После добавления журнала его можно увидеть на главной странице сайта (рис. 5.7) где имеется возможность для администратора или ответственного секретаря журнала изменить его (рис. 5.8) а также просмотреть подробную информацию о нем (рис. 5.9).

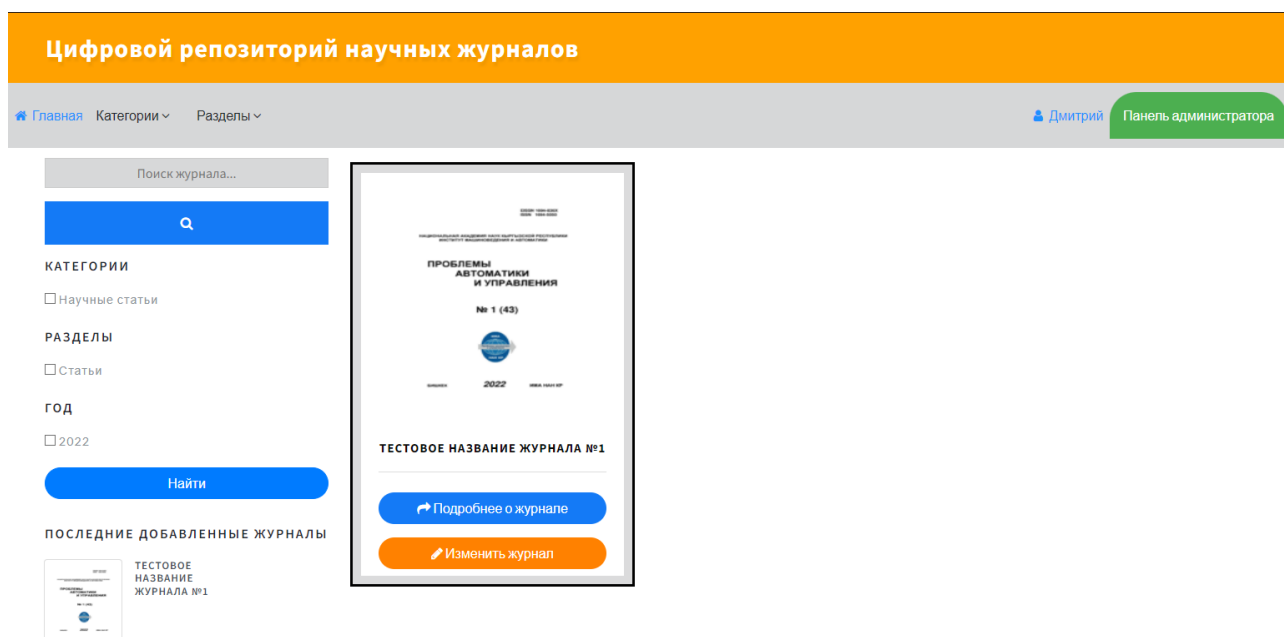


Рисунок 5.7. Главная страница с журналами

## Форма изменение журнала

The screenshot shows a form titled 'Название журнала' for editing journal details. The form includes a text area for 'Описание:' with the placeholder 'Тут будет описание журнала'. Below this are four input fields: 'Категория' (set to 'Научные статьи'), 'Раздел' (set to 'Статьи'), 'Редакционная коллегия' (set to 'Лазарев Дмитрий Денисович' and 'Верзунов Сергей Николаевич'), and 'Год публикации' (set to '2022'). There is also a 'Страна публикации' dropdown set to 'Кыргызстан'. Below these fields are two file upload sections: 'Обложка журнала' and 'Файл', each with a 'На данный момент:' link to a file and an 'Изменить:' button labeled 'Выберите файл'. A 'Черновик' section has a checkbox 'Поставить галочку если да'. At the bottom is a green button labeled 'Внести изменения в журнал'.

Рисунок 5.8. Форма изменения журнала

Изменить данный журнал

Удалить данный журнал

Тестовое название журнала №1

📅 Год выпуска: 2022  
 🌐 Страна: Кыргызстан  
 Описание журнала: Тут будет описание журнала

👤 Редакционная коллегия: Лазарев Дмитрий | Верзунов Сергей |  
 📁 Находится в разделе: Статьи  
 📁 Относится к категории: Научные статьи  
 📎 Файл: [Документ\\_журнала.docx](#)

ИНФОРМАЦИЯ О ЖУРНАЛЕ: ТЕСТОВОЕ НАЗВАНИЕ ЖУРНАЛА №1

Список выпусков в журнале

Название выпуска	Год написания выпуска	Количество статей в выпуске
------------------	-----------------------	-----------------------------

Рисунок 5.9. Подробная информация о журнале

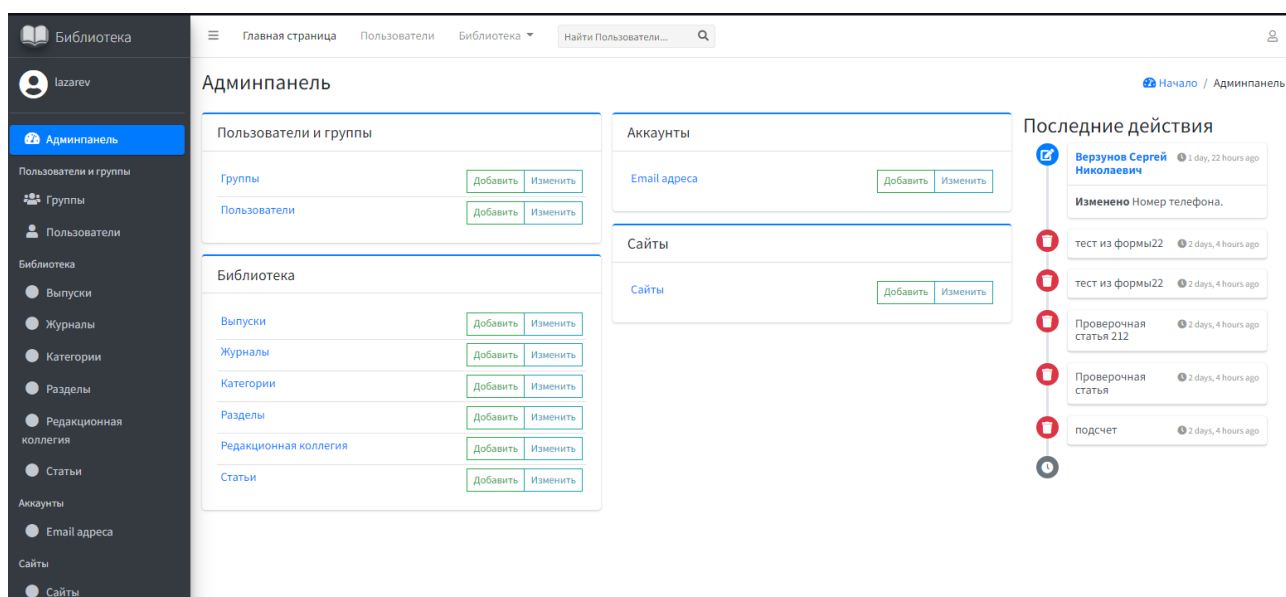


Рисунок 5.10. Панель администратора

## Заключение

В результате выполнения ВКР разработано централизованное хранилище научных статей для обеспечения долговременного и бесперебойного доступа к научным журналам КР при этом, полные тексты статей будут размещаться в открытом доступе для обеспечения свободного использования научной информации для поддержки научных исследований, развития науки и технологического прогресса в нашей стране при обязательном соблюдении прав автора. Данная система выполняет следующие функции:

- Регистрация новых ответственных секретарей журналов
- Авторизация администратора и редакторов;
- Управление пользователями;
- Просмотр каталога проектов;
- Добавление журналов, выпусков и статей;
- Редактирование данных журналов, выпусков и статей;
- Редактированные данных пользователей;
- Удаление журналов, выпусков и статей;
- Добавление журналов, выпусков и статей вручную;
- Просмотр данных статьи.

Средства разработки:

- Среда разработки: PyCharm, Visual Studio Code
- Язык программирования: Python, HTML (JS, CSS)
- Фреймворк: Django
- База данных: MySQL,

Дальнейшее развитие системы включает в себя:

- Импорт полных журналов из системы Open Journal Systems.
- Реализация передачи подготовленных метаданных на проверку ответственному секретарю журнала.

## Список литературы

1. Цифровой репозиторий в информационных научно-образовательных системах / О. А. Федотова, А. М. Федотов, О. Л. Жижимов, М. А. Самбетбаева // Труды ГПНТБ СО РАН. – 2019. – № 3(3). – С. 23-28. – DOI 10.20913/2618-7515-2019-3-23-28. – EDN WNPDOH.
2. <https://studfile.net/preview/8896197/page:2/> (дата обращения 20.06.2022)
3. <https://poznayka.org/s88572t2.html> (дата обращения 20.06.2022)
4. [https://ru.wikipedia.org/wiki/Модульное\\_тестирование](https://ru.wikipedia.org/wiki/Модульное_тестирование) (дата обращения 15.06.2022)
5. Книга The Django Web Framework for the Python programming language:  
<https://django-book.readthedocs.io/en/latest/>
6. Фаулер, Мартин. UML Основы UML Distilled : краткое руководство по стандартному языку объектного моделирования; [пер. с англ. А. Петухова ; предисл. К. Кобрина и др.]. - 3-е изд. - Санкт-Петербург: Символ, 2013. - 184 с
7. Основы объектно-ориентированного представления ПС: Учебник/ С.Орлов. – Спб.:Питер,2016.-186с
8. Леоненков, А. В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose. Курс лекций : учебное пособие для студентов вузов, обучающихся по специальностям в области информационных технологий / А. В. Леоненков. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 318 с.
9. UML диаграммы классов описание: <https://habr.com/ru/post/511798/>
10. ORM, основы Hibernate. Классы–сущности – URL: <https://habr.com/ru/post/29694/>
11. Основы UML. Диаграммы последовательности: <https://pro-prof.com/archives/2769>
12. Основы UML. Диаграммы компонент: <http://khpi-iip.mipk.kharkiv.edu/library/case/leon/gl10/gl10.html>
13. Объектно-ориентированная разработка требований: Учебник/ С.Орлов. – Спб.:Питер,2016.-239с
14. Системное тестирование: <https://coderlessons.com/tutorials/kachestvo-programmnogo-obespecheniia/ruchnoe-testirovanie/testirovanie-sistemy-2>
15. Системные тесты: <https://habr.com/ru/post/81226/>

## **Приложение 1. Глоссарий**

### **1. Введение**

#### **1.1. Цель**

Глоссарий содержит описание терминов, используемых при проектировании цифрового репозитория научных журналов. Определяются основные понятия, непосредственно связанные с конвертацией метаданных.

#### **1.2. Контекст**

Глоссарий создан в рамках разработки Web – портала национальной электронной библиотеки периодических изданий ВАК КР

### **2. Определения**

#### **2.1. Понятия, используемые при описании исходной информации**

##### **Научная статья**

Научная статья – законченное авторское произведение, описывающее результаты оригинального научного исследования (первичная научная статья) или посвящённая рассмотрению ранее опубликованных научных статей, связанных общей темой (обзорная научная статья).

##### **Конвертация**

Конвертация данных - является одним из компонентов технологии обмена данными через формат EnterpriseData. Главное назначение конвертации данных - это создание программного кода модуля менеджера обмена, состоящего из процедур и функций, в которых реализована логика загрузки данных, представленных в формате EnterpriseData, а также логика выгрузки данных в формат.

##### **Ответственный секретарь журнала**

Ответственный секретарь журнала - пользователь системы, наделенный правами на редактирование публикаций.

##### **Администратор**

Администратор - пользователь системы, наделенный правами изменения в системе. Он может добавлять, удалять, изменять информация на сайте

##### **Open Journal Systems**

Open Journal Systems - открытое программное обеспечение для организации рецензируемых научных изданий.



**Web-Платформа** – это многоцелевая платформа для разработки веб-проектов и управления содержимым. Она представляет собой набор комплексных решений, направленных на легкую и успешную разработку веб-сайта и его поддержку.

**Сайт** - информационная система, предоставляющая пользователям сети Интернет доступ к своему содержимому и функционалу в виде упорядоченного набора взаимосвязанных HTML-страниц.

**Читатель** – может зайти на сайт и ознакомиться со всей информацией о журналах, выпусках и статьях

### **Метаданные**

Метаданные раскрывают сведения о признаках и свойствах, характеризующих какие-либо сущности, позволяющие автоматически искать и управлять ими в больших информационных потоках. Метаданные в рамках настоящего проекта – это данные, характеризующие статью: заголовок, аннотация, ключевые слова, текст статьи.

## Приложение 2. Листинг

```
from django.db import models
from django.urls import reverse

COUNTRY_CHOICES = (
    ("Россия", "Россия"),
    ("Кыргызстан", "Кыргызстан"),
    ("Казахстан", "Казахстан"),
    ("Узбекистан", "Узбекистан"),
)

class Category(models.Model):
    "Категории"
    id = models.AutoField(primary_key=True)
    name = models.CharField("Категория", max_length=100)
    def __str__(self):
        return self.name

    class Meta:
        verbose_name = "Категория"
        verbose_name_plural = "Категории"
        ordering = ['id']

class Genre(models.Model):
    "Разделы"
    id = models.AutoField(primary_key=True)
    name = models.CharField("Раздел", max_length=50)

    def __str__(self):
        return self.name

    class Meta:
```

```
verbose_name = "Раздел"
verbose_name_plural = "Разделы"
ordering = ['id']
```

```
class ScientificEditors(models.Model):
```

```
    """Редакционная коллегия"""
```

```
    id = models.AutoField(primary_key=True)
```

```
    name = models.CharField("Имя", blank=True, max_length=50)
```

```
    surname = models.CharField("Фамилия", blank=True, max_length=50)
```

```
    patronymic = models.CharField("Отчество", blank=True, max_length=50)
```

```
    phone_number = models.CharField("Номер телефона", blank=True, max_length=20)
```

```
    image = models.ImageField("Изображение", blank=True, upload_to="Editors/")
```

```
    email = models.EmailField(blank=True)
```

```
    def __str__(self):
```

```
        return self.surname + ' ' + self.name + ' ' + self.patronymic
```

```
    def get_absolute_url(self):
```

```
        return reverse('editor_detail', kwargs={"slug": self.name})
```

```
class Meta:
```

```
    verbose_name = "Редакционная коллегия"
```

```
    verbose_name_plural = "Редакционная коллегия"
```

```
    ordering = ['id']
```

```
class Journal(models.Model):
```

```
    """Журналы"""
```

```
    id = models.AutoField(primary_key=True)
```

```
    name = models.CharField('Название', max_length=100, unique=True)
```

```
    category = models.ManyToManyField(Category, verbose_name='Категория')
```

```
    genres = models.ManyToManyField(Genre, verbose_name="Раздел")
```

```
    description = models.TextField("Описание")
```

```
    image = models.ImageField('Изображение', upload_to='journal/')
```

```

files = models.FileField(upload_to='journal/', blank=False)
publication_date = models.PositiveSmallIntegerField('Дата выхода', default='2022')
country = models.CharField('Страна', choices=COUNTRY_CHOICES)
editors = models.ManyToManyField(ScientificEditors, verbose_name="Редакционная
коллегия", related_name='journal_editors')
draft = models.BooleanField('Черновик', default=False)

def filedir(self):
    return str(self.files).replace('journal/', '')

def __str__(self):
    return self.name

def get_absolute_url(self):
    return reverse('journal_detail', kwargs={"slug": self.name})

def get_update_url(self):
    return reverse('journal_update', kwargs={"slug": self.name})

def get_delete_url(self):
    return reverse('journal_delete', kwargs={"pk": self.id})

class Meta:
    verbose_name = 'Журнал'
    verbose_name_plural = 'Журналы'
    ordering = ['id']

class Issues(models.Model):
    """Выпуски"""
    id = models.AutoField(primary_key=True)
    name = models.CharField('Название', max_length=100)
    journal = models.ForeignKey(Journal, verbose_name='Журнал', on_delete=models.SET_NULL,
null=True)
    description = models.TextField("Описание")

```

```

files = models.FileField(upload_to='Issues/', blank=False)
publication_date = models.PositiveSmallIntegerField('Год выпуска', default='2022')

def __str__(self):
    return self.name

def filedir(self):
    return str(self.files).replace('Issues/', '')

def get_absolute_url(self):
    return reverse('issues_detail', kwargs={'slug': self.name})

def get_update_url(self):
    return reverse('issues_update', kwargs={"slug": self.name})

def get_delete_url(self):
    return reverse('issues_delete', kwargs={"pk": self.id})

@property
def items(self):
    return self.items.set_all().count()

class Meta:
    verbose_name = 'Выпуск'
    verbose_name_plural = 'Выпуски'
    ordering = ['id']

class Items(models.Model):
    """Статьи"""
    id = models.AutoField(primary_key=True)
    name = models.CharField("Имя", blank=True, max_length=50)
    issues = models.ForeignKey(Issues, verbose_name='Выпуск', on_delete=models.SET_NULL,
null=True)

```

```

files = models.FileField(upload_to='Items/', blank=False)
description = models.TextField("Описание")
image = models.ImageField('Изображение', upload_to='items/')
publication_date = models.PositiveSmallIntegerField('Год написания статьи', default='2022')
country = models.CharField('Страна', max_length=30, choices=COUNTRY_CHOICES)
editors = models.ManyToManyField(ScientificEditors, verbose_name="Редактор")


def __str__(self):
    return self.name


def filedir(self):
    return str(self.files).replace('Items/', '')


def get_absolute_url(self):
    return reverse('items_detail', kwargs={"slug": self.name})


def get_update_url(self):
    return reverse('items_update', kwargs={"slug": self.name})


def get_delete_url(self):
    return reverse('items_delete', kwargs={"pk": self.id})


class Meta:
    verbose_name = "Статья"
    verbose_name_plural = "Статьи"
    ordering = ['id']

from django.db.models import Q
from django.shortcuts import render, redirect
from django.views.generic import ListView, DetailView, UpdateView, DeleteView
from .forms import UserRegistrationForm


from library.models import Journal, Category, Genre, ScientificEditors, Issues, Items
from .forms import JournalForm, IssuesForm, ItemsForm

```

```

class JournalFilter:
    """Фильтрация журналов"""
    """По категории"""
    def get_category(self):
        return Category.objects.all()

    """По жанру"""
    def get_genre(self):
        return Genre.objects.all()

    """По годам"""
    def get_years(self):
        return Journal.objects.filter(draft=False).values('publication_date').distinct()

class Genres:
    def get_name(self):
        return Genre.objects.all()

class JournalView(JournalFilter, Genres, ListView):
    """Список журналов"""
    model = Journal
    queryset = Journal.objects.filter(draft=False)
    paginate_by = 6

class JournalDetailView(DetailView):
    """Описание журналов"""
    model = Journal
    slug_field = "name"

def create_journal(request):
    """Создание журнала"""
    error = "
    if request.method == 'POST':

```

```

form = JournalForm(request.POST, request.FILES)
if form.is_valid():
    form.save()
    return redirect('/')
else:
    error = 'Неправильно заполнена форма добавления журнала'

form = JournalForm()

data = {
    'form': form,
    'error': error
}

return render(request, 'library/create_journal.html', data)

class JournalUpdateView(DetailView, UpdateView):
    """изменение журналов"""
    model = Journal
    template_name = 'library/tags/journal_update.html'
    slug_field = "name"
    fields = ['name',
              'category',
              'genres',
              'description',
              'image',
              'files',
              'publication_date',
              'country',
              'editors',
              'draft',
              ]

```



```

class JournalDeleteView(DeleteView):
    """удаление журналов"""
    model = Journal
    success_url = '/'
    template_name = 'library/tags/journal_delete.html'

class IssuesDetailView(DetailView):
    """Описание выпусков"""
    model = Issues
    template_name = 'library/issues_detail.html'
    slug_field = "name"

def create_issues(request):
    """Создание выпуска"""
    error = "
    if request.method == 'POST':
        form = IssuesForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            return redirect('/')
        else:
            error = 'Неправильно заполнена форма добавления выпуска'

    form = IssuesForm()

    data = {
        'form': form,
        'error': error
    }

    return render(request, 'library/create_issues.html', data)

```

```

class IssuesUpdateView(DetailView, UpdateView):
    """изменение выпуска"""
    model = Issues
    template_name = 'library/tags/issues_update.html'
    slug_field = "name"
    fields = ['name',
              'journal',
              'description',
              'files',
              'publication_date',
              ]

```

```

class IssuesDeleteView>DeleteView):
    """удаление выпусков"""
    model = Issues
    success_url = '/'
    template_name = 'library/tags/issues_delete.html'

```

```

class ItemsDetailView(DetailView):
    """Описание статей"""
    model = Items
    template_name = 'library/items_detail.html'
    slug_field = "name"

```

```

def create_items(request):
    """Создание статьи"""
    error = "
    if request.method == 'POST':
        form = ItemsForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            return redirect('/')
    else:

```

```
error = 'Неправильно заполнена форма добавления статьи'
```

```
form = ItemsForm()
```

```
data = {  
    'form': form,  
    'error': error  
}
```

```
return render(request, 'library/create_items.html', data)
```

```
class ItemsUpdateView(DetailView, UpdateView):
```

```
    """изменение статьи"""
```

```
    model = Items
```

```
    template_name = 'library/tags/items_update.html'
```

```
    slug_field = "name"
```

```
    fields = ['name',  
             'issues',  
             'description',  
             'files',  
             'image',  
             'publication_date',  
             'country',  
             'editors',  
             ]
```

```
class ItemsDeleteView(DeleteView):
```

```
    """удаление статей"""
```

```
    model = Items
```

```
    success_url = '/'
```

```
    template_name = 'library/tags/items_delete.html'
```

```

class EditorView(DetailView):
    """Описание редакторов"""
    model = ScientificEditors
    template_name = 'library/Editor.html'
    slug_field = "name"

class Search(JournalFilter, ListView):
    """Поиск"""
    paginate_by = 6

    def get_queryset(self):
        return Journal.objects.filter(name__iregex=self.request.GET.get("q"))

def register(request):
    """Регистрация юзера"""
    if request.method == 'POST':
        user_form = UserRegistrationForm(request.POST)
        if user_form.is_valid():
            new_user = user_form.save(commit=False)
            new_user.set_password(user_form.cleaned_data['password'])
            new_user.save()
            return redirect('/')
        else:
            user_form = UserRegistrationForm()
    return render(request, 'library/create_user.html', {'user_form': user_form})

class FilterJournalView(JournalFilter, ListView):
    """Фильтр"""
    paginate_by = 6

    def get_queryset(self):
        queryset = Journal.objects.filter(
            Q(publication_date__in=self.request.GET.getlist("publication_date")) |

```

```

        Q(category__in=self.request.GET.getlist("category")) |
        Q(genres__in=self.request.GET.getlist("genre"))
    ).distinct()
    return queryset

def showthis(request):
    item = Items.objects.all()

    context = {'item': item, 'item_count': len(item)}

    return render(request, 'library/journal_detail.html', context)

from django.conf import settings
from django.conf.urls.static import static
from django.urls import path

from library import views

urlpatterns = [
    path("", views.JournalView.as_view(), name="home"),
    path("filter/", views.FilterJournalView.as_view(), name="filter"),
    path("create_user/", views.register, name="create_user"),
    path("search/", views.Search.as_view(), name="search"),
    path("create_journal/", views.create_journal, name="create_journal"),
    path("create_issues/", views.create_issues, name="create_issues"),
    path("create_items/", views.create_items, name="create_items"),
    path("<str:slug>/update_issues", views.IssuesUpdateView.as_view(), name="issues_update"),
    path("<str:slug>/update_journal", views.JournalUpdateView.as_view(), name="journal_update"),
    path("<str:slug>/update_items", views.ItemsUpdateView.as_view(), name="items_update"),
    path("<int:pk>/delete_journal", views.JournalDeleteView.as_view(), name="journal_delete"),
    path("<int:pk>/delete_issues", views.IssuesDeleteView.as_view(), name="issues_delete"),
    path("<int:pk>/delete_items", views.ItemsDeleteView.as_view(), name="items_delete"),
    path("<str:slug>/", views.JournalDetailView.as_view(), name="journal_detail"),

```

```

path("journal/<str:slug>/", views.IssuesDetailView.as_view(), name="issues_detail"),
path("issues/<str:slug>/", views.ItemsDetailView.as_view(), name="items_detail"),
path("editor/<str:slug>/", views.EditorView.as_view(), name="editor_detail"),
]

```

```

if settings.DEBUG:

```

```

    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

```

import instance as instance

```

```

from django import forms

```

```

from django.forms import ModelForm, TextInput, Select, FileInput, DateInput, Textarea,
SelectDateWidget, CheckboxInput, \

```

```

    SelectMultiple, DateTimeInput

```

```

from django.contrib.auth.models import User

```

```

from .models import Journal, Items, Issues, Genre, Category

```

```

class JournalForm(forms.ModelForm):

```

```

    def __init__(self, *args, **kwargs):

```

```

        super().__init__(*args, **kwargs)

```

```

        self.fields['category'].empty_label = "Категория не выбрана"

```

```

        self.fields['genres'].empty_label = "Раздел не выбран"

```

```

class Meta:

```

```

    model = Journal

```

```

    fields = ['name',

```

```

               'category',

```

```

               'genres',

```

```

               'description',

```

```

               'image',

```

```

               'files',

```

```

               'publication_date',

```

```

        'country',
        'editors',
        'draft',
    ]
widgets = {
    'name': TextInput(attrs={
        'class': 'form-control',
        'placeholder': 'Введите название журнала...'
    }),
    'category': SelectMultiple(attrs={
        'class': 'form-control',
    }),
    'genres': SelectMultiple(attrs={
        'class': 'form-control',
    }),
    'description': Textarea(attrs={
        'class': 'form-control',
    }),
    'image': DateTimeInput(attrs={
        # 'class': 'form-control',
    }),
    'files': FileInput(attrs={
        'class': 'form-control',
    }),
    'publication_date': TextInput(attrs={
        'class': 'form-control',
    }),

    'editors': SelectMultiple(attrs={
        'class': 'form-control',
    }),
    'draft': CheckboxInput(attrs={
        'class': 'form-check-label',

```

```
    }),  
}
```

```
class IssuesForm(forms.ModelForm):  
    def __init__(self, *args, **kwargs):  
        super().__init__(*args, **kwargs)  
        self.fields['journal'].empty_label = "Журнал не выбран"
```

```
class Meta:  
    model = Issues  
    fields = ['name',  
              'journal',  
              'description',  
              'files',  
              'publication_date',  
              ]
```

```
    widgets = {  
        'name': TextInput(attrs={  
            'class': 'form-control',  
            'placeholder': 'Введите название выпуска...'  
        }),  
        'description': Textarea(attrs={  
            'class': 'form-control',  
        }),  
        'files': FileInput(attrs={  
            'class': 'form-control',  
        }),  
        'publication_date': TextInput(attrs={  
            'class': 'form-control',  
        }),  
    }  
}
```



```

class ItemsForm(forms.ModelForm):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.fields['issues'].empty_label = "Выпуск не выбран"

class Meta:
    model = Items
    fields = ['name',
              'issues',
              'description',
              'files',
              'image',
              'publication_date',
              'country',
              'editors',
              ]

    widgets = {
        'name': TextInput(attrs={
            'class': 'form-control',
            'placeholder': 'Введите название статьи...'
        }),
        'description': Textarea(attrs={
            'class': 'form-control',
        }),
        'files': FileInput(attrs={
            'class': 'form-control',
        }),
        'publication_date': TextInput(attrs={
            'class': 'form-control',
        }),
    }

```

```
class UserRegistrationForm(forms.ModelForm):
    password = forms.CharField(label='Password', widget=forms.PasswordInput)
    password2 = forms.CharField(label='Repeat password', widget=forms.PasswordInput)

    class Meta:
        model = User
        fields = ('username', 'first_name', 'email')

    def clean_password2(self):
        cd = self.cleaned_data
        if cd['password'] != cd['password2']:
            raise forms.ValidationError('Passwords don\'t match.')
        return cd['password2']
```